University of the Philippines

College of Arts and Sciences

Department of Physical Sciences and Mathematics

Around Metro:

A Mobile Public Transport – Geographical Information System for Metro Manila

A special problem in partial fulfillment

of the requirements for the degree of

Bachelor of Science in Computer Science

Submitted by:

Allen Kenneth L. Narvaza

April 2014

Permission is given for the following people to have access to this SP:

Available to the general public	Yes
Available only after consultation with author/SP adviser	No
Available only to those bound by confidentiality agreement	No

ACCEPTANCE SHEET

The Special Problem entitled "Around Metro: A Mobile Public Transport – Geographical Information System" prepared and submitted by Allen Kenneth L. Narvaza in partial fulfilment of the requirements for the degree of Bachelor of Science in Computer Science has been examined and is recommended for acceptance.

_____

Gregorio B. Baes, Ph.D. (*candidate*)
Adviser

EXAMINERS:

		Approved	**Disapproved**
1.	Avegail D. Carpio, M. Sc.	_____	_____
2.	Richard Bryann L. Chua, M. Sc.	_____	_____
3.	Aldrich Colin K. Co, M. Sc. (*candidate*)	_____	_____
4.	Ma. Sheila A. Magboo, M. Sc.	_____	_____
5.	Vincent Peter C. Magboo, M.D., M.Sc.	_____	_____
6.	Geoffrey A. Solano, M. Sc.	_____	_____
7.	Bernie B. Terrado, M. Sc. (*candidate*)	_____	_____

Accepted and approved as partial fulfilment of the requirements for the degree of Bachelor of Science in Computer Science.

_____

Ma. Sheila A. Magboo, M. Sc.
Unit Head
Mathematical and Computing Sciences Unit
Department of Physical Sciences and Mathematics

_____

Marcelina B. Lirazan, Ph. D.
Chair
Department of Physical Sciences
And Mathematics

_____

Alex C. Gonzaga, Ph. D., Dr. Eng.
Dean
College of Arts and Sciences

ABSTRACT

Transportation is a non-separable part of any society. It has greatly influenced people's way of living and the way in which societies are organized. In Metro Manila, thousands of people travel daily for social and economic activities by using public utility vehicles (PUVs). It is a well-known fact that commuting is the most practical way to move from one location to another. However, the efficiency of using PUVs depends on the person's knowledge about the streets. Around Metro is an Android application that generates directions and guides people on how to get to their destinations by using PUVs. It takes into consideration the user's preferences and traffic information in generating trip suggestions. This application is helpful most especially for first-time commuters in Metro Manila.

Keywords: Trip Planner, PUVs, Transit, Graph, Intermodal Transportation

Contents

I. Introduction

A. Background of the Study

Transport plays a major role in developing the social and economic situation of Asian cities [1]. Basically, people need to travel to different places for social and economic activities such as work, education, and medical care. Most people in Asian cities travel by foot and public transport.

Travelling in the Philippines using public transport system, or commonly called commuting, means either riding one or a combination of public utility vehicles or PUVs such as pedicabs, tricycles, jeepneys, FXs, buses, or trains. In Metro Manila, where transport vehicles are everywhere and PUV routes are well-defined, commuting is the most practical way to move from one location to another especially when you know what to ride and where to ride.

With the advancements in technology nowadays, people can be provided with information on how to commute to their destination. This information is sometimes presented by Geographical Information Systems or commonly called GIS. A GIS is a computer-based system that can store, analyse, and display geographically referenced information [2]. GIS can be invaluable during decision-making process and used at the same time in emergency planning, producing floodplain maps, transportation forecasting, and urban planning.

One common function among GIS Applications is to display map layers [3]. There are various mapping services available in the internet and are offered for free. One promising project is the OpenStreetMap which aims to create a digital map of the world that contains as much detailed information as possible [3]. Studies have shown that,

though data are gathered from volunteers, the data accuracy of OpenStreetMap information was good in major cities and acceptable for mid-sized towns [4].

Worldwide sales of mobile devices for the third quarter of 2011 totalled 440.5 million units [5]. Among the top competing phones, Android became the market leader, surpassing both BlackBerry and iPhone. Upon the introduction of smart phones running in Android, a platform marketed by Google, the landscape of mobile market changed [6].

Google offers Android as an open source solution [6]; since then, most developers see this as a great opportunity and prefer Android as their software development platform. Android phones are packaged with built-in activities including e-mail, a Web browser, and a map application.

Incorporating GIS and Android application development to the field of transportation brings a wide range of possible applications that may be beneficial for businesses and citizens. For commuters, this would be very helpful especially in planning their trips. They can have options and make wise decisions based on their imposed constraints.

B. Statement of the Problem

A person can get lost in Metro Manila mainly because of three reasons; it's either he/she is not familiar with the streets or he/she doesn't know what to ride or both. First time commuters in Metro Manila (i.e. tourists and those coming from the province) will most likely get lost not unless accompanied by someone knowledgeable of commuting around the city.

There are available web GIS made to help commuters in traveling. However, none are still known to give directions around Metro Manila except for Google Transit. Google

Transit is an extension of Google Maps that shows the user the most efficient step-by-step transit directions on how to get to his destination with its available public transportation information [7]. However, as of the moment, Google only has information about the metro's four train lines – LRT1, LRT2, MRT, and the PNR Commuter Train [8]. Thus, the efficiency of the results provided by Google Transit is quite questionable because trains are not always the most efficient option. Also, Google Transit did not consider fare which is one of the most important criteria for choosing the most efficient transport route. Nonetheless, it is still accurate in providing the direction using the four metro trains.

In times of doubt, people ask directions from locals or consult a guide book for directions. But sometimes, they tend to get more lost. People who do not know their destination ride taxis to avoid getting lost or resorted to ride a taxi after getting lost. However, riding a taxi is very costly. Nowadays the flag-down rate is Php 40.00. Also there are opportunist taxi drivers that charge commuters twice the normal fare. Thus, it would be very convenient for commuters to have easy access with a reliable mobile application that provides detailed directions.

C. Objectives of the Study

This project aims to create a public transport information system for Metro Manila that would run in Android phones and utilize OpenStreetMap data to cater the needs of commuters for information regarding intermodal transportation. It is also designed to aid the commuter in wisely choosing a transport route depending on the constraint the user will impose. The system has the following functionalities:

1. Allows the user to:
    (a) enter origin and destination
    (b) edit input by manually pinning location in the map
    (c) view the list of set of PUVs the user can ride to get to his destination which is ranked according to the user's preferences
    (d) rank all travel options by least fare, least travel time, or least distance
    (e) view details of intermodal transport route
2. Show different intermodal transport routes
3. Include the traffic condition of the streets in computing the travel time for each transport routes
4. Present necessary transport route details:
    (a) information about the PUV and the PUV route
    (b) instructions on where to ride and where to get off
    (c) breakdown of fare per transport vehicle
    (d) map of transport route

The system allows a System Administrator to:

1. log-in to the web server
2. add, edit, view, or delete routes
3. add, edit, view, or delete public transport trips
4. add, edit, view, or delete public transport stops
5. view and edit fare matrix
6. update the graph used for planning trips

D. Significance of the Study

There are millions of commuters around Metro Manila everyday. On the average, people allot at least 3 hours of their time for commuting. However, if the commuter is provided with essential transportation information, the amount of travelling can be lessened and travel cost may decrease as well. The proposed system will be very helpful in guiding a commuter plan his trips and maximize his time productively.

For first time commuters in Metro Manila, this system will aid them by rendering reliable directions to their destination. Through this, they can avoid getting lost and stay on track.

Available guide books and maps of Metro Manila are sometimes out-dated and have incorrect information. Also, reading these materials during travel can be stressful and may result to more confusion. This system can fill in those lacking information and supply direct instructions for easy and stress-free travel.

The prevalence of smartphones is evident nowadays. Commuters rely on these devices for information such as weather, announcements, and the like. Since this system is a mobile application, users can have easy access to information regarding his trip anytime, anywhere.

E. Scope and Limitations

1. Works only on mobile phones powered by Android.
2. The public utility vehicles considered are buses, jeepneys, UV Express (under LTFRB), and trains that run under LRT, MRT and PNR. Other PUVs such as pedicabs, tricycles, taxies, and ferries are excluded from the system.
3. There are 2 sets of fares for buses (aircon and non-aircon rates). Since the system cannot detect what type of bus the user will choose, the air-conditioned bus rate is used as default bus rate.
4. Real-time traffic information is directly fetched from a hypothetical Traffic Monitoring System using available data scraping software programs.
5. Rerouted transport vehicles will not be considered in the system.
6. Roads not included in the map will not be considered even though these roads are passable in reality.
7. Information about walking especially during mode transfer will not be included.

F. Assumptions

1. The user has an internet connection to use the application.
2. The user is knowledgeable to approximate his location.
3. All roads in the map are passable. In the case of a one-way road, the direction of the one-way road is ignored because in reality there exists a parallel road of opposite direction nearby and the distance between them is insignificant.
4. Public utility vehicles are available all the time.

II. Review of Related Literature

A comprehensible and presentable public transport information system is considered to be invaluable not just for tourists but for local residents as well. For cities with complex structures and diverse transportation systems, it is very helpful in assisting commuters to efficiently use public transport. The public transport enquiry system (PTES), proposed by Pun-Cheng, is an Internet-based public transport searching engine that provides information about different public transport vehicles and transfer details for any given initial location and destination in Hong Kong [9]. This system is characterized by its capability to show details about intermodal transfers where other systems lack.

The PTES is comprised of three bilingual subsystems that form the essential database for route computation; (1) a public transport subsystem, (2) an address subsystem, and (3) a map subsystem. The public transport subsystem is primarily a collection of databases for all route names, stop-to-stop fare for all routes, position of all stops and rail exits, and other useful information such as stop name, address, district, and sub district. The address subsystem is for user selection of origins and destinations. It includes all tourist spots, street names, site names, and building names and landmarks. The map subsystem, on the other hand, is the interactive GIS application of the system.

One interesting feature of PTES is the inclusion of a precomputed transfer table (TT). The TT was described as a matrix table that shows the feasible exchange of routes for nearby stops. It shows the explicit transfer relationship of nodes in the network. This is similar to the precomputed matrices of k-intermediate vertex of the Floyd algorithm.

The TT was used in route computation and helps in improving response time and limiting the database.

The PTES, however, focused on routing alone and excluded real-time traffic conditions. It was because the system was intended for travellers to plan their trips ahead. Thus, traffic information would be unnecessary since traffic conditions cannot be predicted. It would have been better if the system computes routes with real-time traffic information for the user to have better options.

Peng and Wang developed a web-based application that provides intelligent vehicle navigation service that combines web GIS and mobile phone with GPS module [10] and focuses on improving road congestion. Their system takes into consideration "hinder factors," like turnings and traffic lights, and driving speed to find the shortest-time path. The researchers believed that the shortest path is not always the fastest path. There will always be factors that will affect the speed of motor vehicles, which in this case is called hinder factors. The system set different weights to these factors while computing for the optimal route plan for the user. Moreover, the system utilizes open-source tools such as OpenStreetMap for their data source and presentation layer.

Li and Yi integrated digital maps system and mobile phones [11]. The proposed system is based on Google Maps API technology that is capable of browsing and querying electronic maps, searching for bus lines, and local positioning using mobile phones. The system considers the position of the mobile phone through GPS as the origin and the destination is entered by the user. One result produced by the system is the shortest path between the mobile's location and destination.

In a study by Rifat, Motushy, Ahmed, and Ferdous, a location based information system is introduced that uses OpenStreetMap and runs in Android smartphones. The system serves as an audio road guider that will cater not only the normal people but those visually impaired as well [12]. The researchers highlighted the fact that most existing systems are for normal people and few only delve into finding solutions for disabled people. The study also emphasized that they used OpenStreetMap instead of the "constrained" Google map because it is free and does not have technical or illegal restrictions.

Basically, the system was designed to find any location around the area. The *Audio Help* feature is just one of the main features of the application. Through this feature the visually impaired user is guided to find his destination by listening to an audio description of his current location, which is acquired through GPS, and some important details of his destination.

Another study that utilizes Android technology for assisting travellers was presented by Garcia et al [13]. The system was designed to provide users with real time information about public transport buses. Information such as name of the route, stops, next stop, and estimated time for the destination stop of the traveller is presented through text, acoustic warning, or voice messages.

A similar study was also conducted by Nagaraj et al. and caters two kinds of user: the passenger and the system administrator [14]. The system also provides information about all buses, routes, timings of buses and all stops for passengers using Wi-Fi or GPRS technology. The administrator, on the other hand, is responsible for updating, inserting,

and deleting information from the database. The administrator has all rights about database operations.

Maps are frequently used for routing and navigation. Zheng et al. studied routing services provided by free mapping application program interfaces such as CloudMade API and OpenStreetMap [15]. Their paper discussed about the feasibility of providing detailed navigational services for pedestrians, cyclists, and drivers in small towns. The project utilizes also open source products to cut data costs. They used OpenStreetMap because it is free, open and a fast developing map of the world. The CloudMade API was used to provide multi-mode routing services with turn-by-turn descriptions.

One important characteristic of intermodal transport systems is transfer. Liu focused on organizing transfer data according to different constraints of data quantity, precision, and accuracy [16]. The study managed to present three transfer data models with their own advantages and restrictions. It was also noted that multiple applications may be developed by incorporating these models with other functions and algorithms.

Integrating information from various sources poses a great challenge especially in creating a public transport information system. Mead pointed out in his study the challenges of blending data from sources to provide integrated real-time traffic and travel information to influence the passenger modal choice [17]. The data gathered are presented through LCD displays where passengers can immediately see it. Through this system, it is easier for passengers to make informed choices.

Public transport information system can also be used in supporting a green environment. Bliznyuk proposed a mobile application called, "Green Daily Guide," that aims to motivate people use public transport and other alternative means of

transportation instead of using private cars [18]. Green Daily Guide is a unique project that incorporates routing applications in promoting environmental advocacies. Though first functional prototype is still on the way, the idea of applying route-generating applications to projects involving our environment is brilliant and feasible. Through Green Daily Guide, public transport systems may be maximized while environmental hazards are minimized at the same time.

III. Theoretical Framework

A. Land Transportation Franchising and Regulatory Board

Land Transportation Franchising and regulatory Board or commonly called LTFRB is a government agency in the Philippines mandated to promulgate, administer, enforce, and monitor compliance policies, laws, and regulations of public land transportation services [19].

The LTFRB has the power to prescribe and regulate routes of service and the areas of operation of public land transportation services provided by motorized vehicles and approved by the Department of Transportation and Communication. Additionally, the agency has the responsibility to determine, prescribe, approve, and periodically review and adjust reasonable fares, rates and other related charges relative to the operation of motorized vehicles.

B. Light Rail Transit Authority

The Light Rail Transit Authority or LRTA is acknowledged as the premiere rail transit in the Philippines that provide reliable, dependable, and environment-friendly mass rail service for citizens in Metro Manila [20]. It is responsible for construction, operation, maintenance and/or lease of light rail transit in the country.

Currently, LRTA has two functioning rail systems; the LRT Line 1 system and the LRT Line 2 system. The LRT Line 1 system, or the Yellow Line, runs from Taft Avenue to Rizal Avenue between Baclaran, Pasay City, and Bonifacio Monument in Caloocan

City. The LRT Line 2 system, the Purple Line, traverses five cities in Metro Manila namely Pasig, Marikina, Quezon, SanJuan, and Manila. LRT Line 2 runs from Santolan, Pasig City to Recto in Manila.

C. Metro Rail Transit Corporation

The Metro Rail Transit Corporation or MRTC maintains and operates the Metro Rail Transit 3. Commonly called MRT, this rail system stretches along EDSA from North Avenue in Quezon City to Taft Avenue, Pasay City [21]. The MRT is designed to carry 23, 000 passengers to 48, 000 passengers per hour, per direction. The MRT is the cornerstone of Department of Transportation and Communication's strategy to alleviate the chronic traffic congestion along EDSA.

D. Graph

A graph is a collection of vertices and edges. Vertices are representing entities that can have names and properties while edges connect these vertices and represent a relationship between the connected vertices.

As shown in Figure 1, a path from vertex A to D is a list of vertices which successive vertices are connected by edges in the graph. A simple path is which no vertex is repeated. The figure on the left shows a cycle. A cycle is a path where the first and the last vertices are the same.

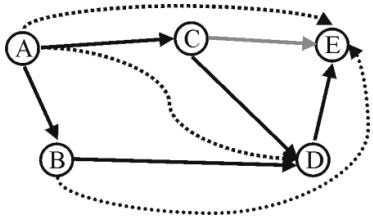


Figure 3.1 Illustration of a path

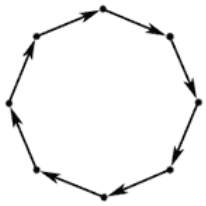


Figure 3.2 Illustration of a cycle

Graphs may be classified as either directed or undirected. An undirected graph, as seen in Figure 3, is a finite set of vertices with a finite set of edges. Both sets may be empty in which case is called an empty graph. The edges are associated with one or more vertices and ordering of vertices is not significant. A directed graph, as shown in Figure 4, has edges that are "one-way": an edge may go from x to y but not from y to x.

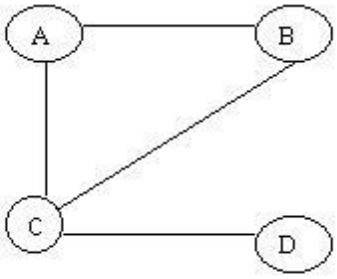


Figure 3.3 An undirected graph

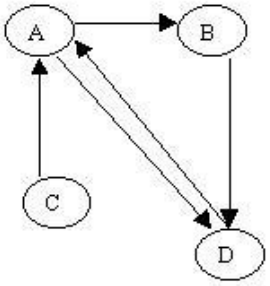


Figure 3.4 A directed graph

Weighted graphs have integers, or commonly known as weights, assigned to each edge to represent distances or costs. Directed weighted graphs are sometimes called networks.

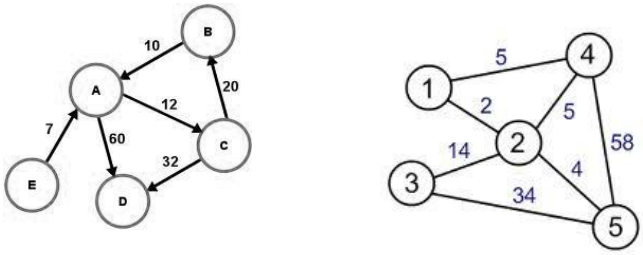


Figure 3.5 A weighted directed graph (right) and a weighted undirected graph (left)

E. Pathfinding Algorithms

Pathfinding strategies are the core of many AI movement systems. Pathfinding strategies are responsible for finding a path from any two coordinates [23]. Various systems use these algorithms to find a series of points that together comprise a path between the starting point and the destination. Usually the path found is the shortest path possible. An analogy of this algorithm would be a simple scenario where a man is about to walk across a room. Rather than finding for other possible path, the person would generally walk in the direction of the destination and avoid any obstructions.

F. A* Algorithm

The A* algorithm is one of the most popular pathfinding algorithms. It is flexible and like other graph-searching algorithms, it can search a huge area of map. It combines the information Dijkstra's algorithm uses (favouring vertices that are close to the origin) and the information that Best-First-Search uses (favouring vertices that are closed to the goal) [24].

As A* algorithm iterate its main loop, it examines the vertex *n* that has the lowest f (n) = g (n) + h (n), where g (n) is the exact cost of the path from the starting point to any vertex n, and h (n) representing the heuristic estimated cost form vertex n to the goal.

G. Intermodal Transportation

Movement of freight is often associated with intermodal transportation. However, intermodal transportation may also refer to movement of people between destinations. A useful definition of intermodal transportation is the movement of goods and people employing more than one form of transformation for a single delivery or trip [25]. The difficulty in intermodal transportation lies in its inter-mode aspect. Seamless movement of goods and people between destinations is the key goal of transportation planners for an efficient and effective intermodal transportation.

H. Geographical Information System

Geographical Information System (GIS) is an information system that allows users to store, maintain, analyse, and display geographic information. The information managed in a GIS has a geographic or spatial context that makes it different form other database systems. GIS combines data from various sources into one common map form based on a common geographic location. Through this representation, the user can clearly visualize the data and see how these data interact. These lead to a better understanding and interpretation of information and a more informed decision process [26].

1. Components of a GIS

a. Digital Data - the most important component of a GIS and often the most expensive. These data may be collected in-house or bought from a commercial data provider [26]. There are two types of data that GIS utilizes [27]: the spatial

data, which describes the absolute and relative location of geographic feature (i.e. coordinate location of a building), and the attribute data which describes the quantitative and/or qualitative characteristics (i.e. name of the building, owner of the building).

b. Software – GIS software provides the functions and tools needed to store, analyse, and display geographic information [26]. Key software components include a database management system (DBMS), tools for the input and manipulation of geographic information, tools that support geographic query, analysis, and visualization, and a graphical user interface (GUI) for easy access of tools.

c. Hardware – this is where the computer where the GIS operates [26]. Nowadays, GIS runs on multiple hardware types, from centralized computer servers to desktop computers used in stand-alone or networked configurations.

d. People - GIS technology is of limited value without the people who deals with the design programming, operation, and management of GIS [26]. Users of GIS range from technical specialists who design and maintain the system.

e. **Procedure –** this component contains the well-designed plan and business rules on how a GIS operates [26]. It includes the manipulation of data, storing them into database, manage, transform, analyse and present them in a final output.

I. Geocoding

Geocoding is the process of assigning locations to addresses so that they can be placed as points on a map, like putting pins on a paper map, and then analysed with other special spatial data [28]. The process includes assigning geographic coordinates to the original data (i.e. latitude and longitude pair (37.423021, 122.083739) to street address 1600 Amphitheatre Parkway, Mountain View, CA).

J. Web Scraping

Web Scraping is a form of data scraping and pertains to various methods used to collect information from across the internet [29]. This is also called Web data extraction, screen scraping or Web harvesting. Generally, the process involves software that simulates human Web surfing to collect valuable information from various websites. This process saves the effort and cost to collect necessary data from the internet.

K. Intermodal Journey Planner

An intermodal journey planner can also be called as a traveller information system. This system usually works as follows: The user indicates his starting point and his destination. The system then computes for optimum trips for private modes such as

car, bicycle, etc. and for all public transport vehicles [30]. It is up to the user to compare the possibilities and determine which one is the most efficient.

L. OpenTripPlanner

OpenTripPlanner (OTP), launched in 2009, is an open-source platform for multi-modal and multi-agency journey planning [22]. It follows a client-server model, providing several map-based web interfaces as well as a REST API for use by third-party applications. The project heavily relies on open data standards such as General Transit Feed Specification for transit and OpenStreetMap for street networks.

M. Database Management System

A database management system (DBMS), sometimes just called a database manager, is a program that lets one or more computer users create and access data in a database. It takes in the requests of users and manages it in a secure and organized way. Familiar user and program interface is the SQL or Structured Query Language, but there is also the ODBMS or the Object Oriented Database Management System.

IV. Design and Implementation

A. Context Diagram

Figure 4.1 shows the context diagram of Around Metro. The system will have two users, the application users and the system administrator. The application users have the capability of viewing directions and route details given that they key in their location and destination. The system administrator, on the other hand, is responsible for updating the database tables and the graph for better accuracy of data. He has the privilege of adding, modifying, and deleting public transport vehicles, routes, trips, stops, and fares.

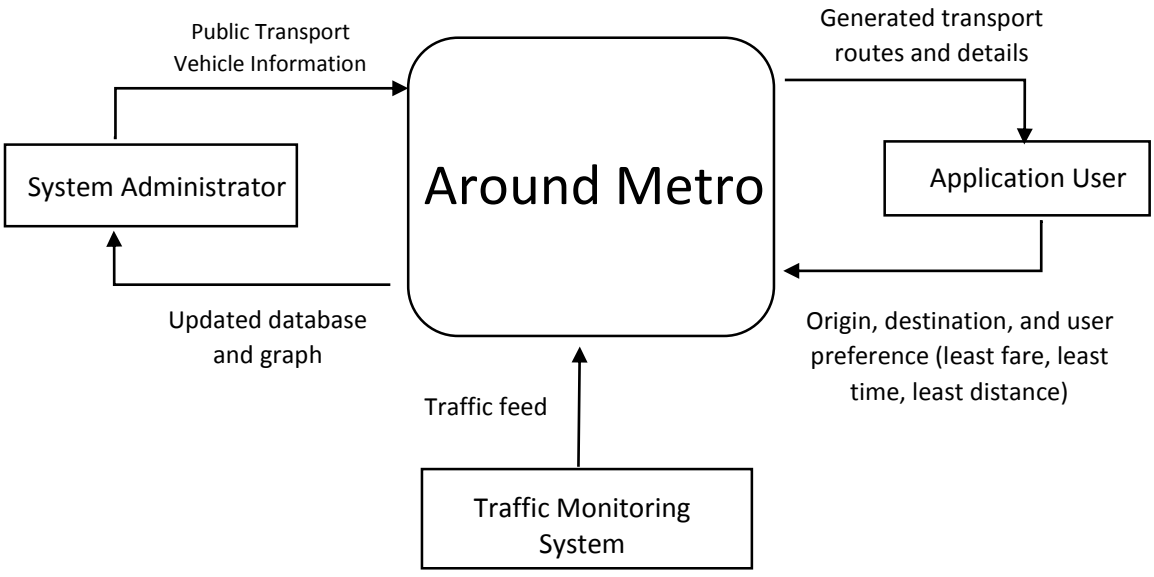


Figure 4.1 Context diagram for Around Metro

B. Data Flow Diagram

The top level data flow diagram for Around Metro, as seen in **Figure 4.2**, shows the different processes for each user; view travel options and view direction details for application users and login, managing PUVs, and updating the graph and database for the system administrator.

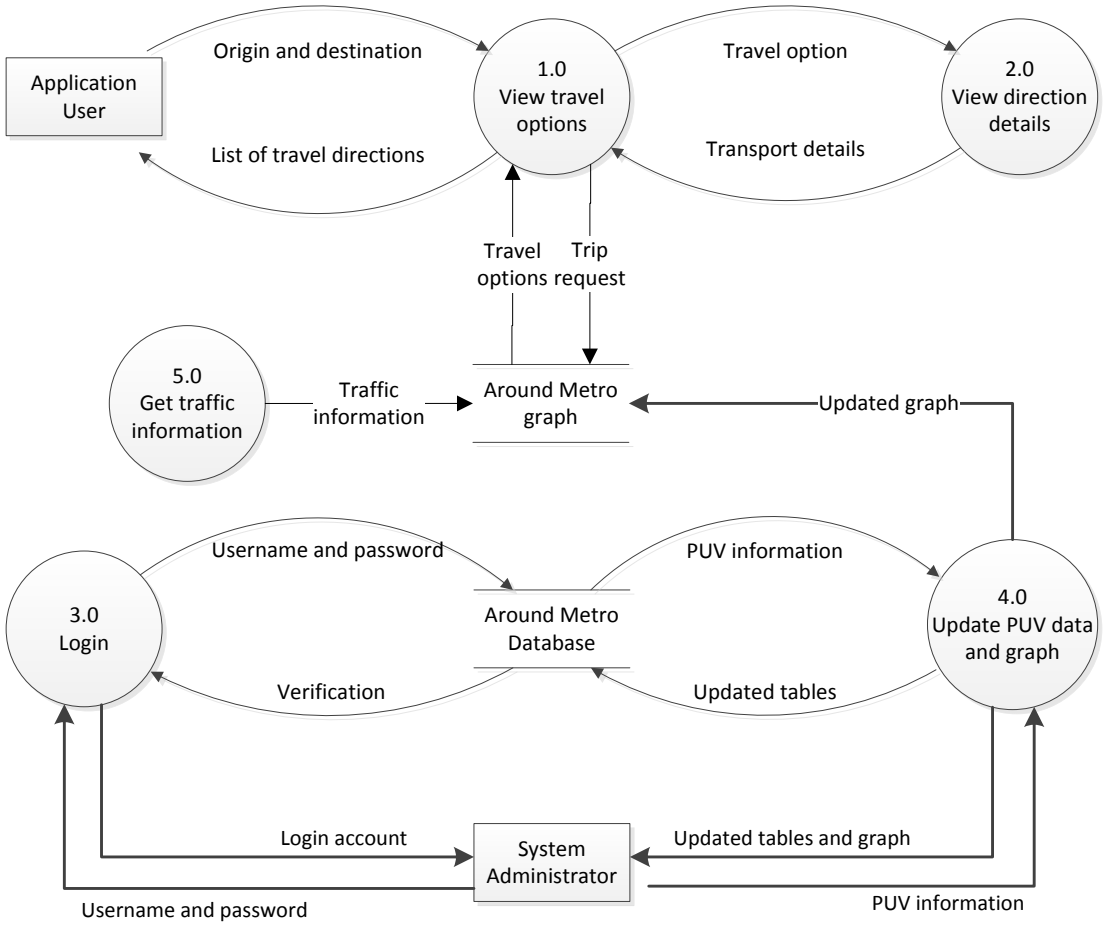


Figure 4.2 Data Flow Diagram for Metro Manila Public Transport Information System

The sub-explosions of processes in **Figure 4.2** will be shown in the following figures:

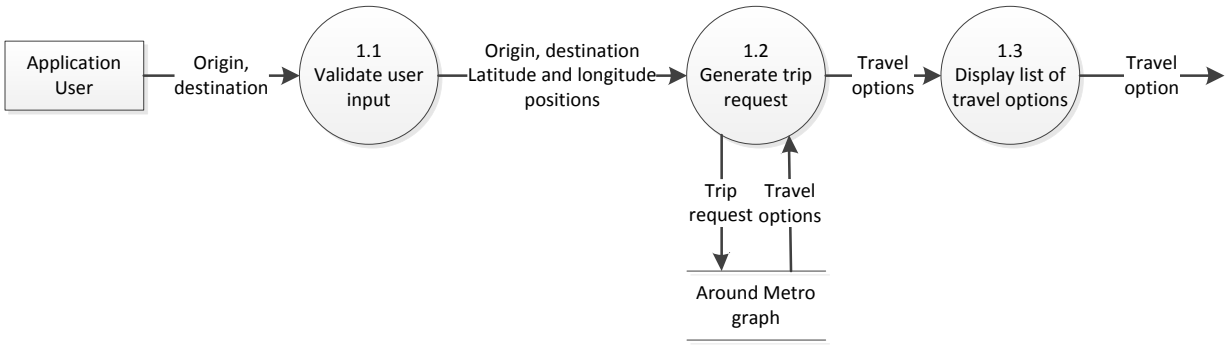


Figure 4.3 Sub-explosion for viewing travel options

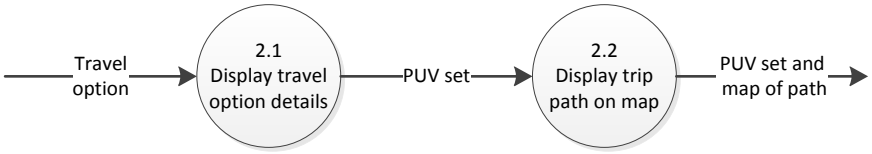


Figure 4.4 Sub-explosion for viewing travel option details

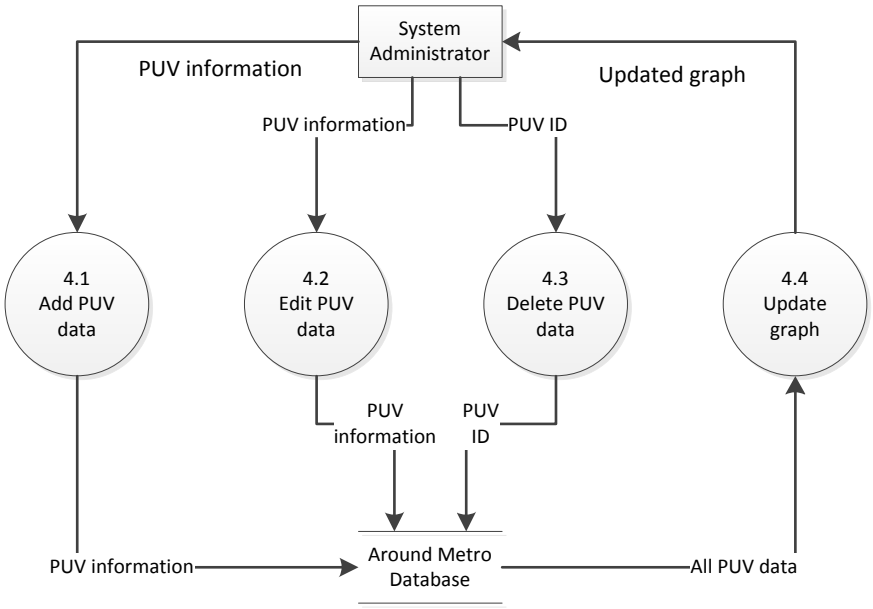


Figure 4.5 Sub-explosion updating PUV data and graph

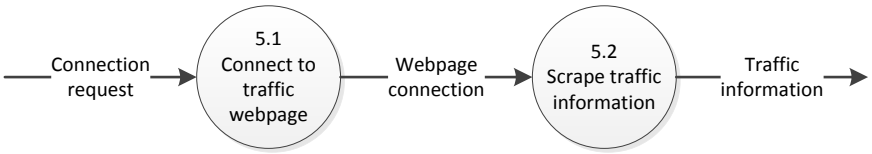


Figure 4.6 Sub-explosion of web scraping traffic information

C. Entity Relationship Diagram

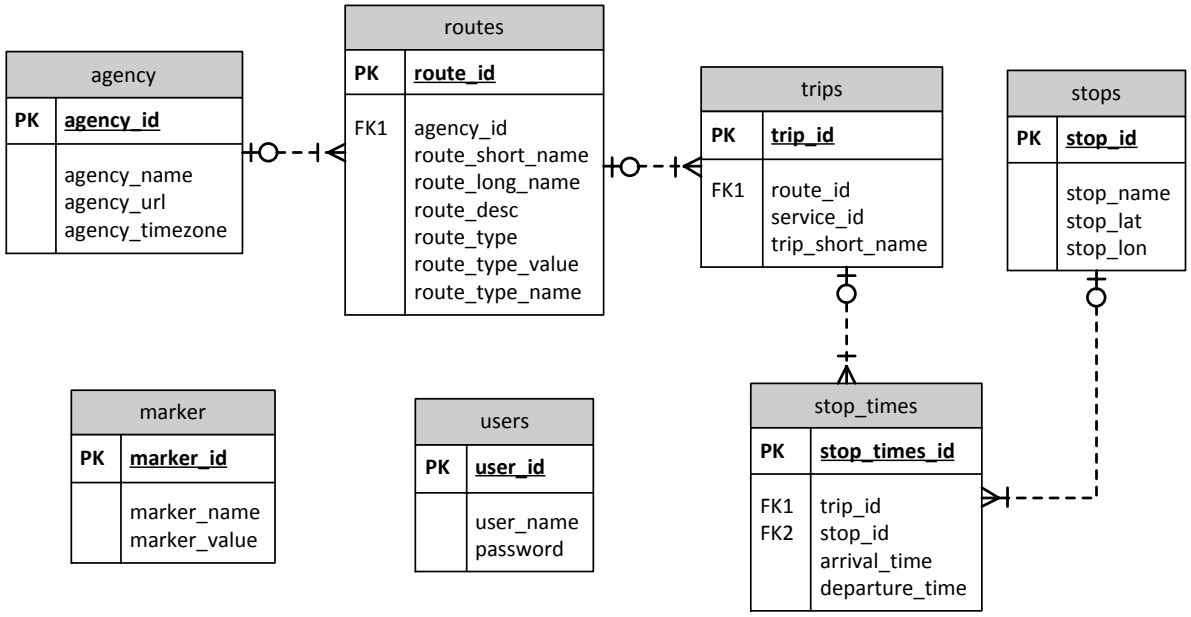


Figure 4.7 Entity Relationship Diagram for Around Metro

D. Data Dictionary

Data Field	Data Type	Description
agency_id (**PK**)	varchar(10)	Identification number that uniquely identifies a transit agency
agency_name	varchar(100)	Full name of the transit agency
agency_url	varchar(100)	Contains the URL of the transit agency
agency_timezone	varchar(30)	Contains the timezone where the transit agency is located

TABLE 4.1: agency contains the information about transit agencies

Data Field	Data Type	Description
route_id (**PK**)	varchar(20)	Contains the ID that uniquely identifies a route
route_short_name	varchar(50)	Contains the short name of a route
route_long_name	varchar	Contains the full name of a route
route_desc	varchar(300)	Contains the description of a route
route_type	int(2)	Type of transportation used on route
route_type_name	varchar(20)	Name of transportation type used on a route
agency_id (**FK**)	varchar(10)	Defines the agency for the specified route

TABLE 4.2: routes contains the information about the transit routes in Metro Manila

Data Field	Data Type	Description
stop_id (**PK**)	varchar(15)	Contains the ID that uniquely identifies a stop or station
stop_name	varchar(300)	Contains the name of a stop or station
stop_lat	decimal(9,6)	Contains the latitude of a stop or station
stop_lon	decimal(9,6)	Contains the longitude of a stop or station

TABLE 4.3: stops contains the information about stops where vehicle pick up or drop off passengers

Data Field	Data Type	Description
trip_id (**PK**)	int(6)	Contains the ID that identifies a trip
route_id	varchar(20)	Contains the id that uniquely identifies a route
trip_short_name	varchar(100)	Contains the text that identify the trip
service_id	int(6)	Contains an ID that uniquely identifies a set of dates when service is available
route_end_long	float(10,6)	Longitude of route end point
PUV_id (**FK**)	int(2)	Identifies the PUV where the route belongs to

TABLE 4.4: trips contains the information about trip/s for each route

Data Field	Data Type	Description
stop_times_id (**PK**)	int(6)	Contains the ID that uniquely identifies a stop or station
trip_id	int(6)	Contains the name of a stop or station
stop_id	varchar(15)	Contains the latitude of a stop or station
arrival_time	time	Contains the longitude of a stop or station
departure_time	time	Contains the longitude of a stop or station

TABLE 4.5: stop_times contains the information about the time a vehicle arrives at and departs from individual stops

Data Field	Data Type	Description
marker_id (**PK**)	int(2)	Contains the ID that uniquely identifies a marker
marker_value	varchar(30)	Contains the value of the marker
marker_name	varchar(20)	Contains the name of the marker

TABLE 4.6: marker is the table that tracks the marker value used for assigning IDs

Data Field	Data Type	Description
user_id (**PK**)	int(3)	Contains the ID that uniquely identifies a user
user_name	varchar(100)	Contains the name of the user
password	varchar(25)	Contains the user's password

TABLE 4.7: user contains the information about user login details

ALGORITHM 1: Main Algorithm (Generic A*)

```
shortest_path_tree spt;                        // collection of paths
priority_queue pq;                             // collection of nodes to
traverse
vertex start, target;                          // origin and destination
locations
state initial_state = new state (start);       // state contains the
information of the vertex (i.e. trip_id, etc)
spt.add (initial_state);
pq.insert (initial_state, 0);                  // initial estimate = 0
while (pq is not empty) {
       state u = pq.extract_min                // get state with lowest weight
       vertex v = get_vertex (u);
       if (v == target) {
              return spt;
       }
       Collection <edges> = connected edges to v;
       foreach (edges : edge) {
              state x = get_state_after_traverse (edge);
              double estimate = weight(x) + estimate distance to target(x,
target);
              if (estimate > maximum weight) { // maximum weight : infinity
                     continue;
              } else if (estimate of (x) is better than estimate of (u)) {
                     spt.add (x);
                     pq.insert (x, estimate);
              }
       }
}
```

ALGORITHM 2: Computing estimate from vertex v to target

```
estimate = Euclidian_distance ( v, target );
```

$$= \sqrt{(v.lat - target.lat)^2 + (v.lng - target.lng)^2}$$

ALGORITHM 3: Computing weight of edge

```
edge_weight = edge_distance * distance_multiplier +
              fare * fare_multiplier +
              (run_time * traffic_points)* time_multiplier +
              Walk_distance* walk_multiplier;
```

ALGORITHM 4: Assigning multipliers

```
if (user preference is least fare) {

     fareMultiplier = 0.2;
     timeMultiplier = 0.3;
     distMultiplier = 0.4;
     walkMultiplier = 0.1;

} else if (user preference is least time) {

     fareMultiplier = 0.4;
     timeMultiplier = 0.2;
     distMultiplier = 0.3;
     walkMultiplier = 0.1;

} else if (user preference is least distance) {

     fareMultiplier = 0.5;
     timeMultiplier = 0.3;
     distMultiplier = 0.2;
     walkMultiplier = 0.1;

}
```

E. Hypothetical Traffic Monitoring System

The Metro Manila Development Authority (MMDA) Traffic Monitoring System provides traffic information on major roads and highways only. It would be insufficient to include those information in routing trips. Thus, a hypothetical traffic monitoring system is created for the sole purpose of having a "real-time" traffic monitoring system on all roads in Metro Manila. The said system mimics how MMDA gathers traffic information.

V. Architecture

A. System Architecture

The system has basically 3 components; the mobile application, the main server, and the web application.

The mobile application serves as the interface that enables the user to communicate with the system. It is built using Java programming language and appropriate Android SDK Tools. The application is currently made to support Android phones with older platform versions starting from Android Ice Cream Sandwich.

The main server is also implemented using Java programming language together with Java Servlet Pages (JSP). It also utilizes libraries such as Log4j for logging, and JSoup library for scraping information from the traffic monitoring system webpage.

The web application is used for maintaining transit data and for updating the graph. It is created using CodeIgniter, a fully-baked PHP MVC Framework, HyperText Markup Language (HTML), JavaScript (JS), and Cascading Style Sheets (CSS). It also uses tools such as jQuery, fancyBox, and Twitter Bootstrap for elegant and easy HTML content display.

B. Technical Architecture

The minimum requirements needed in order for the system to be efficient are the following:

1. 2.00 GHz Intel Core2 Duo CPU
2. At least 4.00 GB RAM

3. More than 2 GB hard drive free space
4. Apache Tomcat 7
5. Java Runtime Environment

The web application needs a MySQL Database and access to Google Maps. It is best viewed in Google Chrome. Any latest browser will also be able to view the application.

The Android application requires the Google Play Services to be installed on the device to access Google Maps and other Google Services.

VI. Results

The system is divided into three parts: the application side, the server side, and the maintenance side. Section A is for the mobile application side and Section B for the web interface for maintenance.

A. Mobile Application

Once the mobile application has been opened, a splash screen will welcome the user for a few seconds then the main screen appears. The main screen has three major components: the map, the text area, and the submit button. This screen serves as the input screen for user's origin and destination.

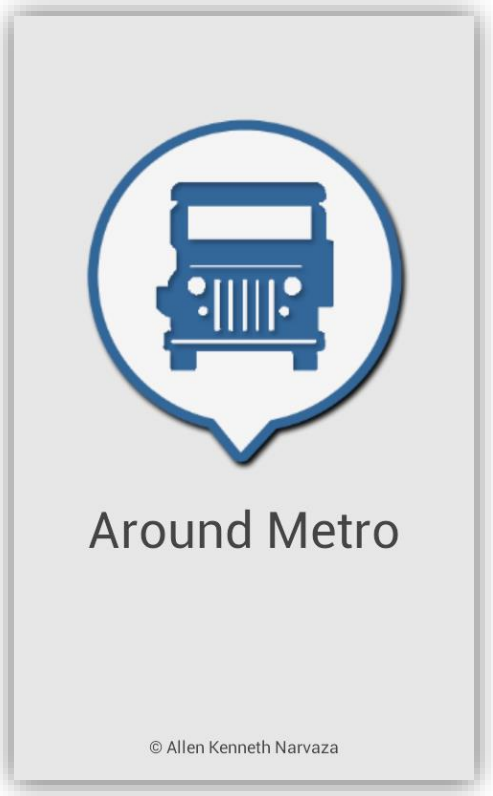


Figure 6.1 Around Metro splash screen

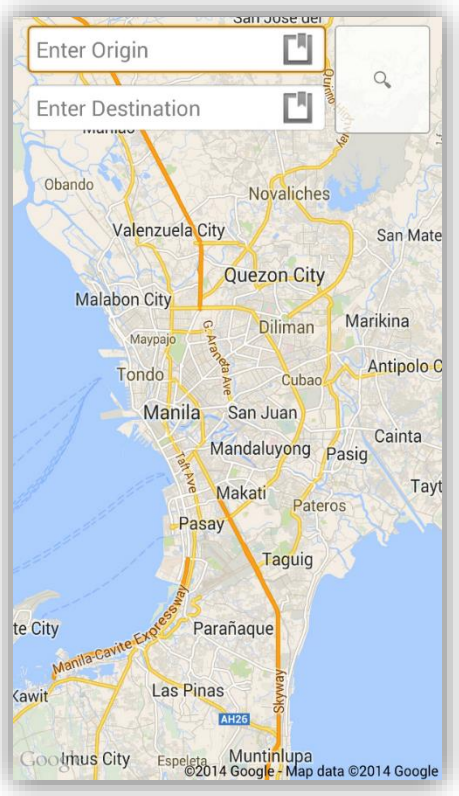


Figure 6.2 Around Metro home screen

There are three options to enter the origin and destination: 1) by pinning on the map through a single tap, 2) type the address in the allotted text area using soft keyboard, or 3) by map long press.

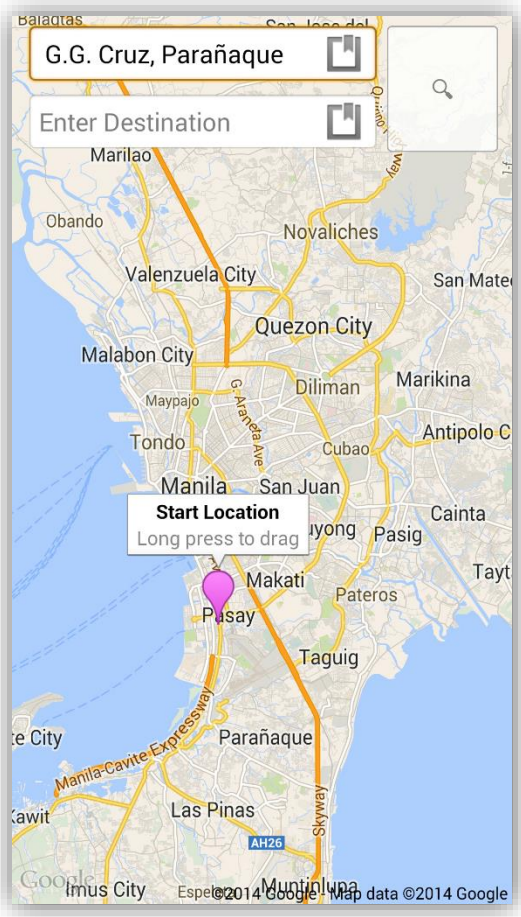


Figure 6.3 Input location by pinning directly on the map through a single click

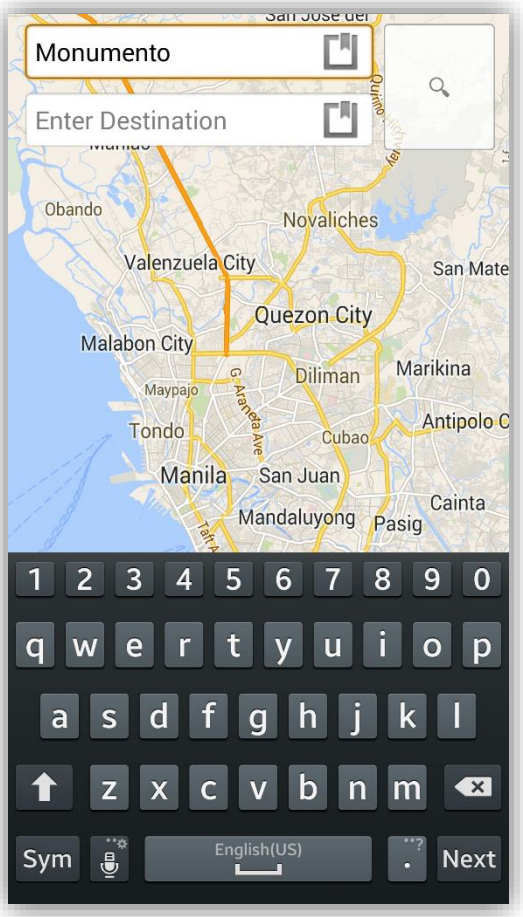


Figure 6.4 Input location by typing the address in the text area

On **Figure 6.6**, the selected point on the map can either be set as start or end location by tapping and holding the map. In the case where the address is entered, a list of suggested

locations is shown, as seen on **Figure 6.5**, if the address has more than one search hits. Otherwise, the address is automatically pinned on the map.

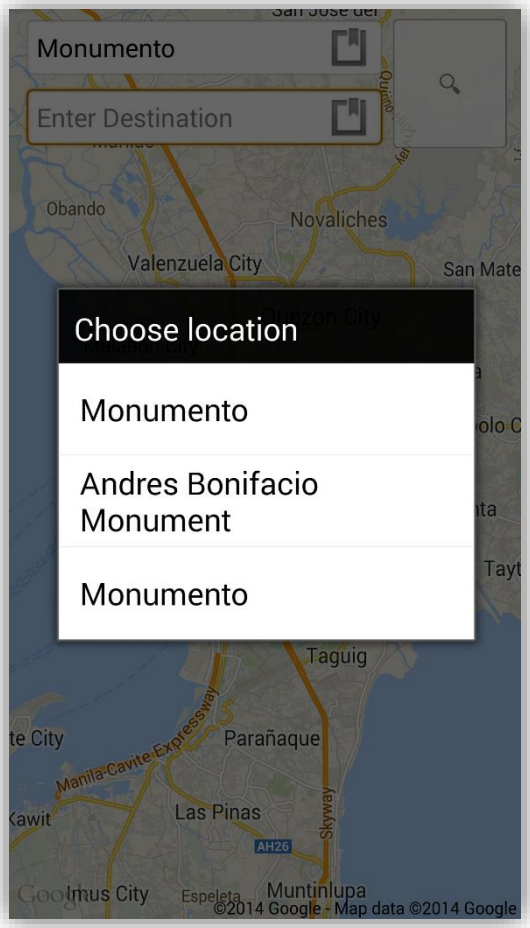


Figure 6.5 Suggested locations based on input address

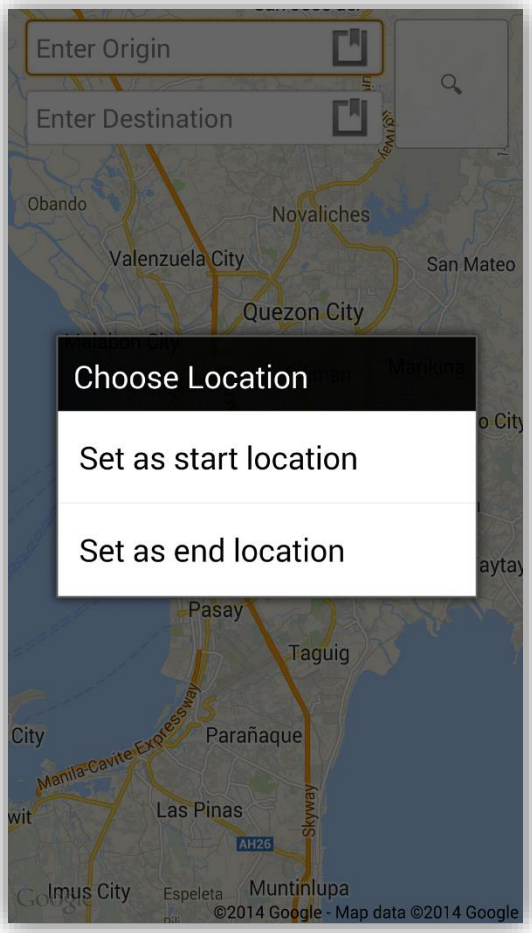


Figure 6.6 Input location by map long press

The markers on the map are draggable in order for the user to edit the location after it has been pinned. The address on the text box also changes after the markers have been moved. If the origin and destination locations are properly placed on the map, the button on the upper right of the screen must be clicked. A prompt will appear, as shown on **Figure 6.7**,

asking about user's preference. The trips generated by the system will greatly rely on the criterion imposed by the user.

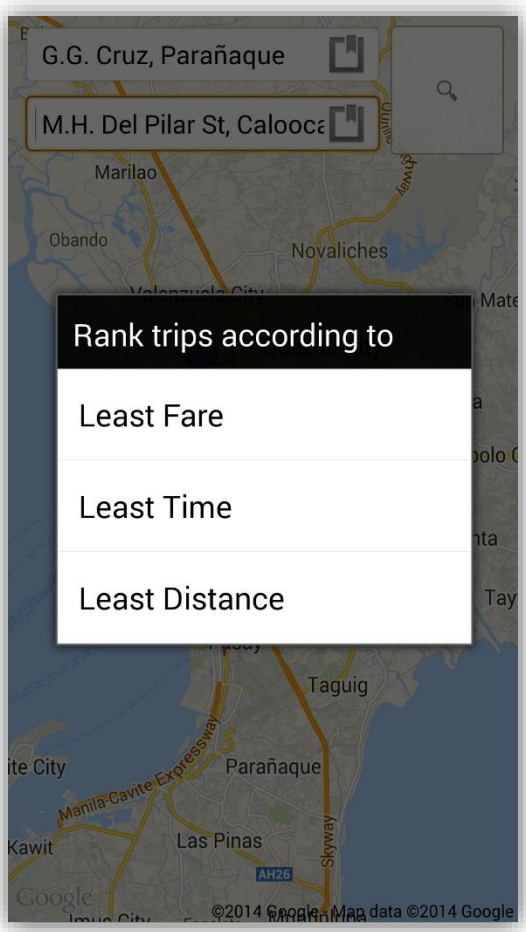


Figure 6.7 Prompt asking user's preference

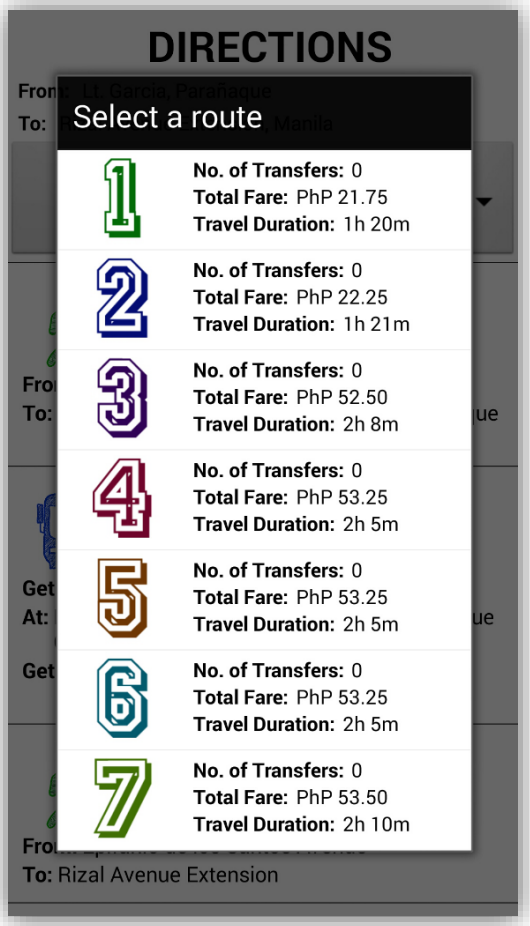


Figure 6.8 List of travel options generated

By default, the first trip on the list is initially shown to the user as this represent the most optimal trip based on the user's preference. To view the complete list of options, which is the one seen on **Figure 6.8**, the row with a number icon must be clicked. Clicking one of these options enable the user to view the details of the chosen trip (see **Figure 6.9**).

Clicking on any of the displayed rows (on **Figure 6.9**), a map will appear with the path to clearly visualize which way the transit vehicle will pass through. Markers can also be clicked to view information regarding the vehicle the marker represent as seen on **Figure 6.10**.

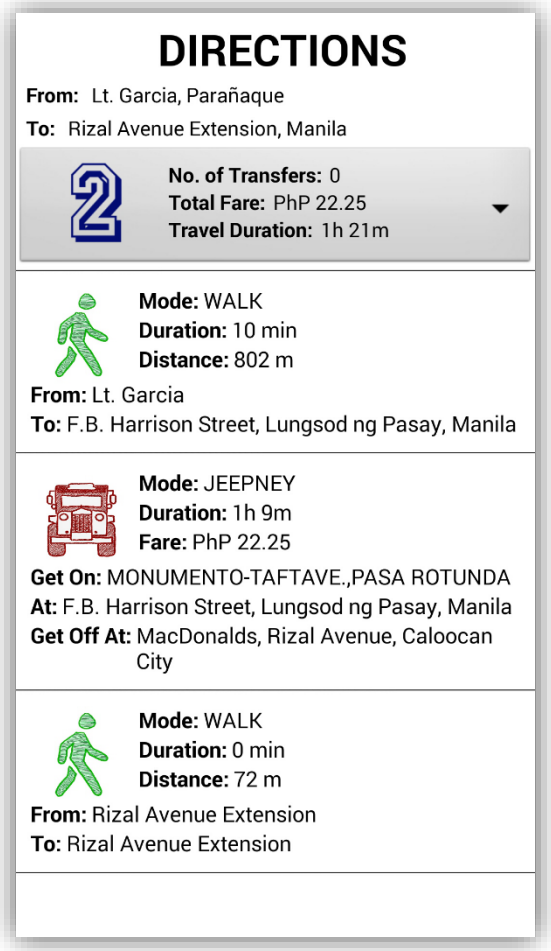


Figure 6.9 Trip details including transport mode, fare, duration, and transit information

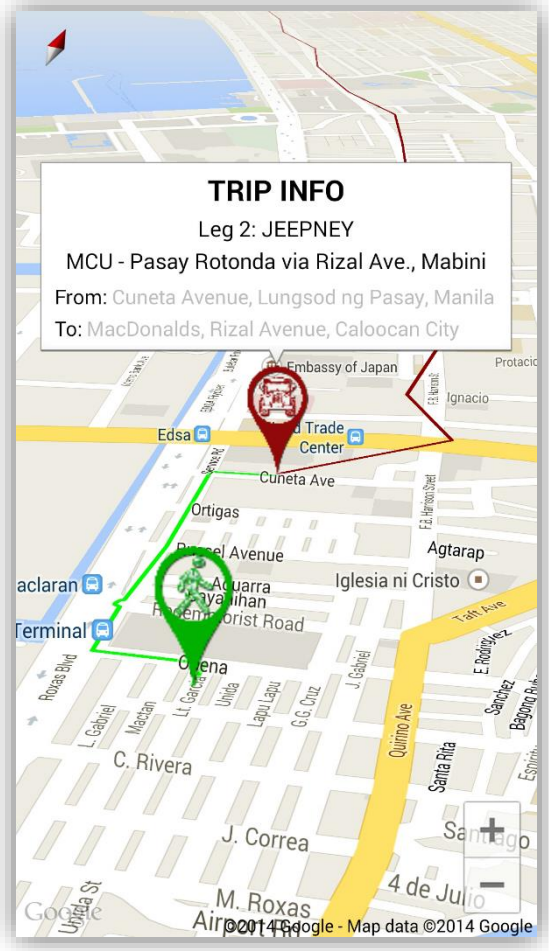


Figure 6.10 Trip details as seen on the map for complete visualization

The map can be zoomed in or zoomed out, tilted, or rotated in order for the user to completely view the whole trip.

B. Web Application

The web application is used for maintaining the system, updating the database, and updating the graph used for generating trips. The system administrator has all the privileges regarding data maintenance; add, edit, view, delete, and graph update.

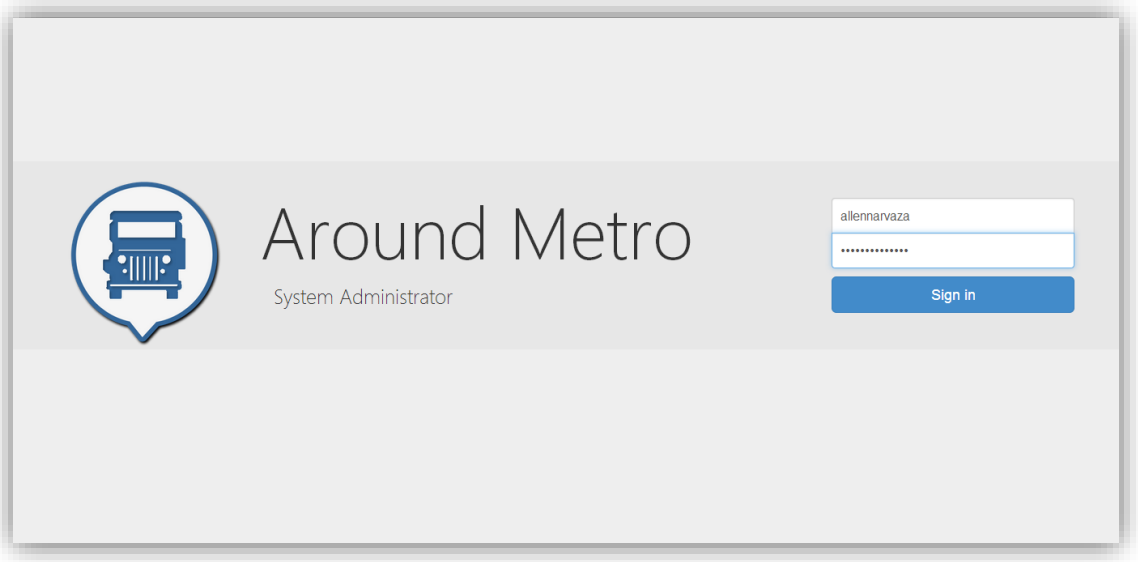


Figure 6.11 Around Metro web interface login page

Once the system administrator enters his login details on the login page (shown on the **Figure above**) and verified, the admin will be brought directly to the routes page where he can add, view, edit, and delete routes (see **Figure 6.12**).

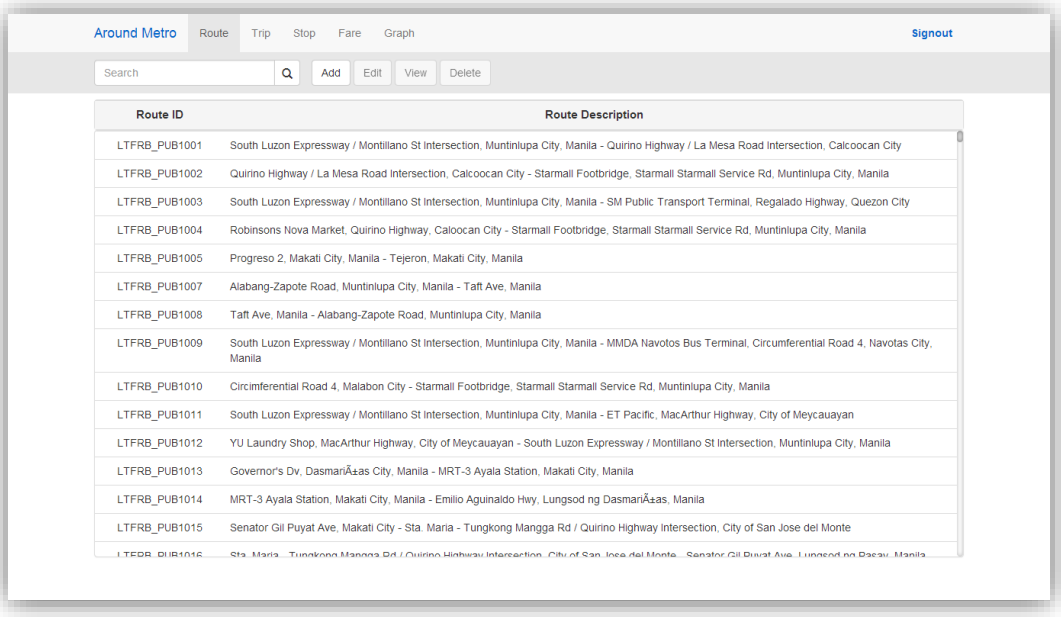


Figure 6.12 Routes page lists all the routes in the database

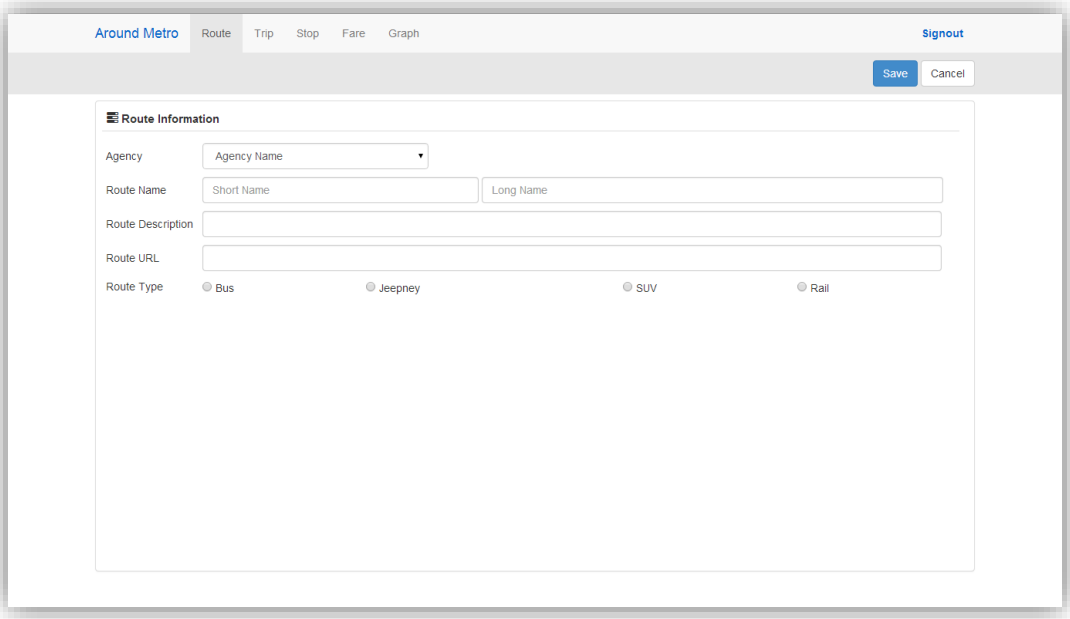
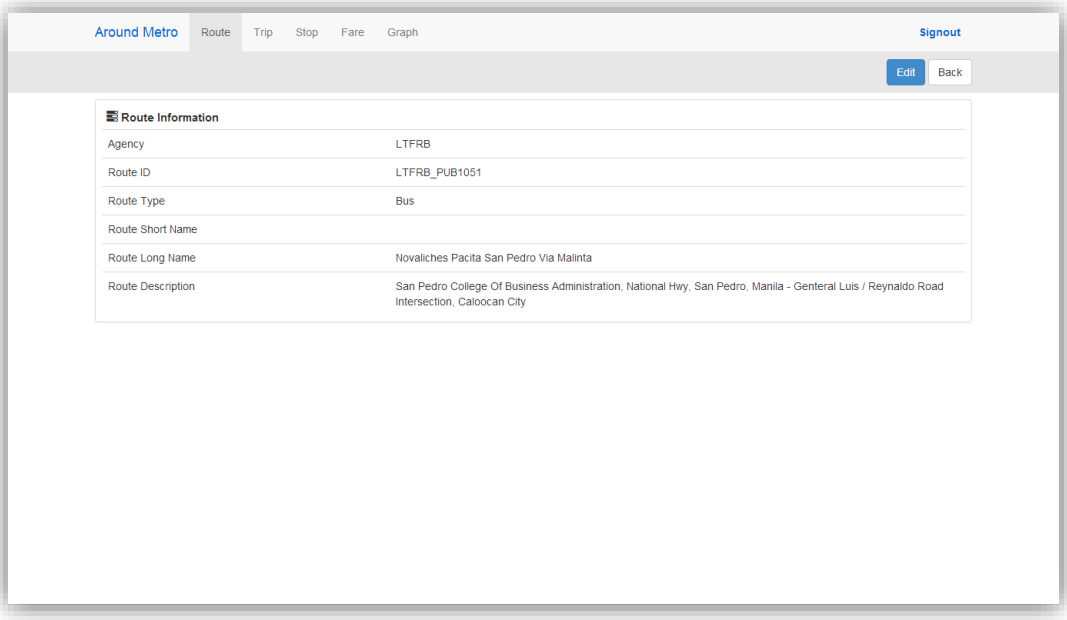


Figure 6.13 Add route page for adding routes



C **Figure 6.14** View route page for viewing route information

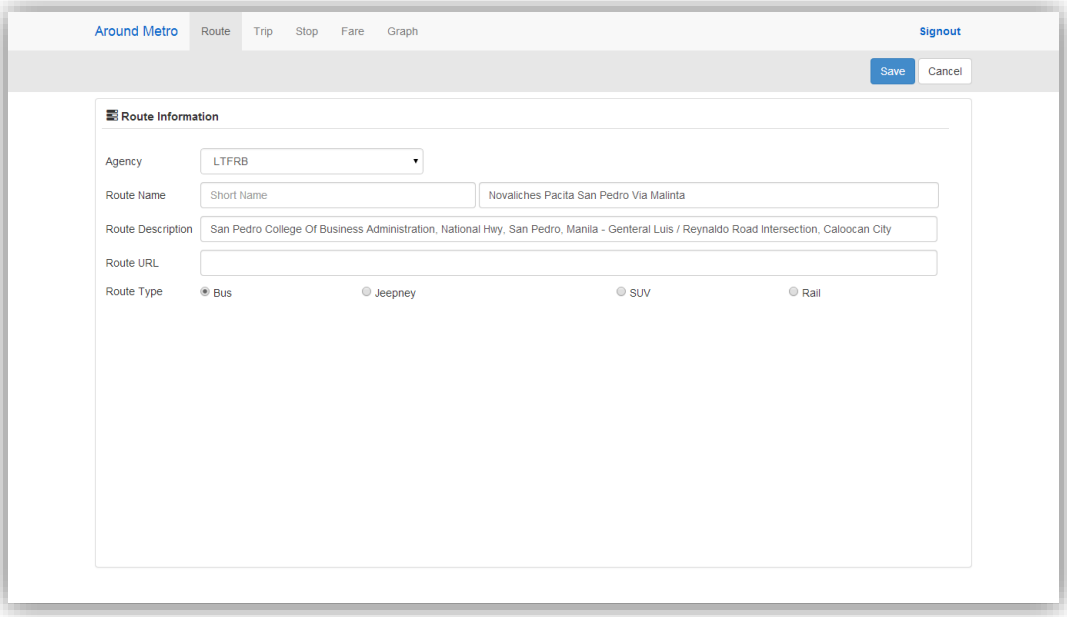


Figure 6.15 Edit route page for editing route information

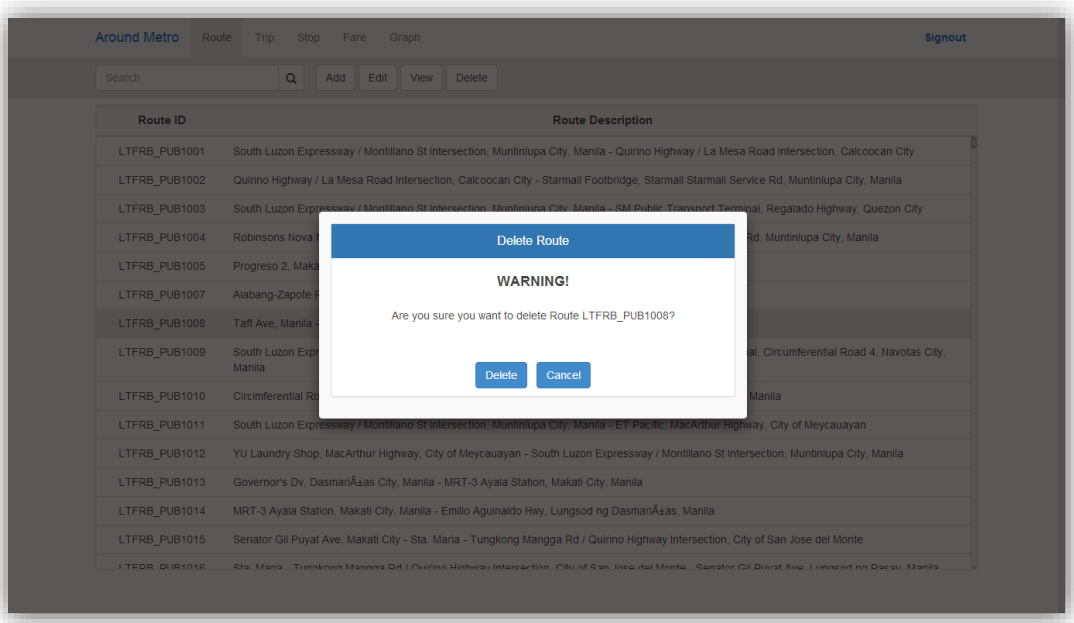


Figure 6.16 Warning prompt for deleting route

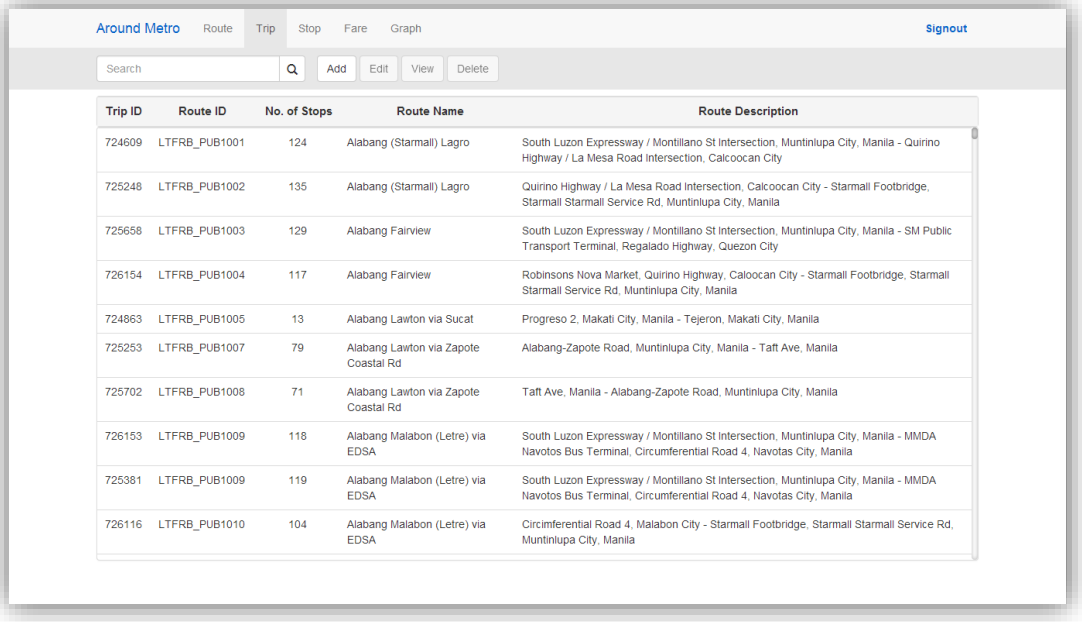


Figure 6.17 Trip page shows the complete list of trips in the database

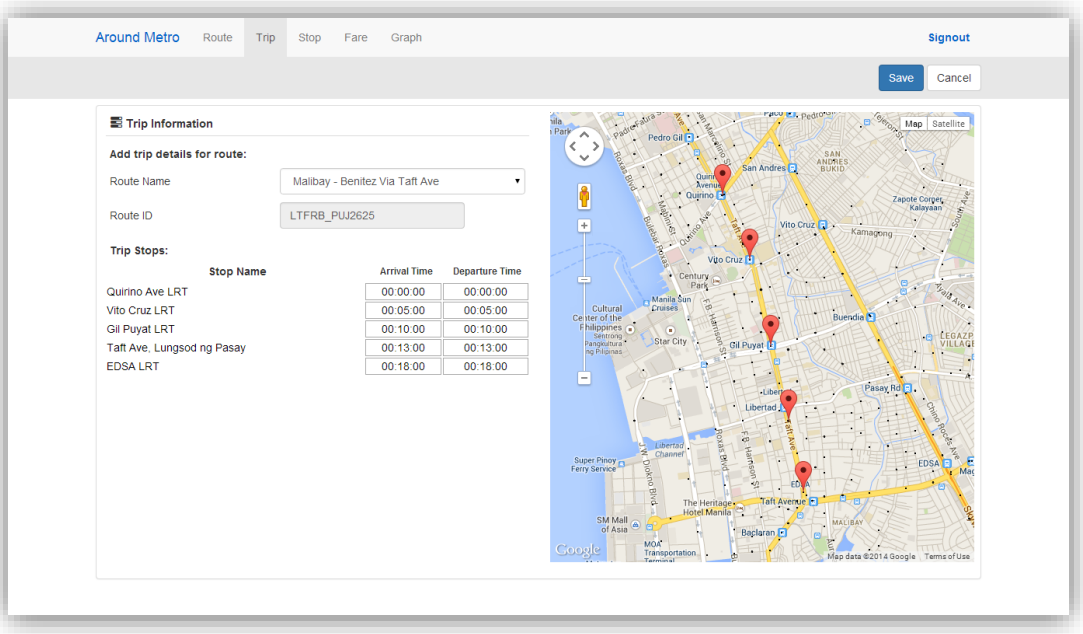


Figure 6.18 Add trip page for adding trips

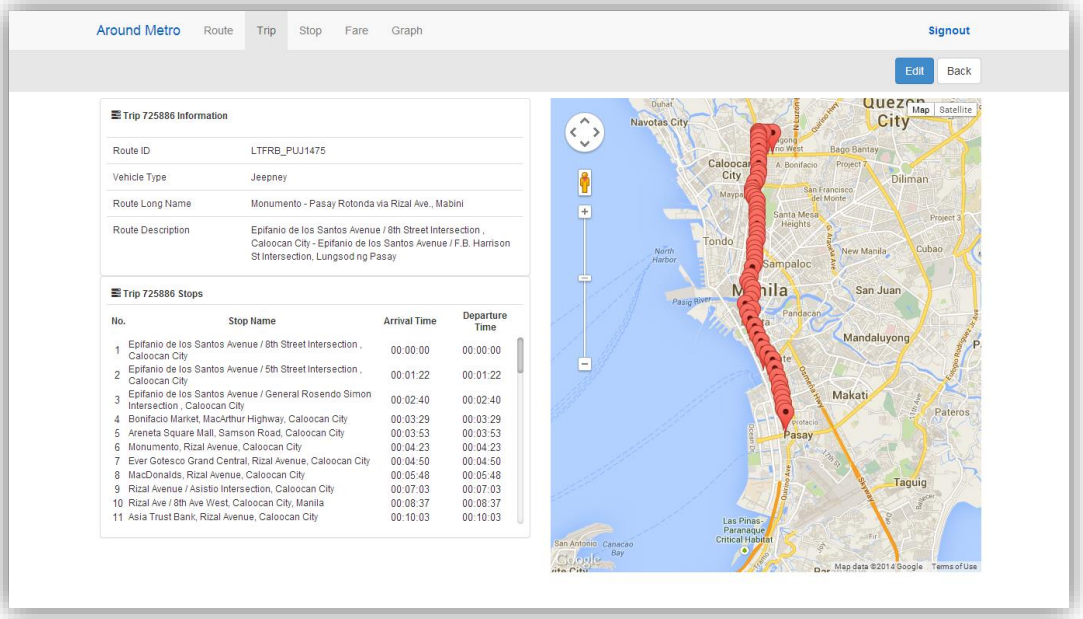


Figure 6.19 View trip page for viewing trip information including all trip stops

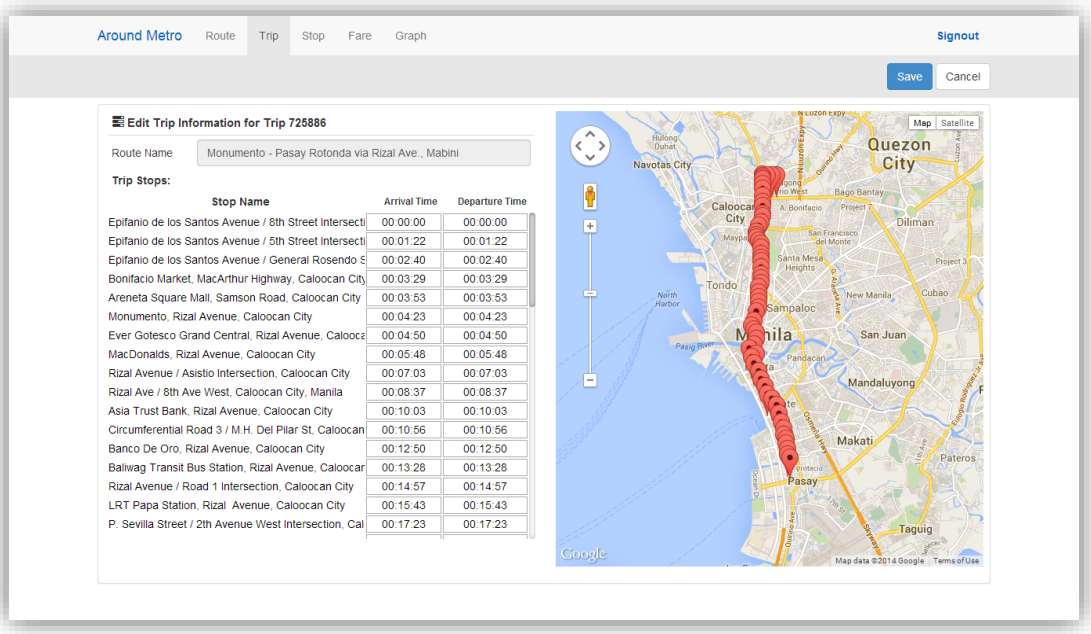


Figure 6.20 Edit trip page for editing trip information especially trip stops

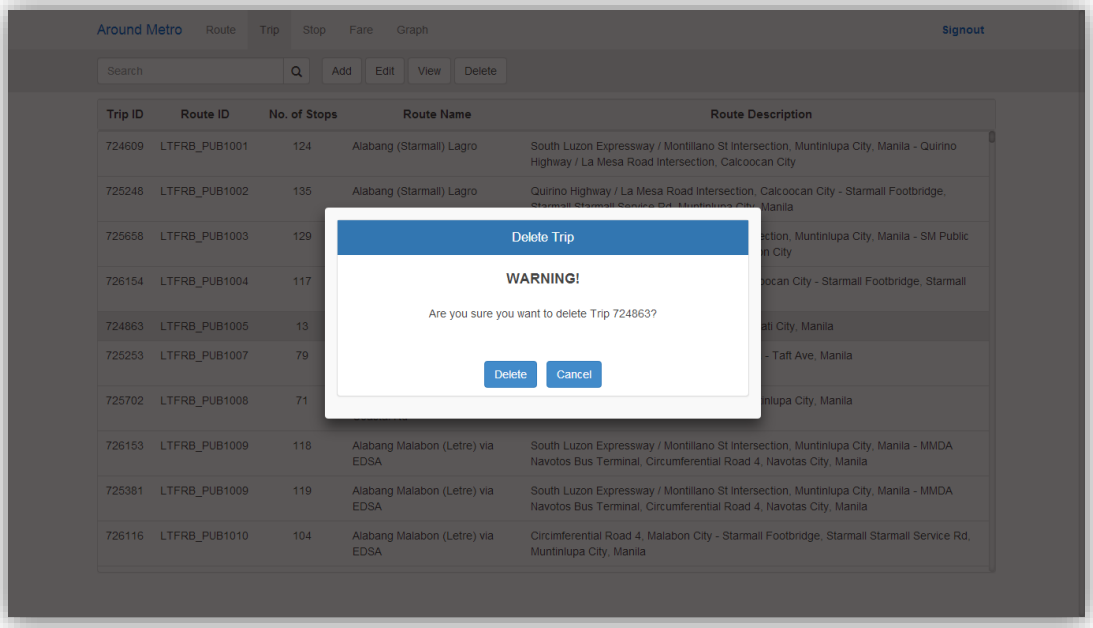


Figure 6.21 Warning prompt for deleting a trip

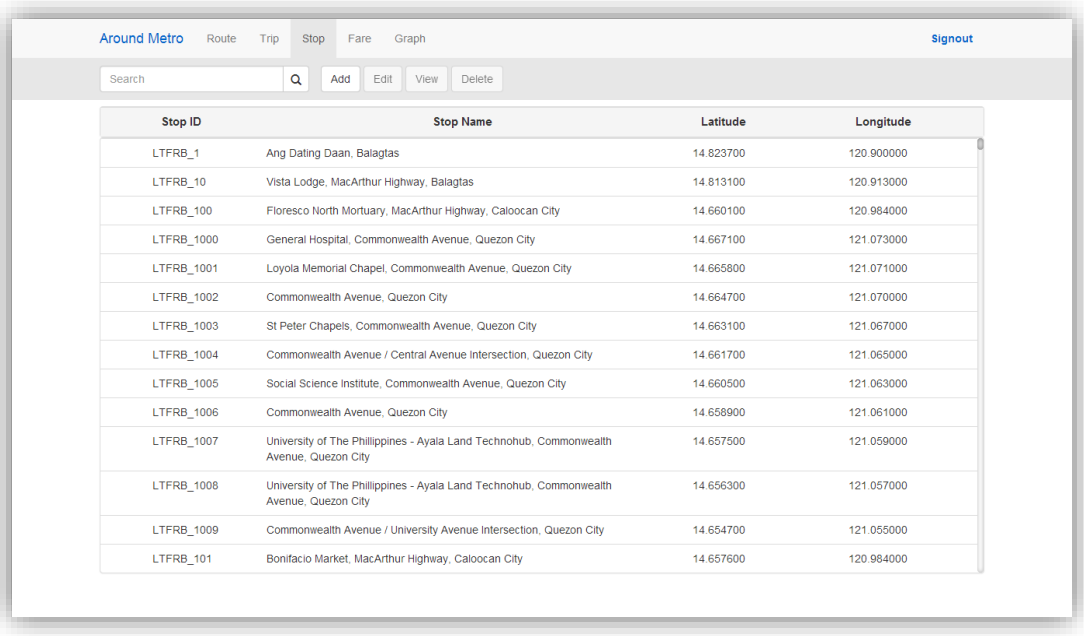


Figure 6.22 Stop page shows all the stops in the database

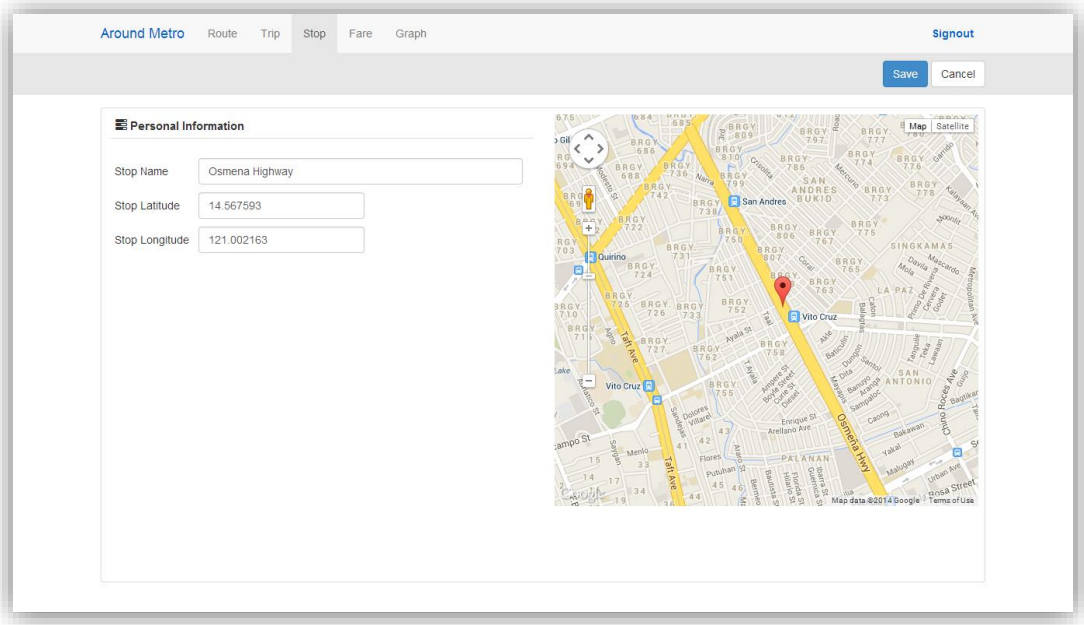


Figure 6.23 Add stop page for adding stop information

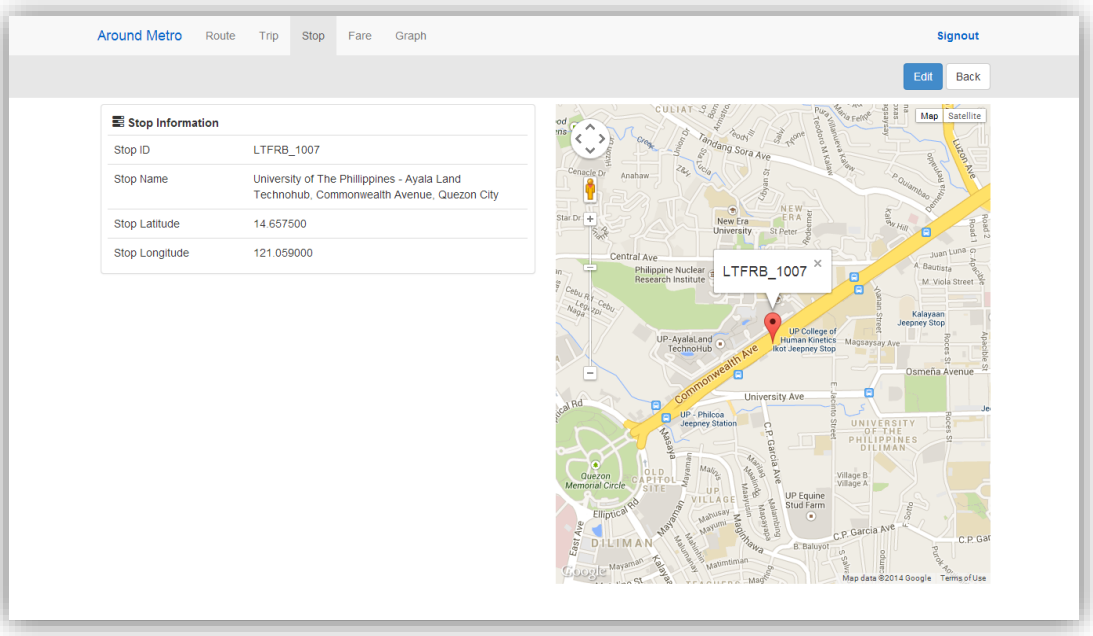


Figure 6.24 View stop page for viewing stop information

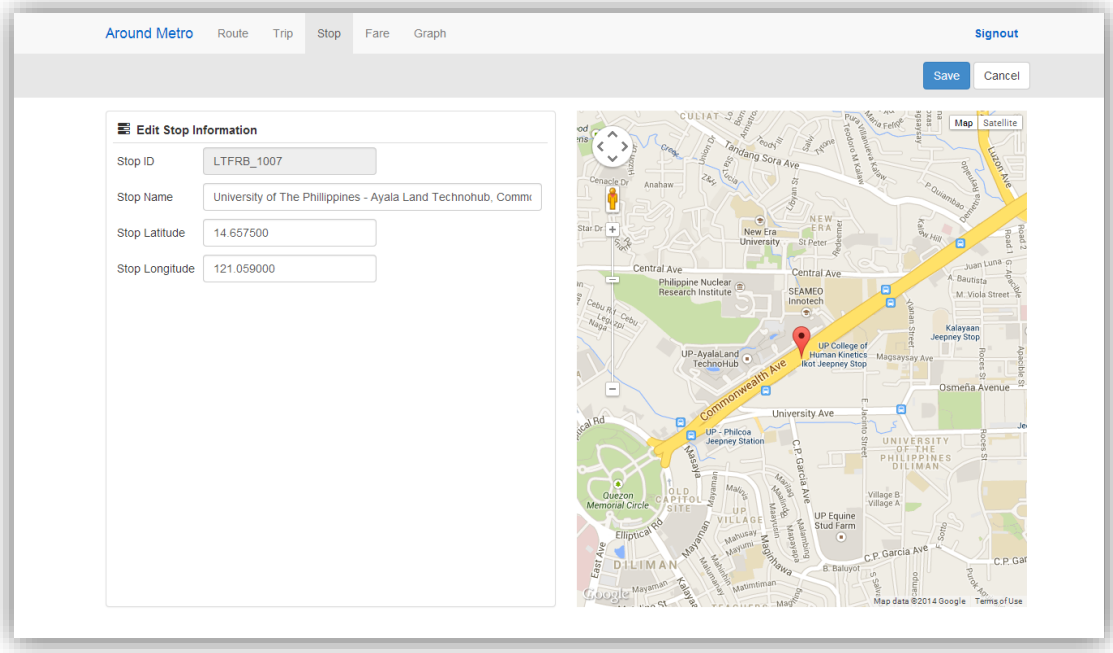


Figure 6.25 Edit stop page for editing stop information

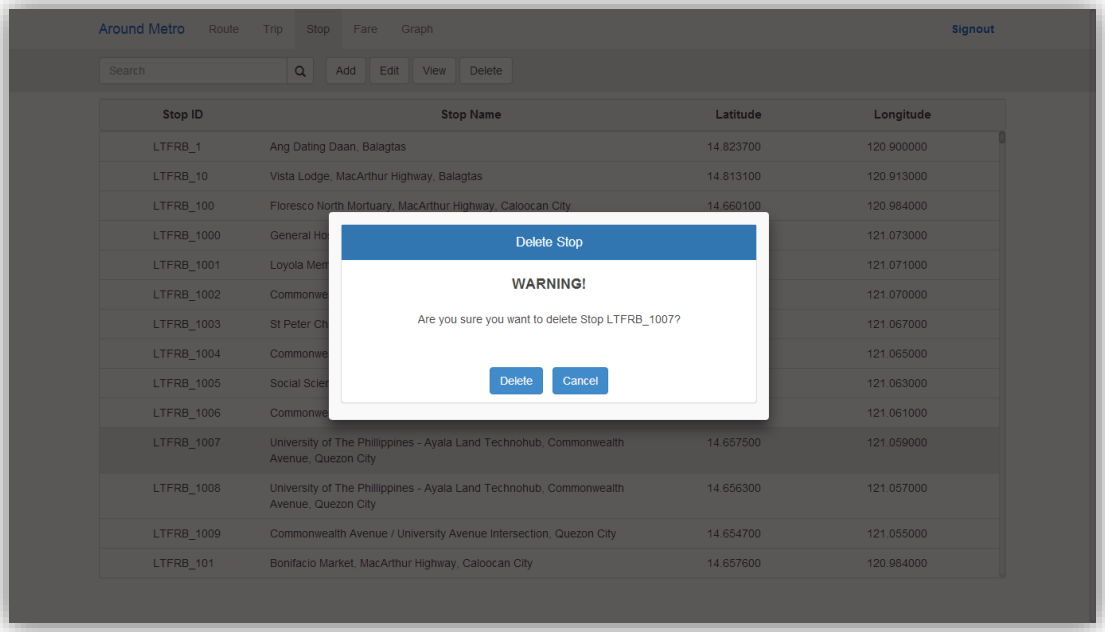


Figure 6.26 Warning prompt for deleting a stop

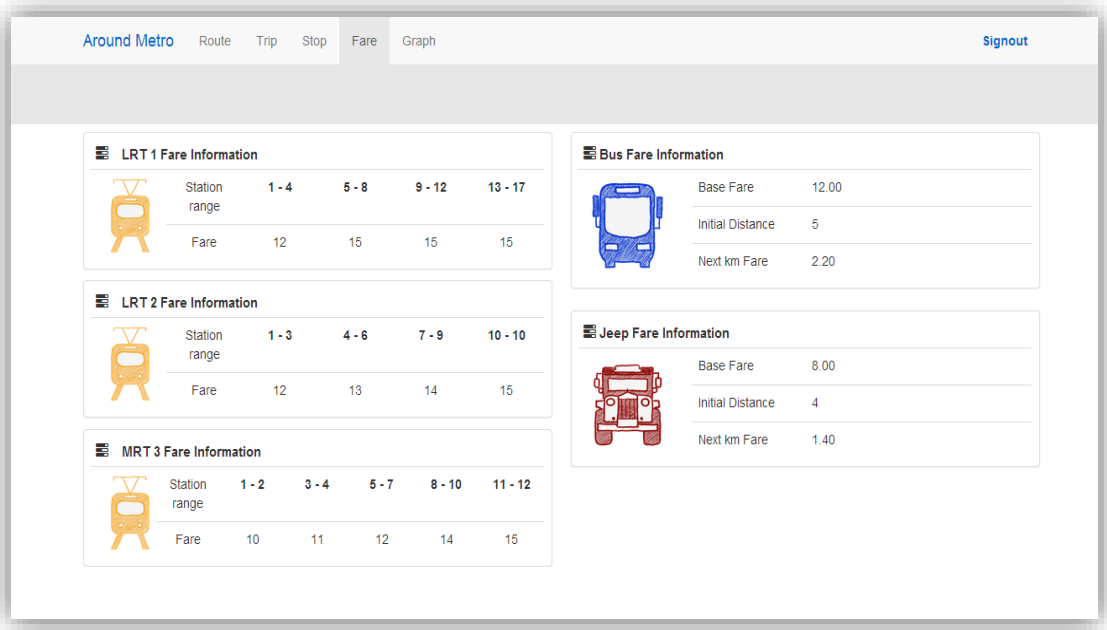


Figure 6.27 Page for viewing fare information

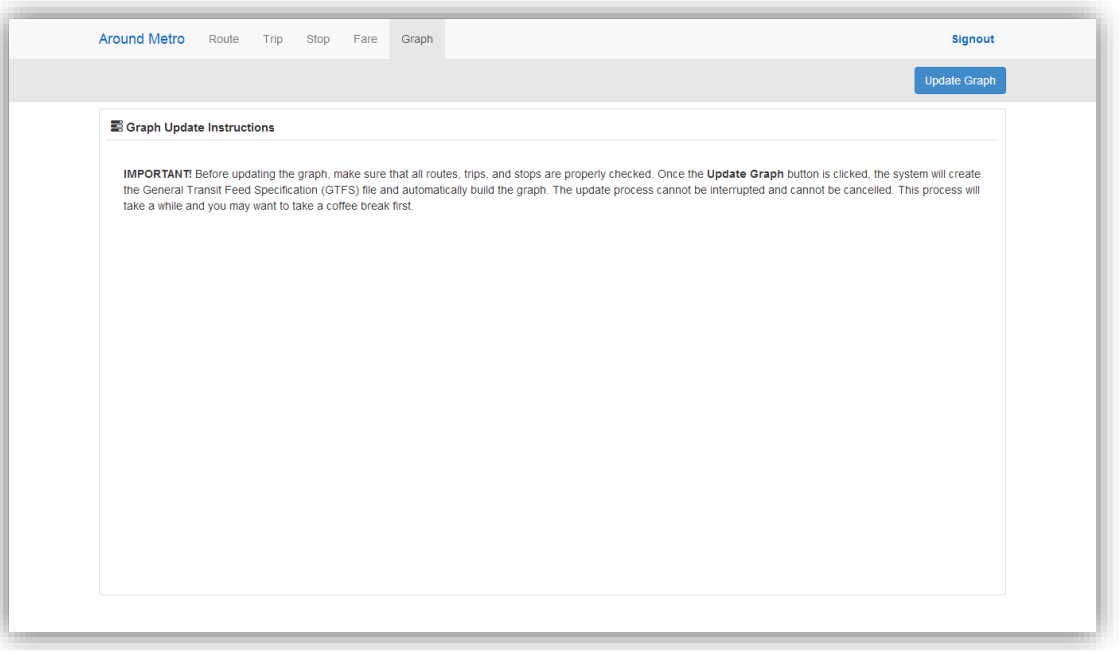


Figure 6.27 Page for updating the graph

The graph update page, as seen on the **Figure above,** is used for updating transit data and the graph used in generating trips. When updating the graph, the Tomcat server must be turned off first and all changes must be properly checked by the System Administrator (Sys Ad). Once all the changes are inspected, the Sys Ad should click the button **Update Graph.** By clicking this button, the system will automatically create all necessary General Transit Feed Specifiction (GTFS) files and update the graph. This process may take a while depending on the traffic in the server. Once finished, the Tomcat server must be restarted.

VII. Discussion

The whole Around Metro system is composed of three parts; the mobile application side, the server side, and the web application side. The system has 2 primary users: the application users and the System Administrator. The mobile side is where the application user interact with the system while the web application side is where all data maintenance and graph updates are made.

The first component is the Android application. It is a simple, user-friendly application used for planning trips with the origin and destination of the user as the main input. The user can input his origin and destination in 3 ways: by pinning directly on the map, typing the address on the text box, or by tapping and holding the map. The user can also impose a constraint on the results either by least fare, least time, or least distance.

The request will be sent over to the server for processing and generating trips. This process may take a while depending on the distance of origin and the destination. The farther they are with each other, the longer the search will be because of numerous possibilities. Once the server is done processing the request, results can be viewed by the user. By default, the first option in the list is displayed as this represents the optimal trip based on user preference. Trip details include number of transfer/s, fare, trip duration, vehicle name, and where to board and alight. User can also view the whole trip on the map for complete visualization of the trip.

Some trips, as seen on the map, may not follow a straight path but a zigzag path instead. This is because of the data incorporated in the graph, specifically the stop locations.

The middle component is the system that receives the request, process the request, and generate trips that are sent back to the users. To limit the user's waiting time, the system is set

to a maximum number of trips. As seen on Figure 6.1, the greater the number of requests, the longer it is for the system to respond to the user. Currently, the app is set to a maximum of 10 trips per request.

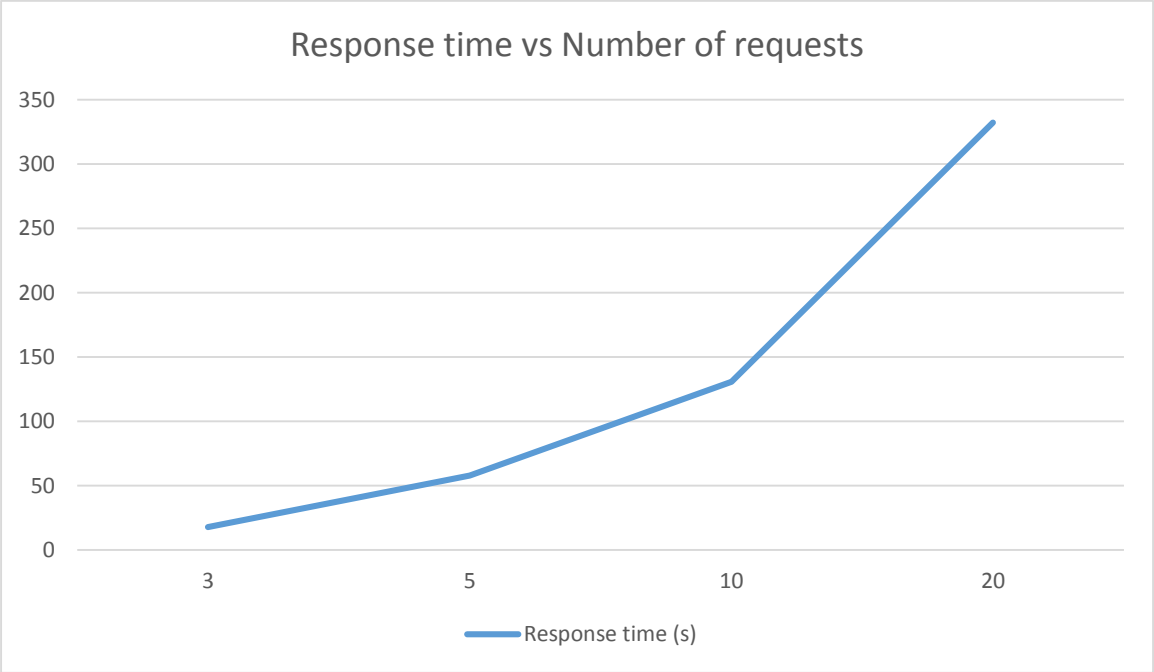


Figure 7.1 The greater the number of requests the longer the response time

In generating trips, the system takes into consideration 5 factors that is believed to affect routing from origin to destination; distance travelled, elapsed time, fare, walk distance, and traffic conditions. Each of these factors, except walk distance and traffic condition, is assigned a different value depending on the preference of the user. The traffic condition is manipulated through the hypothetical traffic monitoring system created for testing the whole system. Through these, the system generate results that are based on what the user wants and at the same time generate results that adjust depending on the traffic condition. Appendix A shows the summary of computations done to get the results based on user preference and traffic condition.

The generic A* Algorithm, as used for generating trips in the system, normally has a time complexity that depends on the heuristic. In the worst case, the number of nodes expanded is exponential in the length of the solution (the shortest path), but it is polynomial when the search space is a tree.

In the process of generating trips, the trips already picked in the previous iteration will be removed from selection on the next iteration. This is to prevent the duplication of trips in the generated results. This process shows the greedy characteristics of A* such that in every iteration, it makes sure that it picks the elements with the lowest weight.

The last component is the maintenance side. It is a web application that handles all data maintenance and graph updates. This side is primarily under the System Administrator. He/she has all the privileges regarding graph updates and data maintenance. These privileges include, adding, viewing, editing, and deleting routes, trips, fare, and stops. He/she also has the right to update the graph. Updating the graph takes a long time to process depending on server traffic.

VIII. Conclusion

The application Around Metro is a very useful mobile application most especially for commuters around Metro Manila. It generates trip options -- a set of available public utility vehicles around Metro Manila – that guide the user from his origin to his destination. It has a very simple interface that can be easily used. It also presents the information clearly through a complete visualization on the map. Through this application, users are guided each step of the way. Also the application gives the user the freedom to choose among the generated results which were based from the user's preference.

The system also has a web interface which is used to update the transit data and the graph used by the system in planning the trips. It is handled by a System Administrator who has all the privileges (adding, viewing, editing, and deleting transit data and updating the graph as well). The web application has made data modification and data visualization easier.

IX. Recommendations

Around Metro heavily relies on the graph – specifically the information inside the graph -- for producing better, if not best, results. Thus in order for the application to generate optimal solutions, data should be improved. One possible data enhancement is the proper positioning of stop locations (i.e. placing stops along the road and not beside the road).

The number of people owning a smartphone has increased rapidly in the past years. However, there are still a lot of people who rely on their conventional cell phones. It would be better if the system could be extended for the benefit those people by creating an SMS version of Around Metro where the user can send their origin and destination as SMS messages and the generated results will be received via SMS as well.

X. Bibliography

[1] Thynell, M., & Aora, A. (2009). Social Impact Assessment of Public Transport in Cities: An approach for people involved in the planning, design, and implementation of public transport systems.

[2] Folger, P. (2009). Geospatial Information and Geographic Information Systems (GIS): Current Issues and Future Challenges. Congressional Research Service.

[3] Haklay, M. (2008). How Good is Volunteered Geographical Information? A comparative study of OpenStreetMap and Ordinance Survey Datasets. Environment and Planning B: Planning and Design.

[4] Zielstra, D., & Zipf, A. (2010). Quantitative Studies on the Data Quality of OpenStreetMap in Germany. .

[5] Peischl, B., & Ziefle, M. H. (2008). A Mobile Information System for Improved Navigation in Public Transport.

[6] Butler, M. (2011). Android: Changing the Mobile Landscape. IEEE Pervasive Computing.

[7] About Google Transit. Retrieved from http://www.scmtd.com/en/routes/about-google-transit

[8] (2012, August 28). Directions by Metro Manila trains now in Google Maps. Retrieved from http://newsbytes.ph/2012/08/28/directions-by-metro-manila-trains-now-in-google-maps/ (n.d.).

[9] Pun-Cheng, L. (2012). An Interactive Web-Based Public Transport Enquiry System With Real-Time Optimal Route Computation. IEEE Transactions on Intelligent Transportation Systems.

[10] Peng, T., & Wang, X. (2012). A Mobile-based Navigation Web Application: Finding the Shortest-tiem Path based on Factor Analysis. Division of Geomatics, Department of Industrial Development, IT and land Management, University of Gavle.

[11] Li, H., & Yi, X. (2010). The Study and Implementation of Mobile Digital Maps System, 2010 2nd International Conference on Information Engineering and Computer Science (ICIECS).

[12] Rifat, R., Moutushy, S., Ahmed, S., & Ferdous, H. (2011). Location based Information System using OpenStreetMap. 2011 IEEE Student Conference on Research and Development (SCOReD) .

[13] Garcia, C., Candela, S., Ginory, J., Arencibia, A., & Alayon, F. (2012). On Route Travel Assistant for public Transport based in Android Technology, 2012 Sixth International Conference on Innovative Mobile and Internet Services in Ubiquitous Computing.

[14] Nagaraj, U., Wakade, R., Gaware, R., Dhame, R., & Alhat, D. (2011). Intelligent Public Transport Information System, International Journal on Computer Science and Engineering (IJCSE).

[15] Zheng, J., Chen, X., Ciephuch, B., Winstanley, A., Mooney, P., & jacob, R. (2010). The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences, Vol. 38, Part II.

[16] Liu, K. (2007). Studies on the Transfer data modeling in the public transport information system, 7th International Conference on ITS Telecommunications.

[17] Mead, J. (2009). Integrated Public Transportation Information System for BAA Southamplton Airport.

[18] Bliznyuk, A. (2011). Green Daily Guide. Easier Environmentally Friendly Transportation with the Help of Mobile Technnologies, 2011 International Conference on Collaboration Technologies and Systems (CTS).

[19] *Land Transportation Franchising and Regulatory Board*(n.d.). Retrieved October 2012, from http://www.ltfrb.gov.ph/main/aboutus

[20] Light Rail Transit Authority Website. Retrieved October 2012, from http://www.lrta.gov.ph/

[21] Metro Rail Transit Authority Website. Retrieved October 2012, from http://dotcmrt3.weebly.com/

[22] OpenTripPlanner. Retrieved March 2014, http://www.opentripplanner.org/

[23] Graham, R., McCabe, H., Sheridan, S. (2002) Pathfinding in Computer Games

[24] *Introduction to A**. (n.d.). Retrieved October 2012, from http://theory.stanford.edu/~amitp/GameProgramming/AStarComparison.html#the-a-star-algorithm

[25] GIS. (n.d.). Retrieved October 2012, from Crow Wing County: http://www.co.crow-wing.mn.us/index.aspx?nid=186

[26] *Components of a GIS*. (n.d.). Retrieved October 2012, from http://www.sfu.ca/rdl/GIS/tour/comp_gis.html

[27] GIS Data Types. (n.d.). Retrieved October 16, 2012, from Bio Diversity GIS: http://bgis.sanbi.org/gis-primer/page_14.htm

[28] *Geocoding Overview and Preparation*. (n.d.). Retrieved October 2012, from http://ocw.tufts.edu/data/54/626652.pdf

[29] *Web Scraping*. (n.d.). Retrieved October 2012, from http://www.techopedia.com/definition/5212/web-scraping

[30] *The importance of computing intermodal roundtrips in multimodal guidance systems*. (n.d.). Retrieved October 2012, from http://www.strc.ch/conferences/2004/Baumann_Torday_Dumont_Computing

XI. Appendix

A. Computation Tables

1. Computation tables for finding trips from UP Manila to UP Diliman where traffic is LIGHT on all areas and user preference is least fare.

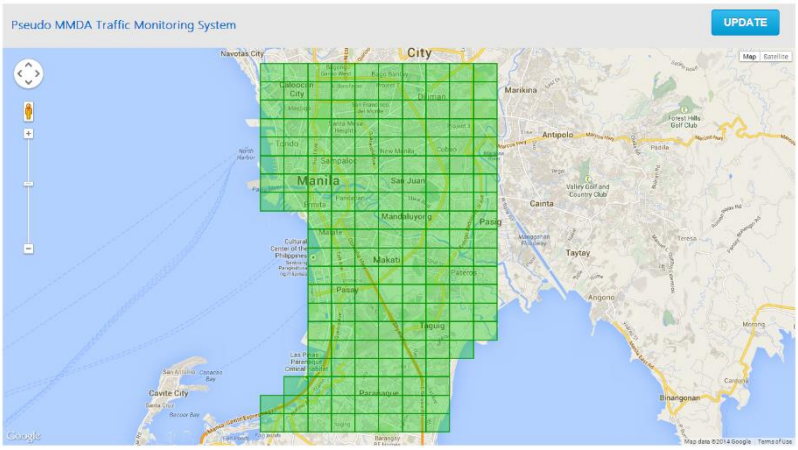


Figure 11.2 Light traffic on all areas

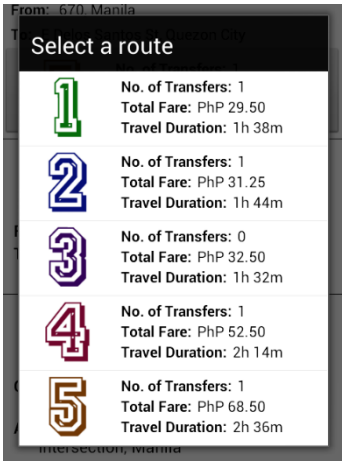


Figure 11.1 Top 5 results

TRIP ID	DISTANCE	TIME	FARE	TRAFFIC SCORE	TIME*TRAFFIC SCORE
LTFRB_724970	217.5172	63	0	1	63
LTFRB_724970	247.0586	76	0	1	76
LTFRB_724970	298.0898	98	0	1	98
LTFRB_724970	237.0979	68	0	1	68
LTFRB_724970	287.7473	84	0	1	84
LTFRB_724970	44.47804	19	0	1	19
LTFRB_724970	180.2093	44	0	1	44
LTFRB_724970	189.0317	58	0	1	58
LTFRB_724970	189.2444	64	0	1	64
LTFRB_724970	405.0377	111	0	1	111
LTFRB_724970	222.3902	68	0	1	68
LTFRB_724970	247.0548	69	0	1	69
LTFRB_724970	189.0317	58	0	1	58
LTFRB_724970	189.0317	59	0	1	59
LTFRB_724970	334.8385	100	0	1	100
LTFRB_724970	162.9078	49	0	1	49

LTFRB_724970	272.2712	72	0	1	72
LTFRB_724970	272.27	72	8.25	1	72
LTFRB_724970	162.9065	56	8.5	1	56
LTFRB_724970	272.268	72	8.75	1	72
LTFRB_724970	180.203	59	9	1	59
LTFRB_724970	265.5994	71	9.5	1	71
LTFRB_724970	301.5684	90	10	1	90
LTFRB_724970	447.4273	159	10.5	1	159
LTFRB_724970	324.5061	100	11	1	100
LTFRB_724970	431.6728	106	11.5	1	106
LTFRB_724970	323.5513	103	12	1	103
LTFRB_724970	337.943	92	12.5	1	92
LTFRB_724970	225.2946	75	12.75	1	75
LTFRB_724970	232.8506	89	13.25	1	89
LTFRB_724970	345.1796	104	13.75	1	104
LTFRB_724970	332.0327	86	14	1	86
LTFRB_724970	337.9372	92	14.5	1	92
LTFRB_724970	286.4219	101	15	1	101
LTFRB_724970	237.0891	74	15.25	1	74
LTFRB_724970	225.2874	68	15.75	1	68
LTFRB_724970	329.5951	81	16	1	81
LTFRB_724970	337.9292	96	16.5	1	96
LTFRB_724970	337.928	101	17.25	1	101
LTFRB_725618	266.8682	81	0	1	81
LTFRB_725618	298.0803	85	0	1	85
LTFRB_725618	255.7487	79	0	1	79
LTFRB_725618	267.2408	76	0	1	76
LTFRB_725618	255.7487	76	0	1	76
LTFRB_725618	391.9448	114	0	1	114
LTFRB_725618	585.7998	176	0	1	176
LTFRB_725618	247.4933	70	0	1	70
LTFRB_725618	279.185	93	0	1	93
LTFRB_725618	253.1711	69	0	1	69
LTFRB_725618	265.5645	86	0	1	86
LTFRB_725618	279.1818	89	0	1	89
LTFRB_725618	253.1676	68	12.25	1	68
TOTAL	14359.69		29.50		4269

Total_weight = 14359.69*0.4 + 29*0.2 + 4269*0.3 + 633.0434*0.1= **7093.40434**

TRIP ID	DISTANCE	TIME	FARE	TRAFFIC SCORE	TIME*TRAFFIC SCORE
LTFRB_725202	217.5172	63	0	1	63
LTFRB_725202	247.0586	76	0	1	76
LTFRB_725202	298.0898	98	0	1	98
LTFRB_725202	237.0979	67	0	1	67
LTFRB_725202	287.7473	85	0	1	85
LTFRB_725202	44.47804	19	0	1	19
LTFRB_725202	180.2093	43	0	1	43
LTFRB_725202	189.0317	59	0	1	59
LTFRB_725202	189.2444	64	0	1	64
LTFRB_725202	405.0377	110	0	1	110
LTFRB_725202	222.3902	69	0	1	69
LTFRB_725202	247.0548	69	0	1	69
LTFRB_725202	189.0317	58	0	1	58
LTFRB_725202	189.0317	59	0	1	59
LTFRB_725202	334.8385	100	0	1	100
LTFRB_725202	162.9078	48	0	1	48
LTFRB_725202	272.2712	72	0	1	72
LTFRB_725202	272.27	73	8.25	1	73
LTFRB_725202	162.9065	56	8.5	1	56
LTFRB_725202	272.268	71	8.75	1	71
LTFRB_725202	180.203	60	9	1	60
LTFRB_725202	265.5994	71	9.5	1	71
LTFRB_725202	301.5684	90	10	1	90
LTFRB_725202	447.4273	158	10.5	1	158
LTFRB_725202	324.5061	101	11	1	101
LTFRB_725202	431.6728	106	11.5	1	106
LTFRB_725202	323.5513	102	12	1	102
LTFRB_725202	337.943	93	12.5	1	93
LTFRB_725202	225.2946	75	12.75	1	75
LTFRB_725202	232.8506	89	13.25	1	89
LTFRB_725202	345.1796	104	13.75	1	104
LTFRB_725202	332.0327	85	14	1	85
LTFRB_725202	337.9372	93	14.5	1	93
LTFRB_725202	431.6621	131	15.25	1	131
LTFRB_725202	325.8279	110	15.75	1	110
LTFRB_725202	430.9441	110	16.25	1	110
LTFRB_725202	112.6452	33	16.5	1	33
LTFRB_725202	237.3183	73	16.75	1	73
LTFRB_725202	374.0589	111	17.25	1	111

LTFRB_725202	219.7365	85	18	1	85
LTFRB_725283	200.1512	62	0	1	62
LTFRB_725283	217.5064	59	0	1	59
LTFRB_725283	247.0491	75	0	1	75
LTFRB_725283	227.2359	66	0	1	66
LTFRB_725283	227.2355	63	0	1	63
LTFRB_725283	100.0756	37	0	1	37
LTFRB_725283	257.103	73	0	1	73
LTFRB_725283	266.8682	81	0	1	81
LTFRB_725283	298.0803	84	0	1	84
LTFRB_725283	255.7487	79	0	1	79
LTFRB_725283	267.2408	76	0	1	76
LTFRB_725283	255.7487	77	0	1	77
LTFRB_725283	391.9448	114	0	1	114
LTFRB_725283	457.6057	136	0	1	136
LTFRB_725283	146.9294	55	0	1	55
LTFRB_725283	524.6934	162	0	1	162
LTFRB_725283	796.693	244	12.25	1	244
LTFRB_725283	253.1676	69	13.25	1	69
TOTAL	15729.52		31.25		4351

Total_weight = 15729.52*0.4 + 31.25*0.2 + 4351*0.3 + 376.5325*0.1= **7641.01125**

TRIP ID	DISTANCE	TIME	FARE	TRAFFIC SCORE	TIME*TRAFFIC SCORE
LTFRB_726297	217.5172	63	0	1	63
LTFRB_726297	247.0586	76	0	1	76
LTFRB_726297	298.0898	98	0	1	98
LTFRB_726297	565.1976	170	0	1	170
LTFRB_726297	489.2584	146	0	1	146
LTFRB_726297	405.0377	111	0	1	111
LTFRB_726297	222.3902	68	0	1	68
LTFRB_726297	247.0548	69	0	1	69
LTFRB_726297	189.0317	58	0	1	58
LTFRB_726297	426.0066	136	0	1	136
LTFRB_726297	162.9078	48	0	1	48
LTFRB_726297	272.2712	72	0	1	72
LTFRB_726297	272.27	73	0	1	73

LTFRB_726297	162.9065	56	0	1	56
LTFRB_726297	272.268	71	0	1	71
LTFRB_726297	180.203	60	0	1	60
LTFRB_726297	265.5994	71	0	1	71
LTFRB_726297	301.5684	90	12.5	1	90
LTFRB_726297	656.7224	209	14	1	209
LTFRB_726297	426.4031	134	14.75	1	134
LTFRB_726297	253.1981	60	15.25	1	60
LTFRB_726297	180.1993	60	15.75	1	60
LTFRB_726297	448.4626	137	16.75	1	137
LTFRB_726297	433.3159	134	17.75	1	134
LTFRB_726297	874.6502	247	19.75	1	247
LTFRB_726297	506.3753	154	20.75	1	154
LTFRB_726297	663.9475	212	22.25	1	212
LTFRB_726297	1139.317	338	24.75	1	338
LTFRB_726297	379.7677	109	25.5	1	109
LTFRB_726297	139.5948	50	25.75	1	50
LTFRB_726297	217.5003	59	26.25	1	59
LTFRB_726297	233.5097	71	26.75	1	71
LTFRB_726297	232.8205	80	27.25	1	80
LTFRB_726297	437.2953	119	28.25	1	119
LTFRB_726297	777.6151	231	30	1	231
LTFRB_726297	544.4624	175	31.25	1	175
LTFRB_726297	253.1676	69	32.50	1	69
TOTAL	15294.96		32.50		4984

Total_weight = 15294.96*0.4 + 32.50*0.2 + 4984*0.3 + 376.5325*0.1 = **7657.33725**

TRIP ID	DISTANCE	TIME	FARE	TRAFFIC SCORE	TIME*TRAFFIC SCORE
LTFRB_725951	624.9991	180	0	1	180
LTFRB_725951	373.5684	107	0	1	107
LTFRB_725951	396.9949	115	0	1	115
LTFRB_725951	382.3997	123	0	1	123
LTFRB_725951	360.4322	96	0	1	96
LTFRB_725951	298.0932	96	0	1	96
LTFRB_725951	277.4708	85	0	1	85
LTFRB_725951	277.4712	85	0	1	85
LTFRB_725951	317.5833	83	0	1	83

LTFRB_725951	318.9349	101	0	1	101
LTFRB_725951	287.7538	90	0	1	90
LTFRB_725951	425.4215	124	0	1	124
LTFRB_725951	586.2019	165	0	1	165
LTFRB_725951	267.2604	83	12.75	1	83
LTFRB_725951	247.0674	77	13.75	1	77
LTFRB_726268	385.873	103	0	1	103
LTFRB_726268	309.5085	98	0	1	98
LTFRB_726268	565.6818	168	0	1	168
LTFRB_726268	171.4313	57	0	1	57
LTFRB_726268	670.6983	188	0	1	188
LTFRB_726268	162.9245	54	0	1	54
LTFRB_726268	866.7911	261	0	1	261
LTFRB_726268	301.6067	93	0	1	93
LTFRB_726268	351.5786	97	0	1	97
LTFRB_726268	298.0931	88	0	1	88
LTFRB_726268	610.0216	191	0	1	191
LTFRB_726268	522.9731	148	12.5	1	148
LTFRB_726268	406.3812	129	13.25	1	129
LTFRB_726268	293.9111	75	14	1	75
LTFRB_726268	171.4215	47	14.5	1	47
LTFRB_726268	637.8603	193	15.75	1	193
LTFRB_726268	144.5536	46	16	1	46
LTFRB_726268	189.2457	49	16.5	1	49
LTFRB_726268	166.7927	53	17	1	53
LTFRB_726268	198.4926	55	17.25	1	55
LTFRB_726268	189.0317	57	17.75	1	57
LTFRB_726268	180.208	46	18.25	1	46
LTFRB_726268	177.9122	54	18.5	1	54
LTFRB_726268	189.0317	57	19	1	57
LTFRB_726268	393.0782	117	19.75	1	117
LTFRB_726268	207.9208	59	20.25	1	59
LTFRB_726268	198.4892	58	20.75	1	58
LTFRB_726268	177.9122	57	21	1	57
LTFRB_726268	207.9196	60	21.5	1	60
LTFRB_726268	198.4879	54	22	1	54
LTFRB_726268	133.4341	43	22.25	1	43
LTFRB_726268	207.9185	58	22.75	1	58
LTFRB_726268	198.4868	56	23.25	1	56
LTFRB_726268	217.5087	61	23.75	1	61

LTFRB_726268	177.9122	60	24	1	60
LTFRB_726268	247.0508	70	24.5	1	70
LTFRB_726268	146.941	36	25	1	36
LTFRB_726268	198.4847	59	25.25	1	59
LTFRB_726268	200.1512	63	25.75	1	63
LTFRB_726268	217.5064	59	26.25	1	59
LTFRB_726268	247.0491	75	26.75	1	75
LTFRB_726268	227.2359	66	27.25	1	66
LTFRB_726268	227.2355	63	27.75	1	63
LTFRB_726268	100.0756	36	28	1	36
LTFRB_726268	257.103	73	28.5	1	73
LTFRB_726268	266.8682	81	29.25	1	81
LTFRB_726268	298.0803	85	29.75	1	85
LTFRB_726268	255.7487	79	30.5	1	79
LTFRB_726268	267.2408	76	31	1	76
LTFRB_726268	255.7487	76	31.5	1	76
LTFRB_726268	533.5748	166	32.75	1	166
LTFRB_726268	233.5097	71	33.25	1	71
LTFRB_726268	232.8205	80	33.75	1	80
LTFRB_726268	437.2953	119	34.75	1	119
LTFRB_726268	524.6934	162	36	1	162
LTFRB_726268	796.693	244	38.25	1	244
LTFRB_726268	253.1676	69	38.75	1	69
TOTAL	22347.02		52.50		6608

Total_weight = 22347.02*0.4 + 52.50*0.2 + 6608*0.3 + 376.5325*0.1 = **10969.3612**

TRIP ID	DISTANCE	TIME	FARE	TRAFFIC SCORE	TIME*TRAFFIC SCORE
LTFRB_725139	4465.706	1364	0	1	1364
LTFRB_725139	432.4597	60	0	1	60
LTFRB_725139	681.7727	204	13.25	1	204
LTFRB_725139	378.5238	109	14.5	1	109
LTFRB_725139	350.5216	110	15.75	1	110
LTFRB_725139	415.8813	126	17.5	1	126
LTFRB_725139	444.7595	131	18.25	1	131
LTFRB_725139	514.2533	159	19.75	1	159
LTFRB_724609	444.7595	131	0	1	131
LTFRB_724609	415.8813	126	0	1	126
LTFRB_724609	350.5216	110	0	1	110

LTFRB_724609	1058.782	313	0	1	313
LTFRB_724609	1862.996	0	0	1	0
LTFRB_724609	2004.355	130	14.5	1	130
LTFRB_724609	385.873	103	15.25	1	103
LTFRB_724609	309.5085	98	16	1	98
LTFRB_724609	301.6167	98	16.75	1	98
LTFRB_724609	265.6516	70	17.25	1	70
LTFRB_724609	171.4313	57	17.75	1	57
LTFRB_724609	272.3147	83	18.25	1	83
LTFRB_724609	398.4721	105	19.25	1	105
LTFRB_724609	162.9245	54	19.5	1	54
LTFRB_724609	866.7911	261	21.5	1	261
LTFRB_724609	301.6067	93	22	1	93
LTFRB_724609	351.5786	97	22.75	1	97
LTFRB_724609	298.0931	88	23.5	1	88
LTFRB_724609	610.0216	191	24.75	1	191
LTFRB_724609	522.9731	148	26	1	148
LTFRB_724609	406.3812	129	27	1	129
LTFRB_724609	293.9111	75	27.5	1	75
LTFRB_724609	171.4215	47	28	1	47
LTFRB_724609	257.1133	79	28.5	1	79
LTFRB_724609	166.7927	53	28.75	1	53
LTFRB_724609	227.2457	62	29.25	1	62
LTFRB_724609	144.5536	46	29.75	1	46
LTFRB_724609	189.2457	49	30	1	49
LTFRB_724609	166.7927	53	30.5	1	53
LTFRB_724609	198.4926	55	30.75	1	55
LTFRB_724609	189.0317	57	31.25	1	57
LTFRB_724609	180.208	46	31.75	1	46
LTFRB_724609	177.9122	54	32	1	54
LTFRB_724609	403.7847	113	33	1	113
LTFRB_724609	177.9122	61	33.25	1	61
LTFRB_724609	207.9208	60	33.75	1	60
LTFRB_724609	198.4892	57	34.25	1	57
LTFRB_724609	177.9122	57	34.75	1	57
LTFRB_724609	207.9196	60	35	1	60
LTFRB_724609	318.9259	97	35.75	1	97
LTFRB_724609	207.9185	58	36.25	1	58
LTFRB_724609	198.4868	56	36.75	1	56
LTFRB_724609	217.5087	62	37.25	1	62

LTFRB_724609	177.9122	59	37.5	1	59
LTFRB_724609	247.0508	70	38	1	70
LTFRB_724609	342.8184	95	38.75	1	95
LTFRB_724609	403.7814	122	39.75	1	122
LTFRB_724609	247.0491	75	40.25	1	75
LTFRB_724609	741.7321	225	42.50	1	225
LTFRB_724609	266.8682	81	43.25	1	81
LTFRB_724609	298.0803	84	43.75	1	84
LTFRB_724609	255.7487	80	44.75	1	80
LTFRB_724609	267.2408	75	45.25	1	75
LTFRB_724609	255.7487	77	46	1	77
LTFRB_724609	391.9448	114	46.75	1	114
LTFRB_724609	585.7998	176	47.50	1	176
LTFRB_724609	247.4933	70	48.75	1	70
TOTAL	28855.18		68.50		7508

Total_weight = 28855.18*0.4 + 68.50*0.2 + 7508*0.3 + 376.5325*0.1 = **13845.825**

	Distance(m)*0.4	**Time (s)*0.3**	**Fare (PhP)*0.2**	**Walk Distance (m)*0.1**	**Total Weight**
Trip 1	5743.876	1280.7	5.9	63.30434	**7093.404**
Trip 2	6291.808	1305.3	6.25	37.65325	**7641.011**
Trip 3	6117.984	1495.2	6.5	37.65325	**7657.337**
Trip 4	8938.808	1982.4	10.5	37.65325	**10969.36**
Trip 5	11542.07	2252.4	13.7	37.65325	**13845.83**

Table 11.1 Summary of computations

2. Computation tables for finding trips from UP Manila to UP Diliman where traffic is HEAVY along EDSA and user preference is least fare.

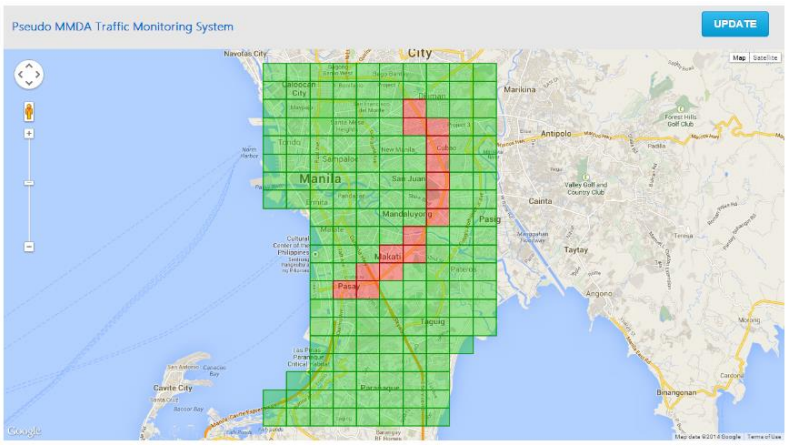


Figure 11.3 Heavy traffic along EDSA

Figure 11.4 Top 5 results

TRIP ID	DISTANCE	TIME	FARE	TRAFFIC SCORE	TIME*TRAFFIC SCORE
LTFRB_726297	217.5172	63	0	1	63
LTFRB_726297	247.0586	76	0	1	76
LTFRB_726297	298.0898	98	0	1	98
LTFRB_726297	565.1976	170	0	1	170
LTFRB_726297	489.2584	146	0	1	146
LTFRB_726297	405.0377	111	0	1	111
LTFRB_726297	222.3902	68	0	1	68
LTFRB_726297	247.0548	69	0	1	69
LTFRB_726297	189.0317	58	0	1	58
LTFRB_726297	426.0066	136	0	1	136
LTFRB_726297	162.9078	48	0	1	48
LTFRB_726297	272.2712	72	0	1	72
LTFRB_726297	272.27	73	0	1	73
LTFRB_726297	162.9065	56	0	1	56
LTFRB_726297	272.268	71	0	1	71
LTFRB_726297	180.203	60	0	1	60
LTFRB_726297	265.5994	71	0	1	71
LTFRB_726297	301.5684	90	12.5	1	90
LTFRB_726297	656.7224	209	14	1	209

LTFRB_726297	426.4031	134	14.75	1	134
LTFRB_726297	253.1981	60	15.25	1	60
LTFRB_726297	180.1993	60	15.75	1	60
LTFRB_726297	448.4626	137	16.75	1	137
LTFRB_726297	433.3159	134	17.75	1	134
LTFRB_726297	874.6502	247	19.75	1	247
LTFRB_726297	506.3753	154	20.75	1	154
LTFRB_726297	663.9475	212	22.25	1	212
LTFRB_726297	1139.317	338	25	1	338
LTFRB_726297	379.7677	109	25.75	1	109
LTFRB_726297	139.5948	50	26	1	50
LTFRB_726297	217.5003	59	26.75	1	59
LTFRB_726297	233.5097	71	27.5	1	71
LTFRB_726297	232.8205	80	28.75	1	80
LTFRB_726297	437.2953	119	29.5	1	119
LTFRB_726297	777.6151	231	31.5	1	231
LTFRB_726297	544.4624	175	32.25	1	175
LTFRB_726297	253.1676	69	32.5	1	69
TOTAL	13994.96		32.5		4184

Total_weight = 13994.96*0.4 + 32.5*0.2 + 4184*0.3 + 376.5325*0.1 = **6897.33725**

TRIP ID	DISTANCE	TIME	FARE	TRAFFIC SCORE	TIME*TRAFFIC SCORE
LTFRB_724970	217.5172	63	0	1	63
LTFRB_724970	247.0586	76	0	1	76
LTFRB_724970	298.0898	98	0	1	98
LTFRB_724970	237.0979	68	0	1	68
LTFRB_724970	287.7473	84	0	1	84
LTFRB_724970	44.47804	19	0	1	19
LTFRB_724970	180.2093	44	0	1	44
LTFRB_724970	189.0317	58	0	1	58
LTFRB_724970	189.2444	64	0	1	64
LTFRB_724970	405.0377	111	0	1	111
LTFRB_724970	222.3902	68	0	1	68
LTFRB_724970	247.0548	69	0	1	69
LTFRB_724970	189.0317	58	0	1	58
LTFRB_724970	189.0317	59	0	1	59
LTFRB_724970	334.8385	100	0	1	100
LTFRB_724970	162.9078	49	0	1	49
LTFRB_724970	272.2712	72	0	1	72

LTFRB_724970	272.27	72	8.25	1	72
LTFRB_724970	162.9065	56	8.5	1	56
LTFRB_724970	272.268	72	8.75	1	72
LTFRB_724970	180.203	59	9	1	59
LTFRB_724970	265.5994	71	9.5	1	71
LTFRB_724970	301.5684	90	10	1	90
LTFRB_724970	447.4273	159	10.5	1	159
LTFRB_724970	324.5061	100	11	1	100
LTFRB_724970	431.6728	106	11.5	1	106
LTFRB_724970	323.5513	103	12	1	103
LTFRB_724970	337.943	92	12.5	1	92
LTFRB_724970	225.2946	75	12.75	1	75
LTFRB_724970	232.8506	89	13.25	1	89
LTFRB_724970	345.1796	104	13.75	1	104
LTFRB_724970	332.0327	86	14	1	86
LTFRB_724970	337.9372	92	14.5	1	92
LTFRB_724970	286.4219	101	15	1	101
LTFRB_724970	237.0891	222	15.25	3	666
LTFRB_724970	225.2874	340	15.75	5	1700
LTFRB_724970	329.5951	405	16	5	2025
LTFRB_724970	337.9292	480	16.5	5	2400
LTFRB_724970	337.928	505	17.5	5	2525
LTFRB_725709	100.0756	180	0	5	900
LTFRB_725709	257.103	365	0	5	1825
LTFRB_725709	266.8682	405	0	5	2025
LTFRB_725709	298.0803	425	0	5	2125
LTFRB_725709	255.7487	395	0	5	1975
LTFRB_725709	267.2408	228	0	3	684
LTFRB_725709	255.7487	76	0	1	76
LTFRB_725709	391.9448	115	0	1	115
LTFRB_725709	1063.391	322	0	1	322
LTFRB_725709	796.693	244	0	1	244
LTFRB_725709	253.1676	68	12	1	68
TOTAL	14666.56		29.5		22362

Total_weight = 14666.56*0.4 + 29.5*0.2 + 22362*0.3 + 901.5028*0.1
= **12671.27428**

TRIP ID	DISTANCE	TIME	FARE	TRAFFIC SCORE	TIME*TRAFFIC SCORE
LTFRB_725202	217.5172	63	0	1	63
LTFRB_725202	247.0586	76	0	1	76
LTFRB_725202	298.0898	98	0	1	98
LTFRB_725202	237.0979	67	0	1	67
LTFRB_725202	287.7473	85	0	1	85
LTFRB_725202	44.47804	19	0	1	19
LTFRB_725202	180.2093	43	0	1	43
LTFRB_725202	189.0317	59	0	1	59
LTFRB_725202	189.2444	64	0	1	64
LTFRB_725202	405.0377	110	0	1	110
LTFRB_725202	222.3902	69	0	1	69
LTFRB_725202	247.0548	69	0	1	69
LTFRB_725202	189.0317	58	0	1	58
LTFRB_725202	189.0317	59	0	1	59
LTFRB_725202	334.8385	100	0	1	100
LTFRB_725202	162.9078	48	0	1	48
LTFRB_725202	272.2712	72	0	1	72
LTFRB_725202	272.27	73	8.25	1	73
LTFRB_725202	162.9065	56	8.5	1	56
LTFRB_725202	272.268	71	8.75	1	71
LTFRB_725202	180.203	60	9	1	60
LTFRB_725202	265.5994	71	9.5	1	71
LTFRB_725202	301.5684	90	10	1	90
LTFRB_725202	447.4273	158	10.5	1	158
LTFRB_725202	324.5061	101	11	1	101
LTFRB_725202	431.6728	106	11.5	1	106
LTFRB_725202	323.5513	102	12	1	102
LTFRB_725202	337.943	93	12.5	1	93
LTFRB_725202	225.2946	75	12.75	1	75
LTFRB_725202	232.8506	89	13.25	1	89
LTFRB_725202	345.1796	104	13.75	1	104
LTFRB_725202	332.0327	85	14	1	85
LTFRB_725202	337.9372	93	14.5	1	93
LTFRB_725202	431.6621	393	15.25	3	1179
LTFRB_725202	325.8279	550	15.75	5	2750
LTFRB_725202	430.9441	550	16.25	5	2750
LTFRB_725202	112.6452	165	16.5	5	825
LTFRB_725202	237.3183	219	16.75	3	657
LTFRB_725202	374.0589	333	17.5	3	999

LTFRB_725202	219.7365	425	18	5	2125
LTFRB_726268	200.1512	315	0	5	1575
LTFRB_726268	217.5064	295	0	5	1475
LTFRB_726268	247.0491	375	0	5	1875
LTFRB_726268	227.2359	330	0	5	1650
LTFRB_726268	227.2355	315	0	5	1575
LTFRB_726268	100.0756	180	0	5	900
LTFRB_726268	257.103	365	0	5	1825
LTFRB_726268	266.8682	405	0	5	2025
LTFRB_726268	298.0803	425	0	5	2125
LTFRB_726268	255.7487	395	0	5	1975
LTFRB_726268	267.2408	228	0	3	684
LTFRB_726268	255.7487	76	0	1	76
LTFRB_726268	533.5748	166	0	1	166
LTFRB_726268	233.5097	71	0	1	71
LTFRB_726268	232.8205	80	0	1	80
LTFRB_726268	437.2953	119	0	1	119
LTFRB_726268	524.6934	162	13.25	1	162
LTFRB_726268	796.693	244	13.75	1	244
LTFRB_726268	253.1676	69	14.25	1	69
TOTAL	16670.24		32.25		32542

Total_weight = 16670.24*0.4 + 32.25*0.2 + 32542*0.3 + 753.065*0.1 = **16506.6475**

TRIP ID	DISTANCE	TIME	FARE	TRAFFIC SCORE	TIME*TRAFFIC SCORE
LRTA_882210	1123.541	170	12	1	170
LRTA_882210	736.7608	90	12	1	90
LRTA_882210	711.2513	90	12	1	90
LRTA_882210	659.2771	80	12	1	80
LRTA_882210	604.1009	80	14	1	80
LRTA_882210	662.954	80	14	1	80
LRTA_882210	907.645	125	14	1	125
LRTA_882210	572.6185	80	15	1	80
LRTA_882210	964.7179	130	15	1	130
LRTA_882210	1099.652	150	15	1	150
LRTA_882210	2167.74	330	15	1	330
LRTA_882210	1885.016	300	15	1	300

LTFRB_724862	441.7904	131	0	1	131
LTFRB_724862	337.8913	127	0	1	127
LTFRB_724862	497.0808	153	0	1	153
LTFRB_724862	37.22968	0	0	1	0
LTFRB_724862	253.718	65	0	1	65
LTFRB_724862	257.0998	83	0	1	83
LTFRB_724862	198.478	58	0	1	58
LTFRB_724862	207.9101	63	0	1	63
LTFRB_724862	198.4788	186	0	3	558
LTFRB_724862	207.9109	310	0	5	1550
LTFRB_724862	198.4796	295	0	5	1475
LTFRB_724862	198.48	295	0	5	1475
LTFRB_724862	189.2321	270	0	5	1350
LTFRB_724862	171.4044	245	0	5	1225
LTFRB_724862	166.7927	290	0	5	1450
LTFRB_724862	180.1967	255	0	5	1275
LTFRB_724862	146.9365	170	12	5	850
LTFRB_725361	266.8682	405	0	5	2025
LTFRB_725361	298.0803	425	0	5	2125
LTFRB_725361	255.7487	395	0	5	1975
LTFRB_725361	267.2408	228	0	3	684
LTFRB_725361	255.7487	76	0	1	76
LTFRB_725361	533.5748	166	0	1	166
LTFRB_725361	233.5097	71	0	1	71
LTFRB_725361	232.8205	80	0	1	80
LTFRB_725361	437.2953	119	0	1	119
LTFRB_725361	777.6151	231	0	1	231
LTFRB_725361	544.4624	175	0	1	175
LTFRB_725361	253.1676	69	12	1	69
TOTAL	20340.52		29		21389

Total_weight = 20340.52*0.4 + 29*0.2 + 21389*0.3 + 429.1699*0.1 = **14601.62499**

TRIP ID	DISTANCE	TIME	FARE	TRAFFIC SCORE	TIME*TRAFFIC SCORE
LTFRB_725735	247.0586	76	0	1	76
LTFRB_725735	298.0898	98	0	1	98
LTFRB_725735	237.0979	68	0	1	68
LTFRB_725735	329.4182	102	0	1	102
LTFRB_725735	180.2093	44	0	1	44
LTFRB_725735	217.5142	65	0	1	65
LTFRB_725735	265.6151	80	0	1	80
LTFRB_725735	405.0377	111	0	1	111
LTFRB_725735	222.3902	69	0	1	69
LTFRB_725735	247.0548	68	0	1	68
LTFRB_725735	189.0317	58	0	1	58
LTFRB_725735	189.0317	59	0	1	59
LTFRB_725735	334.8385	100	0	1	100
LTFRB_725735	162.9078	49	0	1	49
LTFRB_725735	272.2712	72	0	1	72
LTFRB_725735	272.27	72	8	1	72
LTFRB_725735	162.9065	56	8.25	1	56
LTFRB_725735	272.268	72	8.75	1	72
LTFRB_725735	180.203	60	9	1	60
LTFRB_725735	265.5994	70	9.25	1	70
LTFRB_725735	301.5684	90	9.75	1	90
LTFRB_725735	447.4273	159	10.5	1	159
LTFRB_725735	324.5061	101	10.75	1	101
LTFRB_725735	431.6728	106	11.5	1	106
LTFRB_725735	323.5513	102	12	1	102
LTFRB_725735	337.943	93	12.25	1	93
LTFRB_725735	225.2946	75	12.75	1	75
LTFRB_725735	232.8506	88	13	1	88
LTFRB_725735	345.1796	104	13.5	1	104
LTFRB_725735	332.0327	86	14	1	86
LTFRB_725735	337.9372	93	14.5	1	93
LTFRB_725735	431.6621	393	15	3	1179
LTFRB_725735	325.8279	550	15.5	5	2750
LTFRB_725735	430.9441	550	16	5	2750
LTFRB_725735	349.2712	315	16.5	3	945
LTFRB_725735	374.0589	333	17	3	999
LTFRB_725735	219.7365	425	17.75	5	2125
LTFRB_726238	200.1512	310	0	5	1550
LTFRB_726238	217.5064	295	0	5	1475
LTFRB_726238	247.0491	375	0	5	1875

LTFRB_726238	227.2359	330	0	5	1650
LTFRB_726238	227.2355	315	0	5	1575
LTFRB_726238	100.0756	185	0	5	925
LTFRB_726238	257.103	365	0	5	1825
LTFRB_726238	266.8682	405	0	5	2025
LTFRB_726238	298.0803	420	0	5	2100
LTFRB_726238	255.7487	395	0	5	1975
LTFRB_726238	267.2408	228	0	3	684
LTFRB_726238	255.7487	77	0	1	77
LTFRB_726238	533.5748	166	0	1	166
LTFRB_726238	233.5097	71	0	1	71
LTFRB_726238	232.8205	79	0	1	79
LTFRB_726238	437.2953	119	0	1	119
LTFRB_726238	524.6934	163	13.25	1	163
LTFRB_726238	796.693	244	13.75	1	244
LTFRB_726238	253.1676	68	14.25	1	68
TOTAL	16554.08		32		31940

Total_weight = 16554.08*0.4 + 32*0.2 + 31940*0.3 + 1079.6838*0.1 = **16318.00038**

	Distance(m)*0.4	**Time (s)*0.3**	**Fare (PhP)*0.2**	**Walk Distance (m)*0.1**	**Total Weight**
Trip 1	5597.984	1255.2	6.5	37.65325	**6897.337**
Trip 2	5866.624	6708.6	5.9	90.15028	**12671.27**
Trip 3	6668.096	9762.6	6.45	75.3065	**16506.65**
Trip 4	8136.208	6416.7	5.8	42.91699	**14601.62**
Trip 5	6621.632	9582	6.4	107.9684	**16318.00**

Table 11.2 Summary of computations

3. Computation tables for finding trips from UP Manila to UP Diliman where traffic is HEAVY along Quezon Blvd. and user preference is least fare

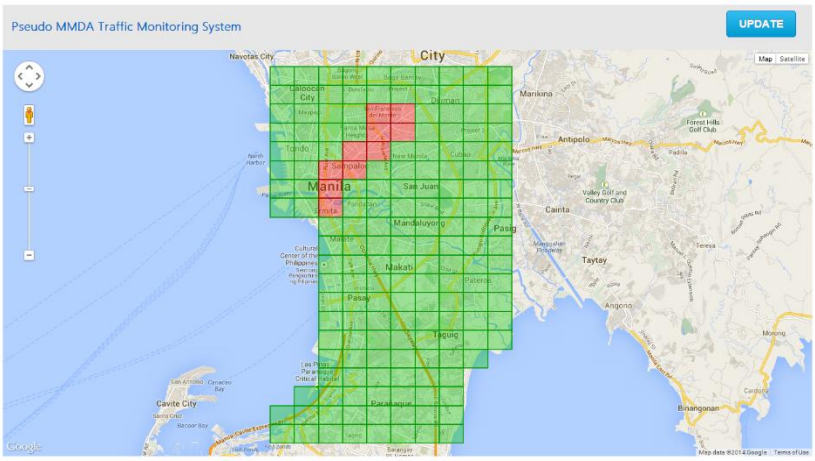


Figure 11.5 Heavy traffic along Quezon Blvd

Figure 11.6 Top 5 results

TRIP ID	DISTANCE	TIME	FARE	TRAFFIC SCORE	TIME*TRAFFIC SCORE
LTFRB_726297	217.5172	63	0	1	63
LTFRB_726297	247.0586	228	0	3	684
LTFRB_726297	298.0898	490	0	5	2450
LTFRB_726297	565.1976	850	0	5	4250
LTFRB_726297	489.2584	730	0	5	3650
LTFRB_726297	405.0377	555	0	5	2775
LTFRB_726297	222.3902	340	0	5	1700
LTFRB_726297	247.0548	345	0	5	1725
LTFRB_726297	189.0317	290	0	5	1450
LTFRB_726297	426.0066	680	0	5	3400
LTFRB_726297	162.9078	240	0	5	1200
LTFRB_726297	272.2712	360	0	5	1800
LTFRB_726297	272.27	365	0	5	1825
LTFRB_726297	162.9065	280	0	5	1400
LTFRB_726297	272.268	355	0	5	1775
LTFRB_726297	180.203	300	0	5	1500
LTFRB_726297	265.5994	355	0	5	1775
LTFRB_726297	301.5684	450	12.5	5	2250
LTFRB_726297	656.7224	1045	14	5	5225
LTFRB_726297	426.4031	670	14.75	5	3350
LTFRB_726297	253.1981	300	15.25	5	1500

LTFRB_726297	180.1993	300	15.75	5	1500
LTFRB_726297	448.4626	685	16.75	5	3425
LTFRB_726297	433.3159	670	17.75	5	3350
LTFRB_726297	874.6502	1235	19.75	5	6175
LTFRB_726297	506.3753	770	20.75	5	3850
LTFRB_726297	663.9475	1060	22.25	5	5300
LTFRB_726297	1139.317	1014	24.75	3	3042
LTFRB_726297	379.7677	109	25.5	1	109
LTFRB_726297	139.5948	50	25.75	1	50
LTFRB_726297	217.5003	59	26.25	1	59
LTFRB_726297	233.5097	71	26.75	1	71
LTFRB_726297	232.8205	80	27.25	1	80
LTFRB_726297	437.2953	119	29.25	1	119
LTFRB_726297	777.6151	231	30.5	1	231
LTFRB_726297	544.4624	175	31.75	1	175
LTFRB_726297	253.1676	69	32.5	1	69
TOTAL	13994.96		32.5		73352

Total_weight = 13994.96*0.4 + 32.5*0.2 + 73352*0.3 + 376.5325*0.1 = **27647.3532**

TRIP ID	DISTANCE	TIME	FARE	TRAFFIC SCORE	TIME*TRAFFIC SCORE
LTFRB_725951	624.9991	180	0	1	180
LTFRB_725951	373.5684	107	0	1	107
LTFRB_725951	396.9949	115	0	1	115
LTFRB_725951	382.3997	123	0	1	123
LTFRB_725951	360.4322	96	0	1	96
LTFRB_725951	298.0932	96	0	1	96
LTFRB_725951	277.4708	85	0	1	85
LTFRB_725951	277.4712	85	0	1	85
LTFRB_725951	317.5833	83	0	1	83
LTFRB_725951	318.9349	101	0	1	101
LTFRB_725951	287.7538	90	0	1	90
LTFRB_725951	425.4215	124	0	1	124
LTFRB_725951	586.2019	165	0	1	165
LTFRB_725951	267.2604	83	12.75	1	83
LTFRB_725951	247.0674	77	13.75	1	77
LTFRB_726268	385.873	103	0	1	103
LTFRB_726268	309.5085	98	0	1	98
LTFRB_726268	565.6818	168	0	1	168
LTFRB_726268	171.4313	57	0	1	57

LTFRB_726268	670.6983	188	0	1	188
LTFRB_726268	162.9245	54	0	1	54
LTFRB_726268	866.7911	261	0	1	261
LTFRB_726268	301.6067	93	0	1	93
LTFRB_726268	351.5786	97	0	1	97
LTFRB_726268	298.0931	88	0	1	88
LTFRB_726268	610.0216	191	0	1	191
LTFRB_726268	522.9731	148	12.5	1	148
LTFRB_726268	406.3812	129	13.25	1	129
LTFRB_726268	293.9111	75	14	1	75
LTFRB_726268	171.4215	47	14.5	1	47
LTFRB_726268	637.8603	193	15.75	1	193
LTFRB_726268	144.5536	46	16	1	46
LTFRB_726268	189.2457	49	16.5	1	49
LTFRB_726268	166.7927	53	17	1	53
LTFRB_726268	198.4926	55	17.25	1	55
LTFRB_726268	189.0317	57	17.75	1	57
LTFRB_726268	180.208	46	18.25	1	46
LTFRB_726268	177.9122	54	18.5	1	54
LTFRB_726268	189.0317	57	19	1	57
LTFRB_726268	393.0782	117	19.75	1	117
LTFRB_726268	207.9208	59	20.25	1	59
LTFRB_726268	198.4892	58	20.75	1	58
LTFRB_726268	177.9122	57	21	1	57
LTFRB_726268	207.9196	60	21.5	1	60
LTFRB_726268	198.4879	54	22	1	54
LTFRB_726268	133.4341	43	22.25	1	43
LTFRB_726268	207.9185	58	22.75	1	58
LTFRB_726268	198.4868	56	23.25	1	56
LTFRB_726268	217.5087	61	23.75	1	61
LTFRB_726268	177.9122	60	24	1	60
LTFRB_726268	247.0508	70	24.5	1	70
LTFRB_726268	146.941	36	25	1	36
LTFRB_726268	198.4847	59	25.25	1	59
LTFRB_726268	200.1512	63	25.75	1	63
LTFRB_726268	217.5064	59	26.25	1	59
LTFRB_726268	247.0491	75	26.75	1	75
LTFRB_726268	227.2359	66	27.25	1	66
LTFRB_726268	227.2355	63	27.75	1	63
LTFRB_726268	100.0756	36	28	1	36
LTFRB_726268	257.103	219	28.5	3	657

LTFRB_726268	266.8682	405	29.25	5	2025
LTFRB_726268	298.0803	425	29.75	5	2125
LTFRB_726268	255.7487	395	30.5	5	1975
LTFRB_726268	267.2408	228	31	3	684
LTFRB_726268	255.7487	76	31.5	1	76
LTFRB_726268	533.5748	166	32.75	1	166
LTFRB_726268	233.5097	71	33.25	1	71
LTFRB_726268	232.8205	80	33.75	1	80
LTFRB_726268	437.2953	119	34.75	1	119
LTFRB_726268	524.6934	162	36.25	1	162
LTFRB_726268	796.693	244	38	1	244
LTFRB_726268	253.1676	69	38.75	1	69
TOTAL	22347.02		52.50		13680

Total_weight = 22347.02*0.4 + 52.50*0.2 + 13680*0.3 + 376.5325*0.1 = **13090.96125**

TRIP ID	DISTANCE	TIME	FARE	TRAFFIC SCORE	TIME*TRAFFIC SCORE
LTFRB_725702	531.3134	152	0	1	152
LTFRB_725702	238.6807	74	0	1	74
LTFRB_725702	406.387	116	0	1	116
LTFRB_725702	585.8316	172	0	1	172
LTFRB_725702	277.9878	85	0	1	85
LTFRB_725702	308.4904	92	0	1	92
LTFRB_725702	591.1283	181	0	1	181
LTFRB_725702	851.909	254	12	1	254
LTFRB_725265	277.9878	86	0	1	86
LTFRB_725265	225.3576	76	0	1	76
LTFRB_725265	437.4893	125	0	1	125
LTFRB_725265	454.1282	162	0	1	162
LTFRB_725265	1160.34	328	0	1	328
LTFRB_725265	363.4098	101	0	1	101
LTFRB_725265	341.4821	105	0	1	105
LTFRB_725265	207.9318	69	0	1	69
LTFRB_725265	537.8793	171	0	1	171
LTFRB_725265	392.0559	123	0	1	123
LTFRB_725265	734.8273	220	12.25	1	220
LTFRB_725265	670.6983	188	13.75	1	188
LTFRB_725265	162.9245	53	14.25	1	53

LTFRB_725265	866.7911	261	16	1	261
LTFRB_725265	301.6067	93	16.75	1	93
LTFRB_725265	351.5786	97	17.5	1	97
LTFRB_725265	298.0931	89	18.25	1	89
LTFRB_725265	610.0216	191	19.5	1	191
LTFRB_725265	522.9731	147	20.5	1	147
LTFRB_725265	406.3812	129	21.5	1	129
LTFRB_725265	293.9111	76	22.25	1	76
LTFRB_725265	171.4215	46	22.5	1	46
LTFRB_725265	257.1133	80	23	1	80
LTFRB_725265	166.7927	52	23.5	1	52
LTFRB_725265	227.2457	63	24	1	63
LTFRB_725265	144.5536	46	24.25	1	46
LTFRB_725265	189.2457	49	24.75	1	49
LTFRB_725265	166.7927	52	25	1	52
LTFRB_725265	198.4926	55	25.5	1	55
LTFRB_725265	189.0317	57	26	1	57
LTFRB_725265	180.208	46	26.25	1	46
LTFRB_725265	177.9122	54	26.75	1	54
LTFRB_725265	189.0317	57	27.25	1	57
LTFRB_725265	227.2425	57	27.75	1	57
LTFRB_725265	177.9122	61	28	1	61
LTFRB_725265	207.9208	60	28.5	1	60
LTFRB_725265	198.4892	57	29	1	57
LTFRB_725265	177.9122	57	29.25	1	57
LTFRB_725265	207.9196	60	29.75	1	60
LTFRB_725265	198.4879	54	30.25	1	54
LTFRB_725265	133.4341	43	30.5	1	43
LTFRB_725265	207.9185	58	31	1	58
LTFRB_725265	198.4868	56	31.5	1	56
LTFRB_725265	217.5087	61	31.75	1	61
LTFRB_725265	177.9122	60	32.25	1	60
LTFRB_725265	247.0508	70	32.75	1	70
LTFRB_725265	146.941	36	33	1	36
LTFRB_725265	198.4847	59	33.5	1	59
LTFRB_725265	200.1512	63	34	1	63
LTFRB_725265	217.5064	59	34.5	1	59
LTFRB_725265	247.0491	75	35	1	75
LTFRB_725265	227.2359	66	35.5	1	66
LTFRB_725265	227.2355	63	36	1	63

LTFRB_725265	100.0756	36	36.25	1	36
LTFRB_725265	257.103	73	36.75	1	73
LTFRB_725265	266.8682	81	37.5	1	81
LTFRB_725265	298.0803	85	38	1	85
LTFRB_725265	255.7487	79	38.5	1	79
LTFRB_725265	267.2408	76	39.25	1	76
LTFRB_725265	255.7487	76	39.75	1	76
LTFRB_725265	533.5748	166	41	1	166
LTFRB_725265	233.5097	71	41.5	1	71
LTFRB_725265	232.8205	80	42	1	80
LTFRB_725265	437.2953	119	43	1	119
LTFRB_725265	524.6934	162	44.5	1	162
LTFRB_725265	796.693	244	46.75	1	244
LTFRB_725265	253.1676	69	47.5	1	69
TOTAL	24422.86		59.5		7265

Total_weight = 24422.86*0.4 + 59.5*0.2 + 7265*0.3 + 753.065*0.1 = **12035.8505**

TRIP ID	DISTANCE	TIME	FARE	TRAFFIC SCORE	TIME*TRAFFIC SCORE
LTFRB_726309	624.8177	190	0	1	190
LTFRB_726309	585.8316	172	0	1	172
LTFRB_726309	277.9878	85	0	1	85
LTFRB_726309	455.8999	139	0	1	139
LTFRB_726309	437.4893	125	0	1	125
LTFRB_726309	225.3576	75	0	1	75
LTFRB_726309	277.9878	87	12	1	87
LTFRB_726167	225.3576	76	0	1	76
LTFRB_726167	437.4893	125	0	1	125
LTFRB_726167	454.1282	162	0	1	162
LTFRB_726167	1160.34	328	0	1	328
LTFRB_726167	363.4098	101	0	1	101
LTFRB_726167	341.4821	105	0	1	105
LTFRB_726167	207.9318	69	0	1	69
LTFRB_726167	537.8793	171	0	1	171
LTFRB_726167	392.0559	123	0	1	123
LTFRB_726167	734.8273	220	0	1	220
LTFRB_726167	670.6983	188	13.25	1	188
LTFRB_726167	162.9245	53	13.5	1	53

LTFRB_726167	866.7911	261	15.5	1	261
LTFRB_726167	301.6067	93	16	1	93
LTFRB_726167	351.5786	97	16.75	1	97
LTFRB_726167	298.0931	89	17.5	1	89
LTFRB_726167	610.0216	191	18.75	1	191
LTFRB_726167	522.9731	147	20	1	147
LTFRB_726167	406.3812	129	21	1	129
LTFRB_726167	293.9111	76	21.5	1	76
LTFRB_726167	171.4215	46	22	1	46
LTFRB_726167	257.1133	80	22.5	1	80
LTFRB_726167	166.7927	52	22.75	1	52
LTFRB_726167	227.2457	63	23.25	1	63
LTFRB_726167	144.5536	46	23.75	1	46
LTFRB_726167	189.2457	49	24	1	49
LTFRB_726167	166.7927	52	24.5	1	52
LTFRB_726167	198.4926	55	25	1	55
LTFRB_726167	189.0317	57	25.25	1	57
LTFRB_726167	180.208	46	25.75	1	46
LTFRB_726167	177.9122	54	26	1	54
LTFRB_726167	189.0317	57	26.5	1	57
LTFRB_726167	227.2425	57	27	1	57
LTFRB_726167	177.9122	61	27.5	1	61
LTFRB_726167	207.9208	60	27.75	1	60
LTFRB_726167	198.4892	57	28.25	1	57
LTFRB_726167	177.9122	57	28.75	1	57
LTFRB_726167	207.9196	60	29.25	1	60
LTFRB_726167	198.4879	54	29.5	1	54
LTFRB_726167	133.4341	43	30	1	43
LTFRB_726167	207.9185	58	30.25	1	58
LTFRB_726167	198.4868	56	30.75	1	56
LTFRB_726167	217.5087	61	31.25	1	61
LTFRB_726167	177.9122	60	31.75	1	60
LTFRB_726167	247.0508	70	32.25	1	70
LTFRB_726167	146.941	36	32.5	1	36
LTFRB_726167	198.4847	59	33	1	59
LTFRB_726167	200.1512	63	33.5	1	63
LTFRB_726167	217.5064	59	33.75	1	59
LTFRB_726167	247.0491	75	34.5	1	75
LTFRB_726167	227.2359	66	35	1	66
LTFRB_726167	227.2355	63	35.5	1	63

LTFRB_726167	100.0756	36	35.75	1	36
LTFRB_726167	257.103	73	36.25	1	73
LTFRB_726167	266.8682	81	36.75	1	81
LTFRB_726167	298.0803	85	37.5	1	85
LTFRB_726167	255.7487	79	38	1	79
LTFRB_726167	267.2408	76	38.5	1	76
LTFRB_726167	255.7487	76	39.25	1	76
LTFRB_726167	533.5748	166	40.25	1	166
LTFRB_726167	233.5097	71	40.75	1	71
LTFRB_726167	232.8205	80	41.25	1	80
LTFRB_726167	437.2953	119	42.25	1	119
LTFRB_726167	524.6934	162	43.75	1	162
LTFRB_726167	796.693	244	45.75	1	244
LTFRB_726167	253.1676	69	46.5	1	69
TOTAL	23238.51		58.5		6926

Total_weight = 23238.51*0.4 + 58.5*0.2 + 6926*0.3 + 753.065*0.1 = **11460.2065**

TRIP ID	DIRECTION	TIME	FARE	TRAFFIC SCORE	TIME*TRAFFIC SCORE
LTFRB_725139	4465.706	1364	0	1	1364
LTFRB_725139	432.4597	60	0	1	60
LTFRB_725139	681.7727	204	13.25	1	204
LTFRB_725139	378.5238	109	14	1	109
LTFRB_725139	350.5216	110	15	1	110
LTFRB_725139	415.8813	126	15.75	1	126
LTFRB_725139	444.7595	131	17.25	1	131
LTFRB_725139	514.2533	159	19.5	1	159
LTFRB_724609	444.7595	131	0	1	131
LTFRB_724609	415.8813	126	0	1	126
LTFRB_724609	350.5216	110	0	1	110
LTFRB_724609	1058.782	313	0	1	313
LTFRB_724609	1862.996	0	0	1	0
LTFRB_724609	2004.355	130	14.5	1	130
LTFRB_724609	385.873	103	15.25	1	103
LTFRB_724609	309.5085	98	16	1	98

LTFRB_724609	301.6167	98	16.75	1	98
LTFRB_724609	265.6516	70	17.25	1	70
LTFRB_724609	171.4313	57	17.75	1	57
LTFRB_724609	272.3147	83	18.25	1	83
LTFRB_724609	398.4721	105	19.25	1	105
LTFRB_724609	162.9245	54	19.5	1	54
LTFRB_724609	866.7911	261	21.5	1	261
LTFRB_724609	301.6067	93	22	1	93
LTFRB_724609	351.5786	97	22.75	1	97
LTFRB_724609	298.0931	88	23.5	1	88
LTFRB_724609	610.0216	191	24.75	1	191
LTFRB_724609	522.9731	148	26	1	148
LTFRB_724609	406.3812	129	27	1	129
LTFRB_724609	293.9111	75	27.5	1	75
LTFRB_724609	171.4215	47	28	1	47
LTFRB_724609	257.1133	79	28.5	1	79
LTFRB_724609	166.7927	53	28.75	1	53
LTFRB_724609	227.2457	62	29.25	1	62
LTFRB_724609	144.5536	46	29.75	1	46
LTFRB_724609	189.2457	49	30	1	49
LTFRB_724609	166.7927	53	30.5	1	53
LTFRB_724609	198.4926	55	30.75	1	55
LTFRB_724609	189.0317	57	31.25	1	57
LTFRB_724609	180.208	46	31.75	1	46
LTFRB_724609	177.9122	54	32	1	54
LTFRB_724609	403.7847	113	33	1	113
LTFRB_724609	177.9122	61	33.25	1	61
LTFRB_724609	207.9208	60	33.75	1	60
LTFRB_724609	198.4892	57	34.25	1	57
LTFRB_724609	177.9122	57	34.75	1	57
LTFRB_724609	207.9196	60	35	1	60
LTFRB_724609	318.9259	97	35.75	1	97
LTFRB_724609	207.9185	58	36.25	1	58
LTFRB_724609	198.4868	56	36.75	1	56
LTFRB_724609	217.5087	62	37.25	1	62
LTFRB_724609	177.9122	59	37.5	1	59
LTFRB_724609	247.0508	70	38	1	70
LTFRB_724609	342.8184	95	38.75	1	95
LTFRB_724609	403.7814	122	39.75	1	122
LTFRB_724609	247.0491	75	40.25	1	75

LTFRB_724609	741.7321	675	42	3	2025
LTFRB_724609	266.8682	405	42.5	5	2025
LTFRB_724609	298.0803	420	43.25	5	2100
LTFRB_724609	255.7487	400	43.75	5	2000
LTFRB_724609	267.2408	225	44.25	3	675
LTFRB_724609	255.7487	77	45	1	77
LTFRB_724609	391.9448	114	45.75	1	114
LTFRB_724609	585.7998	176	48	1	176
LTFRB_724609	247.4933	70	49	1	70
TOTAL	28855.18		68.5		15788

Total_weight = 28855.18*0.4 + 68.5*0.2 + 15788*0.3 + 753.065*0.1 = **16367.4785**

	Distance(m)*0.4	**Time (s)*0.3**	**Fare (PhP)*0.2**	**Walk Distance (m)*0.1**	**Total Weight**
Trip 1	5597.984	22005.6	6.5	37.65325	**27647.35**
Trip 2	8938.808	4104	10.5	37.65325	**13090.96**
Trip 3	9769.144	2179.5	11.9	75.3065	**12035.85**
Trip 4	9295.404	2077.8	11.7	75.3065	**11460.21**
Trip 5	11542.07	4736.4	13.7	75.3065	**16367.48**

Table 11.3 Summary of computations

4. Computation tables for finding trips from UP Manila to UP Diliman where traffic is LIGHT on all areas and user preference is least time

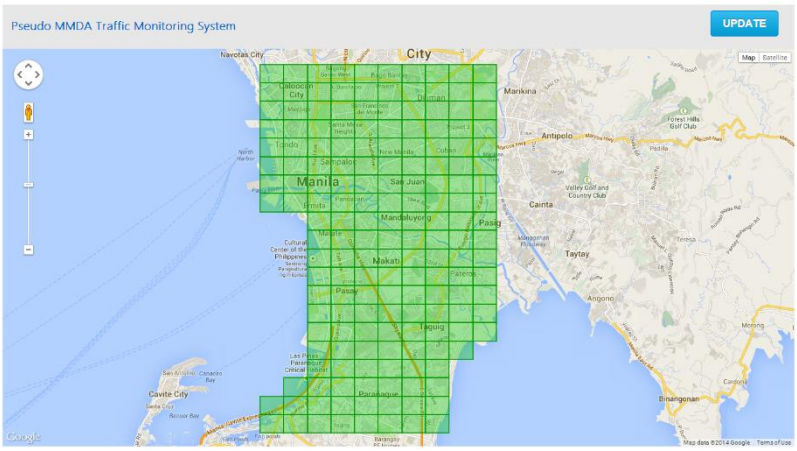


Figure 11.7 Light traffic on all areas

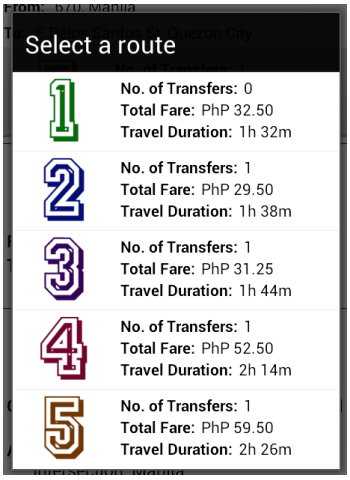


Figure 11.8 Top 5 results

TRIP ID	DISTANCE	TIME	FARE	TRAFFIC SCORE	TIME*TRAFFIC SCORE
LTFRB_726297	217.5172	63	0	1	63
LTFRB_726297	247.0586	76	0	1	76
LTFRB_726297	298.0898	98	0	1	98
LTFRB_726297	565.1976	170	0	1	170
LTFRB_726297	489.2584	146	0	1	146
LTFRB_726297	405.0377	111	0	1	111
LTFRB_726297	222.3902	68	0	1	68
LTFRB_726297	247.0548	69	0	1	69
LTFRB_726297	189.0317	58	0	1	58
LTFRB_726297	426.0066	136	0	1	136
LTFRB_726297	162.9078	48	0	1	48
LTFRB_726297	272.2712	72	0	1	72
LTFRB_726297	272.27	73	0	1	73
LTFRB_726297	162.9065	56	0	1	56
LTFRB_726297	272.268	71	0	1	71
LTFRB_726297	180.203	60	0	1	60
LTFRB_726297	265.5994	71	0	1	71
LTFRB_726297	301.5684	90	12.5	1	90
LTFRB_726297	656.7224	209	14	1	209
LTFRB_726297	426.4031	134	14.75	1	134
LTFRB_726297	253.1981	60	15.25	1	60

LTFRB_726297	180.1993	60	15.75	1	60
LTFRB_726297	448.4626	137	16.75	1	137
LTFRB_726297	433.3159	134	17.75	1	134
LTFRB_726297	874.6502	247	19.75	1	247
LTFRB_726297	506.3753	154	20.75	1	154
LTFRB_726297	663.9475	212	22.25	1	212
LTFRB_726297	1139.317	338	24.75	1	338
LTFRB_726297	379.7677	109	25.5	1	109
LTFRB_726297	139.5948	50	25.75	1	50
LTFRB_726297	217.5003	59	26.25	1	59
LTFRB_726297	233.5097	71	26.75	1	71
LTFRB_726297	232.8205	80	27.25	1	80
LTFRB_726297	437.2953	119	28.25	1	119
LTFRB_726297	777.6151	231	30	1	231
LTFRB_726297	544.4624	175	31.75	1	175
LTFRB_726297	253.1676	69	32.5	1	69
TOTAL	13994.96		32.5		4184

Total_weight = 13994.96*0.3 + 32.5*0.4 + 4184*0.2 + 376.5325*0.1 = **5085.94125**

TRIP ID	DISTANCE	TIME	FARE	TRAFFIC SCORE	TIME*TRAFFIC SCORE
LTFRB_724970	217.5172	63	0	1	63
LTFRB_724970	247.0586	76	0	1	76
LTFRB_724970	298.0898	98	0	1	98
LTFRB_724970	237.0979	68	0	1	68
LTFRB_724970	287.7473	84	0	1	84
LTFRB_724970	44.47804	19	0	1	19
LTFRB_724970	180.2093	44	0	1	44
LTFRB_724970	189.0317	58	0	1	58
LTFRB_724970	189.2444	64	0	1	64
LTFRB_724970	405.0377	111	0	1	111
LTFRB_724970	222.3902	68	0	1	68
LTFRB_724970	247.0548	69	0	1	69
LTFRB_724970	189.0317	58	0	1	58
LTFRB_724970	189.0317	59	0	1	59
LTFRB_724970	334.8385	100	0	1	100
LTFRB_724970	162.9078	49	0	1	49
LTFRB_724970	272.2712	72	0	1	72
LTFRB_724970	272.27	72	8.25	1	72
LTFRB_724970	162.9065	56	8.5	1	56

LTFRB_724970	272.268	72	8.75	1	72
LTFRB_724970	180.203	59	9	1	59
LTFRB_724970	265.5994	71	9.5	1	71
LTFRB_724970	301.5684	90	10	1	90
LTFRB_724970	447.4273	159	10.5	1	159
LTFRB_724970	324.5061	100	11	1	100
LTFRB_724970	431.6728	106	11.5	1	106
LTFRB_724970	323.5513	103	12	1	103
LTFRB_724970	337.943	92	12.5	1	92
LTFRB_724970	225.2946	75	12.75	1	75
LTFRB_724970	232.8506	89	13.25	1	89
LTFRB_724970	345.1796	104	13.75	1	104
LTFRB_724970	332.0327	86	14	1	86
LTFRB_724970	337.9372	92	14.5	1	92
LTFRB_724970	286.4219	101	15	1	101
LTFRB_724970	237.0891	74	15.25	1	74
LTFRB_724970	225.2874	68	15.75	1	68
LTFRB_724970	329.5951	81	16.25	1	81
LTFRB_724970	337.9292	96	16.75	1	96
LTFRB_724970	337.928	101	17.5	1	101
LTFRB_725618	266.8682	81	0	1	81
LTFRB_725618	298.0803	85	0	1	85
LTFRB_725618	255.7487	79	0	1	79
LTFRB_725618	267.2408	76	0	1	76
LTFRB_725618	255.7487	76	0	1	76
LTFRB_725618	391.9448	114	0	1	114
LTFRB_725618	585.7998	176	0	1	176
LTFRB_725618	247.4933	70	0	1	70
LTFRB_725618	279.185	93	0	1	93
LTFRB_725618	253.1711	69	0	1	69
LTFRB_725618	265.5645	86	0	1	86
LTFRB_725618	279.1818	89	0	1	89
LTFRB_725618	253.1676	68	12	1	68
TOTAL	14359.69		29.5		4269

Total_weight = 14359.69*0.3 + 29.5*0.4 + 4269*0.2 + 1009.5755*0.1 = **5274.25755**

TRIP ID	DISTANCE	TIME	FARE	TRAFFIC SCORE	TIME*TRAFFIC SCORE
LTFRB_725202	217.5172	63	0	1	63
LTFRB_725202	247.0586	76	0	1	76
LTFRB_725202	298.0898	98	0	1	98
LTFRB_725202	237.0979	67	0	1	67
LTFRB_725202	287.7473	85	0	1	85
LTFRB_725202	44.47804	19	0	1	19
LTFRB_725202	180.2093	43	0	1	43
LTFRB_725202	189.0317	59	0	1	59
LTFRB_725202	189.2444	64	0	1	64
LTFRB_725202	405.0377	110	0	1	110
LTFRB_725202	222.3902	69	0	1	69
LTFRB_725202	247.0548	69	0	1	69
LTFRB_725202	189.0317	58	0	1	58
LTFRB_725202	189.0317	59	0	1	59
LTFRB_725202	334.8385	100	0	1	100
LTFRB_725202	162.9078	48	0	1	48
LTFRB_725202	272.2712	72	0	1	72
LTFRB_725202	272.27	73	8.25	1	73
LTFRB_725202	162.9065	56	8.5	1	56
LTFRB_725202	272.268	71	8.75	1	71
LTFRB_725202	180.203	60	9	1	60
LTFRB_725202	265.5994	71	9.5	1	71
LTFRB_725202	301.5684	90	10	1	90
LTFRB_725202	447.4273	158	10.5	1	158
LTFRB_725202	324.5061	101	11	1	101
LTFRB_725202	431.6728	106	11.5	1	106
LTFRB_725202	323.5513	102	12	1	102
LTFRB_725202	337.943	93	12.5	1	93
LTFRB_725202	225.2946	75	12.75	1	75
LTFRB_725202	232.8506	89	13.25	1	89
LTFRB_725202	345.1796	104	13.75	1	104
LTFRB_725202	332.0327	85	14	1	85
LTFRB_725202	337.9372	93	14.5	1	93
LTFRB_725202	431.6621	131	15.25	1	131
LTFRB_725202	325.8279	110	15.75	1	110
LTFRB_725202	430.9441	110	16.25	1	110
LTFRB_725202	112.6452	33	16.5	1	33
LTFRB_725202	237.3183	73	17.25	1	73
LTFRB_725202	374.0589	111	17.5	1	111
LTFRB_725202	219.7365	85	18	1	85

LTFRB_725283	200.1512	62	0	1	62
LTFRB_725283	217.5064	59	0	1	59
LTFRB_725283	247.0491	75	0	1	75
LTFRB_725283	227.2359	66	0	1	66
LTFRB_725283	227.2355	63	0	1	63
LTFRB_725283	100.0756	37	0	1	37
LTFRB_725283	257.103	73	0	1	73
LTFRB_725283	266.8682	81	0	1	81
LTFRB_725283	298.0803	84	0	1	84
LTFRB_725283	255.7487	79	0	1	79
LTFRB_725283	267.2408	76	0	1	76
LTFRB_725283	255.7487	77	0	1	77
LTFRB_725283	391.9448	114	0	1	114
LTFRB_725283	457.6057	136	0	1	136
LTFRB_725283	146.9294	55	0	1	55
LTFRB_725283	524.6934	162	12.25	1	162
LTFRB_725283	796.693	244	12.75	1	244
LTFRB_725283	253.1676	69	13.25	1	69
TOTAL	16229.52		31.25		4851

Total_weight = 16229.52*0.3 + 31.25*0.4 + 4851*0.2 + 753.065*0.1 = **5926.8625**

TRIP ID	DISTANCE	TIME	FARE	TRAFFIC SCORE	TIME*TRAFFIC SCORE
LTFRB_725951	624.9991	180	0	1	180
LTFRB_725951	373.5684	107	0	1	107
LTFRB_725951	396.9949	115	0	1	115
LTFRB_725951	382.3997	123	0	1	123
LTFRB_725951	360.4322	96	0	1	96
LTFRB_725951	298.0932	96	0	1	96
LTFRB_725951	277.4708	85	0	1	85
LTFRB_725951	277.4712	85	0	1	85
LTFRB_725951	317.5833	83	0	1	83
LTFRB_725951	318.9349	101	0	1	101
LTFRB_725951	287.7538	90	0	1	90
LTFRB_725951	425.4215	124	0	1	124
LTFRB_725951	586.2019	165	12.5	1	165

LTFRB_725951	267.2604	83	13.25	1	83
LTFRB_725951	247.0674	77	13.75	1	77
LTFRB_726268	385.873	103	0	1	103
LTFRB_726268	309.5085	98	0	1	98
LTFRB_726268	565.6818	168	0	1	168
LTFRB_726268	171.4313	57	0	1	57
LTFRB_726268	670.6983	188	0	1	188
LTFRB_726268	162.9245	54	0	1	54
LTFRB_726268	866.7911	261	0	1	261
LTFRB_726268	301.6067	93	0	1	93
LTFRB_726268	351.5786	97	0	1	97
LTFRB_726268	298.0931	88	0	1	88
LTFRB_726268	610.0216	191	0	1	191
LTFRB_726268	522.9731	148	12.5	1	148
LTFRB_726268	406.3812	129	13.25	1	129
LTFRB_726268	293.9111	75	14	1	75
LTFRB_726268	171.4215	47	14.5	1	47
LTFRB_726268	637.8603	193	15.75	1	193
LTFRB_726268	144.5536	46	16	1	46
LTFRB_726268	189.2457	49	16.5	1	49
LTFRB_726268	166.7927	53	17	1	53
LTFRB_726268	198.4926	55	17.25	1	55
LTFRB_726268	189.0317	57	17.75	1	57
LTFRB_726268	180.208	46	18.25	1	46
LTFRB_726268	177.9122	54	18.5	1	54
LTFRB_726268	189.0317	57	19	1	57
LTFRB_726268	393.0782	117	19.75	1	117
LTFRB_726268	207.9208	59	20.25	1	59
LTFRB_726268	198.4892	58	20.75	1	58
LTFRB_726268	177.9122	57	21	1	57
LTFRB_726268	207.9196	60	21.5	1	60
LTFRB_726268	198.4879	54	22	1	54
LTFRB_726268	133.4341	43	22.25	1	43
LTFRB_726268	207.9185	58	22.75	1	58
LTFRB_726268	198.4868	56	23.25	1	56
LTFRB_726268	217.5087	61	23.75	1	61
LTFRB_726268	177.9122	60	24	1	60
LTFRB_726268	247.0508	70	24.5	1	70
LTFRB_726268	146.941	36	25	1	36
LTFRB_726268	198.4847	59	25.25	1	59
LTFRB_726268	200.1512	63	25.75	1	63

LTFRB_726268	217.5064	59	26.25	1	59
LTFRB_726268	247.0491	75	26.75	1	75
LTFRB_726268	227.2359	66	27.25	1	66
LTFRB_726268	227.2355	63	27.75	1	63
LTFRB_726268	100.0756	36	28	1	36
LTFRB_726268	257.103	73	28.5	1	73
LTFRB_726268	266.8682	81	29.25	1	81
LTFRB_726268	298.0803	85	29.75	1	85
LTFRB_726268	255.7487	79	30.5	1	79
LTFRB_726268	267.2408	76	31	1	76
LTFRB_726268	255.7487	76	31.5	1	76
LTFRB_726268	533.5748	166	32.75	1	166
LTFRB_726268	233.5097	71	33.25	1	71
LTFRB_726268	232.8205	80	33.75	1	80
LTFRB_726268	437.2953	119	34.75	1	119
LTFRB_726268	524.6934	162	36.5	1	162
LTFRB_726268	796.693	244	38.25	1	244
LTFRB_726268	253.1676	69	38.75	1	69
TOTAL	22347.02		52.5		6608

Total_weight = 22347.02*0.3 + 52.5*0.4 + 6608*0.2 + 753.065*0.1 = **8122.0065**

TRIP ID	DISTANCE	TIME	FARE	TRAFFIC SCORE	TIME*TRAFFIC SCORE
LTFRB_725702	531.3134	152	0	1	152
LTFRB_725702	238.6807	74	0	1	74
LTFRB_725702	406.387	116	0	1	116
LTFRB_725702	585.8316	172	0	1	172
LTFRB_725702	277.9878	85	0	1	85
LTFRB_725702	308.4904	92	0	1	92
LTFRB_725702	591.1283	181	0	1	181
LTFRB_725702	851.909	254	12	1	254
LTFRB_725265	277.9878	86	0	1	86
LTFRB_725265	225.3576	76	0	1	76
LTFRB_725265	437.4893	125	0	1	125
LTFRB_725265	454.1282	162	0	1	162
LTFRB_725265	1160.34	328	0	1	328

LTFRB_725265	363.4098	101	0	1	101
LTFRB_725265	341.4821	105	0	1	105
LTFRB_725265	207.9318	69	0	1	69
LTFRB_725265	537.8793	171	0	1	171
LTFRB_725265	392.0559	123	0	1	123
LTFRB_725265	734.8273	220	12.25	1	220
LTFRB_725265	670.6983	188	13.75	1	188
LTFRB_725265	162.9245	53	14.25	1	53
LTFRB_725265	866.7911	261	16	1	261
LTFRB_725265	301.6067	93	16.75	1	93
LTFRB_725265	351.5786	97	17.5	1	97
LTFRB_725265	298.0931	89	18.25	1	89
LTFRB_725265	610.0216	191	19.5	1	191
LTFRB_725265	522.9731	147	20.5	1	147
LTFRB_725265	406.3812	129	21.5	1	129
LTFRB_725265	293.9111	76	22.25	1	76
LTFRB_725265	171.4215	46	22.5	1	46
LTFRB_725265	257.1133	80	23	1	80
LTFRB_725265	166.7927	52	23.5	1	52
LTFRB_725265	227.2457	63	24	1	63
LTFRB_725265	144.5536	46	24.25	1	46
LTFRB_725265	189.2457	49	24.75	1	49
LTFRB_725265	166.7927	52	25	1	52
LTFRB_725265	198.4926	55	25.5	1	55
LTFRB_725265	189.0317	57	26	1	57
LTFRB_725265	180.208	46	26.25	1	46
LTFRB_725265	177.9122	54	26.75	1	54
LTFRB_725265	189.0317	57	27.25	1	57
LTFRB_725265	227.2425	57	27.75	1	57
LTFRB_725265	177.9122	61	28	1	61
LTFRB_725265	207.9208	60	28.5	1	60
LTFRB_725265	198.4892	57	29	1	57
LTFRB_725265	177.9122	57	29.25	1	57
LTFRB_725265	207.9196	60	29.75	1	60
LTFRB_725265	198.4879	54	30.25	1	54
LTFRB_725265	133.4341	43	30.5	1	43
LTFRB_725265	207.9185	58	31	1	58
LTFRB_725265	198.4868	56	31.5	1	56
LTFRB_725265	217.5087	61	31.75	1	61
LTFRB_725265	177.9122	60	32.25	1	60
LTFRB_725265	247.0508	70	32.75	1	70

LTFRB_725265	146.941	36	33	1	36
LTFRB_725265	198.4847	59	33.5	1	59
LTFRB_725265	200.1512	63	34	1	63
LTFRB_725265	217.5064	59	34.5	1	59
LTFRB_725265	247.0491	75	35	1	75
LTFRB_725265	227.2359	66	35.5	1	66
LTFRB_725265	227.2355	63	36	1	63
LTFRB_725265	100.0756	36	36.25	1	36
LTFRB_725265	257.103	73	36.75	1	73
LTFRB_725265	266.8682	81	37.5	1	81
LTFRB_725265	298.0803	85	38	1	85
LTFRB_725265	255.7487	79	38.5	1	79
LTFRB_725265	267.2408	76	39.25	1	76
LTFRB_725265	255.7487	76	39.75	1	76
LTFRB_725265	533.5748	166	41	1	166
LTFRB_725265	233.5097	71	41.5	1	71
LTFRB_725265	232.8205	80	42	1	80
LTFRB_725265	437.2953	119	43	1	119
LTFRB_725265	524.6934	162	44.75	1	162
LTFRB_725265	796.693	244	46.25	1	244
LTFRB_725265	253.1676	69	47.5	1	69
TOTAL	24422.86		59.5		7265

Total_weight = 2442.86*0.3 + 59.5*0.4 + 7265*0.2 + 753.065*0.1 = **8878.9645**

	Distance(m)* 0.3	**Time (s)* 0.2**	**Fare (PhP)* 0.4**	**Walk Distance (m)* 0.1**	**Total Weight**
Trip 1	4198.488	836.8	13	37.65325	**5085.941**
Trip 2	4307.907	853.8	11.8	100.9576	**5274.258**
Trip 3	4868.856	970.2	12.5	75.3065	**5926.863**
Trip 4	6704.106	1321.6	21	75.3065	**8122.007**
Trip 5	732.858	1453	23.8	75.3065	**8878.965**

Table 11.4 Summary of computations

5. Computation tables for finding trips from UP Manila to UP Diliman where traffic is HEAVY along EDSA and user preference is least time

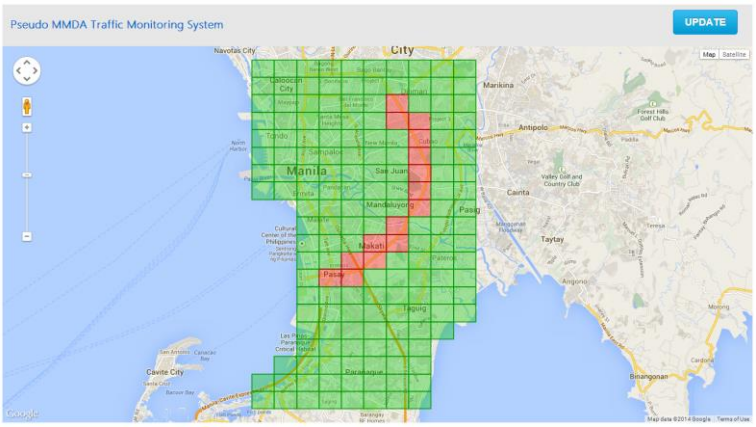


Figure 11.9 Heavy traffic along EDSA

Figure 11.10 Top 5 results

TRIP ID	DISTANCE	TIME	FARE	TRAFFIC SCORE	TIME*TRAFFIC SCORE
LTFRB_726297	217.5172	63	0	1	63
LTFRB_726297	247.0586	76	0	1	76
LTFRB_726297	298.0898	98	0	1	98
LTFRB_726297	565.1976	170	0	1	170
LTFRB_726297	489.2584	146	0	1	146
LTFRB_726297	405.0377	111	0	1	111
LTFRB_726297	222.3902	68	0	1	68
LTFRB_726297	247.0548	69	0	1	69
LTFRB_726297	189.0317	58	0	1	58
LTFRB_726297	426.0066	136	0	1	136
LTFRB_726297	162.9078	48	0	1	48
LTFRB_726297	272.2712	72	0	1	72
LTFRB_726297	272.27	73	0	1	73
LTFRB_726297	162.9065	56	0	1	56
LTFRB_726297	272.268	71	0	1	71
LTFRB_726297	180.203	60	0	1	60
LTFRB_726297	265.5994	71	0	1	71
LTFRB_726297	301.5684	90	12.5	1	90
LTFRB_726297	656.7224	209	14	1	209
LTFRB_726297	426.4031	134	14.75	1	134
LTFRB_726297	253.1981	60	15.25	1	60
LTFRB_726297	180.1993	60	15.75	1	60

LTFRB_726297	448.4626	137	16.75	1	137
LTFRB_726297	874.6502	247	19.75	1	247
LTFRB_726297	506.3753	154	20.75	1	154
LTFRB_726297	663.9475	212	22.25	1	212
LTFRB_726297	1139.317	338	24.75	1	338
LTFRB_726297	379.7677	109	25.5	1	109
LTFRB_726297	139.5948	50	25.75	1	50
LTFRB_726297	217.5003	59	26.25	1	59
LTFRB_726297	233.5097	71	26.75	1	71
LTFRB_726297	232.8205	80	27.25	1	80
LTFRB_726297	437.2953	119	28.25	1	119
LTFRB_726297	777.6151	231	30	1	231
LTFRB_726297	544.4624	175	31.75	1	175
LTFRB_726297	253.1676	69	32.5	1	69
TOTAL	13994.96		32.5		4184

Total_weight = 13994.96*0.3 + 32.5*0.4 + 4184*0.2 + 376.5325*0.1 = 5085.94125

TRIP ID	DISTANCE	TIME	FARE	TRAFFIC SCORE	TIME*TRAFFIC SCORE
LTFRB_724769	217.5172	64	0	1	64
LTFRB_724769	247.0586	76	0	1	76
LTFRB_724769	298.0898	98	0	1	98
LTFRB_724769	237.0979	67	0	1	67
LTFRB_724769	329.4182	103	0	1	103
LTFRB_724769	180.2093	43	0	1	43
LTFRB_724769	189.0317	59	0	1	59
LTFRB_724769	189.2444	64	0	1	64
LTFRB_724769	405.0377	110	0	1	110
LTFRB_724769	222.3902	69	0	1	69
LTFRB_724769	247.0548	69	0	1	69
LTFRB_724769	189.0317	58	0	1	58
LTFRB_724769	189.0317	59	0	1	59
LTFRB_724769	334.8385	100	0	1	100
LTFRB_724769	162.9078	48	0	1	48
LTFRB_724769	272.2712	72	8.25	1	72
LTFRB_724769	272.27	73	8.75	1	73
LTFRB_724689	180.203	60	0	1	60
LTFRB_724689	265.5994	71	0	1	71

LTFRB_724689	301.5684	89	0	1	89
LTFRB_724689	447.4273	159	0	1	159
LTFRB_724689	324.5061	101	0	1	101
LTFRB_724689	431.6728	106	0	1	106
LTFRB_724689	323.5513	102	0	1	102
LTFRB_724689	337.943	93	0	1	93
LTFRB_724689	225.2946	75	0	1	75
LTFRB_724689	232.8506	88	0	1	88
LTFRB_724689	345.1796	104	0	1	104
LTFRB_724689	332.0327	86	0	1	86
LTFRB_724689	337.9372	93	8	1	93
LTFRB_724689	286.4219	101	8.5	1	101
LTFRB_724689	237.0891	219	8.75	3	657
LTFRB_724689	225.2874	340	9.25	5	1700
LTFRB_724689	329.5951	405	9.75	5	2025
LTFRB_724689	337.9292	485	10.5	5	2425
LTFRB_724689	337.928	500	11.25	5	2500
LTFRB_725784	266.8682	405	0	5	2025
LTFRB_725784	298.0803	425	0	5	2125
LTFRB_725784	255.7487	395	0	5	1975
LTFRB_725784	267.2408	228	0	3	684
LTFRB_725784	255.7487	76	0	1	76
LTFRB_725784	391.9448	115	0	1	115
LTFRB_725784	585.7998	175	0	1	175
LTFRB_725784	247.4933	70	0	1	70
LTFRB_725784	279.185	93	0	1	93
LTFRB_725784	253.1711	69	0	1	69
LTFRB_725784	265.5645	86	0	1	86
LTFRB_725784	279.1818	89	0	1	89
LTFRB_725784	253.1676	69	12	1	69
TOTAL	13921.71		32		19518

Total_weight = 13921.71*0.3 + 32*0.4 + 19518*0.2 + 1386.1084*0.1 = **8231.31084**

TRIP ID	DISTANCE	TIME	FARE	TRAFFIC SCORE	TIME*TRAFFIC SCORE
LTFRB_724970	217.5172	63	0	1	63
LTFRB_724970	247.0586	76	0	1	76
LTFRB_724970	298.0898	98	0	1	98
LTFRB_724970	237.0979	68	0	1	68
LTFRB_724970	287.7473	84	0	1	84
LTFRB_724970	44.47804	19	0	1	19
LTFRB_724970	180.2093	44	0	1	44
LTFRB_724970	189.0317	58	0	1	58
LTFRB_724970	189.2444	64	0	1	64
LTFRB_724970	405.0377	111	0	1	111
LTFRB_724970	222.3902	68	0	1	68
LTFRB_724970	247.0548	69	0	1	69
LTFRB_724970	189.0317	58	0	1	58
LTFRB_724970	189.0317	59	0	1	59
LTFRB_724970	334.8385	100	0	1	100
LTFRB_724970	162.9078	49	0	1	49
LTFRB_724970	272.2712	72	0	1	72
LTFRB_724970	272.27	72	8.25	1	72
LTFRB_724970	162.9065	56	8.5	1	56
LTFRB_724970	272.268	72	8.75	1	72
LTFRB_724970	180.203	59	9	1	59
LTFRB_724970	265.5994	71	9.5	1	71
LTFRB_724970	301.5684	90	10	1	90
LTFRB_724970	447.4273	159	10.5	1	159
LTFRB_724970	324.5061	100	11	1	100
LTFRB_724970	431.6728	106	11.5	1	106
LTFRB_724970	323.5513	103	12	1	103
LTFRB_724970	337.943	92	12.5	1	92
LTFRB_724970	225.2946	75	12.75	1	75
LTFRB_724970	232.8506	89	13.25	1	89
LTFRB_724970	345.1796	104	13.75	1	104
LTFRB_724970	332.0327	86	14	1	86
LTFRB_724970	337.9372	92	14.5	1	92
LTFRB_724970	286.4219	101	15	1	101
LTFRB_724970	237.0891	222	15.25	3	666
LTFRB_724970	225.2874	340	15.75	5	1700
LTFRB_724970	329.5951	405	16	5	2025
LTFRB_724970	337.9292	480	16.75	5	2400
LTFRB_724970	337.928	505	17.5	5	2525

LTFRB_725709	100.0756	180	0	5	900
LTFRB_725709	257.103	365	0	5	1825
LTFRB_725709	266.8682	405	0	5	2025
LTFRB_725709	298.0803	425	0	5	2125
LTFRB_725709	255.7487	395	0	5	1975
LTFRB_725709	267.2408	228	0	3	684
LTFRB_725709	255.7487	76	0	1	76
LTFRB_725709	391.9448	115	0	1	115
LTFRB_725709	1063.391	322	0	1	322
LTFRB_725709	796.693	244	0	1	244
LTFRB_725709	253.1676	68	12	1	68
TOTAL	14666.56		29.5		22362

Total_weight = 14666.56*0.3 + 29.5*0.4 + 22362*0.2 + 901.5025*0.1 = 8974.31825

TRIP ID	DISTANCE	TIME	FARE	TRAFFIC SCORE	TIME*TRAFFIC SCORE
LTFRB_725202	217.5172	63	0	1	63
LTFRB_725202	247.0586	76	0	1	76
LTFRB_725202	298.0898	98	0	1	98
LTFRB_725202	237.0979	67	0	1	67
LTFRB_725202	287.7473	85	0	1	85
LTFRB_725202	44.47804	19	0	1	19
LTFRB_725202	180.2093	43	0	1	43
LTFRB_725202	189.0317	59	0	1	59
LTFRB_725202	189.2444	64	0	1	64
LTFRB_725202	405.0377	110	0	1	110
LTFRB_725202	222.3902	69	0	1	69
LTFRB_725202	247.0548	69	0	1	69
LTFRB_725202	189.0317	58	0	1	58
LTFRB_725202	189.0317	59	0	1	59
LTFRB_725202	334.8385	100	0	1	100
LTFRB_725202	162.9078	48	0	1	48
LTFRB_725202	272.2712	72	0	1	72
LTFRB_725202	272.27	73	8.25	1	73
LTFRB_725202	162.9065	56	8.5	1	56
LTFRB_725202	272.268	71	8.75	1	71
LTFRB_725202	180.203	60	9	1	60
LTFRB_725202	265.5994	71	9.5	1	71

LTFRB_725202	301.5684	90	10	1	90
LTFRB_725202	447.4273	158	10.5	1	158
LTFRB_725202	324.5061	101	11	1	101
LTFRB_725202	431.6728	106	11.5	1	106
LTFRB_725202	323.5513	102	12	1	102
LTFRB_725202	337.943	93	12.5	1	93
LTFRB_725202	225.2946	75	12.75	1	75
LTFRB_725202	232.8506	89	13.25	1	89
LTFRB_725202	345.1796	104	13.75	1	104
LTFRB_725202	332.0327	85	14	1	85
LTFRB_725202	337.9372	93	14.5	1	93
LTFRB_725202	431.6621	393	15.25	3	1179
LTFRB_725202	325.8279	550	15.75	5	2750
LTFRB_725202	430.9441	550	16.25	5	2750
LTFRB_725202	112.6452	165	16.5	5	825
LTFRB_725202	237.3183	219	16.75	3	657
LTFRB_725202	374.0589	333	17.5	3	999
LTFRB_725202	219.7365	425	17.75	5	2125
LTFRB_726268	200.1512	315	0	5	1575
LTFRB_726268	217.5064	295	0	5	1475
LTFRB_726268	247.0491	375	0	5	1875
LTFRB_726268	227.2359	330	0	5	1650
LTFRB_726268	227.2355	315	0	5	1575
LTFRB_726268	100.0756	180	0	5	900
LTFRB_726268	257.103	365	0	5	1825
LTFRB_726268	266.8682	405	0	5	2025
LTFRB_726268	298.0803	425	0	5	2125
LTFRB_726268	255.7487	395	0	5	1975
LTFRB_726268	267.2408	228	0	3	684
LTFRB_726268	255.7487	76	0	1	76
LTFRB_726268	533.5748	166	0	1	166
LTFRB_726268	233.5097	71	0	1	71
LTFRB_726268	232.8205	80	0	1	80
LTFRB_726268	437.2953	119	0	1	119
LTFRB_726268	524.6934	162	13.25	1	162
LTFRB_726268	796.693	244	13.75	1	244
LTFRB_726268	253.1676	69	14.25	1	69
TOTAL	16670.24		32		32542

Total_weight = 16670.24*0.3 + 32*0.4 + 32542*0.2 + 753.065*0.1 = **11597.5785**

TRIP ID	DIRECTION	TIME	FARE	TRAFFIC SCORE	TIME*TRAFFIC SCORE
LTFRB_725735	247.0586	76	0	1	76
LTFRB_725735	298.0898	98	0	1	98
LTFRB_725735	237.0979	68	0	1	68
LTFRB_725735	329.4182	102	0	1	102
LTFRB_725735	180.2093	44	0	1	44
LTFRB_725735	217.5142	65	0	1	65
LTFRB_725735	265.6151	80	0	1	80
LTFRB_725735	405.0377	111	0	1	111
LTFRB_725735	222.3902	69	0	1	69
LTFRB_725735	247.0548	68	0	1	68
LTFRB_725735	189.0317	58	0	1	58
LTFRB_725735	189.0317	59	0	1	59
LTFRB_725735	334.8385	100	0	1	100
LTFRB_725735	162.9078	49	0	1	49
LTFRB_725735	272.2712	72	0	1	72
LTFRB_725735	272.27	72	8	1	72
LTFRB_725735	162.9065	56	8.25	1	56
LTFRB_725735	272.268	72	8.75	1	72
LTFRB_725735	180.203	60	9	1	60
LTFRB_725735	265.5994	70	9.25	1	70
LTFRB_725735	301.5684	90	9.75	1	90
LTFRB_725735	447.4273	159	10.5	1	159
LTFRB_725735	324.5061	101	10.75	1	101
LTFRB_725735	431.6728	106	11.5	1	106
LTFRB_725735	323.5513	102	12	1	102
LTFRB_725735	337.943	93	12.25	1	93
LTFRB_725735	225.2946	75	12.75	1	75
LTFRB_725735	232.8506	88	13	1	88
LTFRB_725735	345.1796	104	13.5	1	104
LTFRB_725735	332.0327	86	14	1	86
LTFRB_725735	337.9372	93	14.5	1	93
LTFRB_725735	431.6621	393	15	3	1179
LTFRB_725735	325.8279	550	15.5	5	2750
LTFRB_725735	430.9441	550	16	5	2750
LTFRB_725735	349.2712	315	16.5	3	945
LTFRB_725735	374.0589	333	17.5	3	999
LTFRB_725735	219.7365	425	18	5	2125
LTFRB_726238	200.1512	310	0	5	1550
LTFRB_726238	217.5064	295	0	5	1475

LTFRB_726238	247.0491	375	0	5	1875
LTFRB_726238	227.2359	330	0	5	1650
LTFRB_726238	227.2355	315	0	5	1575
LTFRB_726238	100.0756	185	0	5	925
LTFRB_726238	257.103	365	0	5	1825
LTFRB_726238	266.8682	405	0	5	2025
LTFRB_726238	298.0803	420	0	5	2100
LTFRB_726238	255.7487	395	0	5	1975
LTFRB_726238	267.2408	228	0	3	684
LTFRB_726238	255.7487	77	0	1	77
LTFRB_726238	533.5748	166	0	1	166
LTFRB_726238	233.5097	71	0	1	71
LTFRB_726238	232.8205	79	0	1	79
LTFRB_726238	437.2953	119	0	1	119
LTFRB_726238	524.6934	163	13.25	1	163
LTFRB_726238	796.693	244	14	1	244
LTFRB_726238	253.1676	68	14.25	1	68
TOTAL	16554.08		32.25		31940

Total_weight = 16554.08*0.3 + 32.25*0.4 + 31940*0.2 + 1079.68*0.1 = **11475.092**

	Distance(m) * 0.3	**Time (s) * 0.2**	**Fare (PhP) * 0.4**	**Walk Distance (m) * 0.1**	**Total Weight**
Trip 1	4198.488	836.8	13	37.65325	**5085.941**
Trip 2	4176.513	3903.6	12.8	138.6108	**8231.311**
Trip 3	4399.968	4472.4	11.8	90.15025	**8974.318**
Trip 4	5001.072	6508.4	12.8	75.3065	**11597.58**
Trip 5	4966.224	6388	12.9	107.968	**11475.092**

Table 11.5 Summary of computations

6. Computation tables for finding trips from UP Manila to UP Diliman where traffic is HEAVY along Quezon Blvd. and user preference is least time

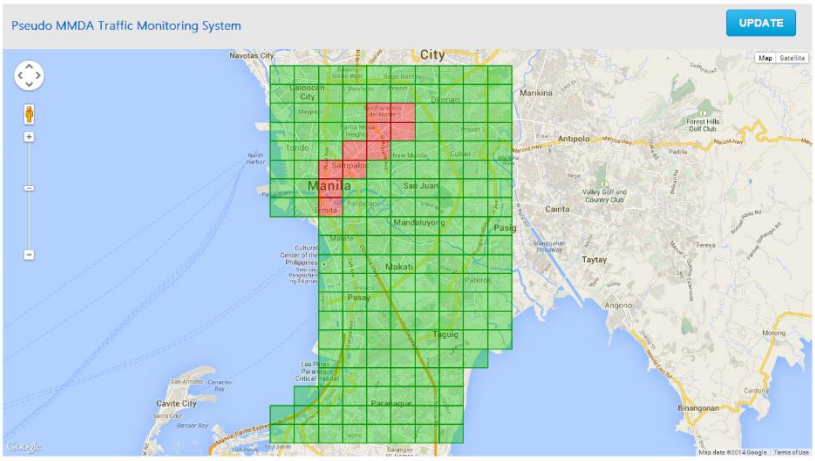


Figure 11.11 Heavy traffic along Quezon Blvd

Figure 11.12 Top 5 results

TRIP ID	DIRECTION	TIME	FARE	TRAFFIC SCORE	TIME*TRAFFIC SCORE
LTFRB_725139	4465.706	1364	12.5	1	1364
LTFRB_725361	309.5085	98	0	1	98
LTFRB_725361	301.6167	98	0	1	98
LTFRB_725361	265.6516	69	0	1	69
LTFRB_725361	171.4313	58	0	1	58
LTFRB_725361	272.3147	82	0	1	82
LTFRB_725361	398.4721	106	0	1	106
LTFRB_725361	162.9245	53	0	1	53
LTFRB_725361	866.7911	261	0	1	261
LTFRB_725361	301.6067	93	0	1	93
LTFRB_725361	351.5786	98	0	1	98
LTFRB_725361	298.0931	88	0	1	88
LTFRB_725361	610.0216	191	0	1	191
LTFRB_725361	522.9731	148	0	1	148
LTFRB_725361	406.3812	128	12.5	1	128
LTFRB_725361	293.9111	76	13.25	1	76
LTFRB_725361	171.4215	46	13.5	1	46
LTFRB_725361	257.1133	80	14	1	80
LTFRB_725361	382.3977	115	15	1	115
LTFRB_725361	189.2457	49	15.75	1	49
LTFRB_725361	166.7927	52	16	1	52

LTFRB_725361	198.4926	55	16.5	1	55
LTFRB_725361	189.0317	57	17	1	57
LTFRB_725361	180.208	46	17.25	1	46
LTFRB_725361	177.9122	54	17.75	1	54
LTFRB_725361	189.0317	58	18	1	58
LTFRB_725361	227.2425	56	18.5	1	56
LTFRB_725361	177.9122	61	19	1	61
LTFRB_725361	207.9208	60	19.5	1	60
LTFRB_725361	198.4892	57	20	1	57
LTFRB_725361	177.9122	57	20.25	1	57
LTFRB_725361	207.9196	60	20.75	1	60
LTFRB_725361	198.4879	54	21.25	1	54
LTFRB_725361	133.4341	43	21.5	1	43
LTFRB_725361	207.9185	58	22	1	58
LTFRB_725361	198.4868	56	22.25	1	56
LTFRB_725361	217.5087	61	22.75	1	61
LTFRB_725361	177.9122	60	23.25	1	60
LTFRB_725361	247.0508	70	23.75	1	70
LTFRB_725361	146.941	37	24	1	37
LTFRB_725361	198.4847	58	24.5	1	58
LTFRB_725361	200.1512	63	25	1	63
LTFRB_725361	217.5064	59	25.5	1	59
LTFRB_725361	247.0491	75	26	1	75
LTFRB_725361	227.2359	66	26.5	1	66
LTFRB_725361	227.2355	63	27	1	63
LTFRB_725361	100.0756	36	27.25	1	36
LTFRB_725361	257.103	73	27.75	1	73
LTFRB_725361	266.8682	81	28.5	1	81
LTFRB_725361	298.0803	85	29	1	85
LTFRB_725361	255.7487	79	29.5	1	79
LTFRB_725361	267.2408	76	30.25	1	76
LTFRB_725361	255.7487	76	30.75	1	76
LTFRB_725361	533.5748	166	32	1	166
LTFRB_725361	233.5097	71	32.5	1	71
LTFRB_725361	232.8205	80	33	1	80
LTFRB_725361	437.2953	119	34	1	119
LTFRB_725361	777.6151	231	35.75	1	231
LTFRB_725361	544.4624	175	37.5	1	175
LTFRB_725361	253.1676	69	38.25	1	69
TOTAL	21001.29		50.75		6260

Total_weight = 21001.29*0.3 + 50.75*0.4 + 6260*0.2 + 873.819*0.1 = **7660.0689**

TRIP ID	DIRECTION	TIME	FARE	TRAFFIC SCORE	TIME*TRAFFIC SCORE
LTFRB_725951	624.9991	180	0	1	180
LTFRB_725951	373.5684	107	0	1	107
LTFRB_725951	396.9949	115	0	1	115
LTFRB_725951	382.3997	123	0	1	123
LTFRB_725951	360.4322	96	0	1	96
LTFRB_725951	298.0932	96	0	1	96
LTFRB_725951	277.4708	85	0	1	85
LTFRB_725951	277.4712	85	0	1	85
LTFRB_725951	317.5833	83	0	1	83
LTFRB_725951	318.9349	101	0	1	101
LTFRB_725951	287.7538	90	0	1	90
LTFRB_725951	425.4215	124	0	1	124
LTFRB_725951	586.2019	165	12.5	1	165
LTFRB_725951	267.2604	83	13.25	1	83
LTFRB_725951	247.0674	77	13.75	1	77
LTFRB_726268	385.873	103	0	1	103
LTFRB_726268	309.5085	98	0	1	98
LTFRB_726268	565.6818	168	0	1	168
LTFRB_726268	171.4313	57	0	1	57
LTFRB_726268	670.6983	188	0	1	188
LTFRB_726268	162.9245	54	0	1	54
LTFRB_726268	866.7911	261	0	1	261
LTFRB_726268	301.6067	93	0	1	93
LTFRB_726268	351.5786	97	0	1	97
LTFRB_726268	298.0931	88	0	1	88
LTFRB_726268	610.0216	191	0	1	191
LTFRB_726268	522.9731	148	12.5	1	148
LTFRB_726268	406.3812	129	13.25	1	129
LTFRB_726268	293.9111	75	14	1	75
LTFRB_726268	171.4215	47	14.5	1	47
LTFRB_726268	637.8603	193	15.75	1	193
LTFRB_726268	144.5536	46	16	1	46
LTFRB_726268	189.2457	49	16.5	1	49
LTFRB_726268	166.7927	53	17	1	53
LTFRB_726268	198.4926	55	17.25	1	55
LTFRB_726268	189.0317	57	17.75	1	57
LTFRB_726268	180.208	46	18.25	1	46
LTFRB_726268	177.9122	54	18.5	1	54
LTFRB_726268	189.0317	57	19	1	57
LTFRB_726268	393.0782	117	19.75	1	117

LTFRB_726268	207.9208	59	20.25	1	59
LTFRB_726268	198.4892	58	20.75	1	58
LTFRB_726268	177.9122	57	21	1	57
LTFRB_726268	207.9196	60	21.5	1	60
LTFRB_726268	198.4879	54	22	1	54
LTFRB_726268	133.4341	43	22.25	1	43
LTFRB_726268	207.9185	58	22.75	1	58
LTFRB_726268	198.4868	56	23.25	1	56
LTFRB_726268	217.5087	61	23.75	1	61
LTFRB_726268	177.9122	60	24	1	60
LTFRB_726268	247.0508	70	24.5	1	70
LTFRB_726268	146.941	36	25	1	36
LTFRB_726268	198.4847	59	25.25	1	59
LTFRB_726268	200.1512	63	25.75	1	63
LTFRB_726268	217.5064	59	26.25	1	59
LTFRB_726268	247.0491	75	26.75	1	75
LTFRB_726268	227.2359	66	27.25	1	66
LTFRB_726268	227.2355	63	27.75	1	63
LTFRB_726268	100.0756	36	28	1	36
LTFRB_726268	257.103	73	28.5	1	73
LTFRB_726268	266.8682	81	29.25	1	81
LTFRB_726268	298.0803	85	29.75	1	85
LTFRB_726268	255.7487	79	30.5	1	79
LTFRB_726268	267.2408	76	31	1	76
LTFRB_726268	255.7487	76	31.5	1	76
LTFRB_726268	533.5748	166	32.75	1	166
LTFRB_726268	233.5097	71	33.25	1	71
LTFRB_726268	232.8205	80	33.75	1	80
LTFRB_726268	437.2953	119	34.75	1	119
LTFRB_726268	524.6934	162	36.75	1	162
LTFRB_726268	796.693	244	37.75	1	244
LTFRB_726268	253.1676	69	38.75	1	69
TOTAL	22347.02		52.5		6608

Total_weight = 22347.02*0.3 + 52.5*0.4 + 6608*0.2 + 753.065*0.1 = **8122.0125**

TRIP ID	DIRECTION	TIME	FARE	TRAFFIC SCORE	TIME*TRAFFIC SCORE
LTFRB_726309	624.8177	190	0	1	190
LTFRB_726309	585.8316	172	0	1	172
LTFRB_726309	277.9878	85	0	1	85
LTFRB_726309	455.8999	139	0	1	139
LTFRB_726309	437.4893	125	0	1	125
LTFRB_726309	225.3576	75	0	1	75
LTFRB_726309	277.9878	87	12	1	87
LTFRB_726167	225.3576	76	0	1	76
LTFRB_726167	437.4893	125	0	1	125
LTFRB_726167	454.1282	162	0	1	162
LTFRB_726167	1160.34	328	0	1	328
LTFRB_726167	363.4098	101	0	1	101
LTFRB_726167	341.4821	105	0	1	105
LTFRB_726167	207.9318	69	0	1	69
LTFRB_726167	537.8793	171	0	1	171
LTFRB_726167	392.0559	123	0	1	123
LTFRB_726167	734.8273	220	0	1	220
LTFRB_726167	670.6983	188	13.25	1	188
LTFRB_726167	162.9245	53	13.5	1	53
LTFRB_726167	866.7911	261	15.5	1	261
LTFRB_726167	301.6067	93	16	1	93
LTFRB_726167	351.5786	97	16.75	1	97
LTFRB_726167	298.0931	89	17.5	1	89
LTFRB_726167	610.0216	191	18.75	1	191
LTFRB_726167	522.9731	147	20	1	147
LTFRB_726167	406.3812	129	21	1	129
LTFRB_726167	293.9111	76	21.5	1	76
LTFRB_726167	171.4215	46	22	1	46
LTFRB_726167	257.1133	80	22.5	1	80
LTFRB_726167	166.7927	52	22.75	1	52
LTFRB_726167	227.2457	63	23.25	1	63
LTFRB_726167	144.5536	46	23.75	1	46
LTFRB_726167	189.2457	49	24	1	49
LTFRB_726167	166.7927	52	24.5	1	52
LTFRB_726167	198.4926	55	25	1	55
LTFRB_726167	189.0317	57	25.25	1	57
LTFRB_726167	180.208	46	25.75	1	46
LTFRB_726167	177.9122	54	26	1	54
LTFRB_726167	189.0317	57	26.5	1	57
LTFRB_726167	227.2425	57	27	1	57

LTFRB_726167	177.9122	61	27.5	1	61
LTFRB_726167	207.9208	60	27.75	1	60
LTFRB_726167	198.4892	57	28.25	1	57
LTFRB_726167	177.9122	57	28.75	1	57
LTFRB_726167	207.9196	60	29.25	1	60
LTFRB_726167	198.4879	54	29.5	1	54
LTFRB_726167	133.4341	43	30	1	43
LTFRB_726167	207.9185	58	30.25	1	58
LTFRB_726167	198.4868	56	30.75	1	56
LTFRB_726167	217.5087	61	31.25	1	61
LTFRB_726167	177.9122	60	31.75	1	60
LTFRB_726167	247.0508	70	32.25	1	70
LTFRB_726167	146.941	36	32.5	1	36
LTFRB_726167	198.4847	59	33	1	59
LTFRB_726167	200.1512	63	33.5	1	63
LTFRB_726167	217.5064	59	33.75	1	59
LTFRB_726167	247.0491	75	34.5	1	75
LTFRB_726167	227.2359	66	35	1	66
LTFRB_726167	227.2355	63	35.5	1	63
LTFRB_726167	100.0756	36	35.75	1	36
LTFRB_726167	257.103	73	36.25	1	73
LTFRB_726167	266.8682	81	36.75	1	81
LTFRB_726167	298.0803	85	37.5	1	85
LTFRB_726167	255.7487	79	38	1	79
LTFRB_726167	267.2408	76	38.5	1	76
LTFRB_726167	255.7487	76	39.25	1	76
LTFRB_726167	533.5748	166	40.25	1	166
LTFRB_726167	233.5097	71	40.75	1	71
LTFRB_726167	232.8205	80	41.25	1	80
LTFRB_726167	437.2953	119	42.25	1	119
LTFRB_726167	524.6934	162	43.5	1	162
LTFRB_726167	796.693	244	45.75	1	244
LTFRB_726167	253.1676	69	46.5	1	69
TOTAL	23238.51		58.5		6926

Total_weight = 23238.51*0.3 + 58.5*0.4 + 6926*0.2 + 753.065*0.1 = **8455.4595**

TRIP ID	DIRECTION	TIME	FARE	TRAFFIC SCORE	TIME*TRAFFIC SCORE
LTFRB_725702	531.3134	152	0	1	152
LTFRB_725702	238.6807	74	0	1	74
LTFRB_725702	406.387	116	0	1	116
LTFRB_725702	585.8316	172	0	1	172
LTFRB_725702	277.9878	85	0	1	85
LTFRB_725702	308.4904	92	0	1	92
LTFRB_725702	591.1283	181	0	1	181
LTFRB_725702	851.909	254	12	1	254
LTFRB_725265	277.9878	86	0	1	86
LTFRB_725265	225.3576	76	0	1	76
LTFRB_725265	437.4893	125	0	1	125
LTFRB_725265	454.1282	162	0	1	162
LTFRB_725265	1160.34	328	0	1	328
LTFRB_725265	363.4098	101	0	1	101
LTFRB_725265	341.4821	105	0	1	105
LTFRB_725265	207.9318	69	0	1	69
LTFRB_725265	537.8793	171	0	1	171
LTFRB_725265	392.0559	123	0	1	123
LTFRB_725265	734.8273	220	12.25	1	220
LTFRB_725265	670.6983	188	13.75	1	188
LTFRB_725265	162.9245	53	14.25	1	53
LTFRB_725265	866.7911	261	16	1	261
LTFRB_725265	301.6067	93	16.75	1	93
LTFRB_725265	351.5786	97	17.5	1	97
LTFRB_725265	298.0931	89	18.25	1	89
LTFRB_725265	610.0216	191	19.5	1	191
LTFRB_725265	522.9731	147	20.5	1	147
LTFRB_725265	406.3812	129	21.5	1	129
LTFRB_725265	293.9111	76	22.25	1	76
LTFRB_725265	171.4215	46	22.5	1	46
LTFRB_725265	257.1133	80	23	1	80
LTFRB_725265	166.7927	52	23.5	1	52
LTFRB_725265	227.2457	63	24	1	63
LTFRB_725265	144.5536	46	24.25	1	46
LTFRB_725265	189.2457	49	24.75	1	49
LTFRB_725265	166.7927	52	25	1	52
LTFRB_725265	198.4926	55	25.5	1	55
LTFRB_725265	189.0317	57	26	1	57
LTFRB_725265	180.208	46	26.25	1	46
LTFRB_725265	177.9122	54	26.75	1	54

LTFRB_725265	189.0317	57	27.25	1	57
LTFRB_725265	227.2425	57	27.75	1	57
LTFRB_725265	177.9122	61	28	1	61
LTFRB_725265	207.9208	60	28.5	1	60
LTFRB_725265	198.4892	57	29	1	57
LTFRB_725265	177.9122	57	29.25	1	57
LTFRB_725265	207.9196	60	29.75	1	60
LTFRB_725265	198.4879	54	30.25	1	54
LTFRB_725265	133.4341	43	30.5	1	43
LTFRB_725265	207.9185	58	31	1	58
LTFRB_725265	198.4868	56	31.5	1	56
LTFRB_725265	217.5087	61	31.75	1	61
LTFRB_725265	177.9122	60	32.25	1	60
LTFRB_725265	247.0508	70	32.75	1	70
LTFRB_725265	146.941	36	33	1	36
LTFRB_725265	198.4847	59	33.5	1	59
LTFRB_725265	200.1512	63	34	1	63
LTFRB_725265	217.5064	59	34.5	1	59
LTFRB_725265	247.0491	75	35	1	75
LTFRB_725265	227.2359	66	35.5	1	66
LTFRB_725265	227.2355	63	36	1	63
LTFRB_725265	100.0756	36	36.25	1	36
LTFRB_725265	257.103	73	36.75	1	73
LTFRB_725265	266.8682	81	37.5	1	81
LTFRB_725265	298.0803	85	38	1	85
LTFRB_725265	255.7487	79	38.5	1	79
LTFRB_725265	267.2408	76	39.25	1	76
LTFRB_725265	255.7487	76	39.75	1	76
LTFRB_725265	533.5748	166	41	1	166
LTFRB_725265	233.5097	71	41.5	1	71
LTFRB_725265	232.8205	80	42	1	80
LTFRB_725265	437.2953	119	43	1	119
LTFRB_725265	524.6934	162	44	1	162
LTFRB_725265	796.693	244	46.25	1	244
LTFRB_725265	253.1676	69	47	1	69
TOTAL	24422.86		59.5		7265

Total_weight = 24422.86*0.3 + 59.5*0.4 + 7265*0.2 + 753.065*0.1 = **8878.9645**

TRIP ID	DIRECTION	TIME	FARE	TRAFFIC SCORE	TIME*TRAFFIC SCORE
LTFRB_726297	217.5172	63	0	1	63
LTFRB_726297	247.0586	228	0	3	684
LTFRB_726297	298.0898	490	0	5	2450
LTFRB_726297	565.1976	850	0	5	4250
LTFRB_726297	489.2584	730	0	5	3650
LTFRB_726297	405.0377	555	0	5	2775
LTFRB_726297	222.3902	340	0	5	1700
LTFRB_726297	247.0548	345	0	5	1725
LTFRB_726297	189.0317	290	0	5	1450
LTFRB_726297	426.0066	680	0	5	3400
LTFRB_726297	162.9078	240	0	5	1200
LTFRB_726297	272.2712	360	0	5	1800
LTFRB_726297	272.27	365	0	5	1825
LTFRB_726297	162.9065	280	0	5	1400
LTFRB_726297	272.268	355	0	5	1775
LTFRB_726297	180.203	300	0	5	1500
LTFRB_726297	265.5994	355	0	5	1775
LTFRB_726297	301.5684	450	12.5	5	2250
LTFRB_726297	656.7224	1045	14	5	5225
LTFRB_726297	426.4031	670	14.75	5	3350
LTFRB_726297	253.1981	300	15.25	5	1500
LTFRB_726297	180.1993	300	15.75	5	1500
LTFRB_726297	448.4626	685	16.75	5	3425
LTFRB_726297	433.3159	670	17.75	5	3350
LTFRB_726297	874.6502	1235	19.75	5	6175
LTFRB_726297	506.3753	770	20.75	5	3850
LTFRB_726297	663.9475	1060	22.25	5	5300
LTFRB_726297	1139.317	1014	24.75	3	3042
LTFRB_726297	379.7677	109	25.5	1	109
LTFRB_726297	139.5948	50	25.75	1	50
LTFRB_726297	217.5003	59	26.25	1	59
LTFRB_726297	233.5097	71	26.75	1	71
LTFRB_726297	232.8205	80	27.25	1	80
LTFRB_726297	437.2953	119	28.25	1	119
LTFRB_726297	777.6151	231	30	1	231
LTFRB_726297	544.4624	175	31.75	1	175
LTFRB_726297	253.1676	69	32.5	1	69
TOTAL	13994.96		32.5		73352

Total_weight = 13994.96*0.3 + 32.5*0.4 + 73352*0.2 + 376.5325*0.1 = **18919.54125**

	Distance(m)*0.3	Time (s)*0.2	Fare (PhP)*0.4	Walk Distance (m)*0.1	Total Weight
Trip 1	6300.387	1252	20.3	87.3819	**7660.069**
Trip 2	6704.106	1321.6	21	75.3065	**8122.013**
Trip 3	6971.553	1385.2	23.4	75.3065	**8455.46**
Trip 4	7326.858	1453	23.8	75.3065	**8878.965**
Trip 5	4198.488	14670.4	13	37.65325	**18919.54**

Table 11.6 Summary of computations

7. Computation tables for finding trips from UP Manila to UP Diliman where traffic is LIGHT on all areas and user preference is least distance

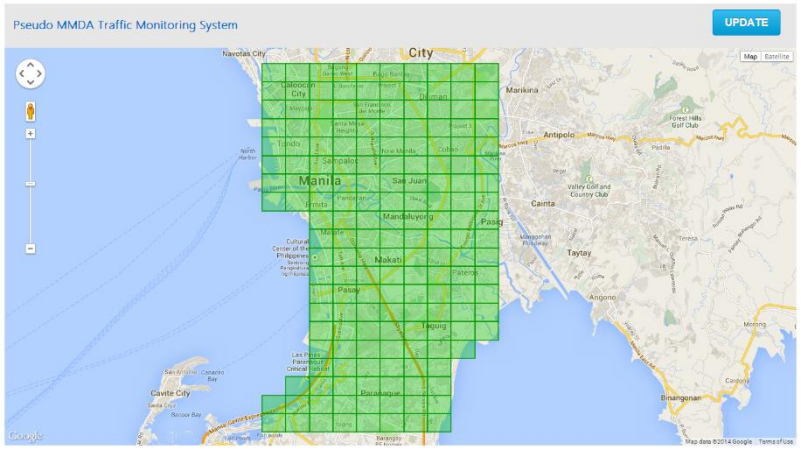


Figure 11.13 Light traffic on all areas

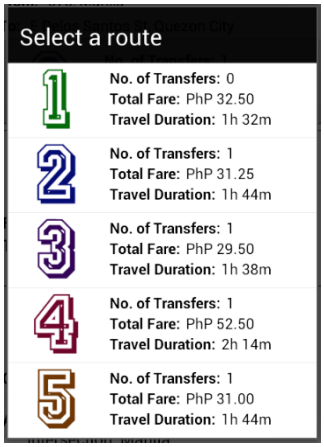


Figure 11.14 Top 5 results

TRIP ID	DIRECTION	TIME	FARE	TRAFFIC SCORE	TIME*TRAFFIC SCORE
LTFRB_726297	217.5172	63	0	1	63
LTFRB_726297	247.0586	76	0	1	76
LTFRB_726297	298.0898	98	0	1	98
LTFRB_726297	565.1976	170	0	1	170
LTFRB_726297	489.2584	146	0	1	146
LTFRB_726297	405.0377	111	0	1	111
LTFRB_726297	222.3902	68	0	1	68
LTFRB_726297	247.0548	69	0	1	69
LTFRB_726297	189.0317	58	0	1	58
LTFRB_726297	426.0066	136	0	1	136

LTFRB_726297	162.9078	48	0	1	48
LTFRB_726297	272.2712	72	0	1	72
LTFRB_726297	272.27	73	0	1	73
LTFRB_726297	162.9065	56	0	1	56
LTFRB_726297	272.268	71	0	1	71
LTFRB_726297	180.203	60	0	1	60
LTFRB_726297	265.5994	71	0	1	71
LTFRB_726297	301.5684	90	12.5	1	90
LTFRB_726297	656.7224	209	14	1	209
LTFRB_726297	426.4031	134	14.75	1	134
LTFRB_726297	253.1981	60	15.25	1	60
LTFRB_726297	180.1993	60	15.75	1	60
LTFRB_726297	448.4626	137	16.75	1	137
LTFRB_726297	433.3159	134	17.75	1	134
LTFRB_726297	874.6502	247	19.75	1	247
LTFRB_726297	506.3753	154	20.75	1	154
LTFRB_726297	663.9475	212	22.25	1	212
LTFRB_726297	1139.317	338	24.75	1	338
LTFRB_726297	379.7677	109	25.5	1	109
LTFRB_726297	139.5948	50	25.75	1	50
LTFRB_726297	217.5003	59	26.25	1	59
LTFRB_726297	233.5097	71	26.75	1	71
LTFRB_726297	232.8205	80	27.25	1	80
LTFRB_726297	437.2953	119	28.25	1	119
LTFRB_726297	777.6151	231	31.25	1	231
LTFRB_726297	544.4624	175	31.75	1	175
LTFRB_726297	253.1676	69	32.5	1	69
TOTAL	13994.96		32.5		4184

Total_weight = 13994.96*0.2 + 32.5*0.4 + 4184*0.3 + 376.5325*0.1 = **4104.84525**

TRIP ID	DIRECTION	TIME	FARE	TRAFFIC SCORE	TIME*TRAFFIC SCORE
LTFRB_725202	217.5172	63	0	1	63
LTFRB_725202	247.0586	76	0	1	76
LTFRB_725202	298.0898	98	0	1	98
LTFRB_725202	237.0979	67	0	1	67
LTFRB_725202	287.7473	85	0	1	85
LTFRB_725202	44.47804	19	0	1	19
LTFRB_725202	180.2093	43	0	1	43
LTFRB_725202	189.0317	59	0	1	59
LTFRB_725202	189.2444	64	0	1	64
LTFRB_725202	405.0377	110	0	1	110
LTFRB_725202	222.3902	69	0	1	69
LTFRB_725202	247.0548	69	0	1	69
LTFRB_725202	189.0317	58	0	1	58
LTFRB_725202	189.0317	59	0	1	59
LTFRB_725202	334.8385	100	0	1	100
LTFRB_725202	162.9078	48	0	1	48
LTFRB_725202	272.2712	72	0	1	72
LTFRB_725202	272.27	73	8.25	1	73
LTFRB_725202	162.9065	56	8.5	1	56
LTFRB_725202	272.268	71	8.75	1	71
LTFRB_725202	180.203	60	9	1	60
LTFRB_725202	265.5994	71	9.5	1	71
LTFRB_725202	301.5684	90	10	1	90
LTFRB_725202	447.4273	158	10.5	1	158
LTFRB_725202	324.5061	101	11	1	101
LTFRB_725202	431.6728	106	11.5	1	106
LTFRB_725202	323.5513	102	12	1	102
LTFRB_725202	337.943	93	12.5	1	93
LTFRB_725202	225.2946	75	12.75	1	75
LTFRB_725202	232.8506	89	13.25	1	89
LTFRB_725202	345.1796	104	13.75	1	104
LTFRB_725202	332.0327	85	14	1	85
LTFRB_725202	337.9372	93	14.5	1	93
LTFRB_725202	431.6621	131	15.25	1	131
LTFRB_725202	325.8279	110	15.75	1	110
LTFRB_725202	430.9441	110	16.25	1	110
LTFRB_725202	112.6452	33	16.5	1	33
LTFRB_725202	237.3183	73	16.75	1	73
LTFRB_725202	374.0589	111	17.75	1	111
LTFRB_725202	219.7365	85	18	1	85
LTFRB_725283	200.1512	62	0	1	62
LTFRB_725283	217.5064	59	0	1	59

LTFRB_725283	247.0491	75	0	1	75
LTFRB_725283	227.2359	66	0	1	66
LTFRB_725283	227.2355	63	0	1	63
LTFRB_725283	100.0756	37	0	1	37
LTFRB_725283	257.103	73	0	1	73
LTFRB_725283	266.8682	81	0	1	81
LTFRB_725283	298.0803	84	0	1	84
LTFRB_725283	255.7487	79	0	1	79
LTFRB_725283	267.2408	76	0	1	76
LTFRB_725283	255.7487	77	0	1	77
LTFRB_725283	391.9448	114	0	1	114
LTFRB_725283	457.6057	136	0	1	136
LTFRB_725283	146.9294	55	0	1	55
LTFRB_725283	524.6934	162	12.25	1	162
LTFRB_725283	796.693	244	12.75	1	244
LTFRB_725283	253.1676	69	13.25	1	69
TOTAL	16229.52		31.25		4851

Total_weight = 16229.52*0.2 + 31.5*0.4 + 4851*0.3 + 753.065*0.1 = **4789.0065**

TRIP ID	DIRECTION	TIME	FARE	TRAFFIC SCORE	TIME*TRAFFIC SCORE
LTFRB_724970	217.5172	63	0	1	63
LTFRB_724970	247.0586	76	0	1	76
LTFRB_724970	298.0898	98	0	1	98
LTFRB_724970	237.0979	68	0	1	68
LTFRB_724970	287.7473	84	0	1	84
LTFRB_724970	44.47804	19	0	1	19
LTFRB_724970	180.2093	44	0	1	44
LTFRB_724970	189.0317	58	0	1	58
LTFRB_724970	189.2444	64	0	1	64
LTFRB_724970	405.0377	111	0	1	111
LTFRB_724970	222.3902	68	0	1	68
LTFRB_724970	247.0548	69	0	1	69
LTFRB_724970	189.0317	58	0	1	58
LTFRB_724970	189.0317	59	0	1	59
LTFRB_724970	334.8385	100	0	1	100
LTFRB_724970	162.9078	49	0	1	49
LTFRB_724970	272.2712	72	0	1	72

LTFRB_724970	272.27	72	8.25	1	72
LTFRB_724970	162.9065	56	8.5	1	56
LTFRB_724970	272.268	72	8.75	1	72
LTFRB_724970	180.203	59	9	1	59
LTFRB_724970	265.5994	71	9.5	1	71
LTFRB_724970	301.5684	90	10	1	90
LTFRB_724970	447.4273	159	10.5	1	159
LTFRB_724970	324.5061	100	11	1	100
LTFRB_724970	431.6728	106	11.5	1	106
LTFRB_724970	323.5513	103	12	1	103
LTFRB_724970	337.943	92	12.5	1	92
LTFRB_724970	225.2946	75	12.75	1	75
LTFRB_724970	232.8506	89	13.25	1	89
LTFRB_724970	345.1796	104	13.75	1	104
LTFRB_724970	332.0327	86	14	1	86
LTFRB_724970	337.9372	92	14.5	1	92
LTFRB_724970	286.4219	101	15	1	101
LTFRB_724970	237.0891	74	15.25	1	74
LTFRB_724970	225.2874	68	15.75	1	68
LTFRB_724970	329.5951	81	16	1	81
LTFRB_724970	337.9292	96	16.5	1	96
LTFRB_724970	337.928	101	17.5	1	101
LTFRB_725618	266.8682	81	0	1	81
LTFRB_725618	298.0803	85	0	1	85
LTFRB_725618	255.7487	79	0	1	79
LTFRB_725618	267.2408	76	0	1	76
LTFRB_725618	255.7487	76	0	1	76
LTFRB_725618	391.9448	114	0	1	114
LTFRB_725618	585.7998	176	0	1	176
LTFRB_725618	247.4933	70	0	1	70
LTFRB_725618	279.185	93	0	1	93
LTFRB_725618	253.1711	69	0	1	69
LTFRB_725618	265.5645	86	0	1	86
LTFRB_725618	279.1818	89	0	1	89
LTFRB_725618	253.1676	68	12	1	68
TOTAL	14359.69		29.5		4269

Total_weight = 14359.69*0.2 + 29.5*0.4 + 4269*0.3 + 1009.5759*0.1 = **4265.39559**

TRIP ID	DIRECTION	TIME	FARE	TRAFFIC SCORE	TIME*TRAFFIC SCORE
LTFRB_725951	624.9991	180	0	1	180
LTFRB_725951	373.5684	107	0	1	107
LTFRB_725951	396.9949	115	0	1	115
LTFRB_725951	382.3997	123	0	1	123
LTFRB_725951	360.4322	96	0	1	96
LTFRB_725951	298.0932	96	0	1	96
LTFRB_725951	277.4708	85	0	1	85
LTFRB_725951	277.4712	85	0	1	85
LTFRB_725951	317.5833	83	0	1	83
LTFRB_725951	318.9349	101	0	1	101
LTFRB_725951	287.7538	90	0	1	90
LTFRB_725951	425.4215	124	0	1	124
LTFRB_725951	586.2019	165	12.5	1	165
LTFRB_725951	267.2604	83	12.75	1	83
LTFRB_725951	247.0674	77	13.75	1	77
LTFRB_726268	385.873	103	0	1	103
LTFRB_726268	309.5085	98	0	1	98
LTFRB_726268	565.6818	168	0	1	168
LTFRB_726268	171.4313	57	0	1	57
LTFRB_726268	670.6983	188	0	1	188
LTFRB_726268	162.9245	54	0	1	54
LTFRB_726268	866.7911	261	0	1	261
LTFRB_726268	301.6067	93	0	1	93
LTFRB_726268	351.5786	97	0	1	97
LTFRB_726268	298.0931	88	0	1	88
LTFRB_726268	610.0216	191	0	1	191
LTFRB_726268	522.9731	148	12.5	1	148
LTFRB_726268	406.3812	129	13.25	1	129
LTFRB_726268	293.9111	75	14	1	75
LTFRB_726268	171.4215	47	14.5	1	47
LTFRB_726268	637.8603	193	15.75	1	193
LTFRB_726268	144.5536	46	16	1	46
LTFRB_726268	189.2457	49	16.5	1	49
LTFRB_726268	166.7927	53	17	1	53
LTFRB_726268	198.4926	55	17.25	1	55
LTFRB_726268	189.0317	57	17.75	1	57
LTFRB_726268	180.208	46	18.25	1	46
LTFRB_726268	177.9122	54	18.5	1	54
LTFRB_726268	189.0317	57	19	1	57
LTFRB_726268	393.0782	117	19.75	1	117

LTFRB_726268	207.9208	59	20.25	1	59
LTFRB_726268	198.4892	58	20.75	1	58
LTFRB_726268	177.9122	57	21	1	57
LTFRB_726268	207.9196	60	21.5	1	60
LTFRB_726268	198.4879	54	22	1	54
LTFRB_726268	133.4341	43	22.25	1	43
LTFRB_726268	207.9185	58	22.75	1	58
LTFRB_726268	198.4868	56	23.25	1	56
LTFRB_726268	217.5087	61	23.75	1	61
LTFRB_726268	177.9122	60	24	1	60
LTFRB_726268	247.0508	70	24.5	1	70
LTFRB_726268	146.941	36	25	1	36
LTFRB_726268	198.4847	59	25.25	1	59
LTFRB_726268	200.1512	63	25.75	1	63
LTFRB_726268	217.5064	59	26.25	1	59
LTFRB_726268	247.0491	75	26.75	1	75
LTFRB_726268	227.2359	66	27.25	1	66
LTFRB_726268	227.2355	63	27.75	1	63
LTFRB_726268	100.0756	36	28	1	36
LTFRB_726268	257.103	73	28.5	1	73
LTFRB_726268	266.8682	81	29.25	1	81
LTFRB_726268	298.0803	85	29.75	1	85
LTFRB_726268	255.7487	79	30.5	1	79
LTFRB_726268	267.2408	76	31	1	76
LTFRB_726268	255.7487	76	31.5	1	76
LTFRB_726268	533.5748	166	32.75	1	166
LTFRB_726268	233.5097	71	33.25	1	71
LTFRB_726268	232.8205	80	33.75	1	80
LTFRB_726268	437.2953	119	34.75	1	119
LTFRB_726268	524.6934	162	36.5	1	162
LTFRB_726268	796.693	244	37.75	1	244
LTFRB_726268	253.1676	69	38.75	1	69
TOTAL	22347.02		52.5		6608

Total_weight = 22347.02*0.2 + 52.5*0.4 + 6608*0.3 + 753.065*0.1 = **6548.1105**

TRIP ID	DIRECTION	TIME	FARE	TRAFFIC SCORE	TIME*TRAFFIC SCORE
LTFRB_725735	247.0586	76	0	1	76
LTFRB_725735	298.0898	98	0	1	98
LTFRB_725735	237.0979	68	0	1	68
LTFRB_725735	329.4182	102	0	1	102
LTFRB_725735	180.2093	44	0	1	44
LTFRB_725735	217.5142	65	0	1	65
LTFRB_725735	265.6151	80	0	1	80
LTFRB_725735	405.0377	111	0	1	111
LTFRB_725735	222.3902	69	0	1	69
LTFRB_725735	247.0548	68	0	1	68
LTFRB_725735	189.0317	58	0	1	58
LTFRB_725735	189.0317	59	0	1	59
LTFRB_725735	334.8385	100	0	1	100
LTFRB_725735	162.9078	49	0	1	49
LTFRB_725735	272.2712	72	0	1	72
LTFRB_725735	272.27	72	8	1	72
LTFRB_725735	162.9065	56	8.25	1	56
LTFRB_725735	272.268	72	8.75	1	72
LTFRB_725735	180.203	60	9	1	60
LTFRB_725735	265.5994	70	9.25	1	70
LTFRB_725735	301.5684	90	9.75	1	90
LTFRB_725735	447.4273	159	10.5	1	159
LTFRB_725735	324.5061	101	10.75	1	101
LTFRB_725735	431.6728	106	11.5	1	106
LTFRB_725735	323.5513	102	12	1	102
LTFRB_725735	337.943	93	12.25	1	93
LTFRB_725735	225.2946	75	12.75	1	75
LTFRB_725735	232.8506	88	13	1	88
LTFRB_725735	345.1796	104	13.5	1	104
LTFRB_725735	332.0327	86	14	1	86
LTFRB_725735	337.9372	93	14.5	1	93
LTFRB_725735	431.6621	131	15	1	131
LTFRB_725735	325.8279	110	15.5	1	110
LTFRB_725735	430.9441	110	16	1	110
LTFRB_725735	349.2712	105	16.5	1	105
LTFRB_725735	374.0589	111	17	1	111
LTFRB_725735	219.7365	85	17.75	1	85
LTFRB_726045	200.1512	63	0	1	63
LTFRB_726045	217.5064	59	0	1	59
LTFRB_726045	247.0491	75	0	1	75

LTFRB_726045	227.2359	66	0	1	66
LTFRB_726045	227.2355	63	0	1	63
LTFRB_726045	100.0756	36	0	1	36
LTFRB_726045	257.103	73	0	1	73
LTFRB_726045	266.8682	81	0	1	81
LTFRB_726045	298.0803	85	0	1	85
LTFRB_726045	255.7487	79	0	1	79
LTFRB_726045	267.2408	76	0	1	76
LTFRB_726045	255.7487	76	0	1	76
LTFRB_726045	391.9448	114	0	1	114
LTFRB_726045	1063.391	323	12.25	1	323
LTFRB_726045	796.693	244	12.75	1	244
LTFRB_726045	253.1676	68	13.25	1	68
TOTAL	16047.52		31		4779

Total_weight = 16047.52*0.2 + 31*0.4 + 4779*0.3 + 1079.68*0.1 = **4763.572**

	Distance(m)*0.3	**Time (s)*0.2**	**Fare (PhP)*0.4**	**Walk Distance(m)*0.1**	**Total Weight**
Trip 1	2798.992	1255.2	13	37.65325	**4104.845**
Trip 2	3245.904	1455.3	12.6	75.3065	**4789.0065**
Trip 3	2871.938	1280.7	11.8	100.9576	**4265.396**
Trip 4	4469.404	1982.4	21	75.3065	**6548.111**
Trip 5	3209.504	1433.7	12.4	107.968	**4763.572**

Table 11.7 Summary of computations

8. Computation tables for finding trips from UP Manila to UP Diliman where traffic is HEAVY along EDSA and user preference is least distance

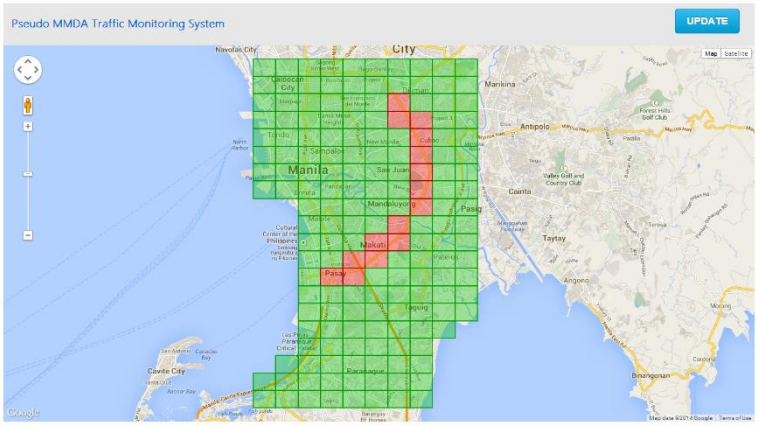


Figure 11.15 Heavy traffic along EDSA

Figure 11.16 Top 5 results

TRIP ID	DIRECTION	TIME	FARE	TRAFFIC SCORE	TIME*TRAFFIC SCORE
LTFRB_726297	217.5172	63	0	1	63
LTFRB_726297	247.0586	76	0	1	76
LTFRB_726297	298.0898	98	0	1	98
LTFRB_726297	565.1976	170	0	1	170
LTFRB_726297	489.2584	146	0	1	146
LTFRB_726297	405.0377	111	0	1	111
LTFRB_726297	222.3902	68	0	1	68
LTFRB_726297	247.0548	69	0	1	69
LTFRB_726297	189.0317	58	0	1	58
LTFRB_726297	426.0066	136	0	1	136
LTFRB_726297	162.9078	48	0	1	48
LTFRB_726297	272.2712	72	0	1	72
LTFRB_726297	272.27	73	0	1	73
LTFRB_726297	162.9065	56	0	1	56
LTFRB_726297	272.268	71	0	1	71
LTFRB_726297	180.203	60	0	1	60
LTFRB_726297	265.5994	71	0	1	71
LTFRB_726297	301.5684	90	12.5	1	90
LTFRB_726297	656.7224	209	14	1	209
LTFRB_726297	426.4031	134	14.75	1	134

LTFRB_726297	253.1981	60	15.25	1	60
LTFRB_726297	180.1993	60	15.75	1	60
LTFRB_726297	448.4626	137	16.75	1	137
LTFRB_726297	433.3159	134	17.75	1	134
LTFRB_726297	874.6502	247	19.75	1	247
LTFRB_726297	506.3753	154	20.75	1	154
LTFRB_726297	663.9475	212	22.25	1	212
LTFRB_726297	1139.317	338	24.75	1	338
LTFRB_726297	379.7677	109	25.5	1	109
LTFRB_726297	139.5948	50	25.75	1	50
LTFRB_726297	217.5003	59	26.25	1	59
LTFRB_726297	233.5097	71	26.75	1	71
LTFRB_726297	232.8205	80	27.25	1	80
LTFRB_726297	437.2953	119	28.25	1	119
LTFRB_726297	777.6151	231	30.75	1	231
LTFRB_726297	544.4624	175	32	1	175
LTFRB_726297	253.1676	69	32.5	1	69
TOTAL	13994.96		32.5		4184

Total_weight = 13994.96*0.2 + 32.5*0.4 + 4184*0.3 + 376.5235*0.1 = **4104.84525**

TRIP ID	DIRECTION	TIME	FARE	TRAFFIC SCORE	TIME*TRAFFIC SCORE
LTFRB_724970	217.5172	63	0	1	63
LTFRB_724970	247.0586	76	0	1	76
LTFRB_724970	298.0898	98	0	1	98
LTFRB_724970	237.0979	68	0	1	68
LTFRB_724970	287.7473	84	0	1	84
LTFRB_724970	44.47804	19	0	1	19
LTFRB_724970	180.2093	44	0	1	44
LTFRB_724970	189.0317	58	0	1	58
LTFRB_724970	189.2444	64	0	1	64
LTFRB_724970	405.0377	111	0	1	111
LTFRB_724970	222.3902	68	0	1	68
LTFRB_724970	247.0548	69	0	1	69
LTFRB_724970	189.0317	58	0	1	58
LTFRB_724970	189.0317	59	0	1	59
LTFRB_724970	334.8385	100	0	1	100
LTFRB_724970	162.9078	49	0	1	49

LTFRB_724970	272.2712	72	0	1	72
LTFRB_724970	272.27	72	8.25	1	72
LTFRB_724970	162.9065	56	8.5	1	56
LTFRB_724970	272.268	72	8.75	1	72
LTFRB_724970	180.203	59	9	1	59
LTFRB_724970	265.5994	71	9.5	1	71
LTFRB_724970	301.5684	90	10	1	90
LTFRB_724970	447.4273	159	10.5	1	159
LTFRB_724970	324.5061	100	11	1	100
LTFRB_724970	431.6728	106	11.5	1	106
LTFRB_724970	323.5513	103	12	1	103
LTFRB_724970	337.943	92	12.5	1	92
LTFRB_724970	225.2946	75	12.75	1	75
LTFRB_724970	232.8506	89	13.25	1	89
LTFRB_724970	345.1796	104	13.75	1	104
LTFRB_724970	332.0327	86	14	1	86
LTFRB_724970	337.9372	92	14.5	1	92
LTFRB_724970	286.4219	101	15	1	101
LTFRB_724970	237.0891	222	15.25	3	666
LTFRB_724970	225.2874	340	15.75	5	1700
LTFRB_724970	329.5951	405	16	5	2025
LTFRB_724970	337.9292	480	16.5	5	2400
LTFRB_724970	337.928	505	17.5	5	2525
LTFRB_725709	100.0756	180	0	5	900
LTFRB_725709	257.103	365	0	5	1825
LTFRB_725709	266.8682	405	0	5	2025
LTFRB_725709	298.0803	425	0	5	2125
LTFRB_725709	255.7487	395	0	5	1975
LTFRB_725709	267.2408	228	0	3	684
LTFRB_725709	255.7487	76	0	1	76
LTFRB_725709	391.9448	115	0	1	115
LTFRB_725709	1063.391	322	0	1	322
LTFRB_725709	796.693	244	0	1	244
LTFRB_725709	253.1676	68	12	1	68
TOTAL	14666.56		29.5		22362

Total_weight = 14666.56*0.2 + 29.5*0.4 + 22362*0.3 + 901.5028*0.1 = **9743.86228**

TRIP ID	DISTANCE	TIME	FARE	TRAFFIC SCORE	TIME*TRAFFIC SCORE
LTFRB_725202	217.5172	63	0	1	63
LTFRB_725202	247.0586	76	0	1	76
LTFRB_725202	298.0898	98	0	1	98
LTFRB_725202	237.0979	67	0	1	67
LTFRB_725202	287.7473	85	0	1	85
LTFRB_725202	44.47804	19	0	1	19
LTFRB_725202	180.2093	43	0	1	43
LTFRB_725202	189.0317	59	0	1	59
LTFRB_725202	189.2444	64	0	1	64
LTFRB_725202	405.0377	110	0	1	110
LTFRB_725202	222.3902	69	0	1	69
LTFRB_725202	247.0548	69	0	1	69
LTFRB_725202	189.0317	58	0	1	58
LTFRB_725202	189.0317	59	0	1	59
LTFRB_725202	334.8385	100	0	1	100
LTFRB_725202	162.9078	48	0	1	48
LTFRB_725202	272.2712	72	0	1	72
LTFRB_725202	272.27	73	8.25	1	73
LTFRB_725202	162.9065	56	8.5	1	56
LTFRB_725202	272.268	71	8.75	1	71
LTFRB_725202	180.203	60	9	1	60
LTFRB_725202	265.5994	71	9.5	1	71
LTFRB_725202	301.5684	90	10	1	90
LTFRB_725202	447.4273	158	10.5	1	158
LTFRB_725202	324.5061	101	11	1	101
LTFRB_725202	431.6728	106	11.5	1	106
LTFRB_725202	323.5513	102	12	1	102
LTFRB_725202	337.943	93	12.5	1	93
LTFRB_725202	225.2946	75	12.75	1	75
LTFRB_725202	232.8506	89	13.25	1	89
LTFRB_725202	345.1796	104	13.75	1	104
LTFRB_725202	332.0327	85	14	1	85
LTFRB_725202	337.9372	93	14.5	1	93
LTFRB_725202	431.6621	393	15.25	3	1179
LTFRB_725202	325.8279	550	15.75	5	2750
LTFRB_725202	430.9441	550	16.25	5	2750
LTFRB_725202	112.6452	165	16.5	5	825
LTFRB_725202	237.3183	219	16.75	3	657
LTFRB_725202	374.0589	333	17.25	3	999

LTFRB_725202	219.7365	425	18	5	2125
LTFRB_726268	200.1512	315	0	5	1575
LTFRB_726268	217.5064	295	0	5	1475
LTFRB_726268	247.0491	375	0	5	1875
LTFRB_726268	227.2359	330	0	5	1650
LTFRB_726268	227.2355	315	0	5	1575
LTFRB_726268	100.0756	180	0	5	900
LTFRB_726268	257.103	365	0	5	1825
LTFRB_726268	266.8682	405	0	5	2025
LTFRB_726268	298.0803	425	0	5	2125
LTFRB_726268	255.7487	395	0	5	1975
LTFRB_726268	267.2408	228	0	3	684
LTFRB_726268	255.7487	76	0	1	76
LTFRB_726268	533.5748	166	0	1	166
LTFRB_726268	233.5097	71	0	1	71
LTFRB_726268	232.8205	80	0	1	80
LTFRB_726268	437.2953	119	0	1	119
LTFRB_726268	524.6934	162	13.25	1	162
LTFRB_726268	796.693	244	13.75	1	244
LTFRB_726268	253.1676	69	14.25	1	69
TOTAL	16670.24		32.25		32542

Total_weight = 16670.24*0.2 + 32.25*0.4 + 32542*0.3 + 753.065*0.1 = **13184.8545**

TRIP ID	DISTANCE	TIME	FARE	TRAFFIC SCORE	TIME*TRAFFIC SCORE
LRTA_882210	1123.541	170	12	1	170
LRTA_882210	736.7608	90	12	1	90
LRTA_882210	711.2513	90	12	1	90
LRTA_882210	659.2771	80	12	1	80
LRTA_882210	604.1009	80	14	1	80
LRTA_882210	662.954	80	14	1	80
LRTA_882210	907.645	125	14	1	125
LRTA_882210	572.6185	80	15	1	80
LRTA_882210	964.7179	130	15	1	130
LRTA_882210	1099.652	150	15	1	150
LRTA_882210	2167.74	330	15	1	330
LRTA_882210	1885.016	300	15	1	300
LTFRB_724862	441.7904	131	0	1	131
LTFRB_724862	337.8913	127	0	1	127

LTFRB_724862	497.0808	153	0	1	153
LTFRB_724862	37.22968	0	0	1	0
LTFRB_724862	253.718	65	0	1	65
LTFRB_724862	257.0998	83	0	1	83
LTFRB_724862	198.478	58	0	1	58
LTFRB_724862	207.9101	63	0	1	63
LTFRB_724862	198.4788	186	0	3	558
LTFRB_724862	207.9109	310	0	5	1550
LTFRB_724862	198.4796	295	0	5	1475
LTFRB_724862	198.48	295	0	5	1475
LTFRB_724862	189.2321	270	0	5	1350
LTFRB_724862	171.4044	245	0	5	1225
LTFRB_724862	166.7927	290	0	5	1450
LTFRB_724862	180.1967	255	0	5	1275
LTFRB_724862	146.9365	170	12	5	850
LTFRB_725361	266.8682	405	0	5	2025
LTFRB_725361	298.0803	425	0	5	2125
LTFRB_725361	255.7487	395	0	5	1975
LTFRB_725361	267.2408	228	0	3	684
LTFRB_725361	255.7487	76	0	1	76
LTFRB_725361	533.5748	166	0	1	166
LTFRB_725361	233.5097	71	0	1	71
LTFRB_725361	232.8205	80	0	1	80
LTFRB_725361	437.2953	119	0	1	119
LTFRB_725361	777.6151	231	0	1	231
LTFRB_725361	544.4624	175	0	1	175
LTFRB_725361	253.1676	69	12	1	69
TOTAL	20340.52		29		21389

Total_weight = 20340.52*0.2 + 29*0.4 + 21389*0.3 + 1287.51*0.1 = **10625.155**

TRIP ID	DISTANCE	TIME	FARE	TRAFFIC SCORE	TIME*TRAFFIC SCORE
LTFRB_725735	247.0586	76	0	1	76
LTFRB_725735	298.0898	98	0	1	98
LTFRB_725735	237.0979	68	0	1	68
LTFRB_725735	329.4182	102	0	1	102
LTFRB_725735	180.2093	44	0	1	44
LTFRB_725735	217.5142	65	0	1	65
LTFRB_725735	265.6151	80	0	1	80
LTFRB_725735	405.0377	111	0	1	111
LTFRB_725735	222.3902	69	0	1	69
LTFRB_725735	247.0548	68	0	1	68
LTFRB_725735	189.0317	58	0	1	58
LTFRB_725735	189.0317	59	0	1	59
LTFRB_725735	334.8385	100	0	1	100
LTFRB_725735	162.9078	49	0	1	49
LTFRB_725735	272.2712	72	0	1	72
LTFRB_725735	272.27	72	8	1	72
LTFRB_725735	162.9065	56	8.25	1	56
LTFRB_725735	272.268	72	8.75	1	72
LTFRB_725735	180.203	60	9	1	60
LTFRB_725735	265.5994	70	9.25	1	70
LTFRB_725735	301.5684	90	9.75	1	90
LTFRB_725735	447.4273	159	10.5	1	159
LTFRB_725735	324.5061	101	10.75	1	101
LTFRB_725735	431.6728	106	11.5	1	106
LTFRB_725735	323.5513	102	12	1	102
LTFRB_725735	337.943	93	12.25	1	93
LTFRB_725735	225.2946	75	12.75	1	75
LTFRB_725735	232.8506	88	13	1	88
LTFRB_725735	345.1796	104	13.5	1	104
LTFRB_725735	332.0327	86	14	1	86
LTFRB_725735	337.9372	93	14.5	1	93
LTFRB_725735	431.6621	393	15	3	1179
LTFRB_725735	325.8279	550	15.5	5	2750
LTFRB_725735	430.9441	550	16	5	2750
LTFRB_725735	349.2712	315	16.5	3	945
LTFRB_725735	374.0589	333	17	3	999
LTFRB_725735	219.7365	425	17.75	5	2125
LTFRB_726238	200.1512	310	0	5	1550
LTFRB_726238	217.5064	295	0	5	1475

LTFRB_726238	247.0491	375	0	5	1875
LTFRB_726238	227.2359	330	0	5	1650
LTFRB_726238	227.2355	315	0	5	1575
LTFRB_726238	100.0756	185	0	5	925
LTFRB_726238	257.103	365	0	5	1825
LTFRB_726238	266.8682	405	0	5	2025
LTFRB_726238	298.0803	420	0	5	2100
LTFRB_726238	255.7487	395	0	5	1975
LTFRB_726238	267.2408	228	0	3	684
LTFRB_726238	255.7487	77	0	1	77
LTFRB_726238	533.5748	166	0	1	166
LTFRB_726238	233.5097	71	0	1	71
LTFRB_726238	232.8205	79	0	1	79
LTFRB_726238	437.2953	119	0	1	119
LTFRB_726238	524.6934	163	13.25	1	163
LTFRB_726238	796.693	244	13.75	1	244
LTFRB_726238	253.1676	68	14.25	1	68
TOTAL	16554.08		32		31940

Total_weight = 16554.08*0.2 + 32*0.4 + 31940*0.3 + 1079.68*0.1 = **13013.584**

	Distance(m) * 0.3	**Time (s) * 0.2**	**Fare (PhP) * 0.4**	**Walk Distance (m) * 0.1**	**Total Weight**
Trip 1	2798.992	1255.2	13	37.65235	**4104.845**
Trip 2	2933.312	6708.6	11.8	90.15028	**9743.862**
Trip 3	3334.048	9762.6	12.9	75.3065	**13184.85**
Trip 4	4068.104	6416.7	11.6	128.751	**10625.16**
Trip 5	3310.816	9582	12.8	107.968	**13013.58**

Table 11.8 Summary of computations

9. Computation tables for finding trips from UP Manila to UP Diliman where traffic is HEAVY along Quezon Blvd. and user preference is least distance

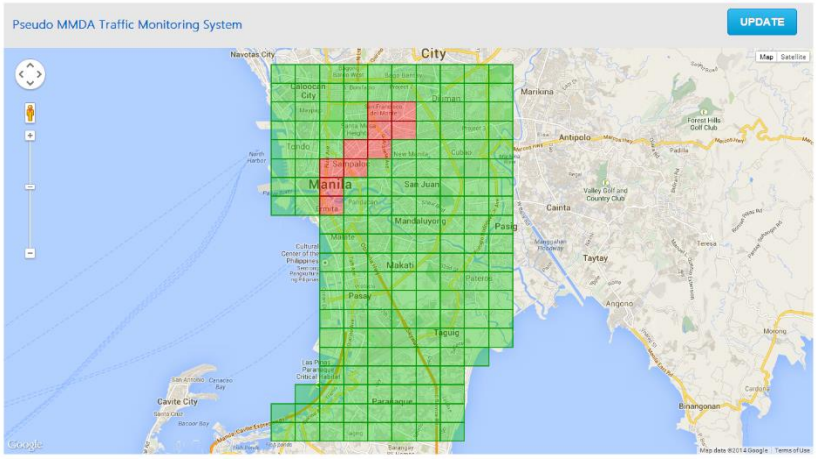


Figure 11.17 Heavy traffic along Quezon Blvd

Figure 11.18 Top 5 Results

TRIP ID	DISTANCE	TIME	FARE	TRAFFIC SCORE	TIME*TRAFFIC SCORE
LTFRB_726297	217.5172	63	0	1	63
LTFRB_726297	247.0586	228	0	3	684
LTFRB_726297	298.0898	490	0	5	2450
LTFRB_726297	565.1976	850	0	5	4250
LTFRB_726297	489.2584	730	0	5	3650
LTFRB_726297	405.0377	555	0	5	2775
LTFRB_726297	222.3902	340	0	5	1700
LTFRB_726297	247.0548	345	0	5	1725
LTFRB_726297	189.0317	290	0	5	1450
LTFRB_726297	426.0066	680	0	5	3400
LTFRB_726297	162.9078	240	0	5	1200
LTFRB_726297	272.2712	360	0	5	1800
LTFRB_726297	272.27	365	0	5	1825
LTFRB_726297	162.9065	280	0	5	1400
LTFRB_726297	272.268	355	0	5	1775
LTFRB_726297	180.203	300	0	5	1500
LTFRB_726297	265.5994	355	0	5	1775
LTFRB_726297	301.5684	450	12.5	5	2250
LTFRB_726297	656.7224	1045	14	5	5225

LTFRB_726297	426.4031	670	14.75	5	3350
LTFRB_726297	253.1981	300	15.25	5	1500
LTFRB_726297	180.1993	300	15.75	5	1500
LTFRB_726297	448.4626	685	16.75	5	3425
LTFRB_726297	433.3159	670	17.75	5	3350
LTFRB_726297	874.6502	1235	19.75	5	6175
LTFRB_726297	506.3753	770	20.75	5	3850
LTFRB_726297	663.9475	1060	22.25	5	5300
LTFRB_726297	1139.317	1014	24.75	3	3042
LTFRB_726297	379.7677	109	25.5	1	109
LTFRB_726297	139.5948	50	25.75	1	50
LTFRB_726297	217.5003	59	26.25	1	59
LTFRB_726297	233.5097	71	26.75	1	71
LTFRB_726297	232.8205	80	27.25	1	80
LTFRB_726297	437.2953	119	28.25	1	119
LTFRB_726297	777.6151	231	30.75	1	231
LTFRB_726297	544.4624	175	32	1	175
LTFRB_726297	253.1676	69	32.5	1	69
TOTAL	13994.96		32.5		73352

Total_weight = 13994.96*0.2 + 32.5*0.4 + 73352*0.3 + 376.5325*0.1 = **24855.24525**

TRIP ID	DISTANCE	TIME	FARE	TRAFFIC SCORE	TIME*TRAFFIC SCORE
LTFRB_725951	624.9991	180	0	1	180
LTFRB_725951	373.5684	107	0	1	107
LTFRB_725951	396.9949	115	0	1	115
LTFRB_725951	382.3997	123	0	1	123
LTFRB_725951	360.4322	96	0	1	96
LTFRB_725951	298.0932	96	0	1	96
LTFRB_725951	277.4708	85	0	1	85
LTFRB_725951	277.4712	85	0	1	85
LTFRB_725951	317.5833	83	0	1	83
LTFRB_725951	318.9349	101	0	1	101
LTFRB_725951	287.7538	90	0	1	90
LTFRB_725951	425.4215	124	0	1	124
LTFRB_725951	586.2019	165	12.5	1	165
LTFRB_725951	267.2604	83	13	1	83

LTFRB_725951	247.0674	77	13.75	1	77
LTFRB_726268	385.873	103	0	1	103
LTFRB_726268	309.5085	98	0	1	98
LTFRB_726268	565.6818	168	0	1	168
LTFRB_726268	171.4313	57	0	1	57
LTFRB_726268	670.6983	188	0	1	188
LTFRB_726268	162.9245	54	0	1	54
LTFRB_726268	866.7911	261	0	1	261
LTFRB_726268	301.6067	93	0	1	93
LTFRB_726268	351.5786	97	0	1	97
LTFRB_726268	298.0931	88	0	1	88
LTFRB_726268	610.0216	191	0	1	191
LTFRB_726268	522.9731	148	12.5	1	148
LTFRB_726268	406.3812	129	13.25	1	129
LTFRB_726268	293.9111	75	14	1	75
LTFRB_726268	171.4215	47	14.5	1	47
LTFRB_726268	637.8603	193	15.75	1	193
LTFRB_726268	144.5536	46	16	1	46
LTFRB_726268	189.2457	49	16.5	1	49
LTFRB_726268	166.7927	53	17	1	53
LTFRB_726268	198.4926	55	17.25	1	55
LTFRB_726268	189.0317	57	17.75	1	57
LTFRB_726268	180.208	46	18.25	1	46
LTFRB_726268	177.9122	54	18.5	1	54
LTFRB_726268	189.0317	57	19	1	57
LTFRB_726268	393.0782	117	19.75	1	117
LTFRB_726268	207.9208	59	20.25	1	59
LTFRB_726268	198.4892	58	20.75	1	58
LTFRB_726268	177.9122	57	21	1	57
LTFRB_726268	207.9196	60	21.5	1	60
LTFRB_726268	198.4879	54	22	1	54
LTFRB_726268	133.4341	43	22.25	1	43
LTFRB_726268	207.9185	58	22.75	1	58
LTFRB_726268	198.4868	56	23.25	1	56
LTFRB_726268	217.5087	61	23.75	1	61
LTFRB_726268	177.9122	60	24	1	60
LTFRB_726268	247.0508	70	24.5	1	70
LTFRB_726268	146.941	36	25	1	36
LTFRB_726268	198.4847	59	25.25	1	59
LTFRB_726268	200.1512	63	25.75	1	63
LTFRB_726268	217.5064	59	26.25	1	59

LTFRB_726268	247.0491	75	26.75	1	75
LTFRB_726268	227.2359	66	27.25	1	66
LTFRB_726268	227.2355	63	27.75	1	63
LTFRB_726268	100.0756	36	28	1	36
LTFRB_726268	257.103	73	28.5	1	73
LTFRB_726268	266.8682	81	29.25	1	81
LTFRB_726268	298.0803	85	29.75	1	85
LTFRB_726268	255.7487	79	30.5	1	79
LTFRB_726268	267.2408	76	31	1	76
LTFRB_726268	255.7487	76	31.5	1	76
LTFRB_726268	533.5748	166	32.75	1	166
LTFRB_726268	233.5097	71	33.25	1	71
LTFRB_726268	232.8205	80	33.75	1	80
LTFRB_726268	437.2953	119	34.75	1	119
LTFRB_726268	524.6934	162	36	1	162
LTFRB_726268	796.693	244	38	1	244
LTFRB_726268	253.1676	69	38.75	1	69
TOTAL	22347.02		52.5		6608

Total_weight = 22347.02*0.2 + 52.5*0.4 + 6608*0.3 + 753.065*0.1 = **6548.1105**

TRIP ID	DISTANCE	TIME	FARE	TRAFFIC SCORE	TIME*TRAFFIC SCORE
LTFRB_725702	531.3134	152	0	1	152
LTFRB_725702	238.6807	74	0	1	74
LTFRB_725702	406.387	116	0	1	116
LTFRB_725702	585.8316	172	0	1	172
LTFRB_725702	277.9878	85	0	1	85
LTFRB_725702	308.4904	92	0	1	92
LTFRB_725702	591.1283	181	0	1	181
LTFRB_725702	851.909	254	12	1	254
LTFRB_725265	277.9878	86	0	1	86
LTFRB_725265	225.3576	76	0	1	76
LTFRB_725265	437.4893	125	0	1	125
LTFRB_725265	454.1282	162	0	1	162
LTFRB_725265	1160.34	328	0	1	328
LTFRB_725265	363.4098	101	0	1	101
LTFRB_725265	341.4821	105	0	1	105
LTFRB_725265	207.9318	69	0	1	69

LTFRB_725265	537.8793	171	0	1	171
LTFRB_725265	392.0559	123	0	1	123
LTFRB_725265	734.8273	220	12.25	1	220
LTFRB_725265	670.6983	188	13.75	1	188
LTFRB_725265	162.9245	53	14.25	1	53
LTFRB_725265	866.7911	261	16	1	261
LTFRB_725265	301.6067	93	16.75	1	93
LTFRB_725265	351.5786	97	17.5	1	97
LTFRB_725265	298.0931	89	18.25	1	89
LTFRB_725265	610.0216	191	19.5	1	191
LTFRB_725265	522.9731	147	20.5	1	147
LTFRB_725265	406.3812	129	21.5	1	129
LTFRB_725265	293.9111	76	22.25	1	76
LTFRB_725265	171.4215	46	22.5	1	46
LTFRB_725265	257.1133	80	23	1	80
LTFRB_725265	166.7927	52	23.5	1	52
LTFRB_725265	227.2457	63	24	1	63
LTFRB_725265	144.5536	46	24.25	1	46
LTFRB_725265	189.2457	49	24.75	1	49
LTFRB_725265	166.7927	52	25	1	52
LTFRB_725265	198.4926	55	25.5	1	55
LTFRB_725265	189.0317	57	26	1	57
LTFRB_725265	180.208	46	26.25	1	46
LTFRB_725265	177.9122	54	26.75	1	54
LTFRB_725265	189.0317	57	27.25	1	57
LTFRB_725265	227.2425	57	27.75	1	57
LTFRB_725265	177.9122	61	28	1	61
LTFRB_725265	207.9208	60	28.5	1	60
LTFRB_725265	198.4892	57	29	1	57
LTFRB_725265	177.9122	57	29.25	1	57
LTFRB_725265	207.9196	60	29.75	1	60
LTFRB_725265	198.4879	54	30.25	1	54
LTFRB_725265	133.4341	43	30.5	1	43
LTFRB_725265	207.9185	58	31	1	58
LTFRB_725265	198.4868	56	31.5	1	56
LTFRB_725265	217.5087	61	31.75	1	61
LTFRB_725265	177.9122	60	32.25	1	60
LTFRB_725265	247.0508	70	32.75	1	70
LTFRB_725265	146.941	36	33	1	36
LTFRB_725265	198.4847	59	33.5	1	59
LTFRB_725265	200.1512	63	34	1	63

LTFRB_725265	217.5064	59	34.5	1	59
LTFRB_725265	247.0491	75	35	1	75
LTFRB_725265	227.2359	66	35.5	1	66
LTFRB_725265	227.2355	63	36	1	63
LTFRB_725265	100.0756	36	36.25	1	36
LTFRB_725265	257.103	73	36.75	1	73
LTFRB_725265	266.8682	81	37.5	1	81
LTFRB_725265	298.0803	85	38	1	85
LTFRB_725265	255.7487	79	38.5	1	79
LTFRB_725265	267.2408	76	39.25	1	76
LTFRB_725265	255.7487	76	39.75	1	76
LTFRB_725265	533.5748	166	41	1	166
LTFRB_725265	233.5097	71	41.5	1	71
LTFRB_725265	232.8205	80	42	1	80
LTFRB_725265	437.2953	119	43	1	119
LTFRB_725265	524.6934	162	44.75	1	162
LTFRB_725265	796.693	244	46.25	1	244
LTFRB_725265	253.1676	69	47.5	1	69
TOTAL	24422.86		59.5		7265

Total_weight = 24422.86*0.2 + 59.5*0.4 + 7265*0.3 + 753.065*0.1 = **7163.1785**

TRIP ID	DISTANCE	TIME	FARE	TRAFFIC SCORE	TIME*TRAFFIC SCORE
LTFRB_726309	624.8177	190	0	1	190
LTFRB_726309	585.8316	172	0	1	172
LTFRB_726309	277.9878	85	0	1	85
LTFRB_726309	455.8999	139	0	1	139
LTFRB_726309	437.4893	125	0	1	125
LTFRB_726309	225.3576	75	0	1	75
LTFRB_726309	277.9878	87	12	1	87
LTFRB_726167	225.3576	76	0	1	76
LTFRB_726167	437.4893	125	0	1	125
LTFRB_726167	454.1282	162	0	1	162
LTFRB_726167	1160.34	328	0	1	328
LTFRB_726167	363.4098	101	0	1	101
LTFRB_726167	341.4821	105	0	1	105

LTFRB_726167	207.9318	69	0	1	69
LTFRB_726167	537.8793	171	0	1	171
LTFRB_726167	392.0559	123	0	1	123
LTFRB_726167	734.8273	220	0	1	220
LTFRB_726167	670.6983	188	13.25	1	188
LTFRB_726167	162.9245	53	13.5	1	53
LTFRB_726167	866.7911	261	15.5	1	261
LTFRB_726167	301.6067	93	16	1	93
LTFRB_726167	351.5786	97	16.75	1	97
LTFRB_726167	298.0931	89	17.5	1	89
LTFRB_726167	610.0216	191	18.75	1	191
LTFRB_726167	522.9731	147	20	1	147
LTFRB_726167	406.3812	129	21	1	129
LTFRB_726167	293.9111	76	21.5	1	76
LTFRB_726167	171.4215	46	22	1	46
LTFRB_726167	257.1133	80	22.5	1	80
LTFRB_726167	166.7927	52	22.75	1	52
LTFRB_726167	227.2457	63	23.25	1	63
LTFRB_726167	144.5536	46	23.75	1	46
LTFRB_726167	189.2457	49	24	1	49
LTFRB_726167	166.7927	52	24.5	1	52
LTFRB_726167	198.4926	55	25	1	55
LTFRB_726167	189.0317	57	25.25	1	57
LTFRB_726167	180.208	46	25.75	1	46
LTFRB_726167	177.9122	54	26	1	54
LTFRB_726167	189.0317	57	26.5	1	57
LTFRB_726167	227.2425	57	27	1	57
LTFRB_726167	177.9122	61	27.5	1	61
LTFRB_726167	207.9208	60	27.75	1	60
LTFRB_726167	198.4892	57	28.25	1	57
LTFRB_726167	177.9122	57	28.75	1	57
LTFRB_726167	207.9196	60	29.25	1	60
LTFRB_726167	198.4879	54	29.5	1	54
LTFRB_726167	133.4341	43	30	1	43
LTFRB_726167	207.9185	58	30.25	1	58
LTFRB_726167	198.4868	56	30.75	1	56
LTFRB_726167	217.5087	61	31.25	1	61
LTFRB_726167	177.9122	60	31.75	1	60
LTFRB_726167	247.0508	70	32.25	1	70
LTFRB_726167	146.941	36	32.5	1	36
LTFRB_726167	198.4847	59	33	1	59

LTFRB_726167	200.1512	63	33.5	1	63
LTFRB_726167	217.5064	59	33.75	1	59
LTFRB_726167	247.0491	75	34.5	1	75
LTFRB_726167	227.2359	66	35	1	66
LTFRB_726167	227.2355	63	35.5	1	63
LTFRB_726167	100.0756	36	35.75	1	36
LTFRB_726167	257.103	73	36.25	1	73
LTFRB_726167	266.8682	81	36.75	1	81
LTFRB_726167	298.0803	85	37.5	1	85
LTFRB_726167	255.7487	79	38	1	79
LTFRB_726167	267.2408	76	38.5	1	76
LTFRB_726167	255.7487	76	39.25	1	76
LTFRB_726167	533.5748	166	40.25	1	166
LTFRB_726167	233.5097	71	40.75	1	71
LTFRB_726167	232.8205	80	41.25	1	80
LTFRB_726167	437.2953	119	42.25	1	119
LTFRB_726167	524.6934	162	43.5	1	162
LTFRB_726167	796.693	244	45.25	1	244
LTFRB_726167	253.1676	69	46.5	1	69
TOTAL	23238.51		58.5		6926

Total_weight = 23238.51*0.2 + 58.5*0.4 + 6926*0.3 + 753.065*0.1 = **6824.2085**

TRIP ID	DISTANCE	TIME	FARE	TRAFFIC SCORE	TIME*TRAFFIC SCORE
LTFRB_725139	4465.706	1364	12.5	1	1364
LTFRB_725361	309.5085	98	0	1	98
LTFRB_725361	301.6167	98	0	1	98
LTFRB_725361	265.6516	69	0	1	69
LTFRB_725361	171.4313	58	0	1	58
LTFRB_725361	272.3147	82	0	1	82
LTFRB_725361	398.4721	106	0	1	106
LTFRB_725361	162.9245	53	0	1	53
LTFRB_725361	866.7911	261	0	1	261
LTFRB_725361	301.6067	93	0	1	93
LTFRB_725361	351.5786	98	0	1	98
LTFRB_725361	298.0931	88	0	1	88

LTFRB_725361	610.0216	191	0	1	191
LTFRB_725361	522.9731	148	0	1	148
LTFRB_725361	406.3812	128	12.5	1	128
LTFRB_725361	293.9111	76	13.25	1	76
LTFRB_725361	171.4215	46	13.5	1	46
LTFRB_725361	257.1133	80	14	1	80
LTFRB_725361	382.3977	115	15	1	115
LTFRB_725361	144.5536	46	15.25	1	46
LTFRB_725361	189.2457	49	15.75	1	49
LTFRB_725361	166.7927	52	16	1	52
LTFRB_725361	198.4926	55	16.5	1	55
LTFRB_725361	189.0317	57	17	1	57
LTFRB_725361	180.208	46	17.25	1	46
LTFRB_725361	177.9122	54	17.75	1	54
LTFRB_725361	189.0317	58	18	1	58
LTFRB_725361	227.2425	56	18.5	1	56
LTFRB_725361	177.9122	61	19	1	61
LTFRB_725361	207.9208	60	19.5	1	60
LTFRB_725361	198.4892	57	20	1	57
LTFRB_725361	177.9122	57	20.25	1	57
LTFRB_725361	207.9196	60	20.75	1	60
LTFRB_725361	198.4879	54	21.25	1	54
LTFRB_725361	133.4341	43	21.5	1	43
LTFRB_725361	207.9185	58	22	1	58
LTFRB_725361	198.4868	56	22.25	1	56
LTFRB_725361	217.5087	61	22.75	1	61
LTFRB_725361	177.9122	60	23.25	1	60
LTFRB_725361	247.0508	70	23.75	1	70
LTFRB_725361	146.941	37	24	1	37
LTFRB_725361	198.4847	58	24.5	1	58
LTFRB_725361	200.1512	63	25	1	63
LTFRB_725361	217.5064	59	25.5	1	59
LTFRB_725361	247.0491	75	26	1	75
LTFRB_725361	227.2359	66	26.5	1	66
LTFRB_725361	227.2355	63	27	1	63
LTFRB_725361	100.0756	36	27.25	1	36
LTFRB_725361	257.103	73	27.75	1	73
LTFRB_725361	266.8682	81	28.5	1	81
LTFRB_725361	298.0803	85	29	1	85
LTFRB_725361	255.7487	79	29.5	1	79
LTFRB_725361	267.2408	76	30.25	1	76

LTFRB_725361	255.7487	76	30.75	1	76
LTFRB_725361	533.5748	166	32	1	166
LTFRB_725361	233.5097	71	32.5	1	71
LTFRB_725361	232.8205	80	33	1	80
LTFRB_725361	437.2953	119	34	1	119
LTFRB_725361	777.6151	231	36	1	231
LTFRB_725361	544.4624	175	37.25	1	175
LTFRB_725361	253.1676	69	38.25	1	69
TOTAL	21001.29		50.75		6260

Total_weight = 21001.29*0.2 + 50.75*0.4 + 6260*0.3 + 873.819*0.1 = **6185.9399**

	Distance(m) * 0.3	**Time (s) * 0.2**	**Fare (PhP) * 0.4**	**Walk Distance (m) * 0.1**	**Total Weight**
Trip 1	2798.992	22005.6	13	37.65325	**24855.25**
Trip 2	4469.404	1982.4	21	75.3065	**6548.111**
Trip 3	4884.572	2179.5	23.8	75.3065	**7163.179**
Trip 4	4647.702	2077.8	23.4	75.3065	**6824.209**
Trip 5	4200.258	1878	20.3	87.3819	**6185.94**

Table 11.9 Summary of computations

B. Source Codes

1. Mobile Application

a. AroundMetro/AndroidManifest.xml

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.aroundmetro"
    android:versionCode="1"
    android:versionName="1.0" >

    <uses-sdk
        android:minSdkVersion="8"
        android:targetSdkVersion="18" />

    <uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION" />
    <uses-permission android:name="android.permission.ACCESS_FINE_LOCATION" />
    <uses-permission android:name="android.permission.ACCESS_WIFI_STATE" />
    <uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />
    <uses-permission android:name="android.permission.INTERNET" />
    <uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE" />
    <uses-permission android:name="com.google.android.providers.gsf.permission.READ_GSERVICES" />

    <application
        android:allowBackup="true"
        android:icon="@drawable/logo"
        android:label="@string/app_name"
        android:theme="@android:style/Theme.Black.NoTitleBar.Fullscreen" >
        <activity
            android:name="com.example.aroundmetro.SplashScreen"
            android:label="@string/app_name" >
             <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
        <activity
            android:name="com.example.aroundmetro.MainActivity"
            android:label="@string/app_name" >
        </activity>
        <meta-data
            android:name="com.google.android.maps.v2.API_KEY"
            android:value="AIzaSyCQR5aGWQdHE8TuMU6J7oAFw0o1I7cNe2k" />
        <meta-data
            android:name="com.google.android.gms.version"
            android:value="@integer/google_play_services_version" />
    </application>

</manifest>
```

b. AroundMetro/src/com/example/aroundmetro/MainActivity.java

```
package com.example.aroundmetro;

import java.util.ArrayList;
import java.util.HashMap;
import java.util.List;
import android.app.Dialog;
import android.graphics.drawable.ColorDrawable;
import android.os.Bundle;
import android.support.v4.app.Fragment;
import android.support.v4.app.FragmentActivity;
import android.support.v4.app.FragmentManager;
import android.support.v4.app.FragmentTransaction;
import android.view.Menu;
import android.view.View;
import android.view.Window;
import com.example.aroundmetro.R;
import com.example.aroundmetro.api.model.Itinerary;
import com.example.aroundmetro.api.model.Leg;
import com.example.aroundmetro.extras.DirectionFragment;
import com.example.aroundmetro.extras.RootFragment;
```

```
import com.example.aroundmetro.extras.ShowDirectionFragment;
import com.example.aroundmetro.listeners.AppFragment;
import com.example.aroundmetro.model.AppBundle;
import com.google.android.gms.maps.model.Marker;

public class MainActivity extends FragmentActivity implements AppFragment{

  private RootFragment rootFragment;
  private List<Itinerary> currentItineraryList = new ArrayList<Itinerary>();
  private AppBundle appBundle;
  private List<Leg> currentItinerary = new ArrayList<Leg>();
  private int currentItineraryIndex = -1;
  private HashMap<String, HashMap<String, String>> trafficData;
  private ArrayList<Marker> trafficMarkers = new ArrayList<Marker>();

  protected void onCreate(Bundle savedInstanceState) {
           super.onCreate(savedInstanceState);
           setContentView(R.layout.activity_main);

           if (savedInstanceState != null)
              rootFragment = (RootFragment) getSupportFragmentManager().findFragmentByTag("rootFragment");

           if (savedInstanceState == null) {
              FragmentTransaction fragmentTransaction = getSupportFragmentManager().beginTransaction();
              rootFragment = new RootFragment();
              fragmentTransaction.replace(R.id.rootFragment, rootFragment, "root_fragment");
              fragmentTransaction.commit();
           }
  }

  public void onCoachMark(View view){

      final Dialog dialog = new Dialog(this);
      dialog.requestWindowFeature(Window.FEATURE_NO_TITLE);
      dialog.getWindow().setBackgroundDrawable(new ColorDrawable(android.graphics.Color.TRANSPARENT));
      dialog.setContentView(R.layout.coach_mark);
      dialog.setCanceledOnTouchOutside(true);

      View masterView = dialog.findViewById(R.id.coach_mark_master_view);
      masterView.setOnClickListener(new View.OnClickListener() {
          @Override
          public void onClick(View view) { dialog.dismiss(); }
      });
      dialog.show();
  }

  public boolean onCreateOptionsMenu(Menu menu) {
           getMenuInflater().inflate(R.menu.main, menu);
           return true;
  }

  public void switchToDirectionsFragment() {
           FragmentManager fm = getSupportFragmentManager();
           FragmentTransaction transaction = fm.beginTransaction();

           Fragment directionFragment = new DirectionFragment();
           transaction.add(R.id.rootFragment, directionFragment);
           transaction.setTransition(FragmentTransaction.TRANSIT_FRAGMENT_FADE);
           transaction.addToBackStack(null);
           transaction.commit();
  }

  public void switchToShowDirectionsFragment() {
           FragmentManager fm = getSupportFragmentManager();
           FragmentTransaction transaction = fm.beginTransaction();

           Fragment showDirectionFragment = new ShowDirectionFragment();
           transaction.add(R.id.rootFragment, showDirectionFragment);
           transaction.setTransition(FragmentTransaction.TRANSIT_FRAGMENT_FADE);
           transaction.addToBackStack(null);
           transaction.commit();
  }

  public void loadItineraries(List<Itinerary> itineraries) {
           currentItineraryList.clear();
           currentItineraryList.addAll(itineraries);
  }

  public List<Itinerary> getCurrentItineraryList() {
```

```
            return currentItineraryList;
    }

    public List<Leg> getCurrentItinerary() { return currentItinerary; }

    public int getCurrentItineraryIndex() { return currentItineraryIndex; }

    public void onItinerarySelected(int i) {
            if (i >= currentItineraryList.size()) return;
            currentItineraryIndex = i;
            currentItinerary.clear();
            currentItinerary.addAll(currentItineraryList.get(i).legs);
    }

    public void setAppBundle(AppBundle appBundle) { this.appBundle = appBundle; }

    public AppBundle getAppBundle() { return appBundle; }

    public void setTrafficInfo(HashMap<String, HashMap<String, String>> traficInfo) {
            trafficData = traficInfo;
    }

    public HashMap<String, HashMap<String, String>> getTrafficInfo() {
            return trafficData;
    }

    public void addTrafficInfoMarkers(ArrayList<Marker> trafficMarkers) {
            this.trafficMarkers = trafficMarkers;
    }

    public ArrayList<Marker> getTrafficInfoMarkers() {
            return trafficMarkers;
    }

}
```

c. AroundMetro/src/com/example/aroundmetro/SplashScreen.java

```
package com.example.aroundmetro;

import android.app.Activity;
import android.content.Intent;
import android.os.Bundle;
import android.os.Handler;

public class SplashScreen extends Activity {

    private static int SPLASH_TIME_OUT = 3000;

    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_splash_screen);

        new Handler().postDelayed(new Runnable() {
            public void run() {
                Intent i = new Intent(SplashScreen.this, MainActivity.class);
                startActivity(i);

                finish();
            }
        }, SPLASH_TIME_OUT);
    }
}
```

d. AroundMetro/resources/com/example/aroundmetro/api/model/AgencyAndId.java

```
package com.example.aroundmetro.api.model;

import java.io.Serializable;

public class AgencyAndId implements Serializable, Comparable<AgencyAndId> {

    private static final long serialVersionUID = 1L;
    private String agencyId;
    private String id;
    public AgencyAndId() {      }
```

```
    public AgencyAndId(String agencyId, String id) {
             this.agencyId = agencyId;
             this.id = id;
    }

    public String getAgencyId() { return agencyId; }

    public void setAgencyId(String agencyId) { this.agencyId = agencyId;}

    public String getId() { return id; }

    public void setId(String id) { this.id = id; }

    public boolean hasValues() { return this.agencyId != null && this.id != null; }

    public int compareTo(AgencyAndId o) {
             int c = this.agencyId.compareTo(o.agencyId);
             if (c == 0)
                      c = this.id.compareTo(o.id);
             return c;
    }

    public static AgencyAndId convertFromString(String value, char separator) {
             int index = value.indexOf(separator);
             if (index == -1) {
                      throw new IllegalStateException("invalid agency-and-id: " + value);
             } else {
                      return new AgencyAndId(value.substring(0, index),
                                        value.substring(index + 1));
             }
    }

    public int hashCode() { return agencyId.hashCode() ^ id.hashCode(); }
    public boolean equals(Object obj) {
             if (this == obj)
                      return true;
             if (obj == null)
                      return false;
             if (!(obj instanceof AgencyAndId))
                      return false;
             AgencyAndId other = (AgencyAndId) obj;
             if (!agencyId.equals(other.agencyId))
                      return false;
             if (!id.equals(other.id))
                      return false;
             return true;
    }

    public String toString() { return agencyId + "_" + id; }
}
```

e. AroundMetro/resources/com/example/aroundmetro/api/model/Itinerary.java

```
package com.example.aroundmetro.api.model;

import java.io.Serializable;
import java.util.ArrayList;
import java.util.Calendar;
import java.util.Iterator;
import java.util.List;
import lombok.Getter;
import lombok.Setter;
import com.example.aroundmetro.routing.core.Fare;

public class Itinerary implements Serializable{
  private static final long serialVersionUID = 8579533939962545543L;
  public long duration = 0;
  public Calendar startTime = null;
  public Calendar endTime = null;
  public long walkTime = 0;
  public long transitTime = 0;
  public long waitingTime = 0;
  public Double walkDistance = 0.0;
  public Double elevationLost = 0.0;
  public Double elevationGained = 0.0;
  public Integer transfers = 0;

  @Getter @Setter public double totalFare;
```

```
  @Getter @Setter public Fare fare;
  @Getter @Setter public Double totalDistance;

  /*@XmlElementWrapper(name = "legs")*/
  /*@XmlElement(name = "leg")*/

  public List<Leg> legs = new ArrayList<Leg>();
  public boolean tooSloped = false;

  public void addLeg(Leg leg) {
           if(leg != null)
                    legs.add(leg);
  }

  public void removeLeg(Leg leg) {
           if(leg != null)
                    legs.remove(leg);
  }

  public void removeBogusLegs() {
           Iterator<Leg> it = legs.iterator();
           while (it.hasNext()) {
              Leg leg = it.next();
              if (leg.isBogusNonTransitLeg()) { it.remove(); }
           }
   }
}
```

f. AroundMetro/resources/com/example/aroundmetro/api/model/Itinerary.java

```
package com.example.aroundmetro.api.model;

import java.io.Serializable;
import java.util.List;

import lombok.Data;
import lombok.Getter;
import lombok.Setter;
import com.example.aroundmetro.routing.core.TraverseMode;
import com.example.aroundmetro.routing.patch.Alerts;
import com.example.aroundmetro.util.model.EncodedPolylineBean;
import com.fasterxml.jackson.annotation.JsonIgnoreProperties;
import com.fasterxml.jackson.annotation.JsonProperty;

@JsonIgnoreProperties(ignoreUnknown = true)
@Data public class Leg implements Serializable{
   private static final long serialVersionUID = -7657410568807464781L;
   public String startTime = null;
   public String endTime = null;
   public long duration;
   public Double distance = null;

   @Getter @Setter public Double fare = null;

   public String mode = TraverseMode.WALK.toString();
   public String route = "";
   public String agencyName;
   public String agencyUrl;
     public int agencyTimeZoneOffset;
     public String routeId = null;
     public String routeTextColor = null;
     public Boolean interlineWithPreviousLeg;
     public String tripShortName = null;
     public String headsign = null;
     public String agencyId = null;
     public String tripId = null;
     public Place from = null;
     public Place to = null;

     /*@XmlElementWrapper(name = "intermediateStops")*/
     @JsonProperty(value="intermediateStops")
     @Getter @Setter public List<Place> stop;

     public EncodedPolylineBean legGeometry;

     /*@XmlElementWrapper(name = "steps")*/
     @Getter @Setter @JsonProperty(value="steps")
```

```
    public List<WalkStep> walkSteps;

    @Getter @Setter private List<Note> notes;
    @Getter @Setter private List<Alerts> alerts;
    public String routeShortName;
    public String routeLongName;
    public String boardRule;
    public String alightRule;
    public Boolean rentedBike;
}
```

g. AroundMetro/resources/com/example/aroundmetro/api/model/Place.java

```
package com.example.aroundmetro.api.model;

import java.io.Serializable;
import java.util.Calendar;

import lombok.Getter;
import lombok.Setter;
import com.example.aroundmetro.api.model.AgencyAndId;
import com.fasterxml.jackson.annotation.JsonIgnoreProperties;

@JsonIgnoreProperties(ignoreUnknown = true)
public class Place implements Serializable{
    private static final long serialVersionUID = 2731076807632135897L;
    public String name = null;
    public AgencyAndId stopId = null;
    public String stopCode = null;
    public Double lon = null;
    public Double lat = null;
    public Calendar arrival = null;
    public Calendar departure = null;

    /*@XmlAttribute*/
    public String orig;

    /*@XmlAttribute*/
    public String zoneId;

    @Getter @Setter public String geometry;

    public Place() {  }

    public Place(Double lon, Double lat, String name) {
        this.lon = lon;
        this.lat = lat;
        this.name = name;
    }

    public Place(Double lon, Double lat, String name, Calendar timeAtState) {
        this(lon, lat, name);
        this.arrival = departure = timeAtState;
    }
}
```

h. AroundMetro/resources/com/example/aroundmetro/api/model/PlannerError.java

```
package com.example.aroundmetro.api.model;

import java.util.ArrayList;
import java.util.List;

public class PlannerError {
    private int    id;
    private String msg;
    private List<String> missing = new ArrayList<String>();
    private boolean noPath = false;

    /** An error where no path has been found, but no points are missing */
    public PlannerError() { noPath = true; }

    public PlannerError(boolean np) { noPath = np; }

    public PlannerError(String msg) { setMsg(msg); }

    public PlannerError(List<String> missing) { this.setMissing(missing); }
```

```
    public String getMsg() { return msg; }

    public void setMsg(String msg) { this.msg = msg; }

    public int getId() { return id; }

    public void setId(int id) { this.id = id; }

    public void setMissing(List<String> missing) { this.missing = missing; }

    public List<String> getMissing() { return missing; }

    public void setNoPath(boolean noPath) { this.noPath = noPath; }

    public boolean getNoPath() { return noPath; }
}
```

i. AroundMetro/resources/com/example/aroundmetro/api/model/PlannerError.java

```
package com.example.aroundmetro.api.model;

import java.util.ArrayList;
import java.util.Date;
import java.util.List;
import lombok.Getter;
import lombok.Setter;
import com.fasterxml.jackson.annotation.JsonProperty;

public class TripPlan {
    public Date date = null;
    public Place from = null;
    public Place to = null;

    /*@XmlElementWrapper(name="itineraries")*/
    @JsonProperty(value="itineraries")
    @Getter @Setter public List<Itinerary> itinerary = new ArrayList<Itinerary>();

    public TripPlan() {}

    public TripPlan(Place from, Place to, Date date) {
        this.from = from;
        this.to = to;
        this.date = date;
    }

    public void addItinerary(Itinerary itinerary) {
        this.itinerary.add(itinerary);
    }
}
```

j. AroundMetro/resources/com/example/aroundmetro/api/ws/Request.java

```
package com.example.aroundmetro.api.ws;

import java.util.ArrayList;
import java.util.Date;
import java.util.HashMap;
import java.util.List;

import com.example.aroundmetro.routing.core.OptimizeType;
import com.example.aroundmetro.routing.core.TraverseMode;
import com.example.aroundmetro.routing.core.TraverseModeSet;
import com.example.aroundmetro.util.DateUtils;

public class Request implements RequestInf {

   private String from;
   private String to;
   private List<String> intermediatePlaces;
   private Double maxWalkDistance = Double.MAX_VALUE;
   private TraverseModeSet modes; // defaults in constructor
   private OptimizeType optimize = OptimizeType.QUICK;
   private Date dateTime = new Date();
   private boolean arriveBy = false;
   private Integer numItineraries = 10;
   private Integer priority;
```

```
private boolean showIntermediateStops = false;
private String[] preferredRoutes;
private String[] unpreferredRoutes;
private final HashMap<String, String> parameters = new HashMap<String, String>();
private Integer minTransferTime;

public Request() {
    modes = new TraverseModeSet("TRANSIT,WALK");
    setIntermediatePlaces(new ArrayList<String>());
}

public HashMap<String, String> getParameters() { return parameters; }

private void paramPush(String param, Object o) {
    if (o != null)
        parameters.put(param, o.toString());
}
public String getFrom() { return from; }

public void setFrom(String from) {
    paramPush(FROM, from);
    this.from = from;
}

public String getTo() { return to; }

public void setTo(String to) {
    paramPush(TO, to);
    this.to = to;
}

public Double getMaxWalkDistance() { return maxWalkDistance; }

public void setMaxWalkDistance(Double walk) {
    paramPush(MAX_WALK_DISTANCE, walk);
    this.maxWalkDistance = walk;
}

public TraverseModeSet getModes() { return modes; }

public String getModesAsStr() {
    String retVal = null;
    for (TraverseMode m : modes.getModes()) {
        if (retVal == null)
            retVal = "";
        else
            retVal += ", ";
        retVal += m;
    }
    return retVal;
}

public void addMode(TraverseMode mode) {
    modes.setMode(mode, true);
    paramPush(MODE, modes);
}

public void addMode(List<TraverseMode> mList) {
    for (TraverseMode m : mList) {
        addMode(m);
    }
    paramPush(MODE, modes);
}

public OptimizeType getOptimize() { return optimize; }

public void setOptimize(OptimizeType opt) {
    optimize = opt;
    paramPush(OPTIMIZE, optimize);
}

public void setPriority(int priority){
  this.priority = priority;
  paramPush("PRIORITY", priority);
}

public Date getDateTime() { return dateTime; }

public void setDateTime(Date dateTime) { this.dateTime = dateTime; }
```

```
    public void setDateTime(String date, String time) {
        paramPush(DATE, date);
        paramPush(TIME, time);
        dateTime = DateUtils.toDate(date, time);
    }

    public boolean isArriveBy() { return arriveBy; }

    public void setArriveBy(boolean arriveBy) {
        paramPush(ARRIVE_BY, arriveBy);
        this.arriveBy = arriveBy;
    }

    public Integer getNumItineraries() { return numItineraries;  }

    public void setNumItineraries(Integer numItineraries) {
        if (numItineraries < 1 || numItineraries > 10)
            numItineraries = 3;
        paramPush(NUMBER_ITINERARIES, numItineraries);
        this.numItineraries = numItineraries;
    }

    public String toHtmlString() { return toString("<br/>"); }

    public String toString() { return toString(" "); }

    public String toString(String sep) {
        return getFrom() + sep + getTo() + sep + getMaxWalkDistance() + sep + getDateTime() + sep
                + isArriveBy() + sep + getOptimize() + sep + getModesAsStr() + sep
                + getNumItineraries();
    }

    public TraverseModeSet getModeSet() { return modes; }

    public void removeMode(TraverseMode mode) {
        modes.setMode(mode, false);
        paramPush(MODE, modes);
    }

    public void setModes(TraverseModeSet modes) {
        this.modes = modes;
        paramPush(MODE, modes);
    }

    public void setIntermediatePlaces(List<String> intermediatePlaces) {
        this.intermediatePlaces = intermediatePlaces;
    }

    public List<String> getIntermediatePlaces() { return intermediatePlaces; }

    public void setShowIntermediateStops(boolean showIntermediateStops) {
        this.showIntermediateStops = showIntermediateStops;
        paramPush(SHOW_INTERMEDIATE_STOPS, showIntermediateStops);
    }

    public boolean getShowIntermediateStops() { return showIntermediateStops; }

    public void setMinTransferTime(Integer minTransferTime) {
        this.minTransferTime = minTransferTime;
    }

    public Integer getMinTransferTime() { return minTransferTime; }

    public void setPreferredRoutes(String[] preferredRoutes) {
        this.preferredRoutes = preferredRoutes;
    }

    public String[] getPreferredRoutes() { return preferredRoutes; }

    public void setUnpreferredRoutes(String[] unpreferredRoutes) {
        this.unpreferredRoutes = unpreferredRoutes;
    }

    public String[] getUnpreferredRoutes() { return unpreferredRoutes; }
}
```

k. AroundMetro/resources/com/example/aroundmetro/api/ws/RequestInf.java

```
package com.example.aroundmetro.api.ws;

import java.util.Date;
import com.example.aroundmetro.routing.core.OptimizeType;
import com.example.aroundmetro.routing.core.TraverseMode;
import com.example.aroundmetro.routing.core.TraverseModeSet;

public interface RequestInf {

  public static String FROM = "fromPlace";
  public static String TO = "toPlace";
  public static String INTERMEDIATE_PLACES = "intermediatePlaces";
  public static String DATE = "date";
  public static String TIME = "time";
  public static String MAX_WALK_DISTANCE = "maxWalkDistance";
  public static String OPTIMIZE = "optimize";
  public static String MODE = "mode";
  public static String NUMBER_ITINERARIES = "numItineraries";
  public static String SHOW_INTERMEDIATE_STOPS = "showIntermediateStops";
  public static String TRIANGLE_TIME_FACTOR = "triangleTimeFactor";
  public static String TRIANGLE_SLOPE_FACTOR = "triangleSlopeFactor";
  public static String TRIANGLE_SAFETY_FACTOR = "triangleSafetyFactor";
  public static String PREFERRED_ROUTES = "preferredRoutes";
  public static String UNPREFERRED_ROUTES = "unpreferredRoutes";
  public static String ARRIVE_BY = "arriveBy";
  public static String WALK_SPEED = "walkSpeed";
  public static String WHEELCHAIR = "wheelchair";
  public static String MIN_TRANSFER_TIME = "minTransferTime";

  public String getFrom();
  public void setFrom(String from);
  public String getTo();
  public void setTo(String to);
  public Double getMaxWalkDistance();
  public void setMaxWalkDistance(Double walk);
  public TraverseModeSet getModes();
  public void addMode(TraverseMode mode);
  public void setModes(TraverseModeSet mode);
  public OptimizeType getOptimize();
  public void setOptimize(OptimizeType opt);
  public Date getDateTime();
  public void setDateTime(Date dateTime);
  public void setDateTime(String date, String time);
  public boolean isArriveBy();
  public void setArriveBy(boolean arriveBy);
  public Integer getNumItineraries();
  public void setNumItineraries(Integer numItineraries);
  public void removeMode(TraverseMode car);
  public void setShowIntermediateStops(boolean showIntermediateStops);
  public boolean getShowIntermediateStops();
}
```

l. AroundMetro/resources/com/example/aroundmetro/api/ws/Response.java

```
package com.example.aroundmetro.api.ws;

import com.example.aroundmetro.api.model.PlannerError;
import com.example.aroundmetro.api.model.TripPlan;
import com.fasterxml.jackson.annotation.JsonIgnoreProperties;

@JsonIgnoreProperties(ignoreUnknown = true)
public class Response {

    private TripPlan plan;
    private PlannerError error = null;

    public Response() {   }

    public TripPlan getPlan() { return plan; }

    public void setPlan(TripPlan plan) { this.plan = plan; }

    public PlannerError getError() { return error; }

    public void setError(PlannerError error) { this.error = error; }
}
```

m. AroundMetro/resources/com/example/aroundmetro/common/model/P2.java

```
package com.example.aroundmetro.common.model;

public class P2<E> extends T2<E, E> {

    private static final long serialVersionUID = 1L;

    public static <E> P2<E> createPair(E first, E second) {
        return new P2<E>(first, second);
    }

    public P2(E first, E second) { super(first, second); }

    public P2(E[] entries) {
        super(entries[0], entries[1]);
        if (entries.length != 2) {
            throw new IllegalArgumentException("This only takes arrays of 2 arguments");
        }
    }

    public String toString() {
        return "P2(" + getFirst() + ", " + getSecond() + ")";
    }
}
```

n. AroundMetro/resources/com/example/aroundmetro/common/model/P2.java

```
package com.example.aroundmetro.common.model;

import java.io.Serializable;

public class T2<E1, E2> implements Serializable {

    private static final long serialVersionUID = 1L;
    private final E1 _first;
    private final E2 _second;
    public T2(E1 first, E2 second) {
        _first = first;
        _second = second;
    }

    public E1 getFirst() { return _first; }

    public E2 getSecond() { return _second; }

    public int hashCode() {
        final int prime = 31;
        int result = 1;
        result = prime * result + ((_first == null) ? 0 : _first.hashCode());
        result = prime * result + ((_second == null) ? 0 : _second.hashCode());
        return result;
    }

    public boolean equals(Object obj) {
        if (this == obj) return true;
        if (obj == null) return false;
        if (getClass() != obj.getClass()) return false;
        T2<?, ?> other = (T2<?, ?>) obj;
        if (_first == null) {
            if (other._first != null) return false;
        } else if (!_first.equals(other._first)) return false;
        if (_second == null) {
            if (other._second != null) return false;
        } else if (!_second.equals(other._second)) return false;

        return true;
    }

    public String toString() {
        return "T2(" + _first + ", " + _second + ")";
    }
}
```

O. AroundMetro/resources/com/example/aroundmetro/extras/DirectionFragment.java

```
package com.example.aroundmetro.extras;

import java.text.DecimalFormat;
import java.util.ArrayList;
import android.app.Activity;
import android.os.Bundle;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.AdapterView;
import android.widget.AdapterView.OnItemClickListener;
import android.widget.ArrayAdapter;
import android.widget.ListView;
import android.widget.Spinner;
import android.widget.TextView;
import com.example.aroundmetro.R;
import com.example.aroundmetro.api.model.Itinerary;
import com.example.aroundmetro.api.model.Leg;
import com.example.aroundmetro.listeners.AppFragment;
import com.example.aroundmetro.model.AppBundle;
import com.example.aroundmetro.model.Direction;
import com.example.aroundmetro.model.DirectionListFragment;
import com.example.aroundmetro.model.GenerateDirections;
import com.example.aroundmetro.util.ConversionUtils;

public class DirectionFragment extends DirectionListFragment{

  private AppFragment fragmentListener;
  private AppBundle appBundle;
  private ListView lv;
  DirectionFragment df;
  View header = null;
  TextView headerFrom;
  TextView headerTo;
  String[] itinerarySummaryList;
  private ArrayList<Direction> directions;

  public void onAttach(Activity activity) {
        super.onAttach(activity);
        try {
            setFragmentListener((AppFragment) activity);
        } catch (ClassCastException e) {
            throw new ClassCastException(
                    activity.toString() + " must implement AppFragment");
        }
    }

  public View onCreateView(LayoutInflater inflater, ViewGroup container, Bundle savedInstanceState) {
           View mainView = inflater.inflate(R.layout.trip_results, container, false);
           header = inflater.inflate(R.layout.trip_results_header, null);
           return mainView;
  }


  public void onActivityCreated(Bundle savedInstanceState){
           super.onActivityCreated(savedInstanceState);

           fragmentListener = this.getFragmentListener();
           df = this;

           headerFrom = (TextView) header.findViewById(R.id.results_title_from);
           headerTo = (TextView) header.findViewById(R.id.results_title_to);

           appBundle = fragmentListener.getAppBundle();
           headerFrom.setText(appBundle.getFromText());
           headerTo.setText(appBundle.getToText());
           fragmentListener.setAppBundle(appBundle);

           ArrayList<Leg> currentItinerary = new ArrayList<Leg>();
           currentItinerary.addAll(fragmentListener.getCurrentItinerary());
           ArrayList<Itinerary> itineraryList = new ArrayList<Itinerary>();
           itineraryList.addAll(fragmentListener.getCurrentItineraryList());

           int currentItineraryIndex = fragmentListener.getCurrentItineraryIndex();
           directions = new ArrayList<Direction>();
           GenerateDirections dirGen = new GenerateDirections(currentItinerary,
```

```
                getActivity().getApplicationContext());
            ArrayList<Direction> tempDirections = dirGen.getDirections();
            if (tempDirections != null && !tempDirections.isEmpty())
                    directions.addAll(tempDirections);

            final Activity activity = this.getActivity();
            Spinner itinerarySelectionSpinner = (Spinner) header.findViewById(R.id.itinerarySelection);

            itinerarySummaryList = new String[itineraryList.size()];

            DecimalFormat currencyFormatter = new DecimalFormat();
            currencyFormatter.setMaximumFractionDigits(2);
            currencyFormatter.setMinimumFractionDigits(2);

            for (int i = 0; i < itinerarySummaryList.length; i++) {
                    Itinerary it = itineraryList.get(i);
                    itinerarySummaryList[i] = it.transfers + "|";
            }

            for (int i = 0; i < itinerarySummaryList.length; i++) {
                    Itinerary it = itineraryList.get(i);
                    itinerarySummaryList[i] += "PhP " +currencyFormatter.format(it.getTotalFare())+ "|";
                    itinerarySummaryList[i] += ConversionUtils
                             .getFormattedDurationTextNoSeconds(it.duration / 1000,
                             getActivity().getApplicationContext()) + "|";
                    itinerarySummaryList[i] += i;
            }

            ArrayAdapter<String> itineraryAdapter = new SpinnerAdapter(activity, R.layout.custom_spinner,
                                                                       itinerarySummaryList);

            itineraryAdapter.setDropDownViewResource(android.R.layout.simple_spinner_dropdown_item);
            itinerarySelectionSpinner.setAdapter(itineraryAdapter);

            AdapterView.OnItemSelectedListener itinerarySpinnerListener
                = new AdapterView.OnItemSelectedListener() {

            public void onItemSelected(AdapterView<?> parent, View view, int position, long id) {
                    fragmentListener.onItinerarySelected(position);
                    directions = new ArrayList<Direction>();
                    GenerateDirections dirGen = new GenerateDirections(
                           fragmentListener.getCurrentItinerary(),
                           getActivity().getApplicationContext());
                    ArrayList<Direction> tempDirections = dirGen.getDirections();
                    if (tempDirections != null && !tempDirections.isEmpty())
                       directions.addAll(tempDirections);

                   Direction direction_data[] = directions.toArray(new Direction[directions.size()]);

                   DirectionListAdapter adapter = new DirectionListAdapter(
                           DirectionFragment.this.getActivity(),
                           R.layout.trip_results_row_details, direction_data);

                   adapter.notifyDataSetChanged();
                   lv.setAdapter(adapter);
            }
};
        itinerarySelectionSpinner.setSelection(currentItineraryIndex);
        itinerarySelectionSpinner.setOnItemSelectedListener(itinerarySpinnerListener);
        lv = getListView();

        Direction direction_data[] = directions.toArray(new Direction[directions.size()]);

        DirectionListAdapter adapter = new DirectionListAdapter( this.getActivity(),
                R.layout.trip_results_row_details, direction_data);

        adapter.notifyDataSetChanged();
        lv.addHeaderView(header);
        lv.setAdapter(adapter);
        lv.setOnItemClickListener(new OnItemClickListener(){

            public void onItemClick(AdapterView<?> adapterView, View view, int item, long id) {
                    AppBundle appBundle = new AppBundle();
                    appBundle.setLeg(fragmentListener.getCurrentItinerary().get(item-1));
                    DirectionFragment.this.getFragmentListener().setAppBundle(appBundle);
                    DirectionFragment.this.getFragmentListener().switchToShowDirectionsFragment();
            }

        });
```

```
  }

  public AppFragment getFragmentListener() { return fragmentListener; }

  public void setFragmentListener(AppFragment fragmentListener){ this.fragmentListener = fragmentListener;}
}
```

p. AroundMetro/resources/com/example/aroundmetro/extras/DirectionListAdapter.java

```
package com.example.aroundmetro.extras;

import android.app.Activity;
import android.content.Context;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.BaseAdapter;
import android.widget.ImageView;
import android.widget.RelativeLayout;
import android.widget.TextView;
import com.example.aroundmetro.R;
import com.example.aroundmetro.model.Direction;

public class DirectionListAdapter extends BaseAdapter {
  Context context;
  int layoutResourceId;
  Direction data[] = null;

  public DirectionListAdapter(Context context, int resource, Direction[] objects) {
           this.layoutResourceId = resource;
           this.context = context;
           this.data = objects;
  }

  public View getView(int position, View convertView, ViewGroup parent) {
           View rowView = convertView;
           DirectionHolder holder = new DirectionHolder();

           if(rowView == null){
                    LayoutInflater inflater = ((Activity) context).getLayoutInflater();
                    rowView = inflater.inflate(layoutResourceId, parent, false);

                    holder.tvMode = (TextView) rowView.findViewById(R.id.direction_mode);
                    holder.tvFare = (TextView) rowView.findViewById(R.id.direction_fare);
                    holder.tvDuration = (TextView) rowView.findViewById(R.id.direction_duration);
                    holder.tvVehicle = (TextView) rowView.findViewById(R.id.direction_vehicle);
                    holder.tvBoard = (TextView) rowView.findViewById(R.id.direction_board);
                    holder.tvAlight = (TextView) rowView.findViewById(R.id.direction_alight);
                    holder.tvDistance = (TextView) rowView.findViewById(R.id.direction_distance);
                    holder.tvWalkFrom = (TextView) rowView.findViewById(R.id.direction_walkFrom);
                    holder.tvWalkTo = (TextView) rowView.findViewById(R.id.direction_walkTo);

                    holder.imgIcon = (ImageView) rowView.findViewById(R.id.imgIcon);
                    holder.transit = (RelativeLayout) rowView.findViewById(R.id.layout_Transit);
                    holder.nonTransit = (RelativeLayout) rowView.findViewById(R.id.layout_nonTransit);
                    holder.fare = (RelativeLayout) rowView.findViewById(R.id.layout_fare);
                    holder.distance = (RelativeLayout) rowView.findViewById(R.id.layout_distance);

                    rowView.setTag(holder);
           }
           else { holder = (DirectionHolder) rowView.getTag(); }

           Direction dir = data[position];

           String[] directions = dir.getDirectionText().split("\\|");
           holder.tvMode.setText(dir.getDirectionMode().toString());

           if(dir.getDirectionMode().isTransit()){
                    holder.nonTransit.setVisibility(View.GONE);
                    holder.distance.setVisibility(View.GONE);
                    holder.tvFare.setText(directions[0]);
                    holder.tvDuration.setText(directions[1]);
                    holder.tvVehicle.setText(directions[2]);
                    holder.tvBoard.setText(directions[3]);
                    holder.tvAlight.setText(directions[4]);
           }
           else{
                    holder.transit.setVisibility(View.GONE);
```

```
                    holder.fare.setVisibility(View.GONE);
                    holder.tvDistance.setText(directions[0]);
                    holder.tvDuration.setText(directions[1]);
                    holder.tvWalkFrom.setText(directions[2]);
                    holder.tvWalkTo.setText(directions[3]);
            }

            holder.imgIcon.setImageResource(dir.getIcon());
            return rowView;
      }

     public int getCount() {
              if(data != null){ return data.length; }
              return 0;
     }

     public Object getItem(int position) { return data[position]; }

     public long getItemId(int position) { return position; }

     static class DirectionHolder {
              TextView tvMode  , tvFare  , tvDuration  , tvVehicle  ,
                               tvBoard  , tvAlight  , tvDistance  , tvWalkFrom  ,
                               tvWalkTo  ;
              RelativeLayout transit  , nonTransit  , distance  , fare  ;
              ImageView imgIcon;
      }
  }
```

q. **AroundMetro/resources/com/example/aroundmetro/extras/FareDataGenerator.java**

```
package com.example.aroundmetro.extras;

import java.io.IOException;
import java.io.StringReader;
import java.util.ArrayList;
import java.util.HashMap;
import javax.xml.parsers.DocumentBuilder;
import javax.xml.parsers.DocumentBuilderFactory;
import javax.xml.parsers.ParserConfigurationException;
import org.apache.http.HttpEntity;
import org.apache.http.HttpResponse;
import org.apache.http.client.methods.HttpPost;
import org.apache.http.impl.client.DefaultHttpClient;
import org.apache.http.util.EntityUtils;
import org.w3c.dom.Document;
import org.w3c.dom.Element;
import org.w3c.dom.Node;
import org.w3c.dom.NodeList;
import org.xml.sax.InputSource;
import org.xml.sax.SAXException;
import android.os.AsyncTask;
import android.util.Log;
import com.example.aroundmetro.listeners.FareDataGeneratorListener;



public class FareDataGenerator extends AsyncTask<String, Void, Long >{

 private FareDataGeneratorListener callback;
 ArrayList<ArrayList<HashMap<String, String>>> rails;
 ArrayList<HashMap<String, String>> jeepBus;

 public FareDataGenerator( FareDataGeneratorListener callback){
          this.callback = callback;
          rails = new ArrayList<ArrayList<HashMap<String, String>>>();
          jeepBus = new ArrayList<HashMap<String, String>>();
 }

 protected Long doInBackground(String... params) {
        Long size = (long) params.length;
        String xml = getXML();
        Document doc = XMLfromString(xml);

        NodeList children = doc.getElementsByTagName("vehicle");

        for (int i = 0; i < children.getLength(); i++) {
          NodeList cases = children.item(i).getChildNodes();
          System.out.println(children.item(i).getNodeName());
```

```
        ArrayList<HashMap<String, String>> railCase = new ArrayList<HashMap<String, String>>();

        for(int x =0; x < cases.getLength(); x++){
           Node node = children.item(i);
           Element e = (Element) node;
           Node node2 = cases.item(x);
           HashMap<String, String> railFare = new HashMap<String, String>();
           HashMap<String, String> jeepBusFare = new HashMap<String, String>();

           if(node2.getNodeType() == Node.ELEMENT_NODE){
                 Element f = (Element) node2;
                    if(e.getAttribute("name").equals("Jeepney") ||
                                          e.getAttribute("name").equals("Bus")){
                          jeepBusFare.put("baseFare", getValue(f, "baseFare"));
                          jeepBusFare.put("firstKm", getValue(f, "firstKm"));
                          jeepBusFare.put("nextKm", getValue(f, "nextKm"));
                          jeepBus.add(jeepBusFare);
                    }
                    else{
                          railFare.put("startRange", getValue(f, "startRange"));
                          railFare.put("endRange", getValue(f, "endRange"));
                          railFare.put("fare", getValue(f, "fare"));
                          railCase.add(railFare);
                    }
              }
        }
         rails.add(railCase);
      }
  return size;
}

public String getXML(){
      String line = null;

      try {

          DefaultHttpClient httpClient = new DefaultHttpClient();
          HttpPost httpPost = new HttpPost("http://agila.upm.edu.ph/~aknarvaza/xml/railfare.xml");
          HttpResponse httpResponse = httpClient.execute(httpPost);
          HttpEntity httpEntity = httpResponse.getEntity();
          line = EntityUtils.toString(httpEntity);
      } catch (Exception e) { line = "Internet Connection Error >> " + e.getMessage(); }
      return line;
}

 public String getValue(Element item, String str) {
         NodeList n = item.getElementsByTagName(str);
         return getElementValue(n.item(0));
 }

 public final Document XMLfromString(String xml){
         Document doc = null;
         DocumentBuilderFactory dbf = DocumentBuilderFactory.newInstance();
         try {
                  DocumentBuilder db = dbf.newDocumentBuilder();
                  InputSource is = new InputSource();
                  is.setCharacterStream(new StringReader(xml));
                  doc = db.parse(is);
      }
         catch (ParserConfigurationException e) { return null; }
         catch (SAXException e) { return null; }
         catch (IOException e) { return null; }
      return doc;
   }

    public final String getElementValue( Node elem ) {
        Node kid;
        if( elem != null){
            if (elem.hasChildNodes()){
                for( kid = elem.getFirstChild(); kid != null; kid = kid.getNextSibling() ){
                    if( kid.getNodeType() == Node.TEXT_NODE  ){
                        return kid.getNodeValue();
                    }
                }
            }
        }
        return "";
    }
```

```
 protected void onPostExecute(Long result) {
          callback.onFareDataRequestComplete(rails, jeepBus);
 }
}
```

r. AroundMetro/resources/com/example/aroundmetro/extras/LocationGeocode.java

```
package com.example.aroundmetro.extras;

import java.io.IOException;
import java.io.InputStream;
import java.io.UnsupportedEncodingException;
import java.lang.ref.WeakReference;
import java.net.URLEncoder;
import java.util.ArrayList;
import org.apache.http.HttpEntity;
import org.apache.http.HttpResponse;
import org.apache.http.client.ClientProtocolException;
import org.apache.http.client.HttpClient;
import org.apache.http.client.methods.HttpGet;
import org.apache.http.impl.client.DefaultHttpClient;
import org.json.JSONArray;
import org.json.JSONException;
import org.json.JSONObject;
import android.annotation.SuppressLint;
import android.app.Activity;
import android.app.ProgressDialog;
import android.content.Context;
import android.location.Address;
import android.location.Geocoder;
import android.os.AsyncTask;
import android.util.Log;
import com.example.aroundmetro.R;
import com.example.aroundmetro.listeners.GeocodingListener;
import com.google.android.gms.maps.model.LatLng;

public class LocationGeocode extends AsyncTask<String, Integer, Long> {

 private boolean geocodingForMarker;
 private boolean isStartTextBox;
 private Context context;
 private ArrayList<Address> geocodedAddress;
 private GeocodingListener callback;
 private LatLng position;
 private String address;
 private ProgressDialog progressDialog;
 private WeakReference<Activity> activity;

 private final double LOWERLEFTLAT = 14.34795;
 private final double LOWERLEFTLNG = 120.90622;
 private final double UPPERRIGHTLAT = 14.705614;
 private final double UPPERRIGHTLNG = 121.090622;

 public LocationGeocode(WeakReference<Activity> activity, boolean geocodingForMarker, Context context,
                                    GeocodingListener callback, boolean isStartTextBox){

          this.geocodingForMarker = geocodingForMarker;
          this.context = context;
          this.callback = callback;
          this.isStartTextBox = isStartTextBox;
          this.activity = activity;
          geocodedAddress = new ArrayList<Address>();
          Activity activityRetrieved = activity.get();
          if (activityRetrieved != null) { progressDialog = new ProgressDialog(activityRetrieved); }
 }

 protected void onPreExecute(){
          if (activity.get() != null) {
            progressDialog.setIndeterminate(true);
            progressDialog.setCancelable(true);
            Activity activityRetrieved = activity.get();
            if (activityRetrieved != null) {
                progressDialog = ProgressDialog.show(activityRetrieved, "",
                        context.getText(R.string.geocode_msg), true);
            }
        }
 }
```

```
@SuppressLint("UseValueOf")
@Override
protected Long doInBackground(String... reqs) {
         long count = reqs.length;

         if(geocodingForMarker){
                  String[] location = reqs[0].split(",");
                  Double lat = new Double(location[0]);
                  Double lng = new Double(location[1]);
                  position = new LatLng(lat, lng);
                  GeocodePoint(position);
                  return count;
         }
         else{
                  address = reqs[0];
                  GeocodeAddress(address);
                  return count;
         }
}

private void GeocodeAddress(String address) {
         Geocoder gc = new Geocoder(context);
         ArrayList<Address> list = null;

         try {
                  list = (ArrayList<Address>) gc.getFromLocationName(address, 5, LOWERLEFTLAT,
                                    LOWERLEFTLNG, UPPERRIGHTLAT, UPPERRIGHTLNG);

                  if(list.size() != 0 && list != null){
                           ArrayList<Address> tempList = new ArrayList<Address>();
                           for(Address adrs: list){
                           LatLng listPoint = new LatLng(adrs.getLatitude(), adrs.getLongitude());
                                    if(insideBounds(listPoint)){
                                             tempList.add(adrs);
                                    }
                           }
                           geocodedAddress = tempList;
                  }
         } catch (IOException e1) {
                  geocodeDirectFromGoogle(true);
         }
}

private void GeocodePoint(LatLng point){
         Geocoder gc = new Geocoder(context);
         ArrayList<Address> list = null;

         try {
                  list = (ArrayList<Address>) gc.getFromLocation(point.latitude, point.longitude, 1);
                  geocodedAddress.addAll(list);
         } catch (IOException e1) { geocodeDirectFromGoogle(false); }
}

private void geocodeDirectFromGoogle(boolean geocodeFromAddress){
         HttpGet httpGet;
         ArrayList<Address> list = new ArrayList<Address>();
         String req = "";
         try {
            if(geocodeFromAddress){
               httpGet = new HttpGet("https://maps.google.com/maps/api/geocode/json?address="+
                                    URLEncoder.encode(address, "UTF-8") + "&sensor=false" + "&key=" +
                                    context.getResources().getString(R.string.api_key));
               req = "https://maps.google.com/maps/api/geocode/json?address="
                                    + URLEncoder.encode(address, "UTF-8") + "&sensor=false"
                                    + "&key=" + context.getResources().getString(R.string.api_key);

            }
            else{
               httpGet = new HttpGet(“https://maps.googleapis.com/maps/api/geocode/json?latlng="
                                    + position.latitude +","+ position.longitude + "&sensor=true"
                                    + "&key=" + context.getResources().getString(R.string.api_key));
               req = "https://maps.googleapis.com/maps/api/geocode/json?latlng="
                                    + position.latitude +","+ position.longitude + "&sensor=true"
                                    + "&key=" + context.getResources().getString(R.string.api_key);
            }
            HttpClient client = new DefaultHttpClient();
            HttpResponse response;
            StringBuilder stringBuilder = new StringBuilder();
            response = client.execute(httpGet);
```

```
                HttpEntity entity = response.getEntity();
                InputStream stream = entity.getContent();
                int b;
                while ((b = stream.read()) != -1) {
                      stringBuilder.append((char) b);
                }

                JSONObject jsonObject = new JSONObject();
                jsonObject = new JSONObject(stringBuilder.toString());

                if(((JSONArray)jsonObject.get("results")).length() > 0){
                   if(geocodeFromAddress){
                      for(int i=0; i < ((JSONArray)jsonObject.get("results")).length(); i++){
                         Address varAddress = new Address(context.getResources().getConfiguration().locale);
                         LatLng loc = new LatLng(((JSONArray) jsonObject.get("results")).getJSONObject(0)
                               .getJSONObject("geometry").getJSONObject("location").getDouble("lat"),
                               ((JSONArray) jsonObject.get("results")).getJSONObject(0)
                               .getJSONObject("geometry").getJSONObject("location").getDouble("lng"));

                         if(insideBounds(loc)){
                               varAddress.setAddressLine(0, ((JSONArray)
                                         jsonObject.get("results")).getJSONObject(0)
                                         .getString("formatted_address").replace(", Philippines", ""));
                               varAddress.setLatitude(loc.latitude);
                               varAddress.setLongitude(loc.longitude);
                               geocodedAddress.add(varAddress);
                         }
                      }
                   }
                   else{
                         Address varAddress = new Address(context.getResources().getConfiguration().locale);
                               varAddress.setAddressLine(0, ((JSONArray)
                                         jsonObject.get("results")).getJSONObject(0)
                                         .getString("formatted_address").replace(", Philippines", ""));
                               varAddress.setLatitude(position.latitude);
                               varAddress.setLongitude(position.longitude);
                               geocodedAddress.add(varAddress);
                         }
                      }
                   else{
                         geocodedAddress = null;
                   }
                }
                catch (JSONException e) { e.printStackTrace(); }
                catch (UnsupportedEncodingException e) { e.printStackTrace(); }
                catch (ClientProtocolException e) { e.printStackTrace(); }
                catch (IOException e) { e.printStackTrace(); }
          }

          private boolean insideBounds(LatLng loc) {
                   if(loc.latitude > LOWERLEFTLAT && loc.longitude > LOWERLEFTLNG
                                     && loc.latitude < UPPERRIGHTLAT && loc.longitude < UPPERRIGHTLNG)
                            return true;
                   return false;
          }


          protected void onPostExecute(Long result){
                   if (activity.get() != null) {
                     try {
                         if (progressDialog != null && progressDialog.isShowing()) {
                             progressDialog.dismiss();
                         }
                     } catch (Exception e) {
                         Log.e("onPostExecute", "Error in TripRequest PostExecute dismissing dialog: " + e);
                     }
                }
                   callback.onGeocodingComplete(isStartTextBox, geocodedAddress, geocodingForMarker);
          }

         }
```

S . AroundMetro/resources/com/example/aroundmetro/extras/RootFragment.java

```
package com.example.aroundmetro.extras;

import java.io.UnsupportedEncodingException;
import java.lang.ref.WeakReference;
import java.net.URLEncoder;
```

```
import java.util.ArrayList;
import java.util.HashMap;
import java.util.List;
import android.annotation.SuppressLint;
import android.app.Activity;
import android.app.AlertDialog;
import android.app.Dialog;
import android.content.Context;
import android.content.DialogInterface;
import android.location.Address;
import android.os.Bundle;
import android.support.v4.app.Fragment;
import android.text.Editable;
import android.util.Log;
import android.view.KeyEvent;
import android.view.LayoutInflater;
import android.view.View;
import android.view.View.OnClickListener;
import android.view.View.OnFocusChangeListener;
import android.view.ViewGroup;
import android.view.inputmethod.EditorInfo;
import android.view.inputmethod.InputMethodManager;
import android.widget.EditText;
import android.widget.ImageButton;
import android.widget.TextView;
import android.widget.TextView.OnEditorActionListener;
import com.example.aroundmetro.R;
import com.example.aroundmetro.api.model.Itinerary;
import com.example.aroundmetro.api.model.Leg;
import com.example.aroundmetro.api.ws.Request;
import com.example.aroundmetro.listeners.AppFragment;
import com.example.aroundmetro.listeners.FareDataGeneratorListener;
import com.example.aroundmetro.listeners.GeocodingListener;
import com.example.aroundmetro.listeners.TripRequestListener;
import com.example.aroundmetro.model.AppBundle;
import com.example.aroundmetro.routing.core.OptimizeType;
import com.example.aroundmetro.routing.core.TraverseMode;
import com.example.aroundmetro.routing.core.TraverseModeSet;
import com.example.aroundmetro.tasks.TripRequest;
import com.google.android.gms.common.ConnectionResult;
import com.google.android.gms.common.GooglePlayServicesUtil;
import com.google.android.gms.maps.CameraUpdateFactory;
import com.google.android.gms.maps.GoogleMap;
import com.google.android.gms.maps.GoogleMap.OnMapClickListener;
import com.google.android.gms.maps.GoogleMap.OnMapLongClickListener;
import com.google.android.gms.maps.GoogleMap.OnMarkerDragListener;
import com.google.android.gms.maps.SupportMapFragment;
import com.google.android.gms.maps.UiSettings;
import com.google.android.gms.maps.model.BitmapDescriptorFactory;
import com.google.android.gms.maps.model.LatLng;
import com.google.android.gms.maps.model.Marker;
import com.google.android.gms.maps.model.MarkerOptions;

public class RootFragment extends Fragment implements GeocodingListener, TripRequestListener,
                                                                    FareDataGeneratorListener{
    private Marker startMarker;
    private Marker endMarker;
    private LatLng startMarkerPosition;
    private LatLng endMarkerPosition;
    private Address startMarkerAddress;
    private Address endMarkerAddress;
    private boolean mMapFailed;
    private boolean isGeocodingStarted;
    private boolean isMarkerPinned;
    private EditText tbStartLocation;
    private EditText tbEndLocation;
    private ImageButton btnPlanTrip;
    private Context applicationContext;
    private GoogleMap mMap;
    private int GPS_ERRDIALOG_REQUEST = 9001;
    private int priority;
    private AppFragment myFragmentListener;
    private ArrayList<ArrayList<HashMap<String, String>>> rails;
    private ArrayList<HashMap<String, String>> jeepBus;

    @Override
    public View onCreateView(LayoutInflater inflater, ViewGroup container, Bundle savedInstanceState){

       final View mainView = inflater.inflate(R.layout.main, container, false);
```

```
      if(mainView != null){
         tbStartLocation = (EditText) mainView.findViewById(R.id.startLocation);
         tbEndLocation = (EditText) mainView.findViewById(R.id.endLocation);
         btnPlanTrip = (ImageButton) mainView.findViewById(R.id.planTrip);

         return mainView;
      }
      else{
         Log.e("AroundMetro",  "Not possible to obtain main view, UI won't be correctly created");
         return null;
      }
 }

 @Override
public void onAttach(Activity activity) {
    super.onAttach(activity);
    try {
        setFragmentListener((AppFragment) activity);
    } catch (ClassCastException e) {
        throw new ClassCastException(activity.toString()
                + " must implement OtpFragment");
    }
}

 public void setFragmentListener(AppFragment fragmentListener) {
          this.myFragmentListener = fragmentListener;
 }

 public AppFragment getFragmentListener() {
          return myFragmentListener;

 }

 @Override
 public void onActivityCreated(Bundle savedInstanceState) {
    super.onActivityCreated(savedInstanceState);

    applicationContext = getActivity().getApplicationContext();
    mMap = retrieveMap(mMap);

    if(!mMapFailed){
          initializeMap(mMap);
          FareDataGenerator fdg = new FareDataGenerator(this);
          fdg.execute();
    }
    else{
          showErrorDialog("Could not initialize map. Please reopen application.");
    }
 }

 private void initializeMap(GoogleMap myMap) {
          LatLng metroManilaCenter = new LatLng(14.606737, 121.016121);
          UiSettings mapUiSettings = myMap.getUiSettings();
          mapUiSettings.setZoomControlsEnabled(false);
          mapUiSettings.setRotateGesturesEnabled(false);
          mapUiSettings.setTiltGesturesEnabled(true);
          myMap.moveCamera(CameraUpdateFactory.newLatLngZoom(metroManilaCenter, 11));
          addMapListeners();
          addInterfaceListeners();
 }

 private void addInterfaceListeners() {

    OnEditorActionListener onEnterKeyPress = new OnEditorActionListener(){

       public boolean onEditorAction(TextView view, int id, KeyEvent event) {
          if(view.getId() == R.id.startLocation){
             if(id == EditorInfo.IME_ACTION_NEXT || event != null
                   && event.getAction() == KeyEvent.ACTION_DOWN
                   && event.getKeyCode() == KeyEvent.KEYCODE_ENTER){

                   CharSequence text = view.getText();
                   if(text != null){
                      isMarkerPinned = false;
                      updateMarkerAddress(true, text.toString(), false);
                   }
             }
          } else if(view.getId() == R.id.endLocation){
             if(id == EditorInfo.IME_ACTION_DONE || event != null
```

```
                    && event.getAction() == KeyEvent.ACTION_DOWN
                    && event.getKeyCode() == KeyEvent.KEYCODE_ENTER){

                    CharSequence text = view.getText();
                    if(text != null){
                       isMarkerPinned = false;
                       updateMarkerAddress(false, text.toString(), false);
                    }
                }
            }
            return false;
        }
    };

tbStartLocation.setOnEditorActionListener(onEnterKeyPress);
tbEndLocation.setOnEditorActionListener(onEnterKeyPress);

   OnFocusChangeListener tbChangeFocusListener = new OnFocusChangeListener() {

      public void onFocusChange(View v, boolean hasFocus) {
         TextView tv = (TextView) v;
         if(!hasFocus){
            String text = tv.getText().toString();

            if(!text.equals("")){
               if(v.getId() == R.id.startLocation && isGeocodingStarted == false
                           && isMarkerPinned == false){
                  updateMarkerAddress(true, text, false);
               }
               else if(v.getId() == R.id.endLocation && isGeocodingStarted == false
                           && isMarkerPinned == false){
                  updateMarkerAddress(false, text, false);
               }
            }
            else{
               if(v.getId() == R.id.startLocation && startMarker != null){
                  startMarker.remove();
                  startMarkerAddress = null;
                  startMarkerPosition = null;
                  startMarker = null;
               }
               else if(v.getId() == R.id.endLocation && endMarker != null){
                  endMarker.remove();
                  endMarkerAddress = null;
                  endMarkerPosition = null;
                  endMarker = null;
               }
            }
         }
      }
   };

tbStartLocation.setOnFocusChangeListener(tbChangeFocusListener);
tbEndLocation.setOnFocusChangeListener(tbChangeFocusListener);

btnPlanTrip.setOnClickListener(new OnClickListener() {

   public void onClick(View v) {
         InputMethodManager imm = (InputMethodManager) RootFragment.this.getActivity()
                        .getSystemService(Context.INPUT_METHOD_SERVICE);
         imm.hideSoftInputFromWindow(tbEndLocation.getWindowToken(), 0);
         imm.hideSoftInputFromWindow(tbStartLocation.getWindowToken(), 0);

         final CharSequence[] items = {"Least Fare", "Least Time", "Least Distance"};

         AlertDialog.Builder builder = new AlertDialog.Builder(RootFragment.this.getActivity());
                  builder.setTitle("Rank trips according to");
                  builder.setItems(items, new DialogInterface.OnClickListener() {

                     public void onClick(DialogInterface dialog, int choice) {
                           priority = choice;
                           processTripRequest();
                     }
                  });

         AlertDialog choose = builder.create();
         choose.show();
         }
   });
```

```
    }

private void addMapListeners() {

    mMap.setOnMapClickListener(new OnMapClickListener() {

       public void onMapClick(LatLng point) {
             InputMethodManager imm = (InputMethodManager) RootFragment.this.getActivity()
                      .getSystemService(Context.INPUT_METHOD_SERVICE);
             imm.hideSoftInputFromWindow(tbEndLocation.getWindowToken(), 0);
             imm.hideSoftInputFromWindow(tbStartLocation.getWindowToken(), 0);

             isMarkerPinned = true;
             if(tbStartLocation.hasFocus() && isGeocodingStarted == false){
                      addMarker(true, point);
                      updateMarkerAddress(true, LatLngToString(point), true);
             }
             else if(tbEndLocation.hasFocus() && isGeocodingStarted == false){
                      addMarker(false, point);
                      updateMarkerAddress(false, LatLngToString(point), true);
             }
       }
    });

    mMap.setOnMarkerDragListener(new OnMarkerDragListener() {

       public void onMarkerDragStart(Marker marker) {
             InputMethodManager imm = (InputMethodManager) RootFragment.this.getActivity()
                       .getSystemService(Context.INPUT_METHOD_SERVICE);
             imm.hideSoftInputFromWindow(tbEndLocation.getWindowToken(), 0);
             imm.hideSoftInputFromWindow(tbStartLocation.getWindowToken(), 0);
       }

       public void onMarkerDragEnd(Marker marker) {
             LatLng dragEndPoint = marker.getPosition();

             if((startMarker != null) && (marker.hashCode() == startMarker.hashCode())){
                      updateMarkerPosition(true, dragEndPoint);
                      updateMarkerAddress(true, LatLngToString(dragEndPoint), true);
             }
             else{
                      updateMarkerPosition(false, dragEndPoint);
                      updateMarkerAddress(false, LatLngToString(dragEndPoint), true);
             }
       }

       public void onMarkerDrag(Marker arg0) { }
    });



    mMap.setOnMapLongClickListener(new OnMapLongClickListener() {

       public void onMapLongClick(LatLng latLng) {
             InputMethodManager imm = (InputMethodManager) RootFragment.this.getActivity()
                      .getSystemService(Context.INPUT_METHOD_SERVICE);
             imm.hideSoftInputFromWindow(tbEndLocation.getWindowToken(), 0);
             imm.hideSoftInputFromWindow(tbStartLocation.getWindowToken(), 0);

             final LatLng point = latLng;
             final CharSequence[] items = {"Set as start location", "Set as end location"};

             AlertDialog.Builder builder = new AlertDialog.Builder(RootFragment.this.getActivity());
                      builder.setTitle("Choose Location");
                      builder.setItems(items, new DialogInterface.OnClickListener() {

                         public void onClick(DialogInterface dialog, int choice) {
                               if(isGeocodingStarted == false){
                                        if(choice == 0){
                                                 addMarker(true, point);
                                                 updateMarkerAddress(true, LatLngToString(point), true);
                                        }
                                        else{
                                                 addMarker(false, point);
                                                 updateMarkerAddress(false, LatLngToString(point), true);
                                        }
                               }
                         }
```

```
                    });
            AlertDialog choose = builder.create();
            choose.show();
        }
    });
}

private void addMarker(boolean isStartMarker, LatLng point) {
    if(isStartMarker){
            if(startMarker == null) startMarker = createMarker(true, point);
            else updateMarkerPosition(true, point);
    }
    else{
            if(endMarker == null) endMarker = createMarker(false, point);
            else updateMarkerPosition(false, point);
    }
}

private void updateMarkerPosition(boolean isStartMarker, LatLng point) {
    if(isStartMarker){
            if(startMarker == null) startMarker = createMarker(true, point);
            else startMarker.setPosition(point);
            startMarkerPosition = point;
    }
    else{
            if(endMarker == null) endMarker = createMarker(false, point);
            else endMarker.setPosition(point);
            endMarkerPosition = point;
    }
}

private Marker createMarker(boolean isStartMarker, LatLng point) {
    MarkerOptions markerOptions = new MarkerOptions()
                                .position(point)
                                .snippet("Long press to drag")
                                .draggable(true);
    if(isStartMarker){
            markerOptions
                    .title("Start Location")
                    .icon(BitmapDescriptorFactory.defaultMarker(BitmapDescriptorFactory.HUE_MAGENTA));
            startMarkerPosition = point;
            return mMap.addMarker(markerOptions);
    }
    else{
            markerOptions
                    .title("End Location")
                    .icon(BitmapDescriptorFactory.defaultMarker(BitmapDescriptorFactory.HUE_BLUE));
            endMarkerPosition = point;
            return mMap.addMarker(markerOptions);
    }
}

private String LatLngToString(LatLng point) {
    return String.valueOf(point.latitude) + "," + String.valueOf(point.longitude);
}

private void updateMarkerAddress(boolean isStartMarker, String location, boolean geocodingForMarker) {
    if(location.equals("")){
       showErrorDialog("Address box is empty!");
       requestTbFocus(isStartMarker);
    }
    else{
       WeakReference<Activity> weakContext = new WeakReference<Activity>(RootFragment.this.getActivity());
       isGeocodingStarted = true;
       LocationGeocode geocoder = new LocationGeocode(weakContext, geocodingForMarker,
                                                      applicationContext, this, isStartMarker);
       geocoder.execute(location);
    }
}

public void onGeocodingComplete(final boolean isStartTextBox, ArrayList<Address> address,
                                                boolean geocodingForMarker) {
    isGeocodingStarted = false;
    if(getActivity() != null){
       if(geocodingForMarker){
          if(address != null && address.size() != 0 && !address.isEmpty()){
             setTextBoxLocation(isStartTextBox, getStringAddress(address.get(0)));
             if(isStartTextBox){
                startMarkerAddress = address.get(0);
```

```
                    startMarkerPosition = new LatLng( address.get(0).getLatitude(),
                                                      address.get(0).getLongitude());
                } else{
                    endMarkerAddress = address.get(0);
                    endMarkerPosition = new LatLng( address.get(0).getLatitude(),
                                                      address.get(0).getLongitude());
                }
            }
            else{
                showErrorDialog("Could not find address for pinned location.");
            }
        } else{
            if(address == null || address.isEmpty()){
                showErrorDialog("Could not find a location for this address.");
                requestTbFocus(isStartTextBox);
            }
            else if(address.size() == 1){
                LatLng addressPoint = new LatLng(address.get(0).getLatitude(),address.get(0).getLongitude());
                if(isStartTextBox){
                        if(startMarker == null){
                                startMarker = createMarker(true, addressPoint);
                        }
                        else{
                                startMarkerPosition = addressPoint;
                                startMarker.setPosition(addressPoint);
                                startMarkerAddress = address.get(0);
                        }
                } else{
                        if(endMarker == null){
                                endMarker = createMarker(false, addressPoint);
                        }
                        else{
                                endMarkerPosition = addressPoint;
                                endMarker.setPosition(addressPoint);
                                endMarkerAddress = address.get(0);
                        }
                }
            } else{
                if(isMarkerPinned == false){
                        AlertDialog.Builder geocoderSelector = new AlertDialog.Builder(getActivity());
                        geocoderSelector.setTitle("Choose location");

                        final CharSequence[] addressesText = new CharSequence[address.size()];
                                for (int i = 0; i < address.size(); i++) {
                                        Address temp = address.get(i);
                                        addressesText[i] = temp.getAddressLine(0);
                                }

                        final ArrayList<Address> addressesTemp = address;
                        geocoderSelector.setItems(addressesText, new DialogInterface.OnClickListener() {
                                public void onClick(DialogInterface dialog, int item) {
                                   Address addressChoice = addressesTemp.get(item);
                                   LatLng choiceLoc = new LatLng(addressChoice.getLatitude(),
                                                                 addressChoice.getLongitude());
                                   updateMarkerPosition(isStartTextBox, choiceLoc);
                                   setTextBoxLocation(isStartTextBox, addressChoice.getAddressLine(0));

                                   if(isStartTextBox) startMarkerAddress = addressChoice;
                                   else endMarkerAddress = addressChoice;
                                }
                        });

                        AlertDialog alertGeocoder = geocoderSelector.create();
                        alertGeocoder.show();
                    }
                  }
            }
        }
}

private void processTripRequest() {
    if(startMarkerPosition == null || startMarkerAddress == null
                        || endMarkerPosition == null || endMarkerAddress == null){
            showErrorDialog("Cannot request trip due to incomplete data.");
    }
    else{ requestTrip();}
}
```

```
private void requestTrip() {
     String startLoc;
     String endLoc;
     startLoc = LatLngToString(startMarkerPosition);
     endLoc = LatLngToString(endMarkerPosition);
     Request request = new Request();
     try {
               request.setFrom(URLEncoder.encode(startLoc, "UTF-8"));
               request.setTo(URLEncoder.encode(endLoc, "UTF-8"));
     } catch (UnsupportedEncodingException e) { e.printStackTrace(); }
     request.setOptimize(OptimizeType.QUICK);
     request.setModes(new TraverseModeSet(TraverseMode.RAIL, TraverseMode.BUS,
                                          TraverseMode.TRANSIT, TraverseMode.WALK));
     request.setShowIntermediateStops(Boolean.TRUE);
     request.setNumItineraries(5);
     request.setPriority(priority);
     WeakReference<Activity> weakContext = new WeakReference<Activity>(RootFragment.this.getActivity());
     new TripRequest(weakContext, applicationContext, this).execute(request);
}

public void onTripRequestCompleted(List<Itinerary> itineraries) {
     saveAppBundle();
     List<Itinerary> itinerariesModified = getFareAndDuration(itineraries);
     getFragmentListener().loadItineraries(itinerariesModified);
     getFragmentListener().onItinerarySelected(0);
     getFragmentListener().switchToDirectionsFragment();
}

private List<Itinerary> getFareAndDuration(List<Itinerary> itineraries) {
     for(Itinerary it: itineraries){
               ArrayList<Leg> legs = (ArrayList<Leg>) it.legs;
               double itFare = 0; long duration =0; Double distance = 0.0;
               for(Leg leg: legs){
                         itFare += getLegFare(leg);
                         duration += leg.duration;
                         distance += leg.distance;
               }
               it.setTotalFare(itFare);
               it.duration = duration;
               it.setTotalDistance(distance);
               System.out.println("itinerary: " + distance);
     }
     return itineraries;
}

@SuppressLint("UseValueOf")
public double getLegFare(Leg leg){
     double legFare = 0, baseFare =0, perKmFare = 0, initial = 0;

     if(leg.mode.equals("BUS")){
               if(leg.routeId.substring(6, 9).equals("PUJ")){
                         leg.mode = TraverseMode.JEEPNEY.toString();
                         baseFare = Double.parseDouble(jeepBus.get(0).get("baseFare"));
                         perKmFare = Double.parseDouble(jeepBus.get(0).get("nextKm"));
                         initial = Double.parseDouble(jeepBus.get(0).get("firstKm"))*1000;
               }
               else{
                         leg.mode = TraverseMode.BUS.toString();
                         baseFare = new Double(jeepBus.get(1).get("baseFare"));
                         perKmFare = new Double(jeepBus.get(1).get("nextKm"));
                         initial = new Double(jeepBus.get(1).get("firstKm"))*1000;
               }
               legFare = computeBusJeepFare(baseFare, perKmFare, initial, leg.distance);
     }
     else if(leg.mode.equals("RAIL")){
               ArrayList<HashMap<String, String>> cases = new ArrayList<HashMap<String, String>>();
               int stops = leg.getStop().size()+1;
               if(leg.route.equals("LRT 2")){
                         cases = rails.get(1);
                         for(int x = 0; x<cases.size(); x++){
                                   if(stops >= Integer.parseInt(cases.get(x).get("startRange"))
                                             && stops <= Integer.parseInt(cases.get(x).get("endRange"))){
                                             legFare = Integer.parseInt(cases.get(x).get("fare"));
                                   }
                         }
               }
               else if(leg.route.equals("LRT 1")){
                         cases = rails.get(0);
                         for(int x = 0; x<cases.size(); x++){
```

```
                                if(stops >= Integer.parseInt(cases.get(x).get("startRange"))
                                        && stops <= Integer.parseInt(cases.get(x).get("endRange"))){
                                        legFare = Integer.parseInt(cases.get(x).get("fare"));
                                }
                        }
                }
                else if(leg.route.equals("MRT-3")){
                        cases = rails.get(2);
                        for(int x = 0; x<cases.size(); x++){
                                if(stops >= Integer.parseInt(cases.get(x).get("startRange"))
                                        && stops <= Integer.parseInt(cases.get(x).get("endRange"))){
                                        legFare = Integer.parseInt(cases.get(x).get("fare"));
                                }
                        }
                }
        }
        leg.setFare(legFare);
        return legFare;
}

private double computeBusJeepFare(double baseFare, double perKmFare, double initial, Double distance) {
        if((distance - initial) < 0){ return baseFare ; }
        else{
                double tmpFare = baseFare + (((distance - initial)/1000)*perKmFare);
                return Math.round(tmpFare * 4.0)/4.0 ;
        }
}

private void saveAppBundle() {
        AppBundle appBundle = new AppBundle();
        appBundle.setFromText(getLocationTbText(true));
        appBundle.setToText(getLocationTbText(false));
        this.getFragmentListener().setAppBundle(appBundle);
}

private void requestTbFocus(boolean isStartTextBox) {
        if(isStartTextBox){ tbStartLocation.requestFocus();}
        else{ tbEndLocation.requestFocus(); }
}

private String getStringAddress(Address address) {
        String result = " ";
        if(address.getMaxAddressLineIndex() >= 0){
                int n = address.getMaxAddressLineIndex();
                for (int i=0; i<=n-1; i++) {
                        if (i!=0) result += ", ";
                        result += address.getAddressLine(i);
                }
                return result;
        }
        else{ return null; }
}

public void setTextBoxLocation(boolean isStartMarker, String text){
        if(isStartMarker){ tbStartLocation.setText(text); }
        else{ tbEndLocation.setText(text); }
}

private String getLocationTbText(boolean isStartMarker){
        Editable locationTbText = null;
        if(isStartMarker){ locationTbText = tbStartLocation.getText(); }
        else{ locationTbText = tbEndLocation.getText(); }
        return locationTbText.toString();
}

private GoogleMap retrieveMap(GoogleMap myMap) {
        mMapFailed = false;
        if(myMap == null){
           myMap = ((SupportMapFragment) getFragmentManager().findFragmentById(R.id.map)).getMap();

           if(myMap == null){
                 Log.w("MAP", "Map is NULL!!");
                 int isAvailable = GooglePlayServicesUtil.isGooglePlayServicesAvailable(applicationContext);

                 if (isAvailable != ConnectionResult.SUCCESS){
                    Log.w("Google Play", "Google Play not available!");
                    Dialog dialog = GooglePlayServicesUtil.getErrorDialog(isAvailable,
                                                        getActivity(), GPS_ERRDIALOG_REQUEST );
                    dialog.show();
```

```
                mMapFailed = true;
            }
        }
    }
    return myMap;
}

private void showErrorDialog(String string) {
    InputMethodManager imm = (InputMethodManager) RootFragment.this.getActivity()
              .getSystemService(Context.INPUT_METHOD_SERVICE);
    imm.hideSoftInputFromWindow(tbEndLocation.getWindowToken(), 0);
    imm.hideSoftInputFromWindow(tbStartLocation.getWindowToken(), 0);

    AlertDialog.Builder errorAlert = new AlertDialog.Builder(RootFragment.this.getActivity());
    errorAlert.setTitle("Error");
    errorAlert.setMessage(string);
    errorAlert.setCancelable(false);
    errorAlert.setPositiveButton("OK", new DialogInterface.OnClickListener() {
            public void onClick(DialogInterface dialog, int which) { }
    });
    AlertDialog error = errorAlert.create();
    error.show();
}

public void onFareDataRequestComplete( ArrayList<ArrayList<HashMap<String, String>>> rails,
                                                    ArrayList<HashMap<String, String>> jeepBus) {
    this.rails = rails;
    this.jeepBus = jeepBus;
}
```

t. **AroundMetro/resources/com/example/aroundmetro/extras/ShowDirectionFragment.java**

```
package com.example.aroundmetro.extras;

import java.util.ArrayList;
import java.util.List;
import android.app.Activity;
import android.content.Context;
import android.content.res.Resources;
import android.graphics.Bitmap;
import android.graphics.BitmapFactory;
import android.graphics.Color;
import android.os.Bundle;
import android.support.v4.app.Fragment;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.TextView;
import com.example.aroundmetro.R;
import com.example.aroundmetro.api.model.Leg;
import com.example.aroundmetro.listeners.AppFragment;
import com.example.aroundmetro.routing.core.TraverseMode;
import com.example.aroundmetro.util.ConversionUtils;
import com.example.aroundmetro.util.LocationUtil;
import com.google.android.gms.maps.CameraUpdateFactory;
import com.google.android.gms.maps.GoogleMap;
import com.google.android.gms.maps.GoogleMap.InfoWindowAdapter;
import com.google.android.gms.maps.SupportMapFragment;
import com.google.android.gms.maps.model.BitmapDescriptorFactory;
import com.google.android.gms.maps.model.CameraPosition;
import com.google.android.gms.maps.model.LatLng;
import com.google.android.gms.maps.model.Marker;
import com.google.android.gms.maps.model.MarkerOptions;
import com.google.android.gms.maps.model.PolylineOptions;

public class ShowDirectionFragment extends Fragment{

    private AppFragment fragmentListener;
    private GoogleMap myMap;
    private Context context;
    private View mainView;
    public String sBoundText = "", nBoundText = "", snippetText = "";

   public void onAttach(Activity activity) {
       super.onAttach(activity);
       try {
           setFragmentListener((AppFragment) activity);
       } catch (ClassCastException e) {
           throw new ClassCastException(
```

```
                activity.toString() + " must implement AppFragment");
    }
}

public View onCreateView(LayoutInflater inflater, ViewGroup container, Bundle savedInstanceState) {
   mainView = inflater.inflate(R.layout.show_direction_layout, container, false);
   return mainView;
}

public void onActivityCreated(Bundle savedInstanceState){
   super.onActivityCreated(savedInstanceState);
   fragmentListener = this.getFragmentListener();
   context = this.getActivity();
   Leg clickedLeg = fragmentListener.getAppBundle().getLeg();
   LatLng targetPoint = null;
   myMap = ((SupportMapFragment) getFragmentManager().findFragmentById(R.id.map_directions)).getMap();

   ArrayList<Leg> itinerary = (ArrayList<Leg>) fragmentListener.getCurrentItinerary();
   LatLng destination = new LatLng(itinerary.get(itinerary.size()-1).to.lat,
                                   itinerary.get(itinerary.size()-1).to.lon);
   myMap.setInfoWindowAdapter(new InfoWindowAdapter() {
         public View getInfoWindow(Marker arg0) { return null; }

         public View getInfoContents(Marker marker) {
            View view;
            view = ((LayoutInflater) context.getSystemService( Context.LAYOUT_INFLATER_SERVICE ))
                                    .inflate(R.layout.info_window_layout_trip, null);
            TextView legInfo = (TextView) view.findViewById(R.id.leg_info);
            TextView legDetails = (TextView) view.findViewById(R.id.leg_details);
            TextView legFrom = (TextView) view.findViewById(R.id.leg_from_text);
            TextView legTo = (TextView) view.findViewById(R.id.leg_to_text);
            TextView legFromLabel = (TextView) view.findViewById(R.id.leg_from_label);
            TextView legToLabel = (TextView) view.findViewById(R.id.leg_to_label);

            String[] info = marker.getSnippet().split("\\|");
            if(info[0].equals("HOME")){
                  legFromLabel.setVisibility(View.GONE);
                  legToLabel.setVisibility(View.GONE);
                  legInfo.setText(info[0]);
                  legDetails.setText(info[1]);
            }
            else{
                  legInfo.setText(info[0]);
                  legDetails.setText(info[1]);
                  legFrom.setText(info[2]);
                  legTo.setText(info[3]);
            }
            return view;
         }
      });

      int index = 0;
      for(Leg leg: itinerary){
         index++;
         String snippetTxt = "";
         List<LatLng> points = LocationUtil.decodePoly(leg.legGeometry.getPoints());
         if(leg == clickedLeg){ targetPoint = points.get(0); }

         MarkerOptions modeMarkerOptions = new MarkerOptions().position(points.get(0));
         TraverseMode traverseMode = TraverseMode.valueOf(leg.mode);

         Bitmap b = decodeSampledBitmapFromResource(getResources(), getPathIcon(leg.mode), 37, 65);
         modeMarkerOptions.icon(BitmapDescriptorFactory.fromBitmap(b));

         if(traverseMode.isTransit()){
            snippetTxt = "Leg " + index + ": " + leg.mode + "|"
                           + leg.route + "|"
                           + leg.from.name + "|"
                           + leg.to.name;
            modeMarkerOptions.title("TRIP INFO");
            modeMarkerOptions.snippet(snippetTxt);
         }
         else{
            snippetTxt = "Leg " + index + ": " + leg.mode + "|"
                           + ConversionUtils.getFormattedDistance(leg.getDistance(), context) + "|"
                           + leg.from.name + "|"
                           + leg.to.name;
            modeMarkerOptions.title("TRIP INFO");
            modeMarkerOptions.snippet(snippetTxt);
```

```
            }

            myMap.addMarker(modeMarkerOptions);
            PolylineOptions options = new PolylineOptions().addAll(points).width(10);

            if(leg.mode.equals("WALK")){ options.color(Color.GREEN); }
            else if(leg.mode.equals("JEEPNEY")){ options.color(Color.rgb(139, 0, 0)); }
            else if(leg.mode.equals("BUS")){ options.color(Color.BLUE); }
            else if(leg.mode.equals("RAIL")){ options.color(Color.rgb(255, 128, 0)); }
            myMap.addPolyline(options);
          }

          Bitmap b = decodeSampledBitmapFromResource(getResources(), R.drawable.pin_home, 37, 65);
          String homeSnippet = "HOME" + "|" + "Trip destination" + "|" + "," + ",";
          myMap.addMarker(new MarkerOptions()
                               .title("TRIP INFO")
                               .snippet(homeSnippet)
                               .position(destination)
                               .icon(BitmapDescriptorFactory.fromBitmap(b)));

          CameraPosition cameraPosition =new CameraPosition.Builder()
                                                 .target(targetPoint)
                                                 .bearing(60)
                                                 .tilt(70)
                                                 .zoom(17)
                                                 .build();
          myMap.animateCamera(CameraUpdateFactory.newCameraPosition(cameraPosition));
    }

    private int getPathIcon(String modeString) {
       TraverseMode mode = TraverseMode.valueOf(modeString);
       int icon;

       if ((mode.compareTo(TraverseMode.BUS) == 0) || (mode.compareTo(TraverseMode.BUSISH) == 0)) {
           icon = R.drawable.pin_bus_2;
       } else if ((mode.compareTo(TraverseMode.RAIL) == 0) || (
               mode.compareTo(TraverseMode.TRAINISH) == 0)) {
           icon = R.drawable.pin_train_2;
       } else if (mode.compareTo(TraverseMode.WALK) == 0) {
           icon = R.drawable.pin_walk_2;
       } else if (mode.compareTo(TraverseMode.JEEPNEY) == 0) {
           icon = R.drawable.pin_jeep_2;
       } else if (mode.compareTo(TraverseMode.TRANSIT) == 0) {
           icon = R.drawable.road;
       } else {
           icon = R.drawable.road;
       }
       return icon;
    }

    public Bitmap decodeSampledBitmapFromResource(Resources res, int resId, int reqWidth, int reqHeight) {

        final BitmapFactory.Options options = new BitmapFactory.Options();
        options.inJustDecodeBounds = true;
        BitmapFactory.decodeResource(res, resId, options);
        options.inSampleSize = calculateInSampleSize(options, reqWidth, reqHeight);
        options.inJustDecodeBounds = false;
        return BitmapFactory.decodeResource(res, resId, options);
    }

    public int calculateInSampleSize(BitmapFactory.Options options, int reqWidth, int reqHeight) {
        final int height = options.outHeight;
        final int width = options.outWidth;
        int inSampleSize = 1;

        if (height > reqHeight || width > reqWidth) {
            final int halfHeight = height / 2;
            final int halfWidth = width / 2;
            while ((halfHeight / inSampleSize) > reqHeight && (halfWidth / inSampleSize) > reqWidth) {
                    inSampleSize *= 2;
            }
        }
        return inSampleSize;
    }

    public void onDestroyView() {
            super.onDestroyView();
            Fragment f = getFragmentManager().findFragmentById(R.id.map_directions);
            if (f != null) getFragmentManager().beginTransaction().remove(f).commit();
```

```
    }

    public AppFragment getFragmentListener() { return fragmentListener; }

    public void setFragmentListener(AppFragment fragmentListener) {
             this.fragmentListener = fragmentListener; }
}
```

U. **AroundMetro/resources/com/example/aroundmetro/extras/SpinnerAdapter.java**

```
package com.example.aroundmetro.extras;

import android.content.Context;
import android.util.Log;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.ArrayAdapter;
import android.widget.ImageView;
import android.widget.TextView;
import com.example.aroundmetro.R;

public class SpinnerAdapter extends ArrayAdapter<String> {
    Context context;
    String[] itinerarySummaryList;
    String[] iconList = {"R.drawable.icon1","R.drawable.icon2","R.drawable.icon3","R.drawable.icon4",
                     "R.drawable.icon5","R.drawable.icon6","R.drawable.icon7","R.drawable.icon8",
                                              "R.drawable.icon9","R.drawable.icon10"};
    public SpinnerAdapter(Context ctx, int txtViewResourceId, String[] objects) {
       super(ctx, txtViewResourceId, objects);
       itinerarySummaryList = objects;
       context = ctx;
    }

    public View getDropDownView(int position, View cnvtView, ViewGroup prnt) {
         return getCustomView(position, cnvtView, prnt);
    }
    public View getView(int pos, View cnvtView, ViewGroup prnt) {
         return getCustomView(pos, cnvtView, prnt);
    }

    public View getCustomView(int position, View convertView, ViewGroup parent) {
      LayoutInflater inflater = (LayoutInflater)context.getSystemService(Context.LAYOUT_INFLATER_SERVICE );
      View mySpinner = inflater.inflate(R.layout.custom_spinner, parent, false);
         String[] itText = itinerarySummaryList[position].split("\\|");
         TextView transferText = (TextView) mySpinner.findViewById(R.id.itinerary_transfer);
         transferText.setText(itText[0]);
         TextView fareText = (TextView) mySpinner.findViewById(R.id.itinerary_fare);
         fareText.setText(itText[1]);
         TextView durationText = (TextView) mySpinner.findViewById(R.id.itinerary_duration);
         durationText.setText(itText[2]);
         ImageView left_icon = (ImageView) mySpinner.findViewById(R.id.left_pic);
         left_icon.setImageResource(getIcon(Integer.parseInt(itText[3])));
         return mySpinner;
     }

     public int getIcon(int index){
              if(index == 0) return R.drawable.icon1;
               else if(index == 1) return R.drawable.icon2;
               else if(index == 2) return R.drawable.icon3;
               else if(index == 3) return R.drawable.icon4;
               else if(index == 4) return R.drawable.icon5;
               else if(index == 5) return R.drawable.icon6;
               else if(index == 6) return R.drawable.icon7;
               else if(index == 7) return R.drawable.icon8;
               else if(index == 8) return R.drawable.icon9;
               else if(index == 9) return R.drawable.icon10;
               return R.drawable.ic_launcher;
     }
}
```

V. **AroundMetro/resources/com/example/aroundmetro/listeners/AppFragment.java**

```
package com.example.aroundmetro.listeners;

import java.util.ArrayList;
import java.util.HashMap;
import java.util.List;
import com.example.aroundmetro.api.model.Itinerary;
```

```
import com.example.aroundmetro.api.model.Leg;
import com.example.aroundmetro.model.AppBundle;
import com.google.android.gms.maps.model.Marker;

public interface AppFragment {
     public void switchToDirectionsFragment();
     public void switchToShowDirectionsFragment();
     public void loadItineraries(List<Itinerary> itineraries);
     public List<Itinerary> getCurrentItineraryList();
     public void setAppBundle(AppBundle appBundle);
     public AppBundle getAppBundle();
     public void onItinerarySelected(int i);
     public List<Leg> getCurrentItinerary();
     public int getCurrentItineraryIndex();
     public void setTrafficInfo(HashMap<String, HashMap<String, String>> traficInfo);
     public HashMap<String, HashMap<String, String>> getTrafficInfo();
     public void addTrafficInfoMarkers(ArrayList<Marker> trafficMarkers);
     public ArrayList<Marker> getTrafficInfoMarkers();
}
```

W. AroundMetro/resources/com/example/aroundmetro/listeners/FareDataGeneratorListener.java

```
package com.example.aroundmetro.listeners;

import java.util.ArrayList;
import java.util.HashMap;

public interface FareDataGeneratorListener {
     public void onFareDataRequestComplete(ArrayList<ArrayList<HashMap<String, String>>> rails,
                                                                    ArrayList<HashMap<String, String>> jeepBus);
}
```

X. AroundMetro/resources/com/example/aroundmetro/listeners/GeocodingListener.java

```
package com.example.aroundmetro.listeners;

import java.util.ArrayList;
import android.location.Address;

public interface GeocodingListener {
     public void onGeocodingComplete(boolean isStartTextBox, ArrayList<Address> address,
                                                                          boolean geocodingForMarker);
}
```

Y. AroundMetro/resources/com/example/aroundmetro/listeners/TripRequestListener.java

```
package com.example.aroundmetro.listeners;

import java.util.List;
import com.example.aroundmetro.api.model.Itinerary;

public interface TripRequestListener {
     public void onTripRequestCompleted(List<Itinerary> itineraries);
}
```

Z. AroundMetro/resources/com/example/aroundmetro/model/AppBundle.java

```
package com.example.aroundmetro.model;

import java.io.Serializable;
import lombok.Getter;
import lombok.Setter;
import com.example.aroundmetro.api.model.Leg;

public class AppBundle implements Serializable{
     private static final long serialVersionUID = 6531154906729143078L;
     private String toText, fromText;
     @Getter @Setter private Leg leg;
     public String getToText() { return toText; }
     public void setToText(String toText) { this.toText = toText; }
     public String getFromText() { return fromText; }
     public void setFromText(String fromText) { this.fromText = fromText; }
}
```

aa. **AroundMetro/resources/com/example/aroundmetro/model/DirectionListFragment.java**

```
package com.example.aroundmetro.model;

import android.content.Context;
import android.os.Bundle;
import android.support.v4.app.Fragment;
import android.view.Gravity;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.view.animation.AnimationUtils;
import android.widget.ExpandableListAdapter;
import android.widget.ExpandableListView;
import android.widget.FrameLayout;
import android.widget.LinearLayout;
import android.widget.ListAdapter;
import android.widget.ListView;
import android.widget.ProgressBar;
import android.widget.TextView;

public class DirectionListFragment extends Fragment {
    static final int INTERNAL_EMPTY_ID = 0x00ff0001;
    static final int INTERNAL_PROGRESS_CONTAINER_ID = 0x00ff0002;
    static final int INTERNAL_LIST_CONTAINER_ID = 0x00ff0003;
    View mExpandableListContainer;
    ListAdapter mAdapter;
    ListView mList;
    View mEmptyView;
    TextView mStandardEmptyView;
    CharSequence mEmptyText;
    boolean mExpandableListShown;
    public DirectionListFragment() {  }

    public View onCreateView(LayoutInflater inflater, ViewGroup container, Bundle savedInstanceState) {
            final Context context = getActivity();
            FrameLayout root = new FrameLayout(context);
            FrameLayout lframe = new FrameLayout(context);
            lframe.setId(INTERNAL_LIST_CONTAINER_ID);
            TextView tv = new TextView(getActivity());
            tv.setId(INTERNAL_EMPTY_ID);
            tv.setGravity(Gravity.CENTER);
            lframe.addView(tv, new FrameLayout.LayoutParams(ViewGroup.LayoutParams.FILL_PARENT,
                                                            ViewGroup.LayoutParams.FILL_PARENT));
            ListView lv = new ListView(getActivity());
            lv.setId(android.R.id.list);
            lv.setDrawSelectorOnTop(false);
            lframe.addView(lv, new FrameLayout.LayoutParams(
                    ViewGroup.LayoutParams.FILL_PARENT, ViewGroup.LayoutParams.FILL_PARENT));

            root.addView(lframe, new FrameLayout.LayoutParams(
                    ViewGroup.LayoutParams.FILL_PARENT, ViewGroup.LayoutParams.FILL_PARENT));
            root.setLayoutParams(new FrameLayout.LayoutParams(
                    ViewGroup.LayoutParams.FILL_PARENT, ViewGroup.LayoutParams.FILL_PARENT));
            return root;
         }

    public void onViewCreated(View view, Bundle savedInstanceState) {
        super.onViewCreated(view, savedInstanceState);
        ensureList();
    }

    private void ensureList() {
            if (mList != null) { return; }
            View root = getView();
            if (root == null) { throw new IllegalStateException("Content view not yet created"); }
            if (root instanceof ListView) { mList = (ListView) root;}
            else {
                mStandardEmptyView = (TextView) root.findViewById(INTERNAL_EMPTY_ID);
                if (mStandardEmptyView == null) { mEmptyView = root.findViewById(android.R.id.empty); }
                 else { mStandardEmptyView.setVisibility(View.GONE); }

                mExpandableListContainer = root.findViewById(INTERNAL_LIST_CONTAINER_ID);
                View rawListView = root.findViewById(android.R.id.list);
                if (!(rawListView instanceof ListView)) {
                    if (rawListView == null) {
                        throw new RuntimeException(
                                "Your content must have a ListView whose id attribute is " +
                                        "'android.R.id.list'");
```

```
                    }
                    throw new RuntimeException(
                            "Content has view with id attribute 'android.R.id.list' "
                                    + "that is not a ListView class");
                }
                mList = (ListView) rawListView;
                if (mEmptyView != null) { mList.setEmptyView(mEmptyView); }
                 else if (mEmptyText != null) {
                    mStandardEmptyView.setText(mEmptyText);
                    mList.setEmptyView(mStandardEmptyView);
                }
            }
            if (mAdapter != null) {
                ListAdapter adapter = mAdapter;
                mAdapter = null;
                setListAdapter(adapter);
            }
        }

    public ListAdapter getListAdapter() { return mAdapter; }

    public void setListAdapter(ListAdapter adapter) {
            boolean hadAdapter = mAdapter != null;
            mAdapter = adapter;
            if (mList != null) {
                mList.setAdapter(adapter);
                if (!mExpandableListShown && !hadAdapter) { setListShown(true); }
            }
     }

     public ListView getListView() {
            ensureList();
            return mList;
     }

     private void setListShown(boolean shown) {
            ensureList();
            if (mExpandableListShown == shown) {return; }
            mExpandableListShown = shown;
            if (shown) { mExpandableListContainer.setVisibility(View.VISIBLE); }
            else { mExpandableListContainer.setVisibility(View.GONE); }
        }
}
```

bb. AroundMetro/resources/com/example/aroundmetro/model/GenerateDirections.java

```
package com.example.aroundmetro.model;

import java.text.DecimalFormat;
import java.util.ArrayList;
import java.util.List;
import android.content.Context;
import android.util.Log;
import com.example.aroundmetro.R;
import com.example.aroundmetro.api.model.Leg;
import com.example.aroundmetro.api.model.Place;
import com.example.aroundmetro.routing.core.TraverseMode;
import com.example.aroundmetro.routing.core.TraverseModeSet;
import com.example.aroundmetro.util.ConversionUtils;

public class GenerateDirections {

    private List<Leg> legs = new ArrayList<Leg>();
    private ArrayList<Direction> directions = new ArrayList<Direction>();
    private Context applicationContext;

    public GenerateDirections(List<Leg> legs, Context applicationContext) {
       this.legs.addAll(legs);
       this.applicationContext = applicationContext;
       convertDirectionToList();
    }

    public ArrayList<Direction> getDirections() { return directions; }

    private void convertDirectionToList(){
        int index=0;
        Direction dir;
        for (Leg leg : legs) {
           index++;
```

```
            dir = generateTransitDirections(leg);
            if (dir == null)  continue;
            dir.setDirectionIndex(index);
            addDirection(dir);
        }
    }

    private Direction generateTransitDirections(Leg leg){
        DecimalFormat currencyFormatter = new DecimalFormat();
        currencyFormatter.setMaximumFractionDigits(2);
        currencyFormatter.setMinimumFractionDigits(2);
        Direction direction = new Direction();
        TraverseMode mode = TraverseMode.valueOf((String) leg.mode);
        int icon = getModeIcon(new TraverseModeSet(mode));
        Place fromPlace = leg.from;
        Place toPlace = leg.to;
        direction.setDirectionMode(mode);
        direction.setIcon(icon);
        String mainDirectionText;
        if(mode.isTransit()){
          mainDirectionText = "PhP " + currencyFormatter.format(leg.fare) + "|";
          mainDirectionText += ConversionUtils.getFormattedDurationTextNoSeconds(leg.duration /
                                                                  1000,applicationContext) + "|";
          mainDirectionText += leg.route + "|";
          mainDirectionText += fromPlace.name == null ? "" : fromPlace.name + "|";
          mainDirectionText += toPlace.name == null ? "" : toPlace.name + "|";
        }
        else{
          mainDirectionText = ConversionUtils.getFormattedDistance(leg.distance,applicationContext) + "|";
          mainDirectionText += ConversionUtils.getFormattedDurationTextNoSeconds(leg.duration /
                                                                  1000,applicationContext) + "|";
          mainDirectionText += fromPlace.name == null ? "" : fromPlace.name + "|";
          mainDirectionText += toPlace.name == null ? "" : toPlace.name + "|";
        }
        direction.setDirectionText(mainDirectionText);
        return direction;
    }

    public void addDirection(Direction dir) {
        if (directions == null) { directions = new ArrayList<Direction>(); }
        directions.add(dir);
    }

    public static int getModeIcon(TraverseModeSet mode) {
        if (mode.contains(TraverseMode.BUSISH)) { return R.drawable.mode_bus; }
        else if (mode.contains(TraverseMode.TRAINISH)) { return R.drawable.mode_train; }
        else if (mode.contains(TraverseMode.TRAM)) { return R.drawable.mode_train; }
        else if (mode.contains(TraverseMode.WALK)) { return R.drawable.mode_walk; }
        else if (mode.contains(TraverseMode.JEEPNEY)) { return R.drawable.mode_jeep; }
        else { return R.drawable.icon; }
    }
}
```

CC. AroundMetro/resources/com/example/aroundmetro/tasks/TripRequest.java

```
package com.example.aroundmetro.tasks;

import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;
import java.io.UnsupportedEncodingException;
import java.lang.ref.WeakReference;
import java.net.HttpURLConnection;
import java.net.URL;
import java.util.Collection;
import java.util.HashMap;
import java.util.Iterator;
import java.util.List;
import android.annotation.SuppressLint;
import android.app.Activity;
import android.app.AlertDialog;
import android.app.ProgressDialog;
import android.content.Context;
import android.content.DialogInterface;
import android.content.DialogInterface.OnCancelListener;
import android.os.AsyncTask;
import android.util.Log;
import com.example.aroundmetro.R;
import com.example.aroundmetro.api.model.Itinerary;
```

```
import com.example.aroundmetro.api.model.PlannerError;
import com.example.aroundmetro.api.ws.Message;
import com.example.aroundmetro.api.ws.Request;
import com.example.aroundmetro.api.ws.Response;
import com.example.aroundmetro.listeners.TripRequestListener;
import com.fasterxml.jackson.databind.ObjectMapper;

public class TripRequest extends AsyncTask<Request, Integer, Response > {

    private Response plan = null;
    private List<Itinerary> itineraries;
    private TripRequestListener callback;
    private WeakReference<Activity> activity;
    private Context context;
    private ProgressDialog progressDialog;

    public TripRequest(WeakReference<Activity> activity, Context context, TripRequestListener callback){
       this.callback = callback;
       this.activity = activity;
       this.context = context;

       final Activity activityRetrieved = activity.get();
       if (activityRetrieved != null) {
             progressDialog = new ProgressDialog(activityRetrieved);
             progressDialog.setCancelable(true);
             progressDialog.setOnCancelListener(new OnCancelListener() {

                public void onCancel(DialogInterface dialog) {
                      try {
                         if (progressDialog != null && progressDialog.isShowing())
                                progressDialog.dismiss();
                      } catch (Exception e) {
                         Log.e("onCancelled", "Error in TripRequest Cancelled dismissing dialog: " + e);
                      }

                      if (activityRetrieved != null) {
                      AlertDialog.Builder geocoderAlert = new AlertDialog.Builder(activityRetrieved);
                      geocoderAlert.setTitle(R.string.tripplanner_result_title)
                                        .setMessage(R.string.tripplanner_error_request_timeout)
                                        .setCancelable(false)
                                        .setPositiveButton(android.R.string.ok, new
                                                                        DialogInterface.OnClickListener() {
                                                  public void onClick(DialogInterface dialog, int id) {}
                                                });

                       AlertDialog alert = geocoderAlert.create();
                       alert.show();
                   }
                }
             });
         }
    }

    protected void onPreExecute(){
             if (activity.get() != null) {
                      progressDialog.setIndeterminate(true);
                      progressDialog.setCancelable(true);
                      Activity activityRetrieved = activity.get();
                      if (activityRetrieved != null) {
                               progressDialog = ProgressDialog.show(activityRetrieved, "",
                                        context.getText(R.string.tripplanner_progress), true);
                      }
             }
    }

    @SuppressLint("UseValueOf")
    @SuppressWarnings("rawtypes")
    @Override
    protected Response doInBackground(Request... params) {
       Request req = params[0];
       HashMap<String, String> temp = req.getParameters();
       String sample = "";
       Collection c = temp.entrySet();
       Iterator itr = c.iterator();
       String parameters = "";
       while (itr.hasNext()) { parameters += "&" + itr.next(); }

       String baseUrl ="http://agila.upm.edu.ph:11076/opentripplanner-api-webapp/ws/plan?";
       String u = baseUrl + parameters;
```

```
            HttpURLConnection urlConnection = null;
            URL url;

            try {
               url = new URL(u);
               urlConnection = (HttpURLConnection) url.openConnection();
               urlConnection.setRequestProperty("Accept", "application/json");
               BufferedReader in = new BufferedReader(new InputStreamReader(urlConnection.getInputStream()));

               String line;
               StringBuffer sb = new StringBuffer();
               while ((line = in.readLine()) != null) { sb.append(line); }
               sample = sb.toString();
               ObjectMapper mapper = new ObjectMapper();
               plan = mapper.readValue(sample, Response.class);
            }
            catch (UnsupportedEncodingException e) { e.printStackTrace();}
            catch (IOException e) { e.printStackTrace(); }
            return plan;
         }

         protected void onPostExecute(Response result) {
            if (activity.get() != null) {
               try {
                    if (progressDialog != null && progressDialog.isShowing()) { progressDialog.dismiss(); }
                } catch (Exception e) {
                    Log.e("onPostExecute", "Error in TripRequest PostExecute dismissing dialog: " + e);
                }
            }

            try{
               itineraries = result.getPlan().getItinerary();
               callback.onTripRequestCompleted(itineraries);
            }
            catch (NullPointerException e){
                  Activity retrievedActivity = activity.get();
                  AlertDialog.Builder serverResponse = new AlertDialog.Builder(retrievedActivity);
                  serverResponse.setTitle(context.getResources().
                            getString(R.string.tripplanner_error_dialog_title));
                  serverResponse.setNeutralButton("OK", null);
                  String msg = "";
                  PlannerError err = result.getError();
                  if(err != null){
                     int errCode = err.getId();
                     if(result != null && result.getError() != null && errCode != Message.PLAN_OK.getId()){
                           msg = getErrorMessage(errCode);
                     }
                  }
                  serverResponse.setMessage(msg);
                  serverResponse.create().show();
            }
         }

         private String getErrorMessage(int errCode) {
            if(errCode == Message.SYSTEM_ERROR.getId()){
                  return (context.getResources().getString(R.string.tripplanner_error_system));
            } else if(errCode == Message.OUTSIDE_BOUNDS.getId()){
                  return (context.getResources().getString(R.string.tripplanner_error_outside_bounds));
            } else if(errCode == Message.PATH_NOT_FOUND.getId()){
                  return (context.getResources().getString(R.string.tripplanner_error_path_not_found));
            } else if(errCode == Message.REQUEST_TIMEOUT.getId()){
                  return (context.getResources().getString(R.string.tripplanner_error_request_timeout));
            } else if(errCode == Message.NO_TRANSIT_TIMES.getId()){
                  return (context.getResources().getString(R.string.tripplanner_error_no_transit_times));
            } else if(errCode == Message.BOGUS_PARAMETER.getId()){
                  return (context.getResources().getString(R.string.tripplanner_error_bogus_parameter));
            } else if(errCode == Message.TOO_CLOSE.getId()){
                  return (context.getResources().getString(R.string.tripplanner_error_too_close));
            } else if(errCode == Message.LOCATION_NOT_ACCESSIBLE.getId()){
                 return (context.getResources().getString(R.string.tripplanner_error_location_not_accessible));
            } else if(errCode == Message.GEOCODE_FROM_AMBIGUOUS.getId()){
                  return (context.getResources().getString(R.string.tripplanner_error_geocode_from_ambiguous));
            } else if(errCode == Message.GEOCODE_TO_AMBIGUOUS.getId()){
                  return (context.getResources().getString(R.string.tripplanner_error_geocode_to_ambiguous));
            } else if(errCode == Message.GEOCODE_FROM_TO_AMBIGUOUS.getId()){
               return (context.getResources().getString(R.string.tripplanner_error_geocode_from_to_ambiguous));
            } else{ return null;  }
         }
      }
```

dd. AroundMetro/resources/com/example/aroundmetro/util/ProcessJson.java

```
package com.example.aroundmetro.util;

import com.example.aroundmetro.api.ws.Response;
import com.fasterxml.jackson.databind.DeserializationFeature;
import com.fasterxml.jackson.databind.ObjectMapper;
import com.fasterxml.jackson.databind.ObjectReader;

public class ProcessJson {
    private static ObjectMapper mapper = null;
    private static ObjectReader reader = null;
    private ProcessJson(){ }

    public synchronized static ObjectReader getObjectReaderInstance() {
        if (reader == null) { reader = initObjectMapper().reader(Response.class); }
        return reader;
    }

    private static ObjectMapper initObjectMapper() {
        if (mapper == null) {
            mapper = new ObjectMapper();
            mapper.configure(DeserializationFeature.ACCEPT_SINGLE_VALUE_AS_ARRAY, true);
            mapper.configure(DeserializationFeature.ACCEPT_EMPTY_STRING_AS_NULL_OBJECT, true);
            mapper.configure(DeserializationFeature.USE_JAVA_ARRAY_FOR_JSON_ARRAY, true);
            mapper.configure(DeserializationFeature.READ_ENUMS_USING_TO_STRING, true);
            mapper.configure(DeserializationFeature.FAIL_ON_UNKNOWN_PROPERTIES, false);
        }
        return mapper;
    }
}
```

2. Trip Planner (Server Side)

a. opentripplanner-routing/src/main/java/org/opentripplanner/routing/core/RoutingRequest.java

```
package org.opentripplanner.routing.core;

import java.io.Serializable;
import java.util.ArrayList;
import java.util.Arrays;
import java.util.Date;
import java.util.HashMap;
import java.util.HashSet;
import java.util.List;
import java.util.Locale;
import java.util.Map;
import java.util.TimeZone;
import lombok.Getter;
import lombok.Setter;
import org.onebusaway.gtfs.model.AgencyAndId;
import org.onebusaway.gtfs.model.Route;
import org.onebusaway.gtfs.model.Trip;
import org.opentripplanner.common.MavenVersion;
import org.opentripplanner.common.model.GenericLocation;
import org.opentripplanner.common.model.NamedPlace;
import org.opentripplanner.gtfs.GtfsLibrary;
import org.opentripplanner.routing.graph.Edge;
import org.opentripplanner.routing.graph.Graph;
import org.opentripplanner.routing.graph.Vertex;
import org.opentripplanner.routing.request.BannedStopSet;
import org.opentripplanner.util.DateUtils;
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;

@Getter @Setter
public class RoutingRequest implements Cloneable, Serializable {

    private static final long serialVersionUID = MavenVersion.VERSION.getUID();
    private static final Logger LOG = LoggerFactory.getLogger(RoutingRequest.class);
    private static final int CLAMP_ITINERARIES = 20;
    @Setter
    private IntersectionTraversalCostModel traversalCostModel = new SimpleIntersectionTraversalCostModel();
    public String routerId = "";
    @Getter @Setter
    public GenericLocation from;
    @Getter @Setter
    public GenericLocation to;
```

```
@Getter @Setter
public List<GenericLocation> intermediatePlaces;
public boolean intermediatePlacesOrdered;
public double maxWalkDistance = 1000;
public long worstTime = Long.MAX_VALUE;
public double maxWeight = Double.MAX_VALUE;
public TraverseModeSet modes = new TraverseModeSet("TRANSIT,WALK"); // defaults in constructor
public OptimizeType optimize = OptimizeType.QUICK;
public long dateTime = new Date().getTime() / 1000;
public boolean arriveBy = false;
public boolean wheelchairAccessible = false;
public int numItineraries = 20;
public double fareMultiplier = 0.0;
public double distMultiplier = 0.0;
public double timeMultiplier = 0.0;
public double walkMultiplier = 0.0;
public int priority = 0;
public double maxSlope = 0.0833333333333; // ADA max wheelchair ramp slope is a good default.
public boolean showIntermediateStops = false;
private double walkSpeed;
private double railSpeed;
private double busSpeed;
private Locale locale = new Locale("en", "US");
public int transferPenalty = 500;
public double walkReluctance = 2.0;
public double stairsReluctance = 2.0;
@Setter
public double turnReluctance = 1.0;
public double waitReluctance = 0.95;
public double waitAtBeginningFactor = 0.2;
protected int walkBoardCost = 60 * 10;
public RouteMatcher bannedRoutes = RouteMatcher.emptyMatcher();
public HashSet<String> bannedAgencies = new HashSet<String>();
public HashMap<AgencyAndId, BannedStopSet> bannedTrips = new HashMap<AgencyAndId, BannedStopSet>();
public RouteMatcher preferredRoutes = RouteMatcher.emptyMatcher();
public HashSet<String> preferredAgencies = new HashSet<String>();
public int otherThanPreferredRoutesPenalty = 300;
protected int railBoardCost = 60 * 2;
protected int busBoardCost = 60 * 5;
public RouteMatcher unpreferredRoutes = RouteMatcher.emptyMatcher();
public HashSet<String> unpreferredAgencies = new HashSet<String>();
public int useUnpreferredRoutesPenalty = 300;
private int transferSlack = 0;
private int boardSlack = 0;
private int alightSlack = 0;
public int maxTransfers = 2;
public Map<Object, Object> extensions = new HashMap<Object, Object>();
public int nonpreferredTransferPenalty = 180;
public boolean reverseOptimizing = false;
public boolean batch = false;
public RoutingContext rctx;
private AgencyAndId startingTransitStopId;

public RoutingRequest() {
    walkSpeed = 1.33; // 1.33 m/s ~ 3mph, avg. human speed
    railSpeed = 30;
    busSpeed = 15;
    setModes(new TraverseModeSet(new TraverseMode[] { TraverseMode.WALK, TraverseMode.TRANSIT }));
    bikeWalkingOptions = this;

    from = new GenericLocation();
    to = new GenericLocation();
}

public RoutingRequest(TraverseModeSet modes) {
    this();
    this.setModes(modes);
}

public RoutingRequest(TraverseMode mode) {
    this();
    this.setModes(new TraverseModeSet(mode));
}

public RoutingRequest(TraverseMode mode, OptimizeType optimize) {
    this(new TraverseModeSet(mode), optimize);
}

public RoutingRequest(TraverseModeSet modeSet, OptimizeType optimize) {
```

```
    this();
    this.optimize = optimize;
    this.setModes(modeSet);
}

public boolean transitAllowed() { return modes.isTransit(); }

public long getSecondsSinceEpoch() { return dateTime; }

public void setArriveBy(boolean arriveBy) {
    this.arriveBy = arriveBy;
    bikeWalkingOptions.arriveBy = arriveBy;
    if (worstTime == Long.MAX_VALUE || worstTime == 0)
        worstTime = arriveBy ? 0 : Long.MAX_VALUE;
}

public void setMultipliers(){
 if(priority == 1){                  //priority is least fare
          fareMultiplier = 0.2;
          timeMultiplier = 0.3;
          distMultiplier = 0.5;
 }
 else if(priority == 2){             //priority is least time
          fareMultiplier = 0.5;
          timeMultiplier = 0.2;
          distMultiplier = 0.3;
 }
 else if(priority == 3){             //priority is least distance
          fareMultiplier = 0.5;
          timeMultiplier = 0.3;
          distMultiplier = 0.2;
 }
}

public void setMode(TraverseMode mode) {
    setModes(new TraverseModeSet(mode));
}

public void setOptimize(OptimizeType optimize) {
    this.optimize = optimize;
    bikeWalkingOptions.optimize = optimize;
}

public void putExtension(Object key, Object value) {
    extensions.put(key, value);
}

public boolean containsExtension(Object key) {
    return extensions.containsKey(key);
}

@SuppressWarnings("unchecked")
public <T> T getExtension(Object key) {
    return (T) extensions.get(key);
}

public IntersectionTraversalCostModel getIntersectionTraversalCostModel() {
    return traversalCostModel;
}

public double getMaxWalkDistance() {
    if (!getModes().isTransit()) return Double.MAX_VALUE;
    else return maxWalkDistance;
}

public void setPreferredAgencies(String s) {
    if (s != null && !s.equals(""))
        preferredAgencies = new HashSet<String>(Arrays.asList(s.split(",")));
}

public void setPreferredRoutes(String s) {
    if (s != null && !s.equals("")) preferredRoutes = RouteMatcher.parse(s);
    else preferredRoutes = RouteMatcher.emptyMatcher();
}

public void setOtherThanPreferredRoutesPenalty(int penalty) {
    if(penalty < 0) penalty = 0;
    this.otherThanPreferredRoutesPenalty = penalty;
}
```

```
public void setUnpreferredAgencies(String s) {
    if (s != null && !s.equals(""))
        unpreferredAgencies = new HashSet<String>(Arrays.asList(s.split(",")));
}

public void setUnpreferredRoutes(String s) {
    if (s != null && !s.equals(""))
        unpreferredRoutes = RouteMatcher.parse(s);
    else
        unpreferredRoutes = RouteMatcher.emptyMatcher();
}

public void setBannedRoutes(String s) {
    if (s != null && !s.equals("")) bannedRoutes = RouteMatcher.parse(s);
    else bannedRoutes = RouteMatcher.emptyMatcher();
}

public void setBannedAgencies(String s) {
    if (s != null && !s.equals("")) bannedAgencies = new HashSet<String>(Arrays.asList(s.split(",")));
}

public void setFromString(String from) {
    this.from = GenericLocation.fromOldStyleString(from);
}

public void setToString(String to) {
    this.to = GenericLocation.fromOldStyleString(to);
}

public void clearModes() { modes.clear(); }

public void addMode(TraverseMode mode) { modes.setMode(mode, true); }

public void addMode(List<TraverseMode> mList) {
    for (TraverseMode m : mList) { addMode(m); }
}

public Date getDateTime() { return new Date(dateTime * 1000); }

public void setDateTime(Date dateTime) { this.dateTime = dateTime.getTime() / 1000; }

public void setDateTime(String date, String time, TimeZone tz) {
    Date dateObject = DateUtils.toDate(date, time, tz);
    setDateTime(dateObject);
}

public int getNumItineraries() {
    if (getModes().isTransit()) { return numItineraries; }
    else return 1;
}

public void setNumItineraries(int numItineraries) {
    if (numItineraries > CLAMP_ITINERARIES) {
        numItineraries = CLAMP_ITINERARIES;
    } else if (numItineraries < 1) {
        numItineraries = 1;
    }
    this.numItineraries = numItineraries;
}

public String toHtmlString() { return toString("<br/>"); }

public String toString() { return toString(" "); }

public String toString(String sep) {
    return getFrom() + sep + getTo() + sep + getMaxWalkDistance() + sep + getDateTime() + sep
            + isArriveBy() + sep + getOptimize() + sep + modes.getAsStr() + sep
            + getNumItineraries();
}

public void removeMode(TraverseMode mode) { modes.setMode(mode, false); }

public void setIntermediatePlacesFromStrings(List<String> intermediates) {
    this.intermediatePlaces = new ArrayList<GenericLocation>(intermediates.size());
    for (String place : intermediates) {
        intermediatePlaces.add(GenericLocation.fromOldStyleString(place));
    }
}
```

```
public void clearIntermediatePlaces() {
    if (this.intermediatePlaces != null) {
        this.intermediatePlaces.clear();
    }
}

public boolean hasIntermediatePlaces() {
    return this.intermediatePlaces != null && this.intermediatePlaces.size() > 0;
}

public void addIntermediatePlace(GenericLocation location) {
    if (this.intermediatePlaces == null) {
        this.intermediatePlaces = new ArrayList<GenericLocation>();
    }
    this.intermediatePlaces.add(location);
}

public boolean intermediatesEffectivelyOrdered() {
    boolean exactlyOneIntermediate = (this.intermediatePlaces != null
                                            && this.intermediatePlaces.size() == 1);
    return this.intermediatePlacesOrdered || exactlyOneIntermediate;
}

public void setMaxTransfers(int maxTransfers) { this.maxTransfers = maxTransfers; }

public NamedPlace getFromPlace() { return this.from.getNamedPlace(); }

public NamedPlace getToPlace() { return this.to.getNamedPlace(); }

@SuppressWarnings("unchecked")
@Override
public RoutingRequest clone() {
    try {
        RoutingRequest clone = (RoutingRequest) super.clone();
        clone.bannedRoutes = bannedRoutes.clone();
        clone.bannedTrips = (HashMap<AgencyAndId, BannedStopSet>) bannedTrips.clone();
        return clone;
    } catch (CloneNotSupportedException e) {
        throw new RuntimeException(e);
    }
}

public RoutingRequest reversedClone() {
    RoutingRequest ret = this.clone();
    ret.setArriveBy(!ret.isArriveBy());
    ret.reverseOptimizing = !ret.reverseOptimizing; // this is not strictly correct
    ret.useBikeRentalAvailabilityInformation = false;
    return ret;
}

public void setRoutingContext(Graph graph) {
    if (rctx == null) {
        this.rctx = new RoutingContext(this, graph);
        this.rctx.check();
    } else {
        if (rctx.graph == graph) {
            LOG.debug("keeping existing routing context");
            return;
        } else {
            LOG.error("attempted to reset routing context using a different graph");
            return;
        }
    }
}

public void setRoutingContext(Graph graph, Vertex from, Vertex to) {
    setRoutingContext(graph, null, from, to);
}

public RoutingContext getRoutingContext() { return this.rctx; }

@Override
public boolean equals(Object o) {
    if (!(o instanceof RoutingRequest))
        return false;
    RoutingRequest other = (RoutingRequest) o;
    if (this.batch != other.batch)
        return false;
    boolean endpointsMatch;
```

```
    if (this.batch) {
        if (this.arriveBy) {
            endpointsMatch = to.equals(other.to);
        } else {
            endpointsMatch = from.equals(other.from);
        }
    } else {
        endpointsMatch = ((from == null && other.from == null) || from.equals(other.from))
                && ((to == null && other.to == null) || to.equals(other.to));
    }
    return endpointsMatch
            && dateTime == other.dateTime
            && isArriveBy() == other.isArriveBy()
            && numItineraries == other.numItineraries // should only apply in non-batch?
            && walkSpeed == other.walkSpeed
            && railSpeed == other.railSpeed
            && busSpeed == other.busSpeed
            && maxWeight == other.maxWeight
            && worstTime == other.worstTime
            && maxTransfers == other.maxTransfers
            && fareMultiplier == other.fareMultiplier
            && distMultiplier== other.distMultiplier
            && timeMultiplier == other.timeMultiplier
            && walkMultiplier == other.walkMultiplier
            && getModes().equals(other.getModes())
            && wheelchairAccessible == other.wheelchairAccessible
            && optimize.equals(other.optimize)
            && maxWalkDistance == other.maxWalkDistance
            && transferPenalty == other.transferPenalty
            && maxSlope == other.maxSlope
            && walkReluctance == other.walkReluctance
            && waitReluctance == other.waitReluctance
            && walkBoardCost == other.walkBoardCost
            && busBoardCost == other.busBoardCost
            && railBoardCost == other.railBoardCost
            && bannedRoutes.equals(other.bannedRoutes)
            && bannedTrips.equals(other.bannedTrips)
            && preferredRoutes.equals(other.preferredRoutes)
            && unpreferredRoutes.equals(other.unpreferredRoutes)
            && transferSlack == other.transferSlack
            && boardSlack == other.boardSlack
            && alightSlack == other.alightSlack
            && nonpreferredTransferPenalty == other.nonpreferredTransferPenalty
            && extensions.equals(other.extensions)
            && clampInitialWait == other.clampInitialWait
}

public void cleanup() {
    if (this.rctx == null)
        LOG.warn("routing context was not set, cannot destroy it.");
    else {
        int nRemoved = this.rctx.destroy();
        LOG.debug("routing context destroyed ({} temporary edges removed)", nRemoved);
    }
}

public double getSpeed(TraverseMode mode) {
    switch (mode) {
    case WALK:
        return walkSpeed;
    case RAIL:
          return railSpeed;
    case BUS:
          return busSpeed;
    default:
        break;
    }
    throw new IllegalArgumentException("getSpeed(): Invalid mode " + mode);
}

public double getStreetSpeedUpperBound() {
    if(modes.getBus())
          return busSpeed;
    if(modes.getRail())
          return railSpeed;
    return walkSpeed;
}
```

```
public int getBoardCost(TraverseMode mode) {
    if (mode == TraverseMode.RAIL)
          return railBoardCost;
    else if(mode == TraverseMode.BUS)
          return busBoardCost;

    return walkBoardCost;
}

public int getBoardCostLowerBound() {
    if (modes.getWalk())
        return walkBoardCost;
    else if(modes.getRail())
          return railBoardCost;
    else if(modes.getBus())
          return busBoardCost;
    return bikeBoardCost;
}

private String getRouteOrAgencyStr(HashSet<String> strings) {
 StringBuilder builder = new StringBuilder();
    for (String agency : strings) {
        builder.append(agency);
        builder.append(",");
    }
    if (builder.length() > 0) {
        // trim trailing comma
        builder.setLength(builder.length() - 1);
    }
    return builder.toString();
}

public String getPreferredRouteStr() {
    return preferredRoutes.asString();
}

public String getPreferredAgenciesStr() {
 return getRouteOrAgencyStr(preferredAgencies);
}

public String getUnpreferredRouteStr() {
    return unpreferredRoutes.asString();
}

public String getUnpreferredAgenciesStr() {
 return getRouteOrAgencyStr(unpreferredAgencies);
}

public String getBannedRouteStr() {
    return bannedRoutes.asString();
}

public String getBannedAgenciesStr() {
 return getRouteOrAgencyStr(bannedAgencies);
}

public void setMaxWalkDistance(double maxWalkDistance) {
    if (maxWalkDistance == 0)
        return;
    this.maxWalkDistance = maxWalkDistance;
    if (bikeWalkingOptions != null && bikeWalkingOptions != this) {
        this.bikeWalkingOptions.setMaxWalkDistance(maxWalkDistance);
    }
}

public void banTrip(AgencyAndId trip) {
    bannedTrips.put(trip, BannedStopSet.ALL);
}

public boolean tripIsBanned(Trip trip) {
    if (bannedAgencies != null) {
        if (bannedAgencies.contains(trip.getId().getAgencyId())) {
            return true;
        }
    }

    if (bannedRoutes != null) {
        Route route = trip.getRoute();
        if (bannedRoutes.matches(route)) {
```

```
                return true;
            }
        }

        return false;
    }

    public long preferencesPenaltyForTrip(Trip trip) {
     long preferences_penalty = 0;

     Route route = trip.getRoute();
     String agencyID = route.getId().getAgencyId();

     if (preferredRoutes != null || (preferredAgencies != null && !preferredAgencies.isEmpty())) {
              boolean isPreferedRoute = preferredRoutes != null && preferredRoutes.matches(route);
              boolean isPreferedAgency = preferredAgencies != null && preferredAgencies.contains(agencyID);
              if (!isPreferedRoute && !isPreferedAgency) {
                       preferences_penalty += otherThanPreferredRoutesPenalty;
              }
              else {
                       preferences_penalty = 0;
              }
     }

     boolean isUnpreferedRoute = unpreferredRoutes != null && unpreferredRoutes.matches(route);
     boolean isUnpreferedAgency = unpreferredAgencies != null && unpreferredAgencies.contains(agencyID);
     if (isUnpreferedRoute || isUnpreferedAgency) {
              preferences_penalty += useUnpreferredRoutesPenalty;
     }

     return preferences_penalty;
    }
}
```

b. opentripplanner-routing/src/main/java/org/opentripplanner/routing/core/RoutingContext.java

```
package org.opentripplanner.routing.core;

import java.util.ArrayList;
import java.util.HashMap;
import java.util.HashSet;
import java.util.Map;
import java.util.Set;

import org.onebusaway.gtfs.model.AgencyAndId;
import org.onebusaway.gtfs.model.calendar.ServiceDate;
import org.onebusaway.gtfs.services.calendar.CalendarService;
import org.opentripplanner.common.geometry.DistanceLibrary;
import org.opentripplanner.common.geometry.GeometryUtils;
import org.opentripplanner.common.geometry.SphericalDistanceLibrary;
import org.opentripplanner.common.model.GenericLocation;
import org.opentripplanner.routing.algorithm.strategies.RemainingWeightHeuristic;
import org.opentripplanner.routing.algorithm.strategies.TrivialRemainingWeightHeuristic;
import org.opentripplanner.routing.edgetype.PartialPlainStreetEdge;
import org.opentripplanner.routing.edgetype.PlainStreetEdge;
import org.opentripplanner.routing.edgetype.TimetableResolver;
import org.opentripplanner.routing.error.TransitTimesException;
import org.opentripplanner.routing.error.VertexNotFoundException;
import org.opentripplanner.routing.graph.Edge;
import org.opentripplanner.routing.graph.Graph;
import org.opentripplanner.routing.graph.Vertex;
import org.opentripplanner.routing.impl.DefaultRemainingWeightHeuristicFactoryImpl;
import org.opentripplanner.routing.location.StreetLocation;
import org.opentripplanner.routing.pathparser.PathParser;
import org.opentripplanner.routing.services.RemainingWeightHeuristicFactory;
import org.opentripplanner.routing.services.TransitIndexService;
import org.opentripplanner.routing.vertextype.StreetVertex;
import org.opentripplanner.routing.vertextype.TransitStop;
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import com.google.common.collect.Iterables;
import com.vividsolutions.jts.geom.Geometry;
import com.vividsolutions.jts.geom.LineString;

/**
 * A RoutingContext holds information needed to carry out a search for a particular TraverseOptions, on a
specific graph.
* @author abyrd
*/
```

```
public class RoutingContext implements Cloneable {

    private static final Logger LOG = LoggerFactory.getLogger(RoutingContext.class);

    private static RemainingWeightHeuristicFactory heuristicFactory = new
                                                    DefaultRemainingWeightHeuristicFactoryImpl();
    public RoutingRequest opt; // not final so we can reverse-clone
    public final Graph graph;
    public final Vertex fromVertex;
    public final Vertex toVertex;
    public final Vertex origin;
    public final Vertex target;
    public Edge originBackEdge;
    public final ArrayList<Vertex> intermediateVertices = new ArrayList<Vertex>();
    public final CalendarService calendarService;
    public final Map<AgencyAndId, Set<ServiceDate>> serviceDatesByServiceId =
                                                    new HashMap<AgencyAndId, Set<ServiceDate>>();

    public RemainingWeightHeuristic remainingWeightHeuristic;
    public final TransferTable transferTable;
    public final TimetableResolver timetableSnapshot;
    public ArrayList<ServiceDay> serviceDays;
    public long searchAbortTime = 0;
    public PathParser[] pathParsers = new PathParser[] {};
    public Vertex startingStop;

    public RoutingContext(RoutingRequest routingRequest, Graph graph) {
        this(routingRequest, graph, null, null, true);
    }

    public RoutingContext(RoutingRequest routingRequest, Graph graph, Vertex from, Vertex to) {
        this(routingRequest, graph, from, to, false);
    }

    private Set<PlainStreetEdge> overlappingPlainStreetEdges(Vertex u, Vertex v) {
        Set<Integer> vIds = new HashSet<Integer>();
        Set<Integer> uIds = new HashSet<Integer>();
        for (Edge e : Iterables.concat(v.getIncoming(), v.getOutgoing())) {
            vIds.add(e.getId());
        }
        for (Edge e : Iterables.concat(u.getIncoming(), u.getOutgoing())) {
            uIds.add(e.getId());
        }

        uIds.retainAll(vIds);
        Set<Integer> overlappingIds = uIds;
        Set<PlainStreetEdge> overlap = new HashSet<PlainStreetEdge>();
        for (Integer id : overlappingIds) {
            Edge e = graph.getEdgeById(id);
            if (e == null || !(e instanceof PlainStreetEdge)) { continue; }
            overlap.add((PlainStreetEdge) e);
        }
        return overlap;
    }

    private PartialPlainStreetEdge makePartialEdgeAlong(PlainStreetEdge e,StreetVertex u,StreetVertex v) {
        DistanceLibrary dLib = SphericalDistanceLibrary.getInstance();

        Vertex head = e.getFromVertex();
        double uDist = dLib.fastDistance(head.getCoordinate(), u.getCoordinate());
        double vDist = dLib.fastDistance(head.getCoordinate(), v.getCoordinate());

        StreetVertex first = u;
        StreetVertex second = v;
        if (vDist < uDist) {
            first = v;
            second = u;
        }

        Geometry parentGeom = e.getGeometry();
        LineString myGeom = GeometryUtils.getInteriorSegment(parentGeom, first.getCoordinate(),
                second.getCoordinate());

        double lengthRatio = myGeom.getLength() / parentGeom.getLength();
        double length = e.getLength() * lengthRatio;

        String name = first.getLabel() + " to " + second.getLabel();
        return new PartialPlainStreetEdge(e, first, second, myGeom, name, length);
    }
```

```
private RoutingContext(RoutingRequest routingRequest, Graph graph, Vertex from, Vertex to,
        boolean findPlaces) {
    this.opt = routingRequest;
    this.graph = graph;

    Edge fromBackEdge = null;
    Edge toBackEdge = null;
    if (findPlaces) {
        if (!opt.batch || opt.arriveBy) {
            toVertex = graph.streetIndex.getVertexForLocation(opt.getTo(), opt);
            if (opt.getTo().hasEdgeId()) {
                toBackEdge = graph.getEdgeById(opt.getTo().getEdgeId());
            }
        } else {
            toVertex = null;
        }
        if (!opt.batch || !opt.arriveBy) {
            fromVertex = graph.streetIndex.getVertexForLocation(opt.getFrom(), opt, toVertex);
            if (opt.getFrom().hasEdgeId()) {
                fromBackEdge = graph.getEdgeById(opt.getFrom().getEdgeId());
            }
        } else {
            fromVertex = null;
        }
        if (opt.intermediatePlaces != null) {
            for (GenericLocation intermediate : opt.intermediatePlaces) {
                Vertex vertex = graph.streetIndex.getVertexForLocation(intermediate, opt);
                intermediateVertices.add(vertex);
            }
        }
    } else {
        fromVertex = from;
        toVertex = to;
    }
    if (fromVertex instanceof StreetLocation && toVertex instanceof StreetLocation) {
        StreetVertex fromStreetVertex = (StreetVertex) fromVertex;
        StreetVertex toStreetVertex = (StreetVertex) toVertex;
        Set<PlainStreetEdge> overlap = overlappingPlainStreetEdges(fromStreetVertex,
                toStreetVertex);

        for (PlainStreetEdge pse : overlap) {
            makePartialEdgeAlong(pse, fromStreetVertex, toStreetVertex);
        }
    }

    if (opt.getStartingTransitStopId() != null) {
        TransitIndexService tis = graph.getService(TransitIndexService.class);
        if (tis == null) {
            throw new RuntimeException("Next/Previous/First/Last trip "
                    + "functionality depends on the transit index. Rebuild "
                    + "the graph with TransitIndexBuilder");
        }
        AgencyAndId stopId = opt.getStartingTransitStopId();
        startingStop = tis.getPreBoardEdge(stopId).getToVertex();
    }
    origin = opt.arriveBy ? toVertex : fromVertex;
    originBackEdge = opt.arriveBy ? toBackEdge : fromBackEdge;
    target = opt.arriveBy ? fromVertex : toVertex;
    calendarService = graph.getCalendarService();
    transferTable = graph.getTransferTable();
    if (graph.timetableSnapshotSource != null)
        timetableSnapshot = graph.timetableSnapshotSource.getSnapshot();
    else
        timetableSnapshot = null;
    setServiceDays();
    if (opt.batch)
        remainingWeightHeuristic = new TrivialRemainingWeightHeuristic();
    else
        remainingWeightHeuristic = heuristicFactory.getInstanceForSearch(opt);

    if (this.origin != null) {
        LOG.debug("Origin vertex inbound edges {}", this.origin.getIncoming());
        LOG.debug("Origin vertex outbound edges {}", this.origin.getOutgoing());
    }
    LOG.debug("Target vertex {}", this.target);
    if (this.target != null) {
        LOG.debug("Destination vertex inbound edges {}", this.target.getIncoming());
        LOG.debug("Destination vertex outbound edges {}", this.target.getOutgoing());
    }
```

```
    }

    public void check() {
        ArrayList<String> notFound = new ArrayList<String>();
        if (!(opt.batch && opt.arriveBy))
            if (fromVertex == null)
                notFound.add("from");
        if (!opt.batch || opt.arriveBy)
            if (toVertex == null)
                notFound.add("to");
        for (int i = 0; i < intermediateVertices.size(); i++) {
            if (intermediateVertices.get(i) == null) {
                notFound.add("intermediate." + i);
            }
        }
        if (notFound.size() > 0) {
            throw new VertexNotFoundException(notFound);
        }
        if (opt.getModes().isTransit() && !graph.transitFeedCovers(opt.dateTime)) {
            throw new TransitTimesException();
        }
    }

    public void setServiceDays() {
        final long SEC_IN_DAY = 60 * 60 * 24;
        final long time = opt.getSecondsSinceEpoch();
        this.serviceDays = new ArrayList<ServiceDay>(3);
        if (calendarService == null && graph.getCalendarService() != null
                && (opt.getModes() == null || opt.getModes().contains(TraverseMode.TRANSIT))) {
            LOG.warn("RoutingContext has no CalendarService. Transit will never be boarded.");
            return;
        }
        for (String agency : graph.getAgencyIds()) {
            addIfNotExists(this.serviceDays, new ServiceDay(graph, time - SEC_IN_DAY,
                    calendarService, agency));
            addIfNotExists(this.serviceDays, new ServiceDay(graph, time, calendarService, agency));
            addIfNotExists(this.serviceDays, new ServiceDay(graph, time + SEC_IN_DAY,
                    calendarService, agency));
        }
    }

    private static <T> void addIfNotExists(ArrayList<T> list, T item) {
        if (!list.contains(item)) {
            list.add(item);
        }
    }

    public boolean isWheelchairAccessible(Vertex v) {
        if (v instanceof TransitStop) {
            TransitStop ts = (TransitStop) v;
            return ts.hasWheelchairEntrance();
        } else if (v instanceof StreetLocation) {
            StreetLocation sl = (StreetLocation) v;
            return sl.isWheelchairAccessible();
        }
        return true;
    }

    public int destroy() {
        int nRemoved = 0;
        if (origin != null)
            nRemoved += origin.removeTemporaryEdges();
        if (target != null)
            nRemoved += target.removeTemporaryEdges();
        for (Vertex v : intermediateVertices)
            nRemoved += v.removeTemporaryEdges();
        return nRemoved;
    }

}
```

c. opentripplanner-routing/src/main/java/org/opentripplanner/routing/core/State.java

```
package org.opentripplanner.routing.core;

import java.util.Arrays;
import java.util.Date;
import java.util.Set;
```

```
import org.onebusaway.gtfs.model.AgencyAndId;
import org.onebusaway.gtfs.model.Trip;
import org.opentripplanner.routing.algorithm.NegativeWeightException;
import org.opentripplanner.routing.automata.AutomatonState;
import org.opentripplanner.routing.edgetype.OnBoardForwardEdge;
import org.opentripplanner.routing.edgetype.TablePatternEdge;
import org.opentripplanner.routing.edgetype.PlainStreetEdge;
import org.opentripplanner.routing.edgetype.StreetEdge;
import org.opentripplanner.routing.edgetype.TransitBoardAlight;
import org.opentripplanner.routing.edgetype.TripPattern;
import org.opentripplanner.routing.graph.Edge;
import org.opentripplanner.routing.graph.Vertex;
import org.opentripplanner.routing.patch.Alert;
import org.opentripplanner.routing.pathparser.PathParser;
import org.opentripplanner.routing.trippattern.TripTimes;
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;

public class State implements Cloneable {

    protected double distance;
    protected long time;
    protected double weight;
    protected Vertex vertex;
    protected State backState;
    protected Edge backEdge;
    protected State next;
    public StateData stateData;
    protected double walkDistance;
    protected int[] pathParserStates;
    private static final Logger LOG = LoggerFactory.getLogger(State.class);

    public State(RoutingRequest opt) {
        this(opt.rctx.origin, opt.rctx.originBackEdge, opt.getSecondsSinceEpoch(), opt);
    }

    public State(Vertex vertex, RoutingRequest opt) {
        this(vertex, opt.getSecondsSinceEpoch(), opt);
    }

    public State(Vertex vertex, long timeSeconds, RoutingRequest options) {
        this(vertex, null, timeSeconds, options);
    }

    public State(Vertex vertex, Edge backEdge, long timeSeconds, RoutingRequest options) {
        this.weight = 0;
        this.distance = 0;
        this.vertex = vertex;
        this.backEdge = backEdge;
        this.backState = null;
        this.stateData = new StateData(options);
        this.stateData.opt = options;
        this.stateData.startTime = timeSeconds;
        this.stateData.usingRentedBike = false;
        this.walkDistance = 0;
        this.time = timeSeconds * 1000;
        if (options.rctx != null) {
            this.pathParserStates = new int[options.rctx.pathParsers.length];
            Arrays.fill(this.pathParserStates, AutomatonState.START);
        }
        stateData.routeSequence = new AgencyAndId[0];
    }

    public StateEditor edit(Edge e) { return new StateEditor(this, e); }

    protected State clone() {
        State ret;
        try {
            ret = (State) super.clone();
        } catch (CloneNotSupportedException e1) {
            throw new IllegalStateException("This is not happening");
        }
        return ret;
    }

    public Object getExtension(Object key) {
        if (stateData.extensions == null) return null;
        return stateData.extensions.get(key);
    }
```

```
public String toString() {
    return "<State " + new Date(getTimeInMillis()) + " [" + weight + "] " + vertex + ">";
}

public String toStringVerbose() {
    return "<State " + new Date(getTimeInMillis()) +
            " w=" + this.getWeight() +
            " t=" + this.getElapsedTimeSeconds() +
            " d=" + this.getWalkDistance() +
            " b=" + this.getNumBoardings() + ">";
}

public long getTimeSeconds() { return time / 1000; }

public long getElapsedTimeSeconds() { return Math.abs(getTimeSeconds() - stateData.startTime); }

public TripTimes getTripTimes() { return stateData.tripTimes; }

public long getActiveTime () {
    long clampInitialWait = stateData.opt.clampInitialWait;
    long initialWait = stateData.initialWaitTime;
    if (clampInitialWait >= 0 && initialWait > clampInitialWait)
        initialWait = clampInitialWait;
    long activeTime = getElapsedTimeSeconds() - initialWait;
    if (activeTime < 0) {
        LOG.warn("initial wait was greater than elapsed time.");
        activeTime = getElapsedTimeSeconds();
    }
    return activeTime;
}

public AgencyAndId getTripId() { return stateData.tripId; }

public String getZone() { return stateData.zone; }

public AgencyAndId getRoute() { return stateData.route; }

public int getNumBoardings() { return stateData.numBoardings; }

public boolean isAlightedLocal() { return stateData.alightedLocal; }

public boolean isEverBoarded() { return stateData.everBoarded; }

public boolean isBikeRenting() { return stateData.usingRentedBike; }

public Vertex getPreviousStop() { return stateData.previousStop; }

public long getLastAlightedTimeSeconds() { return stateData.lastAlightedTime; }

public double getWalkDistance() { return walkDistance; }

public Vertex getVertex() { return this.vertex; }

public int getLastNextArrivalDelta () { return stateData.lastNextArrivalDelta; }

public boolean dominates(State other) {
    if (other.weight == 0) {
        return false;
    }
    if (isBikeRenting() != other.isBikeRenting()) return false;

    if (backEdge != other.getBackEdge() && ((backEdge instanceof PlainStreetEdge)
            && (!((PlainStreetEdge) backEdge).getTurnRestrictions().isEmpty())))
        return false;

    if (this.similarRouteSequence(other)) { return this.weight <= other.weight; }

    double weightDiff = this.weight / other.weight;
    return walkDistance <= other.getWalkDistance() * 1.05
            && (weightDiff < 1.02 && this.weight - other.weight < 30)
            && this.getElapsedTimeSeconds() - other.getElapsedTimeSeconds() <= 30;
}

public boolean betterThan(State other) { return this.weight < other.weight; }

public double getWeight() { return this.weight; }

public double getDistance() { return this.distance; }
```

```
public int getTimeDeltaSeconds() {
    return (int) (getTimeSeconds() - backState.getTimeSeconds());
}

public int getAbsTimeDeltaSeconds() {
    return (int) Math.abs(getTimeSeconds() - backState.getTimeSeconds());
}

public double getWalkDistanceDelta () {
    if (backState != null) return Math.abs(this.walkDistance - backState.walkDistance);
    else return 0.0;
}

public double getWeightDelta() { return this.weight - backState.weight; }

public void checkNegativeWeight() {
    double dw = this.weight - backState.weight;
    if (dw < 0) {
        throw new NegativeWeightException(String.valueOf(dw) + " on edge " + backEdge);
    }
}

public boolean isOnboard() { return this.backEdge instanceof OnBoardForwardEdge; }

public State getBackState() { return this.backState; }

public TraverseMode getBackMode () { return stateData.backMode; }

public boolean isBackWalkingBike () { return stateData.backWalkingBike; }

public Set<Alert> getBackAlerts () { return stateData.notes; }

public String getBackDirection () {
    if (backEdge instanceof TablePatternEdge) {
        return stateData.tripTimes.getHeadsign(((TablePatternEdge)backEdge).getStopIndex());
    }
    else {
        return backEdge.getDirection();
    }
}

public Trip getBackTrip () {
    if (backEdge instanceof TablePatternEdge) { return stateData.tripTimes.getTrip(); }
    else { return backEdge.getTrip(); }
}

public Edge getBackEdge() { return this.backEdge; }

public boolean exceedsWeightLimit(double maxWeight) { return weight > maxWeight; }

public long getStartTimeSeconds() { return stateData.startTime; }

public State getNextResult() { return next; }

public State addToExistingResultChain(State existingResultChain) {
    if (this.getNextResult() != null)
        throw new IllegalStateException("this result already has a next result set");
    next = existingResultChain;
    return this;
}

public State detachNextResult() {
    State ret = this.next;
    this.next = null;
    return ret;
}

public RoutingContext getContext() { return stateData.opt.rctx; }

public RoutingRequest getOptions () { return stateData.opt; }

public TraverseMode getNonTransitMode() { return stateData.nonTransitMode; }

public TraverseMode getTransitMode() { return stateData.transitMode; }

public State reversedClone() {
    State newState = new State(this.vertex, getTimeSeconds(), stateData.opt.reversedClone());
    newState.stateData.tripTimes = stateData.tripTimes;
```

```
    newState.stateData.initialWaitTime = stateData.initialWaitTime;
    return newState;
}

public void dumpPath() {
    System.out.printf("---- FOLLOWING CHAIN OF STATES ----\n");
    State s = this;
    while (s != null) {
        System.out.printf("%s via %s by %s\n", s, s.backEdge, s.getBackMode());
        s = s.backState;
    }
    System.out.printf("---- END CHAIN OF STATES ----\n");
}

public long getTimeInMillis() { return time; }

public boolean similarRouteSequence(State that) {
    AgencyAndId[] rs0 = this.stateData.routeSequence;
    AgencyAndId[] rs1 = that.stateData.routeSequence;
    if (rs0 == rs1)
        return true;
    int n = rs0.length < rs1.length ? rs0.length : rs1.length;
    for (int i = 0; i < n; i++)
        if (rs0[i] != rs1[i])
            return false;
    return true;
}

public double getWalkSinceLastTransit() { return walkDistance - stateData.lastTransitWalk; }

public double getWalkAtLastTransit() { return stateData.lastTransitWalk; }

public boolean multipleOptionsBefore() {
    boolean foundAlternatePaths = false;
    TraverseMode requestedMode = getNonTransitMode();
    for (Edge out : backState.vertex.getOutgoing()) {
        if (out == backEdge) { continue; }
        if (!(out instanceof StreetEdge)) { continue; }
        State outState = out.traverse(backState);
        if (outState == null) { continue; }
        if (!outState.getBackMode().equals(requestedMode)) { continue; }

        Vertex tov = outState.getVertex();
        boolean found = false;
        for (Edge out2 : tov.getOutgoing()) {
            State outState2 = out2.traverse(outState);
            if (outState2 != null && !outState2.getBackMode().equals(requestedMode)) {
                continue;
            }
            found = true;
            break;
        }
        if (!found) { continue; }
        foundAlternatePaths = true;
        break;
    }
    return foundAlternatePaths;
}

public boolean allPathParsersAccept() {
   PathParser[] parsers = this.stateData.opt.rctx.pathParsers;
   for (int i = 0; i < parsers.length; i++)
          if ( ! parsers[i].accepts(pathParserStates[i]))
                  return false;
   return true;
}

public String getPathParserStates() {
          StringBuilder sb = new StringBuilder();
          sb.append("( ");
          for (int i : pathParserStates)
                  sb.append(String.format("%02d ", i));
          sb.append(")");
          return sb.toString();
 }

public TripPattern getLastPattern() { return stateData.lastPattern; }
```

```
public State optimizeOrReverse (boolean optimize, boolean forward) {
    State orig = this;
    State unoptimized = orig;
    State ret = orig.reversedClone();
    long newInitialWaitTime = this.stateData.initialWaitTime;
    PathParser pathParsers[];

    pathParsers = stateData.opt.rctx.pathParsers;
    stateData.opt.rctx.pathParsers = new PathParser[0];

    Edge edge = null;

    while (orig.getBackState() != null) {
        edge = orig.getBackEdge();

        if (optimize) {
            if (edge instanceof TransitBoardAlight &&
                    forward &&
                    orig.getNumBoardings() == 1 &&
                    (
                            (((TransitBoardAlight) edge).isBoarding() &&
                                    !stateData.opt.isArriveBy()) ||
                            (!((TransitBoardAlight) edge).isBoarding() &&
                                    stateData.opt.isArriveBy())
                     )
                ) {
                ret = ((TransitBoardAlight) edge).traverse(ret, orig.getBackState().getTimeSeconds());
                newInitialWaitTime = ret.stateData.initialWaitTime;
            }
            else
                ret = edge.traverse(ret);

            if (ret == null) {
                LOG.warn("Cannot reverse path at edge: " + edge +
                         ", returning unoptimized path. If edge is a " +
                         "PatternInterlineDwell or if there is a time-dependent turn " +
                         "restriction here, this is not totally unexpected; " +
                         "otherwise, you might want to look into it");

                stateData.opt.rctx.pathParsers = pathParsers;

                if (forward)
                    return this;
                else
                    return unoptimized.reverse();
            }
        }
        else {
            StateEditor editor = ret.edit(edge);
            editor.setFromState(orig);
            editor.incrementTimeInSeconds(orig.getAbsTimeDeltaSeconds());
            editor.incrementWeight(orig.getWeightDelta());
            editor.incrementWalkDistance(orig.getWalkDistanceDelta());
            editor.setBackMode(orig.getBackMode());
            editor.addAlerts(orig.getBackAlerts());
            if (orig.isBikeRenting() != orig.getBackState().isBikeRenting())
                editor.setBikeRenting(!orig.isBikeRenting());
            ret = editor.makeState();
        }
        orig = orig.getBackState();
    }

    stateData.opt.rctx.pathParsers = pathParsers;

    if (forward) {
        State reversed = ret.reverse();
        if (getWeight() <= reversed.getWeight())
            LOG.warn("Optimization did not decrease weight: before " + this.getWeight()
                    + " after " + reversed.getWeight());
        if (getElapsedTimeSeconds() != reversed.getElapsedTimeSeconds())
            LOG.warn("Optimization changed time: before " + this.getElapsedTimeSeconds() + " after "
                    + reversed.getElapsedTimeSeconds());
        if (getActiveTime() <= reversed.getActiveTime())
            LOG.warn("Optimization did not decrease active time: before "
                    + this.getActiveTime() + " after " + reversed.getActiveTime()
                    + ", boardings: " + this.getNumBoardings());
        if (reversed.getWeight() < this.getBackState().getWeight())
            LOG.warn("Weight has been reduced enough to make it run backwards, now:"
                    + reversed.getWeight() + " backState " + getBackState().getWeight() + ", "
```

```
                    + "number of boardings: " + getNumBoardings());
            if (getTimeSeconds() != reversed.getTimeSeconds())
                LOG.warn("Times do not match");
            if (Math.abs(getWeight() - reversed.getWeight()) > 1
                    && newInitialWaitTime == stateData.initialWaitTime)
                LOG.warn("Weight is changed (before: " + getWeight() + ", after: "
                        + reversed.getWeight() + "), initial wait times " + "constant at "
                        + newInitialWaitTime);
            if (newInitialWaitTime != reversed.stateData.initialWaitTime)
                LOG.warn("Initial wait time not propagated: is "
                        + reversed.stateData.initialWaitTime + ", should be " + newInitialWaitTime);
            reversed.initializeFieldsFrom(this);
            return reversed;
        }
        else
            return ret;
    }

    public State optimize() { return optimizeOrReverse(true, false); }

    public State reverse() { return optimizeOrReverse(false, false); }

    private void initializeFieldsFrom (State o) {
        StateData currentStateData = this.stateData;
        this.stateData = o.stateData.clone();
        this.stateData.initialWaitTime = currentStateData.initialWaitTime;
        this.stateData.lastNextArrivalDelta = -1;
    }

    public boolean getReverseOptimizing () { return stateData.opt.reverseOptimizing; }

    public double getOptimizedElapsedTimeSeconds() {
        return getElapsedTimeSeconds() - stateData.initialWaitTime;
    }
}
```

d. opentripplanner-routing/src/main/java/org/opentripplanner/routing/core/StateData.java

```
package org.opentripplanner.routing.core;

import java.util.HashMap;
import java.util.Set;
import org.onebusaway.gtfs.model.AgencyAndId;
import org.opentripplanner.routing.edgetype.TripPattern;
import org.opentripplanner.routing.graph.Vertex;
import org.opentripplanner.routing.trippattern.TripTimes;
import org.opentripplanner.routing.patch.Alert;

public class StateData implements Cloneable {

    protected long startTime;
    protected TripTimes tripTimes;
    protected AgencyAndId tripId;
    protected double lastTransitWalk = 0;
    protected String zone;
    protected AgencyAndId route;
    protected int numBoardings;
    protected boolean alightedLocal;
    protected boolean everBoarded;
    protected Vertex previousStop;
    protected long lastAlightedTime;
    protected AgencyAndId[] routeSequence;
    protected HashMap<Object, Object> extensions;
    protected RoutingRequest opt;
    protected TripPattern lastPattern;
    protected ServiceDay serviceDay;
    protected TraverseMode nonTransitMode;
    protected TraverseMode transitMode;
    protected long initialWaitTime = 0;
    protected int lastNextArrivalDelta;

    protected Set<Alert> notes = null;
    protected TraverseMode backMode;

    public StateData(RoutingRequest options) {
        TraverseModeSet modes = options.getModes();
        if (modes.getCar())
            nonTransitMode = TraverseMode.CAR;
        else if (modes.getCustomMotorVehicle())
```

```
            nonTransitMode = TraverseMode.CUSTOM_MOTOR_VEHICLE;
        else if (modes.getWalk())
            nonTransitMode =  TraverseMode.WALK;
        else
              nonTransitMode = null;

        if (modes.getBus())
            transitMode = TraverseMode.BUS;
        else if (modes.getRail())
            transitMode = TraverseMode.RAIL;
        else
            transitMode = null;
    }

    protected StateData clone() {
        try {
            return (StateData) super.clone();
        } catch (CloneNotSupportedException e1) {
            throw new IllegalStateException("This is not happening");
        }
    }

}
```

e. opentripplanner-routing/src/main/java/org/opentripplanner/routing/core/StateDataEditor.java

```
package org.opentripplanner.routing.core;

import java.util.Arrays;
import java.util.HashMap;
import java.util.Set;
import java.util.HashSet;
import java.util.List;
import org.onebusaway.gtfs.model.AgencyAndId;
import org.opentripplanner.routing.automata.AutomatonState;
import org.opentripplanner.routing.edgetype.TripPattern;
import org.opentripplanner.routing.graph.Edge;
import org.opentripplanner.routing.graph.Vertex;
import org.opentripplanner.routing.patch.Alert;
import org.opentripplanner.routing.patch.Patch;
import org.opentripplanner.routing.pathparser.PathParser;
import org.opentripplanner.routing.trippattern.TripTimes;
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;

/**
 * This class is a wrapper around a new State that provides it with setter and increment methods,
 * allowing it to be modified before being put to use.
 *
 * By virtue of being in the same package as States, it can modify their package private fields.
 *
 * @author andrewbyrd
 */
public class StateEditor {

    private static final Logger LOG = LoggerFactory.getLogger(StateEditor.class);
    protected State child;
    private boolean extensionsModified = false;
    private boolean spawned = false;
    private boolean defectiveTraversal = false;
    private boolean traversingBackward;
    private Set<Alert> notes = null;

    protected StateEditor() {}

    public StateEditor(RoutingRequest options, Vertex v) {
        child = new State(v, options);
        child.stateData = new StateData(options);
    }

    public StateEditor(State parent, Edge e) {
        child = parent.clone();
        child.backState = parent;
        child.backEdge = e;
        child.next = null;
        if (e == null) {
            child.backState = null;
            child.vertex = parent.vertex;
            child.stateData = child.stateData.clone();
```

```
    } else {
       if (e.getFromVertex().equals(e.getToVertex())
                && parent.vertex.equals(e.getFromVertex())) {
            traversingBackward = parent.getOptions().isArriveBy();
            child.vertex = e.getToVertex();
        } else if (parent.vertex.equals(e.getFromVertex())) {
            traversingBackward = false;
            child.vertex = e.getToVertex();
        } else if (parent.vertex.equals(e.getToVertex())) {
            traversingBackward = true;
            child.vertex = e.getFromVertex();
        } else {
            LOG.warn("Edge is not connected to parent state: {}", e);
            LOG.warn("   from   vertex: {}", e.getFromVertex());
            LOG.warn("   to     vertex: {}", e.getToVertex());
            LOG.warn("   parent vertex: {}", parent.vertex);
            defectiveTraversal = true;
        }
        if (traversingBackward != parent.getOptions().isArriveBy()) {
        LOG.error("Actual traversal direction does not match traversal direction in TraverseOptions.");
            defectiveTraversal = true;
        }
    }
}

public State makeState() {
    if (spawned)
        throw new IllegalStateException("A StateEditor can only be used once.");
    if (defectiveTraversal) {
        LOG.error("Defective traversal flagged on edge " + child.backEdge);
        return null;
    }

    if (child.backState != null) {
        child.checkNegativeWeight();
        if (traversingBackward ? (child.getTimeDeltaSeconds() > 0):(child.getTimeDeltaSeconds() < 0)) {
            LOG.trace("Time was incremented the wrong direction during state editing. {}",
                    child.backEdge);
            return null;
        }

        if(!applyPatches()) { return null; }
    }
    if ( ! parsePath(this.child)) return null;

    if (this.notes != child.stateData.notes) {
        cloneStateDataAsNeeded();
        child.stateData.notes = this.notes;
    }
    spawned = true;
    return child;
}

public boolean weHaveWalkedTooFar(RoutingRequest options) {
    if (!options.getModes().isTransit())
        return false;
    return child.walkDistance >= options.maxWalkDistance;
}

public String toString() { return "<StateEditor " + child + ">"; }

@SuppressWarnings("unchecked")
public void setExtension(Object key, Object value) {
    cloneStateDataAsNeeded();
    if (!extensionsModified) {
        HashMap<Object, Object> newExtensions;
        if (child.stateData.extensions == null)
            newExtensions = new HashMap<Object, Object>(4);
        else
            newExtensions = (HashMap<Object, Object>) child.stateData.extensions.clone();
        child.stateData.extensions = newExtensions;
        extensionsModified = true;
    }
    child.stateData.extensions.put(key, value);
}

public void blockTraversal() { this.defectiveTraversal = true; }
```

```
public void addAlert(Alert notes) {
    if (notes == null) return;
    if (this.notes == null) this.notes = new HashSet<Alert>();
    this.notes.add(notes);
}

public void addAlerts(Iterable<Alert> alerts) {
    if (alerts == null) return;
    for (Alert alert : alerts) {
        this.addAlert(alert);
    }
}

public void incrementWeight(double weight) {
    if (Double.isNaN(weight)) {
        LOG.warn("A state's weight is being incremented by NaN while traversing edge "
                + child.backEdge);
        defectiveTraversal = true;
        return;
    }
    if (weight < 0) {
        LOG.warn("A state's weight is being incremented by a negative amount while traversing edge "
                + child.backEdge);
        defectiveTraversal = true;
        return;
    }
    child.weight += weight;
}

public void incrementDistance(double distance) { child.distance += distance; }

public void incrementTimeInSeconds(int seconds) { incrementTimeInMilliseconds(seconds * 1000);  }

public void incrementTimeInMilliseconds(int milliseconds) {
    if (milliseconds < 0) {
        LOG.warn("A state's time is being incremented by a negative amount while traversing edge "
                + child.getBackEdge());
        defectiveTraversal = true;
        return;
    }
    child.time += (traversingBackward ? -milliseconds : milliseconds);
}

public void incrementWalkDistance(double length) {
    if (length < 0) {
        LOG.warn("A state's walk distance is being incremented by a negative amount.");
        defectiveTraversal = true;
        return;
    }
    child.walkDistance += length;
}

public void incrementNumBoardings() {
    cloneStateDataAsNeeded();
    child.stateData.numBoardings++;
}

public void setTripTimes(TripTimes tripTimes) {
    cloneStateDataAsNeeded();
    child.stateData.tripTimes = tripTimes;
}

public void setTripId(AgencyAndId tripId) {
    cloneStateDataAsNeeded();
    child.stateData.tripId = tripId;
}

public void setInitialWaitTimeSeconds(long initialWaitTimeSeconds) {
    cloneStateDataAsNeeded();
    child.stateData.initialWaitTime = initialWaitTimeSeconds;
}

public void setBackMode(TraverseMode mode) {
    if (mode == child.stateData.backMode)
        return;

    cloneStateDataAsNeeded();
    child.stateData.backMode = mode;
}
```

```
public void setBackWalkingBike (boolean walkingBike) {
    if (walkingBike == child.stateData.backWalkingBike)
        return;
    cloneStateDataAsNeeded();
    child.stateData.backWalkingBike = walkingBike;
}

public void setLastNextArrivalDelta (int lastNextArrivalDelta) {
    cloneStateDataAsNeeded();
    child.stateData.lastNextArrivalDelta = lastNextArrivalDelta;
}

public void setWalkDistance(double walkDistance) { child.walkDistance = walkDistance; }

public void setZone(String zone) {
    if (zone == null) {
        if (child.stateData.zone != null) {
            cloneStateDataAsNeeded();
            child.stateData.zone = zone;
        }
    } else if (!zone.equals(child.stateData.zone)) {
        cloneStateDataAsNeeded();
        child.stateData.zone = zone;
    }
}

public void setRoute(AgencyAndId routeId) {
    cloneStateDataAsNeeded();
    child.stateData.route = routeId;
    if (routeId != null) {
        AgencyAndId[] oldRouteSequence = child.stateData.routeSequence;
        int oldLength = oldRouteSequence.length;
        child.stateData.routeSequence = Arrays.copyOf(oldRouteSequence, oldLength + 1);
        child.stateData.routeSequence[oldLength] = routeId;
    }
}

public void setNumBoardings(int numBoardings) {
    cloneStateDataAsNeeded();
    child.stateData.numBoardings = numBoardings;
}

public void setAlightedLocal(boolean alightedLocal) {
    cloneStateDataAsNeeded();
    child.stateData.alightedLocal = alightedLocal;
}

public void setEverBoarded(boolean everBoarded) {
    cloneStateDataAsNeeded();
    child.stateData.everBoarded = everBoarded;
}

public void setPreviousStop(Vertex previousStop) {
    cloneStateDataAsNeeded();
    child.stateData.previousStop = previousStop;
}

public void setLastAlightedTimeSeconds(long lastAlightedTimeSeconds) {
    cloneStateDataAsNeeded();
    child.stateData.lastAlightedTime = lastAlightedTimeSeconds;
}

public void setTimeSeconds(long seconds) { child.time = seconds * 1000; }

public void setStartTimeSeconds(long seconds) {
    cloneStateDataAsNeeded();
    child.stateData.startTime = seconds;
}

public void setFromState(State state) {
    cloneStateDataAsNeeded();
    child.stateData.route = state.stateData.route;
    child.stateData.tripTimes = state.stateData.tripTimes;
    child.stateData.tripId = state.stateData.tripId;
    child.stateData.zone = state.stateData.zone;
    child.stateData.extensions = state.stateData.extensions;
    child.stateData.usingRentedBike = state.stateData.usingRentedBike;
}
```

```
public Object getExtension(Object key) { return child.getExtension(key); }

public long getTimeSeconds() { return child.getTimeSeconds(); }

public long getElapsedTimeSeconds() { return child.getElapsedTimeSeconds(); }

public AgencyAndId getTripId() { return child.getTripId(); }

public String getZone() { return child.getZone(); }

public AgencyAndId getRoute() { return child.getRoute(); }

public int getNumBoardings() { return child.getNumBoardings(); }

public boolean isAlightedLocal() { return child.isAlightedLocal(); }

public boolean isEverBoarded() { return child.isEverBoarded(); }

public Vertex getPreviousStop() { return child.getPreviousStop(); }

public long getLastAlightedTimeSeconds() { return child.getLastAlightedTimeSeconds(); }

public double getWalkDistance() { return child.getWalkDistance(); }

public Vertex getVertex() { return child.getVertex(); }

private boolean applyPatches() {
    List<Patch> patches = child.backEdge.getPatches();
    boolean display = false, active = false;

    if (patches != null) {
        for (Patch patch : patches) {
            active  = false;
            display = patch.displayDuring(child.stateData.opt, child.getStartTimeSeconds(),
                                          child.getTimeSeconds());

            if(!display) {
                active = patch.activeDuring(child.stateData.opt, child.getStartTimeSeconds(),
                                            child.getTimeSeconds());
            }

            if(display || active) {
                if(!patch.filterTraverseResult(this, display))
                    return false;
            }
        }
    }
    return true;
}

private void cloneStateDataAsNeeded() {
    if (child.backState != null && child.stateData == child.backState.stateData)
        child.stateData = child.stateData.clone();
}

public boolean parsePath(State state) {
    if (state.stateData.opt.rctx == null)
        return true;
    PathParser[] parsers = state.stateData.opt.rctx.pathParsers;
    int[] parserStates = state.pathParserStates;
    boolean accept = true;
    boolean modified = false;
    int i = 0;
    for (PathParser parser : parsers) {
        int terminal = parser.terminalFor(state);
        int oldState = parserStates[i];
        int newState = parser.transition(oldState, terminal);
        if (newState != oldState) {
            if (!modified) {
                parserStates = parserStates.clone();
                modified = true;
            }
            parserStates[i] = newState;
            if (newState == AutomatonState.REJECT)
                accept = false;
        }
        i++;
    }
```

```
        if (modified) state.pathParserStates = parserStates;
        return accept;
    }

    public void alightTransit() {
        cloneStateDataAsNeeded();
        child.stateData.lastTransitWalk = child.getWalkDistance();
    }

    public void setLastPattern(TripPattern pattern) {
        cloneStateDataAsNeeded();
        child.stateData.lastPattern = pattern;
    }
    public void setOptions(RoutingRequest options) {
        cloneStateDataAsNeeded();
        child.stateData.opt = options;
    }

    public void setServiceDay(ServiceDay day) {
        cloneStateDataAsNeeded();
        child.stateData.serviceDay = day;
    }

    public void setBikeRentalNetwork(Set<String> networks) {
        cloneStateDataAsNeeded();
        child.stateData.bikeRentalNetworks = networks;
    }
}
```

f. opentripplanner-routing/src/main/java/org/opentripplanner/routing/edgetype/FrequencyHop.java

```
package org.opentripplanner.routing.edgetype;

import java.io.BufferedReader;
import java.io.File;
import java.io.FileReader;
import java.io.IOException;
import org.onebusaway.gtfs.model.Stop;
import org.onebusaway.gtfs.model.Trip;
import org.opentripplanner.common.geometry.SphericalDistanceLibrary;
import org.opentripplanner.common.geometry.GeometryUtils;
import org.opentripplanner.gtfs.GtfsLibrary;
import org.opentripplanner.model.Fare;
import org.opentripplanner.routing.core.RoutingRequest;
import org.opentripplanner.routing.core.State;
import org.opentripplanner.routing.core.StateEditor;
import org.opentripplanner.routing.core.TraverseMode;
import org.opentripplanner.routing.graph.Edge;
import org.opentripplanner.routing.vertextype.TransitVertex;
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import com.vividsolutions.jts.geom.Coordinate;
import com.vividsolutions.jts.geom.LineString;

public class FrequencyHop extends Edge implements OnBoardForwardEdge, OnBoardReverseEdge, HopEdge {

    private static final long serialVersionUID = 2389378459266920841L;
    private static final Logger LOG = LoggerFactory.getLogger(FrequencyHop.class);
    private Fare fareDetails;
    private RoutingRequest options;
    private FrequencyBasedTripPattern pattern;
    private Stop start;
    private Stop end;
    private int stopIndex;
    private LineString geometry;
    private double jeepBase, jeepFirst, jeepPer;
    private double busBase, busFirst, busPer;
    private double railBase, railPer;
    private int railFirst;

    public FrequencyHop(TransitVertex depart, TransitVertex arrive, Stop s0, Stop s1,
            int stopIndex, FrequencyBasedTripPattern pattern) {
        super(depart, arrive);
        this.pattern = pattern;
        this.start = s0;
        this.end = s1;
        this.stopIndex = stopIndex;
    }
```

```
public double getDistance() {
    return SphericalDistanceLibrary.getInstance().distance(start.getLat(),
                                                start.getLon(), end.getLat(),end.getLon());
}

public TraverseMode getMode() {
    return GtfsLibrary.getTraverseMode(pattern.getTrip().getRoute());
}

public String getName() {
    return GtfsLibrary.getRouteName(pattern.getTrip().getRoute());
}

@Override
public Trip getTrip() { return pattern.getTrip(); }

public String getDirection () { return pattern.getHeadsign(stopIndex); }

public State optimisticTraverse(State state0) {
    int runningTime = pattern.getRunningTime(stopIndex);
    StateEditor s1 = state0.edit(this);
    s1.setBackMode(getMode());
    s1.incrementTimeInSeconds(runningTime);
    s1.incrementWeight(runningTime);
    return s1.makeState();
}

@Override
public double timeLowerBound(RoutingRequest options) {
    return pattern.getRunningTime(stopIndex);
}

@Override
public double weightLowerBound(RoutingRequest options) {
    return timeLowerBound(options);
}

public State traverse(State s0) {
    options = new RoutingRequest();
    initialize();
    int runningTime = pattern.getRunningTime(stopIndex);
    double stopFare = getFare(s0.getDistance() + getDistance());
    double trafficPoints = getTrafficPoints();
    StateEditor s1 = s0.edit(this);
    s1.incrementTimeInSeconds(runningTime*(int)trafficPoints);
    if (s0.getOptions().isArriveBy())
        s1.setZone(getStartStop().getZoneId());
    else
        s1.setZone(getEndStop().getZoneId());
    s1.setRoute(pattern.getTrip().getRoute().getId());
    s1.setBackMode(getMode());
    s1.incrementWeight(((getDistance()/100)*options.distMultiplier)
                           + (stopFare*options.fareMultiplier)
                           + ((runningTime*getTrafficPoints())*options.timeMultiplier)
                           + s0.getWalkDistance()*options.walkMultiplier);
    s1.incrementDistance(getDistance());
    return s1.makeState();
}

public double getFare( double distance){
    double fare = 0;

    String[] routeId = pattern.getTrip().getRoute().getId().toString().split("\\_");
    if(routeId[2].substring(0, 3).equals("PUJ")){
          fare = computeBusJeepFare(jeepBase, jeepPer, jeepFirst, distance);
    }
    else if(routeId[2].substring(0, 3).equals("PUB")){
          fare = computeBusJeepFare(busBase, busPer, busFirst, distance);
    }
    else if(getMode().toString().equals("RAIL")){
          fare = computeRailFare(railBase, railFirst, railPer);
    }
    return fare;
}

private double computeBusJeepFare(double baseFare, double perKmFare, double initial, Double distance) {
          if((distance - initial) < 0){ return baseFare ; }
          else{
                   double tmpFare = baseFare + (((distance - initial)/1000)*perKmFare);
```

```
                    return Math.round(tmpFare * 4.0)/4.0 ;
          }
 }

private double computeRailFare(double baseFare, int firstStops, double perStopFare) {
 if(pattern.getStops() != null){
          if(pattern.getStops().size() <= firstStops){ return baseFare ; }
          else{
                    double tmpFare = baseFare + (pattern.getStops().size()*perStopFare);
                    return Math.round(tmpFare * 4.0)/4.0 ;
          }
 }
 else return baseFare;
 }

public void setGeometry(LineString geometry) { this.geometry = geometry; }

public LineString getGeometry() {
    if (geometry == null) {
        Coordinate c1 = new Coordinate(start.getLon(), start.getLat());
        Coordinate c2 = new Coordinate(end.getLon(), end.getLat());
        geometry = GeometryUtils.getGeometryFactory().createLineString(new Coordinate[] { c1, c2 });
    }
    return geometry;
}

@Override
public Stop getEndStop() { return end; }

@Override
public Stop getStartStop() { return start; }

public String toString() { return "PatternHop(" + getFromVertex() + ", " + getToVertex() + ")"; }

@Override
public int getStopIndex() { return stopIndex; }

public void initialize(){
    fareDetails = new Fare();
           try {

             File file = new File("/public_html/otp/fare.txt");
             FileReader fr = new FileReader(file);
             BufferedReader br = new BufferedReader(fr);
             String data;
             while((data = br.readLine()) != null)
             {
                 //data = br.readLine( );
                    String[] info = data.split("\\|");
                    if(info[0].equals("JEEPNEY")){
                              jeepBase = new Double(info[1]);
                              jeepFirst = new Double(info[2]);
                              jeepPer = new Double(info[3]);
                    }
                    else if(info[0].equals("BUS")){
                              busBase = new Double(info[1]);
                              busFirst = new Double(info[2]);
                              busPer = new Double(info[3]);
                    }
                    else if(info[0].equals("RAIL")){
                              railBase = new Double(info[1]);
                              railFirst = new Integer(info[2]);
                              railPer = new Double(info[3]);
                    }
             }
         } catch(IOException e) {
             System.out.println("Cannot read Text File.");
         }
 }

public double getTrafficPoints(){
    double startLat = start.getLat();
    double startLon = start.getLon();
    double endLat = end.getLat();
    double endLon = end.getLon();
    int startScore = 1, endScore = 1;

    if(getMode().equals("RAIL")){
          return 1;
```

```
    }

    try {

           File file = new File("/public_html/otp/traffic.txt");
           FileReader fr = new FileReader(file);
           BufferedReader br = new BufferedReader(fr);
           String data;
           while((data = br.readLine()) != null)
           {
             String[] info = data.split("\\|");
             String topBound[] = info[2].split("\\,");
             String botBound[] = info[3].split("\\,");

             double topBoundLat = new Double(topBound[0]);
             double topBoundLon = new Double(topBound[1]);

             double botBoundLat = new Double(botBound[0]);
             double botBoundLon = new Double(botBound[1]);

             if(startLat <= topBoundLat && startLat >= botBoundLat
                           && startLon >= topBoundLon && startLon <= botBoundLon){
                    startScore = getScore(info[1]);
             }

             if(endLat <= topBoundLat && endLat >= botBoundLat
                           && endLon >= topBoundLon && endLon <= botBoundLon){
                    endScore = getScore(info[1]);
             }

           }
        } catch(IOException e) {
            System.out.println("Cannot read Text File.");
        }

    return (startScore + endScore)/2;

    }

    public int getScore(String infoTxt){
        int score =0;
        if(infoTxt.equals("LIGHT TRAFFIC")){
              score = 1;
        }
        else if(infoTxt.equals("MODERATE TRAFFIC")){
              score = 3;
        }
        else if(infoTxt.equals("HEAVY TRAFFIC")){
              score = 5;
        }

        return score;
    }

}
```

g. opentripplanner-routing/src/main/java/org/opentripplanner/routing/edgetype/FrequencyBoard.java

```
package org.opentripplanner.routing.edgetype;

import org.onebusaway.gtfs.model.Trip;
import org.opentripplanner.routing.core.RoutingContext;
import org.opentripplanner.routing.core.RoutingRequest;
import org.opentripplanner.routing.core.ServiceDay;
import org.opentripplanner.routing.core.State;
import org.opentripplanner.routing.core.StateEditor;
import org.opentripplanner.routing.core.TraverseMode;
import org.opentripplanner.routing.core.TraverseModeSet;
import org.opentripplanner.routing.graph.Edge;
import org.opentripplanner.routing.vertextype.TransitVertex;
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import com.vividsolutions.jts.geom.LineString;

public class FrequencyBoard extends Edge implements OnBoardForwardEdge, PatternEdge {
    private static final long serialVersionUID = 7919511656529752927L;
    private static final Logger LOG = LoggerFactory.getLogger(FrequencyBoard.class);
    private int stopIndex;
    private FrequencyBasedTripPattern pattern;
```

```
private int modeMask;
private int serviceId;

public FrequencyBoard(TransitVertex from, TransitVertex to,
        FrequencyBasedTripPattern pattern, int stopIndex, TraverseMode mode, int serviceId) {
    super(from, to);
    this.pattern = pattern;
    this.stopIndex = stopIndex;
    this.modeMask = new TraverseModeSet(mode).getMask();
    this.serviceId = serviceId;
}

@Override
public Trip getTrip() { return pattern.getTrip(); }

public String getDirection() { return pattern.getHeadsign(stopIndex); }

public double getDistance() { return 0; }

public LineString getGeometry() { return null; }

public TraverseMode getMode() { return TraverseMode.BOARDING; }

public String getName() { return "leave street network for transit network"; }

public State traverse(State state0) {
    RoutingContext rctx = state0.getContext();
    RoutingRequest options = state0.getOptions();
    int transferPenalty = options.transferPenalty;
    Trip trip = pattern.getTrip();

    if (options.isArriveBy()) {
        if (state0.getBackEdge() instanceof TransitBoardAlight &&
                !((TransitBoardAlight) state0.getBackEdge()).isBoarding()) {
            return null;
        }
        StateEditor s1 = state0.edit(this);
        int type = pattern.getBoardType(stopIndex);
        if (TransitUtils.handleBoardAlightType(s1, type)) {
            return null;
        }
        s1.setTripId(null);
        s1.setLastAlightedTimeSeconds(state0.getTimeSeconds());
        s1.setBackMode(TraverseMode.BOARDING);
        s1.setPreviousStop(fromv);
        return s1.makeState();
    } else {
        if (!options.getModes().get(modeMask)) {
            return null;
        }
        long currentTime = state0.getTimeSeconds();
        int bestWait = -1;
        TraverseMode mode = state0.getNonTransitMode();
        TraverseMode mode2 = null;
        if(state0.getTransitMode() != null)
           mode2 = state0.getTransitMode();

        if (options.bannedTrips.containsKey(trip.getId())) { return null; }
        for (ServiceDay sd : rctx.serviceDays) {
            int secondsSinceMidnight = sd.secondsSinceMidnight(currentTime);
            if (secondsSinceMidnight < 0) continue;
            if (sd.serviceIdRunning(serviceId)) {
                int startTime = pattern.getNextDepartureTime(stopIndex, secondsSinceMidnight,
                        options.wheelchairAccessible, mode == TraverseMode.BICYCLE, true);
                if (startTime >= 0) {
                    int wait = (int) (sd.time(startTime) - currentTime);
                    if (wait < 0)
                        LOG.error("negative wait time on board");
                    if (bestWait < 0 || wait < bestWait) {
                        bestWait = wait;
                    }
                }

            }
        }
        if (bestWait < 0) { return null; }

        if (options.tripIsBanned(trip)) return null;
```

```
            long preferences_penalty = options.preferencesPenaltyForTrip(trip);

            StateEditor s1 = state0.edit(this);
            int type = pattern.getBoardType(stopIndex);
            if (TransitUtils.handleBoardAlightType(s1, type)) { return null; }
            s1.incrementTimeInSeconds(bestWait);
            s1.incrementNumBoardings();
            s1.setTripId(trip.getId());
            s1.setZone(pattern.getZone(stopIndex));
            s1.setRoute(trip.getRoute().getId());
            s1.setBackMode(TraverseMode.BOARDING);

            long wait_cost = bestWait;
            if (state0.getNumBoardings() == 0) { wait_cost *= options.waitAtBeginningFactor; }
            else { wait_cost *= options.waitReluctance; }
            String id = trip.getId().toString();
            s1.incrementWeight(preferences_penalty);
            s1.incrementWeight(wait_cost + transferPenalty);
            return s1.makeState();
        }
    }

    public State optimisticTraverse(State state0) {
        StateEditor s1 = state0.edit(this);
        s1.setBackMode(TraverseMode.BOARDING);
        return s1.makeState();
    }

    public double timeLowerBound(RoutingContext rctx) {
        if (rctx.opt.isArriveBy()) {
            if (! rctx.opt.getModes().get(modeMask)) {
                return Double.POSITIVE_INFINITY;
            }
            for (ServiceDay sd : rctx.serviceDays)
                if (sd.serviceIdRunning(serviceId))
                    return 0;
            return Double.POSITIVE_INFINITY;
        } else {
            return 0;
        }
    }

    public double weightLowerBound(RoutingRequest options) {
        if (options.isArriveBy()) return timeLowerBound(options);
        else return options.getBoardCostLowerBound();
    }

    public int getStopIndex() { return stopIndex; }

    public String toString() {
        return "FrequencyBoard(" + getFromVertex() + ", " + getToVertex() + ")";
    }

    public FrequencyBasedTripPattern getPattern() { return pattern; }
}
```

h. opentripplanner-routing/src/main/java/org/opentripplanner/routing/impl/RetryingPathServiceImpl.java

```
package org.opentripplanner.routing.impl;

import java.io.BufferedReader;
import java.io.File;
import java.io.FileReader;
import java.io.IOException;
import java.io.PrintWriter;
import java.util.ArrayList;
import java.util.LinkedList;
import java.util.List;
import java.util.Queue;
import org.jsoup.Jsoup;
import org.jsoup.nodes.Document;
import org.jsoup.nodes.Element;
import org.jsoup.select.Elements;
import org.onebusaway.gtfs.model.AgencyAndId;
import org.opentripplanner.model.Fare;
import org.opentripplanner.routing.core.RoutingRequest;
import org.opentripplanner.routing.pathparser.BasicPathParser;
import org.opentripplanner.routing.pathparser.NoThruTrafficPathParser;
import org.opentripplanner.routing.pathparser.PathParser;
```

```
import org.opentripplanner.routing.services.GraphService;
import org.opentripplanner.routing.services.PathService;
import org.opentripplanner.routing.services.SPTService;
import org.opentripplanner.routing.spt.GraphPath;
import org.opentripplanner.routing.spt.ShortestPathTree;
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import org.springframework.beans.factory.annotation.Autowired;

public class RetryingPathServiceImpl implements PathService {

    private static final Logger LOG = LoggerFactory.getLogger(RetryingPathServiceImpl.class);
    private static final int MAX_TIME_FACTOR = 2;
    private static final int MAX_WEIGHT_FACTOR = 2;
    private static final double MAX_WALK_MULTIPLE = 16;

    @Autowired
    private GraphService graphService;
    @Autowired
    private SPTService sptService;

    private double firstPathTimeout = 0; // seconds
    private double multiPathTimeout = 0; // seconds

    public void setTimeout (double seconds) {
        firstPathTimeout = seconds;
        multiPathTimeout = seconds;
    }

    public void setFirstPathTimeout (double seconds) { firstPathTimeout = seconds; }

    public void setMultiPathTimeout (double seconds) { multiPathTimeout = seconds; }

    @Override
    public List<GraphPath> getPaths(RoutingRequest options) {
        ArrayList<GraphPath> paths = new ArrayList<GraphPath>();
        parseTxtFile();
        getTrafficInfo();

        if (options.rctx == null) {
            options.setRoutingContext(graphService.getGraph(options.getRouterId()));
            options.rctx.pathParsers = new PathParser[] { new BasicPathParser(),
                    new NoThruTrafficPathParser() };
        }

        long searchBeginTime = System.currentTimeMillis();
        Queue<RoutingRequest> optionQueue = new LinkedList<RoutingRequest>();
        optionQueue.add(options);

        double maxWeight = Double.MAX_VALUE;
        double maxWalk = options.getMaxWalkDistance();
        double initialMaxWalk = maxWalk;
        long maxTime = options.isArriveBy() ? 0 : Long.MAX_VALUE;
        RoutingRequest currOptions;
        while (paths.size() < options.numItineraries) {
            currOptions = optionQueue.poll();
            if (currOptions == null) {
                LOG.debug("Ran out of options to try.");
                break;
            }
            currOptions.setMaxWalkDistance(maxWalk);
            double timeout = paths.isEmpty() ? firstPathTimeout : multiPathTimeout;
            long subsearchBeginTime = System.currentTimeMillis();
            ShortestPathTree spt = sptService.getShortestPathTree(currOptions, timeout);
            if (spt == null) break;
            List<GraphPath> somePaths = spt.getPaths();
            LOG.debug("END SUBSEARCH ({} msec of {} msec total)",
                    System.currentTimeMillis() - subsearchBeginTime,
                    System.currentTimeMillis() - searchBeginTime);
            if (somePaths == null) {
                LOG.warn("Aborting search. {} paths found, elapsed time {} sec",
                        paths.size(), (System.currentTimeMillis() - searchBeginTime) / 1000.0);
                break;
            }
            if (maxWeight == Double.MAX_VALUE && maxWalk == Double.MAX_VALUE) {
                if (somePaths.isEmpty()) { return null; }
                GraphPath path = somePaths.get(0);
                long duration = path.getDuration();
                LOG.debug("Setting max time and weight for subsequent searches.");
```

```
            LOG.debug("First path start time:  {}", path.getStartTime());
            maxTime = path.getStartTime() +
                              MAX_TIME_FACTOR * (currOptions.isArriveBy() ? -duration : duration);
            LOG.debug("First path duration:  {}", duration);
            LOG.debug("Max time set to:  {}", maxTime);
            maxWeight = path.getWeight() * MAX_WEIGHT_FACTOR;
            LOG.debug("Max weight set to:  {}", maxWeight);
            if (path.getWalkDistance() > maxWalk) {
                maxWalk = path.getWalkDistance() * 1.25;
            }
        }
        if (somePaths.isEmpty()) {
            LOG.debug("No paths were found.");
            if (maxWalk > initialMaxWalk * MAX_WALK_MULTIPLE || maxWalk >= Double.MAX_VALUE)
                break;
            maxWalk *= 2;
            LOG.debug("Doubled walk distance to {}", maxWalk);
            optionQueue.add(currOptions);
            continue;
        }
        for (GraphPath path : somePaths) {
            if (!paths.contains(path)) {
                if (path.getWalkDistance() > maxWalk) {
                    maxWalk = path.getWalkDistance() * 1.25;
                }
                paths.add(path);
                LOG.debug("New trips: {}", path.getTrips());
                RoutingRequest newOptions = currOptions.clone();
                for (AgencyAndId trip : path.getTrips()) {
                    newOptions.banTrip(trip);
                }
                if (!optionQueue.contains(newOptions)) {
                    optionQueue.add(newOptions);
                }
            }
        }
        LOG.debug("{} / {} itineraries", paths.size(), currOptions.numItineraries);
    }
    if (paths.size() == 0) { return null; }
    return paths;
}

public GraphService getGraphService() { return graphService; }

public void setGraphService(GraphService graphService) { this.graphService = graphService; }

public SPTService getSptService() { return sptService; }

public void setSptService(SPTService sptService) { this.sptService = sptService; }

public void parseTxtFile(){
 Fare fare = new Fare();
           try {
             File file = new File("/public_html/otp/fare.txt");
             FileReader fr = new FileReader(file);
             BufferedReader br = new BufferedReader(fr);
             String data;
             while((data = br.readLine()) != null)
             {
                String[] info = data.split("\\|");
                if(info[0].equals("JEEPNEY")){
                    fare.setJeepFare(new Double(info[1]), new Double(info[2]), new Double(info[3]));
                }
                else if(info[0].equals("BUS")){
                    fare.setBusFare(new Double(info[1]), new Double(info[2]), new Double(info[3]));
                }
                else if(info[0].equals("RAIL")){
                    fare.setRailFare(new Double(info[1]), new Integer(info[2]), new Double(info[3]));
                }
             }
         } catch(IOException e) {
             System.out.println("Cannot read Text File.");
         }
 }

public void getTrafficInfo(){
    try{
          String url = "http://localhost/mmda/viewTraffic.php";
          Document document = Jsoup.connect(url)
```

```
                        .timeout(0)
                        .userAgent("Mozilla/5.0 (Windows; U; WindowsNT 5.1; en-US; rv1.8.1.6) Gecko/20070725
                                                                        Firefox/2.0.0.6")
                                .get();
                Elements areaInfo = document.getElementsByClass("areaTDview");
                Elements trafficInfo = document.getElementsByClass("info");

                PrintWriter writer = new PrintWriter("/public_html/otp/traffic.txt", "UTF-8");
                String[] areaId = new String[200];
                int count=0, x =0;
                for(Element a: areaInfo){
                        areaId[count] = a.id();
                        count++;
                }

                for(Element s: trafficInfo){
                        String toWrite = areaId[x];
                        toWrite += "|" + s.text() + "|" + getBounds(areaId[x]);
                        writer.println(toWrite);
                        x++;
                }

                writer.close();
           }
           catch(IOException e){
                e.printStackTrace();
           }
      }

      public String getBounds(String id){
                String[] x = id.split("_");
                String txt = "";
                int row = Integer.parseInt(x[0]);
                int col = Integer.parseInt(x[1]);
                double upLat = 14.665761;
                double upLon = 120.950546;
                double xDif = 0.0138359;
                double yDif = 0.0104315;

                double topLat = upLat - (yDif*(row-1));
                double topLon = upLon + (xDif*(col-1));
                txt = topLat + "," + topLon + "|";

                double botLat = upLat - (yDif*(row));
                double botLon = upLon + (xDif*(col));
                txt += botLat + "," + botLon;

                return txt;
       }

}
```

i. opentripplanner-routing/src/main/java/org/opentripplanner/routing/algorithm/GenericAStar.java

```
package org.opentripplanner.routing.algorithm;

import java.util.Collection;
import org.opentripplanner.common.pqueue.BinHeap;
import org.opentripplanner.common.pqueue.OTPPriorityQueue;
import org.opentripplanner.common.pqueue.OTPPriorityQueueFactory;
import org.opentripplanner.routing.algorithm.strategies.RemainingWeightHeuristic;
import org.opentripplanner.routing.algorithm.strategies.SearchTerminationStrategy;
import org.opentripplanner.routing.algorithm.strategies.SkipTraverseResultStrategy;
import org.opentripplanner.routing.algorithm.strategies.TrivialRemainingWeightHeuristic;
import org.opentripplanner.routing.core.RoutingContext;
import org.opentripplanner.routing.core.RoutingRequest;
import org.opentripplanner.routing.core.State;
import org.opentripplanner.routing.graph.Edge;
import org.opentripplanner.routing.graph.Vertex;
import org.opentripplanner.routing.services.SPTService;
import org.opentripplanner.routing.spt.DefaultShortestPathTreeFactory;
import org.opentripplanner.routing.spt.ShortestPathTree;
import org.opentripplanner.routing.spt.ShortestPathTreeFactory;
import org.opentripplanner.util.DateUtils;
import org.opentripplanner.util.monitoring.MonitoringStore;
import org.opentripplanner.util.monitoring.MonitoringStoreFactory;
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
```

```
public class GenericAStar implements SPTService {

    private static final Logger LOG = LoggerFactory.getLogger(GenericAStar.class);
    private static final MonitoringStore store = MonitoringStoreFactory.getStore();
    private boolean _verbose = false;
    private ShortestPathTreeFactory _shortestPathTreeFactory = new DefaultShortestPathTreeFactory();
    private SkipTraverseResultStrategy _skipTraversalResultStrategy;
    private SearchTerminationStrategy _searchTerminationStrategy;
    private TraverseVisitor traverseVisitor;

    public void setShortestPathTreeFactory(ShortestPathTreeFactory shortestPathTreeFactory) {
        _shortestPathTreeFactory = shortestPathTreeFactory;
    }

    public void setSkipTraverseResultStrategy(SkipTraverseResultStrategy skipTraversalResultStrategy) {
        _skipTraversalResultStrategy = skipTraversalResultStrategy;
    }

    public void setSearchTerminationStrategy(SearchTerminationStrategy searchTerminationStrategy) {
        _searchTerminationStrategy = searchTerminationStrategy;
    }

    public ShortestPathTree getShortestPathTree(RoutingRequest req) {
        return getShortestPathTree(req, -1, _searchTerminationStrategy);
    }

    public ShortestPathTree getShortestPathTree(RoutingRequest req, double timeoutSeconds) {
        return this.getShortestPathTree(req, timeoutSeconds, _searchTerminationStrategy);
    }

    public ShortestPathTree getShortestPathTree(RoutingRequest options, double relTimeout,
            SearchTerminationStrategy terminationStrategy) {

        RoutingContext rctx = options.getRoutingContext();
        long abortTime = DateUtils.absoluteTimeout(relTimeout);

        ShortestPathTree spt = createShortestPathTree(options);

        final RemainingWeightHeuristic heuristic = options.batch ?
                new TrivialRemainingWeightHeuristic() : rctx.remainingWeightHeuristic;

        State initialState = new State(options);
        heuristic.initialize(initialState, rctx.target);
        spt.add(initialState);
        OTPPriorityQueueFactory qFactory = BinHeap.FACTORY;
        int initialSize = rctx.graph.getVertices().size();
        initialSize = (int) Math.ceil(2 * (Math.sqrt((double) initialSize + 1)));
        OTPPriorityQueue<State> pq = qFactory.create(initialSize);
        pq.insert(initialState, 0);
        int nVisited = 0;

        while (!pq.empty()) { // Until the priority queue is empty:
            if (_verbose) {
                double w = pq.peek_min_key();
                System.out.println("pq min key = " + w);
            }

            if (abortTime < Long.MAX_VALUE  && System.currentTimeMillis() > abortTime) {
                LOG.warn("Search timeout. origin={} target={}", rctx.origin, rctx.target);
                storeMemory();
                return null; // throw timeout exception
            }

            // get the lowest-weight state in the queue
            State u = pq.extract_min();

            // check that this state has not been dominated
            // and mark vertex as visited
            if (!spt.visit(u)) {
                // state has been dominated since it was added to the priority queue, so it is
                // not in any optimal path. drop it on the floor and try the next one.
                continue;
            }

            if (traverseVisitor != null) {
                traverseVisitor.visitVertex(u);
            }

            Vertex u_vertex = u.getVertex();
```

```
        if (_verbose)
            System.out.println("   vertex " + u_vertex);

        if (terminationStrategy != null) {
            if (!terminationStrategy.shouldSearchContinue(
                rctx.origin, rctx.target, u, spt, options))
                break;

        } else if (!options.batch && u_vertex == rctx.target && u.isFinal()
                            && u.allPathParsersAccept()) {
            LOG.debug("total vertices visited {}", nVisited);
            storeMemory();
            return spt;
        }

        Collection<Edge> edges = options.isArriveBy() ?
                                    u_vertex.getIncoming() : u_vertex.getOutgoing();

        nVisited += 1;
        for (Edge edge : edges) {

            for (State v = edge.traverse(u); v != null; v = v.getNextResult()) {

                if (traverseVisitor != null) { traverseVisitor.visitEdge(edge, v); }

                if (_skipTraversalResultStrategy != null
                        && _skipTraversalResultStrategy.shouldSkipTraversalResult(
                                rctx.origin, rctx.target, u, v, spt, options)) {
                    continue;
                }

                double remaining_w = computeRemainingWeight(heuristic, v, rctx.target, options);
                if (remaining_w < 0 || Double.isInfinite(remaining_w) ) { continue; }
                double estimate = v.getWeight() + remaining_w;

                if (_verbose) {
                    System.out.println("      edge " + edge);
                    System.out.println("      " + u.getWeight() + " -> " + v.getWeight()
                            + "(w) + " + remaining_w + "(heur) = " + estimate + " vert = "
                            + v.getVertex());
                }

                if (estimate > options.maxWeight) {
                    if (_verbose)
                        System.out.println("too expensive to reach. estimated weight = " + estimate);
                } else if (isWorstTimeExceeded(v, options)) {
                   if (_verbose)
                        System.out.println("too much time to reach. time = " + v.getTimeSeconds());
                } else {
                    if (spt.add(v)) {
                        if (traverseVisitor != null)
                            traverseVisitor.visitEnqueue(v);
                        pq.insert(v, estimate);
                    }
                }
            }
        }
    }
    storeMemory();
    return spt;
}

private void storeMemory() {
    if (store.isMonitoring("memoryUsed")) {
        System.gc();
        long memoryUsed = Runtime.getRuntime().totalMemory() -
                Runtime.getRuntime().freeMemory();
        store.setLongMax("memoryUsed", memoryUsed);
    }
}

private double computeRemainingWeight(final RemainingWeightHeuristic heuristic, State v,
        Vertex target, RoutingRequest options) {

 if (options.isArriveBy()) {
        return heuristic.computeReverseWeight(v, target);
    } else {
        return heuristic.computeForwardWeight(v, target);
    }
```

```
    }

    private boolean isWorstTimeExceeded(State v, RoutingRequest opt) {
        if (opt.isArriveBy())
            return v.getTimeSeconds() < opt.worstTime;
        else
            return v.getTimeSeconds() > opt.worstTime;
    }

    private ShortestPathTree createShortestPathTree(RoutingRequest opts) {
        return _shortestPathTreeFactory.create(opts);
    }

    public void setTraverseVisitor(TraverseVisitor traverseVisitor) {
        this.traverseVisitor = traverseVisitor;
    }
}
```

j. opentripplanner-routing/src/main/java/org/opentripplanner/routing/algorithm/strategies/DefaultRemainingWeightHeuristic.java

```
package org.opentripplanner.routing.algorithm.strategies;

import org.opentripplanner.common.geometry.DistanceLibrary;
import org.opentripplanner.common.geometry.SphericalDistanceLibrary;
import org.opentripplanner.routing.core.OptimizeType;
import org.opentripplanner.routing.core.State;
import org.opentripplanner.routing.core.TraverseMode;
import org.opentripplanner.routing.core.RoutingRequest;
import org.opentripplanner.routing.graph.Graph;
import org.opentripplanner.routing.graph.Vertex;
import org.opentripplanner.routing.vertextype.IntersectionVertex;
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;

/**
 * A Euclidean remaining weight strategy that takes into account transit boarding costs where applicable.
 *
 */
public class DefaultRemainingWeightHeuristic implements RemainingWeightHeuristic {

    private static final long serialVersionUID = -5172878150967231550L;
    private RoutingRequest options;
    private boolean useTransit = false;
    private double maxSpeed;
    private DistanceLibrary distanceLibrary = SphericalDistanceLibrary.getInstance();
    private static Logger LOG = LoggerFactory.getLogger(LBGRemainingWeightHeuristic.class);
    private TransitLocalStreetService localStreetService;
    private double targetX;
    private double targetY;

    @Override
    public void initialize(State s, Vertex target) {
        this.options = s.getOptions();
        this.useTransit = options.getModes().isTransit();
        this.maxSpeed = getMaxSpeed(options);
        Graph graph = options.rctx.graph;
        localStreetService = graph.getService(TransitLocalStreetService.class);
        targetX = target.getX();
        targetY = target.getY();
    }


    @Override
    public double computeForwardWeight(State s, Vertex target) {
        Vertex sv = s.getVertex();
        double euclideanDistance = distanceLibrary.fastDistance(sv.getY(), sv.getX(), targetY, targetX);
        if (useTransit) {
            double streetSpeed = options.getStreetSpeedUpperBound();
            if (s.isAlightedLocal() || euclideanDistance < target.getDistanceToNearestTransitStop()) {
                if (euclideanDistance + s.getWalkDistance() > options.getMaxWalkDistance()) {
                    return -1; // impossible to reach destination
                }
                return options.walkReluctance * euclideanDistance / streetSpeed;
            }

            int boardCost;
            if (s.isOnboard()) {
                boardCost = 0;
```

```
            } else {
                boardCost = options.getBoardCostLowerBound();
                if (s.isEverBoarded()) {
                    boardCost += options.transferPenalty;
                    if (localStreetService != null) {
                        if (options.getMaxWalkDistance() - s.getWalkDistance() < euclideanDistance
                                && sv instanceof IntersectionVertex
                                && !localStreetService.transferrable(sv)) {
                            return Double.POSITIVE_INFINITY;
                        }
                    }
                }
            }
            double mandatoryWalkDistance = target.getDistanceToNearestTransitStop()
                    + sv.getDistanceToNearestTransitStop();
            double transitCost = (euclideanDistance - mandatoryWalkDistance) / maxSpeed + boardCost;
            double transitStreetCost = mandatoryWalkDistance * options.walkReluctance / streetSpeed;
            return Math.min(transitCost + transitStreetCost,
                            options.walkReluctance * euclideanDistance / streetSpeed);
        } else {
            return options.walkReluctance * euclideanDistance / maxSpeed;
        }
    }

    @Override
    public double computeReverseWeight(State s, Vertex target) { return computeForwardWeight(s, target); }

    public static double getMaxSpeed(RoutingRequest options) {
        if (options.getModes().contains(TraverseMode.TRANSIT)) {
            return 10;
        } else {
            if (options.optimize == OptimizeType.QUICK) {
                return options.getStreetSpeedUpperBound();
            } else {
                return options.getStreetSpeedUpperBound() * 10;
            }
        }
    }

    @Override
    public void reset() {}

    @Override
    public void abort() {}
}
```

k. opentripplanner-routing/src/main/java/org/opentripplanner/routing/graph/Graph.java

```
package org.opentripplanner.routing.graph;

import static org.opentripplanner.common.IterableLibrary.filter;
import java.io.BufferedInputStream;
import java.io.BufferedOutputStream;
import java.io.File;
import java.io.FileInputStream;
import java.io.FileOutputStream;
import java.io.IOException;
import java.io.InputStream;
import java.io.InvalidClassException;
import java.io.ObjectInputStream;
import java.io.ObjectOutputStream;
import java.io.ObjectStreamClass;
import java.io.Serializable;
import java.util.ArrayList;
import java.util.Collection;
import java.util.Collections;
import java.util.Date;
import java.util.HashMap;
import java.util.HashSet;
import java.util.LinkedList;
import java.util.List;
import java.util.Map;
import java.util.Set;
import java.util.TimeZone;
import java.util.concurrent.ConcurrentHashMap;
import com.google.common.collect.Iterables;
import org.onebusaway.gtfs.impl.calendar.CalendarServiceImpl;
import org.onebusaway.gtfs.model.Agency;
import org.onebusaway.gtfs.model.AgencyAndId;
```

```
import org.onebusaway.gtfs.model.Stop;
import org.onebusaway.gtfs.model.calendar.CalendarServiceData;
import org.onebusaway.gtfs.model.calendar.ServiceDate;
import org.onebusaway.gtfs.services.calendar.CalendarService;
import org.opentripplanner.common.MavenVersion;
import org.opentripplanner.gbannotation.GraphBuilderAnnotation;
import org.opentripplanner.gbannotation.NoFutureDates;
import org.opentripplanner.model.GraphBundle;
import org.opentripplanner.routing.core.MortonVertexComparatorFactory;
import org.opentripplanner.routing.core.TransferTable;
import org.opentripplanner.routing.edgetype.StreetEdge;
import org.opentripplanner.routing.edgetype.TimetableSnapshotSource;
import org.opentripplanner.routing.impl.DefaultStreetVertexIndexFactory;
import org.opentripplanner.routing.services.StreetVertexIndexFactory;
import org.opentripplanner.routing.services.StreetVertexIndexService;
import org.opentripplanner.routing.vertextype.TransitStop;
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import com.google.common.collect.HashMultiset;
import com.google.common.collect.Lists;
import com.google.common.collect.Multiset;
import com.vividsolutions.jts.geom.Envelope;

/**
 * A graph is really just one or more indexes into a set of vertexes. It used to keep edgelists for each
vertex, but those are in the vertex now.
 */
public class Graph implements Serializable {

    private static final long serialVersionUID = MavenVersion.VERSION.getUID();
    private final MavenVersion mavenVersion = MavenVersion.VERSION;
    private static final Logger LOG = LoggerFactory.getLogger(Graph.class);
    private long transitServiceStarts = Long.MAX_VALUE;
    private long transitServiceEnds = 0;
    private Map<Class<?>, Object> _services = new HashMap<Class<?>, Object>();
    private TransferTable transferTable = new TransferTable();
    private GraphBundle bundle;
    private transient Map<String, Vertex> vertices;
    private transient CalendarService calendarService;
    private boolean debugData = true;
    private transient Map<Integer, Vertex> vertexById;
    private transient Map<Integer, Edge> edgeById;
    public transient StreetVertexIndexService streetIndex;
    public transient TimetableSnapshotSource timetableSnapshotSource = null;
    private transient List<GraphBuilderAnnotation> graphBuilderAnnotations = new
LinkedList<GraphBuilderAnnotation>(); // initialize for tests
    private Collection<String> agenciesIds = new HashSet<String>();
    private Collection<Agency> agencies = new HashSet<Agency>();
    private transient Set<Edge> temporaryEdges;
    private VertexComparatorFactory vertexComparatorFactory = new MortonVertexComparatorFactory();
    private transient TimeZone timeZone = null;
    private List<Stop> stops;
    public Graph(Graph basedOn) {
        this();
        this.bundle = basedOn.getBundle();
    }

    public Graph() {
        this.vertices = new ConcurrentHashMap<String, Vertex>();
        this.temporaryEdges = Collections.newSetFromMap(new ConcurrentHashMap<Edge, Boolean>());
        this.edgeById = new ConcurrentHashMap<Integer, Edge>();
        this.vertexById = new ConcurrentHashMap<Integer, Vertex>();
    }

    public List<Stop> getAllStops() {
        if (stops == null) {
            stops = Lists.newArrayList();
            for (TransitStop stopv : Iterables.filter(vertices.values(), TransitStop.class)) {
                stops.add(stopv.getStop());
            }
        }
        return stops;
    }

    public void addVertex(Vertex v) {
        Vertex old = vertices.put(v.getLabel(), v);
        if (old != null) {
            if (old == v)
                LOG.error("repeatedly added the same vertex: {}", v);
```

```
        else
            LOG.error("duplicate vertex label in graph (added vertex to graph anyway): {}", v);
    }
}

public void removeVertex(Vertex v) {
    if (vertices.remove(v.getLabel()) != v) {
        LOG.error(
                "attempting to remove vertex that is not in graph (or mapping value was null): {}",
                v);
    }
}

@Deprecated
public Vertex getVertex(String label) { return vertices.get(label);  }

public Vertex getVertexById(int id) { return this.vertexById.get(id); }

public Collection<Vertex> getVertices() { return this.vertices.values(); }

public Edge getEdgeById(int id) { return edgeById.get(id); }

public Collection<Edge> getEdges() {
    Set<Edge> edges = new HashSet<Edge>();
    for (Vertex v : this.getVertices()) {
        edges.addAll(v.getOutgoing());
    }
    return edges;
}

public Collection<StreetEdge> getStreetEdges() {
    Collection<Edge> allEdges = this.getEdges();
    return Lists.newArrayList(filter(allEdges, StreetEdge.class));
}

public boolean containsVertex(Vertex v) { return (v != null) && vertices.get(v.getLabel()) == v; }

@SuppressWarnings("unchecked")
public <T> T putService(Class<T> serviceType, T service) {
    return (T) _services.put(serviceType, service);
}

public boolean hasService(Class<?> serviceType) {
    return _services.containsKey(serviceType);
}

@SuppressWarnings("unchecked")
public <T> T getService(Class<T> serviceType) {
    return (T) _services.get(serviceType);
}

public void remove(Vertex vertex) {
    vertices.remove(vertex.getLabel());
}

public void removeVertexAndEdges(Vertex vertex) {
    if (!containsVertex(vertex)) {
        throw new IllegalStateException("attempting to remove vertex that is not in graph.");
    }
    for (Edge e : vertex.getIncoming()) {
        temporaryEdges.remove(e);
    }
    for (Edge e : vertex.getOutgoing()) {
        temporaryEdges.remove(e);
    }
    vertex.removeAllEdges();
    this.remove(vertex);
}

public Envelope getExtent() {
    Envelope env = new Envelope();
    for (Vertex v : getVertices()) {
        env.expandToInclude(v.getCoordinate());
    }
    return env;
}

public TransferTable getTransferTable() {
    return transferTable;
```

```
}

public void updateTransitFeedValidity(CalendarServiceData data) {
    long now = new Date().getTime() / 1000;
    final long SEC_IN_DAY = 24 * 60 * 60;
    HashSet<String> agenciesWithFutureDates = new HashSet<String>();
    HashSet<String> agencies = new HashSet<String>();
    for (AgencyAndId sid : data.getServiceIds()) {
        agencies.add(sid.getAgencyId());
        for (ServiceDate sd : data.getServiceDatesForServiceId(sid)) {
            long t = sd.getAsDate().getTime() / 1000;
            if (t > now) {
                agenciesWithFutureDates.add(sid.getAgencyId());
            }
            long u = t + SEC_IN_DAY;
            if (t < this.transitServiceStarts)
                this.transitServiceStarts = t;
            if (u > this.transitServiceEnds)
                this.transitServiceEnds = u;
        }
    }
    for (String agency : agencies) {
        if (!agenciesWithFutureDates.contains(agency)) {
            LOG.warn(this.addBuilderAnnotation(new NoFutureDates(agency)));
        }
    }
}

public boolean transitFeedCovers(long t) {
    return t >= this.transitServiceStarts && t < this.transitServiceEnds;
}

public GraphBundle getBundle() { return bundle; }

public void setBundle(GraphBundle bundle) { this.bundle = bundle; }

public int countVertices() { return vertices.size(); }

public int countEdges() {
    int ne = 0;
    for (Vertex v : getVertices()) {
        ne += v.getDegreeOut();
    }
    return ne;
}

private void addEdgesToIndex(Collection<Edge> es) {
    for (Edge e : es) {
        this.edgeById.put(e.getId(), e);
    }
}

public void rebuildVertexAndEdgeIndices() {
    this.vertexById = new HashMap<Integer, Vertex>(AbstractVertex.getMaxIndex());
    Collection<Vertex> vertices = getVertices();
    for (Vertex v : vertices) {
        vertexById.put(v.getIndex(), v);
    }

    this.edgeById = new HashMap<Integer, Edge>();
    for (Vertex v : vertices) {
        if (v == null) {
            continue;
        }
        addEdgesToIndex(v.getOutgoing());
    }
}

private void readObject(ObjectInputStream inputStream) throws ClassNotFoundException, IOException {
    inputStream.defaultReadObject();
    temporaryEdges = Collections.newSetFromMap(new ConcurrentHashMap<Edge, Boolean>());
}

public String addBuilderAnnotation(GraphBuilderAnnotation gba) {
    String ret = gba.getMessage();
    if (this.graphBuilderAnnotations != null)
        this.graphBuilderAnnotations.add(gba);
    return ret;
}
```

```
public List<GraphBuilderAnnotation> getBuilderAnnotations() {
    return this.graphBuilderAnnotations;
}

public enum LoadLevel {
    BASIC, FULL, DEBUG;
}

public static Graph load(File file, LoadLevel level) throws IOException, ClassNotFoundException {
    LOG.info("Reading graph " + file.getAbsolutePath() + " ...");
    ObjectInputStream in = new ObjectInputStream(new FileInputStream(file));
    return load(in, level);
}

public static Graph load(ClassLoader classLoader, File file, LoadLevel level)
                                        throws IOException, ClassNotFoundException {
    LOG.info("Reading graph " + file.getAbsolutePath() + " with alternate classloader ...");
    ObjectInputStream in = new GraphObjectInputStream(new BufferedInputStream(
            new FileInputStream(file)), classLoader);
    return load(in, level);
}

public static Graph load(InputStream is, LoadLevel level) throws ClassNotFoundException, IOException {
    return load(new ObjectInputStream(is), level);
}

public static Graph load(ObjectInputStream in, LoadLevel level) throws IOException,
        ClassNotFoundException {
    return load(in, level, new DefaultStreetVertexIndexFactory());
}

@SuppressWarnings("unchecked")
public static Graph load(ObjectInputStream in, LoadLevel level,
        StreetVertexIndexFactory indexFactory) throws IOException, ClassNotFoundException {
    try {
        Graph graph = (Graph) in.readObject();
        LOG.debug("Basic graph info read.");
        if (graph.graphVersionMismatch())
            throw new RuntimeException("Graph version mismatch detected.");
        if (level == LoadLevel.BASIC)
            return graph;
        LOG.debug("Loading edges...");
        List<Edge> edges = (ArrayList<Edge>) in.readObject();
        graph.vertices = new HashMap<String, Vertex>();

        for (Edge e : edges) {
            graph.vertices.put(e.getFromVertex().getLabel(), e.getFromVertex());
            graph.vertices.put(e.getToVertex().getLabel(), e.getToVertex());
        }
        for (Vertex v : graph.getVertices())
            v.compact();
        LOG.info("Main graph read. |V|={} |E|={}", graph.countVertices(), graph.countEdges());

        graph.streetIndex = indexFactory.newIndex(graph);
        LOG.debug("street index built.");

        LOG.debug("Rebuilding edge and vertex indices");
        graph.rebuildVertexAndEdgeIndices();

        if (level == LoadLevel.FULL) {
            return graph;
        }

        if (graph.debugData) {
            graph.graphBuilderAnnotations = (List<GraphBuilderAnnotation>) in.readObject();
            LOG.debug("Debug info read.");
        } else {
            LOG.warn("Graph file does not contain debug data.");
        }
        return graph;
    } catch (InvalidClassException ex) {
        LOG.error("Stored graph is incompatible with this version of OTP, please rebuild it.");
        throw new IllegalStateException("Stored Graph version error", ex);
    }
}

private boolean graphVersionMismatch() {
    MavenVersion v = MavenVersion.VERSION;
```

```
    MavenVersion gv = this.mavenVersion;
    LOG.info("Graph version: {}", gv);
    LOG.info("OTP version:   {}", v);
    if (!v.equals(gv)) {
        LOG.error("This graph was built with a different version of OTP. Please rebuild it.");
        return true;
    } else if (!v.commit.equals(gv.commit)) {
        if (v.qualifier.equals("SNAPSHOT")) {
            LOG.warn("This graph was built with the same SNAPSHOT version of OTP, but a "
                    + "different commit. Please rebuild the graph if you experience incorrect "
                    + "behavior. ");
            return false;
        } else {
            LOG.error("Commit mismatch in non-SNAPSHOT version. This implies a problem with "
                    + "the build or release process.");
            return true;
        }
    } else {
        LOG.info("This graph was built with the currently running version and commit of OTP.");
        return false;
    }
}

public void save(File file) throws IOException {
    LOG.info("Main graph size: |V|={} |E|={}", this.countVertices(), this.countEdges());
    LOG.info("Writing graph " + file.getAbsolutePath() + " ...");
    ObjectOutputStream out = new ObjectOutputStream(new BufferedOutputStream(
            new FileOutputStream(file)));
    try {
        save(out);
        out.close();
    } catch (RuntimeException e) {
        out.close();
        file.delete(); // remove half-written file
        throw e;
    }
}

public void save(ObjectOutputStream out) throws IOException {
    LOG.debug("Consolidating edges...");
    List<Edge> edges = new ArrayList<Edge>(this.countEdges());
    for (Vertex v : getVertices()) {
        edges.addAll(v.getOutgoing());
        if (v.getDegreeOut() + v.getDegreeIn() == 0)
            LOG.debug("vertex {} has no edges, it will not survive serialization.", v);
    }
    LOG.debug("Assigning vertex/edge ID numbers...");
    this.rebuildVertexAndEdgeIndices();
    LOG.debug("Writing edges...");
    out.writeObject(this);
    out.writeObject(edges);
    if (debugData) {
        LOG.debug("Writing debug data...");
        out.writeObject(this.graphBuilderAnnotations);
        out.writeObject(this.vertexById);
        out.writeObject(this.edgeById);
    } else {
        LOG.debug("Skipping debug data.");
    }
    LOG.info("Graph written.");
}

private static class GraphObjectInputStream extends ObjectInputStream {
    ClassLoader classLoader;

    public GraphObjectInputStream(InputStream in, ClassLoader classLoader) throws IOException {
        super(in);
        this.classLoader = classLoader;
    }

    @Override
    public Class<?> resolveClass(ObjectStreamClass osc) {
        try {
            return Class.forName(osc.getName(), false, classLoader);
        } catch (ClassNotFoundException e) {
            throw new RuntimeException(e);
        }
    }
}
```

```
public Integer getIdForEdge(Edge edge) {
    return edge.getId();
}

public CalendarService getCalendarService() {
    if (calendarService == null) {
        CalendarServiceData data = this.getService(CalendarServiceData.class);
        if (data != null) {
            CalendarServiceImpl calendarService = new CalendarServiceImpl();
            calendarService.setData(data);
            this.calendarService = calendarService;
        }
    }
    return this.calendarService;
}

public int removeEdgelessVertices() {
    int removed = 0;
    List<Vertex> toRemove = new LinkedList<Vertex>();
    for (Vertex v : this.getVertices())
        if (v.getDegreeOut() + v.getDegreeIn() == 0)
            toRemove.add(v);
    for (Vertex v : toRemove) {
        this.remove(v);
        removed += 1;
        LOG.trace("removed edgeless vertex {}", v);
    }
    return removed;
}

public Collection<String> getAgencyIds() { return agenciesIds; }

public Collection<Agency> getAgencies() { return agencies; }

public void addAgency(Agency agency) {
    agencies.add(agency);
    agenciesIds.add(agency.getId());
}

public void addTemporaryEdge(Edge edge) { temporaryEdges.add(edge); }

public void removeTemporaryEdge(Edge edge) {
    if (edge.getFromVertex() == null || edge.getToVertex() == null) { return; }
    temporaryEdges.remove(edge);
}

public Collection<Edge> getTemporaryEdges() { return temporaryEdges; }

public VertexComparatorFactory getVertexComparatorFactory() { return vertexComparatorFactory; }

public void setVertexComparatorFactory(VertexComparatorFactory vertexComparatorFactory) {
    this.vertexComparatorFactory = vertexComparatorFactory;
}

public TimeZone getTimeZone() {
    if (timeZone == null) {
        Collection<String> agencyIds = this.getAgencyIds();
        if (agencyIds.size() == 0) {
            timeZone = TimeZone.getTimeZone("GMT");
            LOG.warn("graph contains no agencies; API request times will be interpreted as GMT.");
        } else {
            CalendarService cs = this.getCalendarService();
            for (String agencyId : agencyIds) {
                TimeZone tz = cs.getTimeZoneForAgencyId(agencyId);
                if (timeZone == null) {
                    LOG.debug("graph time zone set to {}", tz);
                    timeZone = tz;
                } else if (!timeZone.equals(tz)) {
                    LOG.error("agency time zone differs from graph time zone: {}", tz);
                }
            }
        }
    }
    return timeZone;
}

public void summarizeBuilderAnnotations() {
    List<GraphBuilderAnnotation> gbas = this.graphBuilderAnnotations;
```

```
        Multiset<Class<? extends GraphBuilderAnnotation>> classes = HashMultiset.create();
        LOG.info("Summary (number of each type of annotation):");
        for (GraphBuilderAnnotation gba : gbas)
            classes.add(gba.getClass());
        for (Multiset.Entry<Class<? extends GraphBuilderAnnotation>> e : classes.entrySet()) {
            String name = e.getElement().getSimpleName();
            int count = e.getCount();
            LOG.info("    {} - {}", name, count);
        }
    }

}
```

l. opentripplanner-routing/src/main/java/org/opentripplanner/routing/graph/Vertex.java

```
package org.opentripplanner.routing.graph;

import java.io.Serializable;
import java.util.Collection;
import java.util.List;
import org.opentripplanner.routing.graph.AbstractVertex.ValidEdgeTypes;
import com.vividsolutions.jts.geom.Coordinate;

/**
 * A vertex in the graph. Each vertex has a longitude/latitude location, as well as a set of
 * incoming and outgoing edges.
 */
public interface Vertex extends Serializable, Cloneable {

    public abstract String toString();
    public abstract Collection<Edge> getOutgoing();
    public abstract void addOutgoing(Edge ee);
    public abstract boolean removeOutgoing(Edge ee);
    public abstract int getDegreeOut();
    public abstract Collection<Edge> getIncoming();
    public abstract void addIncoming(Edge ee);
    public abstract boolean removeIncoming(Edge ee);
    public abstract int getDegreeIn();

    public abstract double getDistanceToNearestTransitStop();
    public abstract void setDistanceToNearestTransitStop(double distance);
    public abstract double getX();
    public abstract double getY();
    public abstract String getLabel();
    public abstract String getName();
    public void setStreetName(String streetName);
    public abstract Coordinate getCoordinate();
    public abstract double azimuthTo(Coordinate other);
    public abstract double azimuthTo(Vertex other);
    public abstract int getIndex();
    public abstract void setIndex(int index);
    public abstract void setGroupIndex(int groupIndex);
    public abstract int getGroupIndex();
    public abstract List<Edge> getOutgoingStreetEdges();
    public abstract void mergeFrom(Graph graph, Vertex other);
    public abstract void removeAllEdges();
    public abstract int removeTemporaryEdges();
    public abstract void compact();
    public abstract ValidEdgeTypes getValidOutgoingEdgeTypes();
    public abstract ValidEdgeTypes getValidIncomingEdgeTypes();
    public abstract boolean edgeTypesValid();
}
```

m. opentripplanner-routing/src/main/java/org/opentripplanner/routing/graph/AbstractVertex.java

```
package org.opentripplanner.routing.graph;

import java.io.IOException;
import java.io.ObjectInputStream;
import java.io.ObjectOutputStream;
import java.util.ArrayList;
import java.util.Collection;
import java.util.List;
import java.util.Set;
import java.util.concurrent.CopyOnWriteArraySet;
import javax.xml.bind.annotation.XmlTransient;
import org.opentripplanner.common.MavenVersion;
import org.opentripplanner.common.geometry.DirectionUtils;
import org.opentripplanner.routing.edgetype.StreetEdge;
```

```
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import com.vividsolutions.jts.geom.Coordinate;

public abstract class AbstractVertex implements Vertex {

    private static final long serialVersionUID = MavenVersion.VERSION.getUID();
    private static final Logger LOG = LoggerFactory.getLogger(AbstractVertex.class);
    private static int maxIndex = 0;
    private int index;
    private int groupIndex = -1;
    private final String label;
    private String name;
    private final double x;
    private final double y;
    private double distanceToNearestTransitStop = 0;
    private transient Set<Edge> incoming = new CopyOnWriteArraySet<Edge>();
    private transient Set<Edge> outgoing = new CopyOnWriteArraySet<Edge>();

    public AbstractVertex(Graph g, String label, double x, double y) {
        this.label = label;
        this.x = x;
        this.y = y;
        this.index = maxIndex  ++;
        if (g != null)
            g.addVertex(this);
        this.name = "(no name provided)";
    }

    protected AbstractVertex(Graph g, String label, double x, double y, String name) {
        this(g, label, x, y);
        this.name = name;
    }

    @Override
    public String toString() {
        StringBuilder sb = new StringBuilder();
        sb.append("<").append(this.getLabel());
        if (this.getCoordinate() != null) {
            sb.append(" lat,lng=").append(this.getCoordinate().y);
            sb.append(",").append(this.getCoordinate().x);
        }
        sb.append(">");
        return sb.toString();
    }

    public int hashCode() {
        return index;
    }

    @Override
    public void addOutgoing(Edge ee) {
        if (outgoing.contains(ee)) {
            LOG.error("repeatedly added edge {} to vertex {}", ee, this);
        } else {
            outgoing.add(ee);
        }
    }

    @Override
    public boolean removeOutgoing(Edge ee) {
        if (!outgoing.contains(ee)) {
            LOG.error("Removing edge which isn't connected to this vertex");
        }
        boolean removed = outgoing.remove(ee);
        if (outgoing.contains(ee)) {
            LOG.error("edge {} still in edgelist of {} after removed. there must have been multiple
copies.");
        }
        return removed;
    }

    @Override
    public Collection<Edge> getOutgoing() { return outgoing; }

    @Override
    public void addIncoming(Edge ee) {
        if (incoming.contains(ee)) {
            LOG.error("repeatedly added edge {} to vertex {}", ee, this);
```

```
        } else {
            incoming.add(ee);
        }
    }

    @Override
    public boolean removeIncoming(Edge ee) {
        if (!incoming.contains(ee)) {
            LOG.error("Removing edge which isn't connected to this vertex");
        }
        boolean removed = incoming.remove(ee);
        if (incoming.contains(ee)) {
            LOG.error("edge {} still in edgelist of {} after removed. there must have been multiple
copies.");
        }
        return removed;
    }

    @Override
    public Collection<Edge> getIncoming() { return incoming; }

    @Override
    @XmlTransient
    public int getDegreeOut() { return outgoing.size(); }

    @Override
    @XmlTransient
    public int getDegreeIn() { return incoming.size(); }

    @Override
    public void setDistanceToNearestTransitStop(double distance) {
        distanceToNearestTransitStop = distance;
    }

    @Override
    public double getDistanceToNearestTransitStop() { return distanceToNearestTransitStop; }

    @Override
    public double getX() { return x; }

    @Override
    public double getY() { return y; }

    public double getLon() { return x; }

    public double getLat() { return y; }

    @Override
    public void setGroupIndex(int groupIndex) { this.groupIndex = groupIndex; }

    @Override
    @XmlTransient
    public int getGroupIndex() { return groupIndex; }

    @Override
    public String getName() { return this.name; }

    @Override
    public void setStreetName(String name) { this.name = name; }

    @Override
    public String getLabel() { return label; }

    @XmlTransient
    public Coordinate getCoordinate() { return new Coordinate(getX(), getY()); }

    @Override
    public double azimuthTo(Coordinate other) {
              return DirectionUtils.getAzimuth(getCoordinate(), other);
    }

    @Override
    public double azimuthTo(Vertex other) { return azimuthTo(other.getCoordinate()); }

    @XmlTransient
    public int getIndex() { return index; }
```

```
@Override
public void setIndex(int index) { this.index = index; }

public static int getMaxIndex() { return maxIndex; }

private void writeObject(ObjectOutputStream out) throws IOException { out.defaultWriteObject();}

private void readObject(ObjectInputStream in) throws IOException, ClassNotFoundException {
    in.defaultReadObject();
    this.incoming = new CopyOnWriteArraySet<Edge>();
    this.outgoing = new CopyOnWriteArraySet<Edge>();
    index = maxIndex++;
}

@Override
@XmlTransient
public List<Edge> getOutgoingStreetEdges() {
    List<Edge> result = new ArrayList<Edge>();
    for (Edge out : this.getOutgoing()) {
        if (!(out instanceof StreetEdge)) {
            continue;
        }
        result.add((StreetEdge) out);
    }
    return result;
}

@Override
public void mergeFrom(Graph graph, Vertex other) {
    List<Edge> edges = new ArrayList<Edge>();
    edges.addAll(this.getIncoming());
    edges.addAll(this.getOutgoing());
    edges.addAll(other.getIncoming());
    edges.addAll(other.getOutgoing());

    for (Edge e : edges) {
        Vertex from = e.getFromVertex();
        Vertex to = e.getToVertex();
        if ((from==this && to==other) || (from==other && to==this)) {
            e.detach();
        } else if (from == other) {
            e.attach(this, to);
        } else if (to == other) {
            e.attach(from, this);
        }
    }
    graph.removeVertex(other);
}

@Override
public void removeAllEdges() {
    for (Edge e : outgoing) {
        Vertex target = e.getToVertex();
        if (target != null) {
            target.removeIncoming(e);
        }
    }
    for (Edge e : incoming) {
        Vertex source = e.getFromVertex();
        if (source != null) {
            source.removeOutgoing(e);
        }
    }
    incoming = new CopyOnWriteArraySet<Edge>();
    outgoing = new CopyOnWriteArraySet<Edge>();
}


@SuppressWarnings("unchecked")
private static final ValidEdgeTypes VALID_EDGE_TYPES = new ValidEdgeTypes(Edge.class);

@XmlTransient
@Override
public ValidEdgeTypes getValidOutgoingEdgeTypes() { return VALID_EDGE_TYPES; }

@XmlTransient
@Override
public ValidEdgeTypes getValidIncomingEdgeTypes() { return VALID_EDGE_TYPES ; }
```

```
    @Override
    public boolean edgeTypesValid() {
        ValidEdgeTypes validOutgoingTypes = getValidOutgoingEdgeTypes();
        for (Edge e : getOutgoing())
            if (!validOutgoingTypes.isValid(e))
                return false;
        ValidEdgeTypes validIncomingTypes = getValidIncomingEdgeTypes();
        for (Edge e : getIncoming())
            if (!validIncomingTypes.isValid(e))
                return false;
        return true;
    }

    public static final class ValidEdgeTypes {
        private final Class<? extends Edge>[] classes;
        public ValidEdgeTypes (Class<? extends Edge>... classes) {
            this.classes = classes;
        }
        public boolean isValid (Edge e) {
            for (Class<? extends Edge> c : classes) {
                if (c.isInstance(e))
                    return true;
            }
            return false;
        }
    }

    @Override public int removeTemporaryEdges() {
        return 0;
    }

}
```

n. opentripplanner-routing/src/main/java/org/opentripplanner/routing/graph/Edge.java

```
package org.opentripplanner.routing.graph;

import java.io.IOException;
import java.io.ObjectInputStream;
import java.io.ObjectOutputStream;
import java.io.Serializable;
import java.util.ArrayList;
import java.util.Collections;
import java.util.List;
import javax.xml.bind.annotation.XmlTransient;
import lombok.Getter;
import org.onebusaway.gtfs.model.Trip;
import org.opentripplanner.common.MavenVersion;
import org.opentripplanner.routing.core.RoutingRequest;
import org.opentripplanner.routing.core.State;
import org.opentripplanner.routing.edgetype.PlainStreetEdge;
import org.opentripplanner.routing.patch.Patch;
import org.opentripplanner.routing.util.IncrementingIdGenerator;
import org.opentripplanner.routing.util.UniqueIdGenerator;
import com.vividsolutions.jts.geom.LineString;

/**
 * This is the standard implementation of an edge with fixed from and to Vertex instances; all standard OTP
edges are subclasses of this.
 */
public abstract class Edge implements Serializable {

    private static final long serialVersionUID = MavenVersion.VERSION.getUID();
    private static final UniqueIdGenerator<Edge> idGenerator = new IncrementingIdGenerator<Edge>();
    @Getter
    private int id;
    protected Vertex fromv;
    protected Vertex tov;
    private List<Patch> patches;
    protected Edge(Vertex v1, Vertex v2) {
        if (v1 == null || v2 == null) {
            String err = String.format("%s constructed with null vertex : %s %s", this.getClass(),
                    v1, v2);
            throw new IllegalStateException(err);
        }
        this.fromv = v1;
        this.tov = v2;
        this.id = idGenerator.getId(this);
```

```
    fromv.addOutgoing(this);
    tov.addIncoming(this);
}

public Vertex getFromVertex() { return fromv; }

public Vertex getToVertex() { return tov; }

public boolean isPartial() { return false; }

public boolean isEquivalentTo(Edge e) { return this == e; }

public boolean isReverseOf(Edge e) {
    return (this.getFromVertex() == e.getToVertex() &&
            this.getToVertex() == e.getFromVertex());
}

public void attachFrom(Vertex fromv) {
    detachFrom();
    if (fromv == null)
        throw new IllegalStateException("attaching to fromv null");
    this.fromv = fromv;
    fromv.addOutgoing(this);
}

public void attachTo(Vertex tov) {
    detachTo();
    if (tov == null)
        throw new IllegalStateException("attaching to tov null");
    this.tov = tov;
    tov.addIncoming(this);
}

public void attach(Vertex fromv, Vertex tov) {
    attachFrom(fromv);
    attachTo(tov);
}

public String getDirection() { return null; }

protected boolean detachFrom() {
    boolean detached = false;
    if (fromv != null) {
        detached = fromv.removeOutgoing(this);
        fromv = null;
    }
    return detached;
}

protected boolean detachTo() {
    boolean detached = false;
    if (tov != null) {
        detached = tov.removeIncoming(this);
        tov = null;
    }
    return detached;
}

public int detach() {
    int nDetached = 0;
    if (detachFrom()) {
        ++nDetached;
    }
    if (detachTo()) {
        ++nDetached;
    }
    return nDetached;
}

public Trip getTrip() { return null; }

@Override
public int hashCode() { return fromv.hashCode() * 31 + tov.hashCode(); }

public boolean isRoundabout() { return false; }

public abstract State traverse(State s0);

public State optimisticTraverse(State s0) {
```

```
    return this.traverse(s0);
}

public double weightLowerBound(RoutingRequest options) { return 0; }

public double timeLowerBound(RoutingRequest options) { return 0; }

public void addPatch(Patch patch) {
    if (patches == null) {
        patches = new ArrayList<Patch>();
    }
    if (!patches.contains(patch)) {
        patches.add(patch);
    }
}

public List<Patch> getPatches() {
    if (patches == null) {
        return Collections.emptyList();
    }
    return patches;
}

public void removePatch(Patch patch) {
    if (patches == null || patches.size() == 1) {
        patches = null;
    } else {
        patches.remove(patch);
    }
}

public abstract String getName();

public boolean hasBogusName() {
    return false;
}

public String toString() {
    if (id >= 0) {
        return String.format("%s:%s (%s -> %s)", getClass().getName(), id, fromv, tov);
    }
    return String.format("%s (%s -> %s)", getClass().getName(), fromv, tov);
}

public LineString getGeometry() { return null; }

public double getAzimuth() {
    return getFromVertex().azimuthTo(getToVertex());
}

public double getDistance() {
    return 0;
}

private void readObject(ObjectInputStream in) throws IOException, ClassNotFoundException {
    in.defaultReadObject();
    fromv.addOutgoing(this);
    tov.addIncoming(this);
}

private void writeObject(ObjectOutputStream out) throws IOException, ClassNotFoundException {
    if (fromv == null) {
        if (this instanceof PlainStreetEdge)
            System.out.println(((PlainStreetEdge) this).getGeometry());
        System.out.printf("fromv null %s \n", this);
    }
    if (tov == null) {
        if (this instanceof PlainStreetEdge)
            System.out.println(((PlainStreetEdge) this).getGeometry());
        System.out.printf("tov null %s \n", this);
    }
    out.defaultWriteObject();
}

@SuppressWarnings("unchecked")
private static final ValidVertexTypes VALID_VERTEX_TYPES = new ValidVertexTypes(Vertex.class,
        Vertex.class);

@XmlTransient
```

```
    public ValidVertexTypes getValidVertexTypes() {
        return VALID_VERTEX_TYPES;
    }

    public final boolean vertexTypesValid() {
        return getValidVertexTypes().isValid(fromv, tov);
    }

    public static final class ValidVertexTypes {
        private final Class<? extends Vertex>[] classes;
        public ValidVertexTypes(Class<? extends Vertex>... classes) {
            if (classes.length % 2 != 0) {
                throw new IllegalStateException("from/to/from/to...");
            } else {
                this.classes = classes;
            }
        }

        public boolean isValid(Vertex from, Vertex to) {
            for (int i = 0; i < classes.length; i += 2) {
                if (classes[i].isInstance(from) && classes[i + 1].isInstance(to))
                    return true;
            }
            return false;
        }
    }

}
```

3. Web Application

a. aroundmetro/application/views /template.php

```
<head>
<!-- Bootstrap -->
      <link href=<?php echo base_url("css/bootstrap.min.css");?> rel="stylesheet">
      <link href=<?php echo base_url("css/style.css");?> rel="stylesheet">
      <script src=<?php echo base_url("/js/jquery-1.10.2.min.js"); ?>></script>
      <script src=<?php echo base_url("/js/bootstrap.min.js"); ?>></script>
      <script type="text/javascript" src=<?php echo base_url("/fancybox/jquery.fancybox.js?v=2.1.5");
                                                                                    ?>></script>
      <link rel="stylesheet" type="text/css" href=<?php echo
                              base_url("fancybox/jquery.fancybox.css?v=2.1.5");?> media="screen" />
      <link rel="icon" href="<?php echo base_url()?>/favicon.gif" type="image/gif">

      <title><?php echo $title ?></title>
</head>

<?php
      if($this->session->userdata('logged_in')==FALSE && $this->session->userdata('page')!='login'){
               redirect('login');
      }

      if($this->session->userdata('page')!=='login'){
      <div style="position:fixed; top: 2%; width:100%; text-align:right; padding-right: 15%; z-index: 1031">
                        Welcome back, <?php echo $this->session->userdata('username');?>!
      </div>
               <?php
               $page=$this->session->userdata('page');
               if($page=='route' || $page=='trip' || $page=='stop'  || $page=='fare'|| $page=='graph'){
                        $this->load->view('navBar');

                        $action=$this->session->userdata('action');
                        if(!$action){
                                 $this->load->view($page.'/submenu');
                        }
                        else{
                           switch($action){
                                 case 'view': $this->load->view($page.'/viewSubmenu'); break;
                                 case 'add': case 'edit': $this->load->view($page.'/addSubmenu'); break;
                                 case 'referral': $this->load->view($page.'/referralSubmenu'); break;
                                 case 'permission': $this->load->view($page.'/permissionSubmenu'); break;
                           }
                        }
               }
      }
      $this->load->view($main_content);
?>
```

b. aroundmetro/application/views/user/login_view.php

```
<?php

echo form_open('login/checkFields'); ?>
     <link href=<?php echo base_url("css/bootstrap.min.css")?> rel="stylesheet">
     <link href=<?php echo base_url("css/login.css")?> rel="stylesheet">

     <div class="container wrapper">
              <div class="project">
                       <div class="logo" style="height: 17%; width: 25%;">
                                <img style="height: 210%;" src=<?php echo base_url("images/logo.png")?>>
                       </div>
                       <div class="content">
                                <div class="title"> <p>Around Metro</p> </div>
                                <div class="subtitle"> <p>System Administrator</p> </div>
                       </div>
              </div>

              <?php if (validation_errors()) { ?>
                       <div class="error"> <label><?php echo validation_errors(); ?></label></div>
              <?php } ?>
                       <!-- end div for errors-->

              <div class="form-signin"> <!--for errors-->
              <input type="text" class="form-control" name="username" placeholder="Username"
                                        autofocus="autofocus" required>
              <input type="password" class="form-control" name="password" placeholder="Password" required>
                      <button class="btn btn-lg btn-primary btn-block" type="submit">Sign in</button>
              </div>

    <script src=<?php echo base_url("/js/bootstrap.min.js")?>></script>

     </div>
<?php echo form_close(); ?>
```

c. aroundmetro/application/views/route/addRoute_view.php

```
<div class="container">
   <div class="routes-table">
      <div class="panel panel-default data">
         <table class="details" style='font-size: 14px; width: 100%' >
            <tr>
              <th colspan="5"> <span class="glyphicon glyphicon-tasks"></span> Route Information</th>
            </tr>

            <tr><td><input type="hidden" name="routeId" id="routeId"></td></tr>

             <tr>
              <td>Agency</td>
              <td colspan="4">
                       <select name="agency" class="form-control" id='agency' style="width: 30%;" required>
                  <option value='0'>Agency Name</option>
                       <?php
                                foreach($agency as $u){
                                   echo "<option value='".$u['id']."'>".$u['name']."</option>";
                                }
                       ?>
                       </select>
              </td>
             </tr>
             <tr>
              <td>Route Name</td>
              <td colspan="4"><input name="routeShortName" type="text" class="form-control"
                                                          placeholder="Short Name" size="50%" required>
                  <input name="routeLongName" type="text" class="form-control" placeholder="Long Name"
                                                                        size="90%" required>
              </td>
             </tr>
             <tr>
              <td>Route Description</td>
              <td colspan="4"><input name="routeDescription" type="text" class="form-control" size="150%"
                                                                                    required>
             </tr>
             <tr>
              <td>Route URL</td>
```

```
            <td colspan="4"><input name="routeUrl" type="text" class="form-control" size="150%">
            </tr>
            <tr id ="rt">
               <td>Route Type</td>
               <td><input type="radio" name="routeTypeVal" value="1"> Bus</td>
               <td><input type="radio" name="routeTypeVal" value="2"> Jeepney</td>
               <td colspan="2"><input type="radio" name="routeTypeVal" value="4"> Rail</td>
            </tr>
            <tr>
               <td><input type="hidden" name="routeType" id="routeType"> </td>
               <td><input type="hidden" name="routeTypeName" id="routeTypeName"> </td>
               <td><input type="hidden" name="busMarkerVal" id="busMarkerVal"
                                                value="<?php echo $marker[1]['value']?>"> </td>
               <td><input type="hidden" name="jeepMarkerVal" id="jeepMarkerVal"
                                                value="<?php echo $marker[2]['value']?>"> </td>
               <td><input type="hidden" name="railMarkerVal" id="railMarkerVal"
                                                value="<?php echo $marker[4]['value']?>"> </td>
               <td><input type="hidden" name="SUVMarkerVal" id="suvMarkerVal"
                                                value="<?php echo $marker[3]['value']?>"> </td>
            </tr>
         </table>
      </div><!-- /panel panel-default -->
   </div><!-- /user-table -->
</div>

<script>
    $(document).ready(function(){

            $('#thisForm').on('submit', function() {
                    var site_url='<?php echo site_url();?>';
                    $("#thisForm").attr("action",site_url+"route/saveRouteToDatabase");
                    $("#thisForm").attr("method","POST");

                    var agency = $("#agency").val();
                    if(agency == 0){
                            alert("Agency name is required.");
                            $("#agency").focus();
                            return false;
                    }

                    var routeTypeVal= $("input:radio[name='routeTypeVal']:checked").val();
                    var routeMarker;
                    if (!routeTypeVal) {
                            alert("Please select a route type.");
                            $("#rt").addClass("highlight");
                            return false;
                    }else{
                            if(routeTypeVal == 1){
                                    $("#routeType").val("3");
                                    $("#routeTypeName").val("Bus");
                                    routeMarker = "PUB" + $("#busMarkerVal").val();
                            }
                            else if(routeTypeVal == 2){
                                    $("#routeType").val("3");
                                    $("#routeTypeName").val("Jeepney");
                                    routeMarker = "PUJ" + $("#jeepMarkerVal").val();
                            }
                            else if(routeTypeVal == 3){
                                    $("#routeType").val("3");
                                    $("#routeTypeName").val("SUV");
                                    routeMarker = "SUV" + $("#suvMarkerVal").val();
                            }
                            else if(routeTypeVal == 4){
                                    $("#routeType").val("2");
                                    $("#routeTypeName").val("Rail");
                                    routeMarker = $("#railMarkerVal").val();
                            }
                    }
                    var routeId = agency + "_" + routeMarker;
                    $("#routeId").val(routeId);
                    console.log("route id: " + routeId);
            });
    });
</script>
```

d. aroundmetro/application/views/route/addSubmenu.php

```
<div class="submenu">
     <div class="container">
               <div align="right">
                        <form id='thisForm' style="margin-bottom: 0;">
                        <input type=submit class="btn btn-primary"  id="saveButton" value='Save'>
                        <a class="btn btn-default" role="button" id="cancelButton">Cancel</a>
               </div><!-- /tools -->
     </div>
</div><!-- /submenu -->

<script>
     $(document).ready(function(){
               $("#cancelButton").click(function() {
                        window.location.replace("<?php echo site_url('route'); ?>");
               });
     });
</script>
```

e. aroundmetro/application/views/route/editRoute_view.php

```
<div class="container">
   <div class="routes-table">
      <div class="panel panel-default data">
         <table class="details" style='font-size: 14px; width: 100%' >
               <tr>
                  <th colspan="5"> <span class="glyphicon glyphicon-tasks"></span> Route Information</th>
               </tr>

               <tr><td><input type="hidden" name="routeId" id="routeId"></td></tr>
               <tr><td><input type="hidden" name="prevRouteId" id="prevRouteId"
                                                         value="<?php echo $routeId;?>"></td></tr>
               <tr>
                  <td>Agency</td>
                  <td colspan="4">
                     <select name="agency" class="form-control" id='agency' style="width: 30%;"required>
                     <option value='0'>Agency Name</option>
                        <?php
                                 foreach($agency as $u){
                                          echo "<option value='".$u['id']."' ";
                                          if($route['agency'] == $u['id']){
                                                   echo "selected='selected'";
                                          }
                                          echo ">".$u['name']."</option>";
                                   }
                        ?>
                        </select>
                  </td>
               </tr>
               <tr>
                  <td>Route Name</td>
                  <td colspan="4">
                     <input name="routeShortName" type="text" class="form-control" placeholder="Short Name"
                                 value="<?php echo $route['routeShortName'];?>" size="50%" required>
                     <input name="routeLongName" type="text" class="form-control" placeholder="Long Name"
                                 size="90%" value="<?php echo $route['routeLongName'];?>" required>
                  </td>
               </tr>
               <tr>
                  <td>Route Description</td>
                  <td colspan="4"><input name="routeDescription" type="text" class="form-control"
                                 size="150%" value="<?php echo $route['routeDesc'];?>" required>
               </tr>
               <tr>
                  <td>Route URL</td>
                  <td colspan="4"><input name="routeUrl" type="text" class="form-control"
                                                      value="<?php echo $route['routeUrl'];?>" size="150%">
               </tr>
               <tr id ="rt">
                  <td>Route Type</td>
                  <td><input type="radio" name="routeTypeVal" value="1" <?php if($route['routeTypeValue'] ==
                                          "1"){ echo "checked"; }?>> Bus</td>
                  <td><input type="radio" name="routeTypeVal" value="2" <?php if($route['routeTypeValue'] ==
                                             "2"){ echo "checked"; }?>> Jeepney</td>
                  <td><input type="radio" name="routeTypeVal" value="4" <?php if($route['routeTypeValue'] ==
                                                      "4"){ echo "checked"; }?>> Rail</td>
```

```
            </tr>
            <tr>
               <td><input type="hidden" name="routeType" id="routeType"> </td>
               <td><input type="hidden" name="routeTypeName" id="routeTypeName"> </td>
               <td><input type="hidden" name="prevRouteTypeVal" id="prevRouteTypeVal" value="<?php echo
                                                   $route['routeTypeValue'];?>"> </td>

               <td><input type="hidden" name="busMarkerVal" id="busMarkerVal"
                                                   value="<?php echo $marker[1]['value']?>"> </td>
               <td><input type="hidden" name="jeepMarkerVal" id="jeepMarkerVal"
                                                   value="<?php echo $marker[2]['value']?>"> </td>
               <td><input type="hidden" name="railMarkerVal" id="railMarkerVal"
                                                   value="<?php echo $marker[4]['value']?>"> </td>
               <td><input type="hidden" name="SUVMarkerVal" id="suvMarkerVal"
                                                   value="<?php echo $marker[3]['value']?>"> </td>
            </tr>
         </table>
      </div><!-- /panel panel-default -->
   </div><!-- /user-table -->
</div>
</form>

<script>
    $(document).ready(function(){
            var site_url='<?php echo site_url();?>';
            $("#thisForm").attr("action",site_url+"route/update");
            $("#thisForm").attr("method","POST");

            var agency = $("#agency").val();
                    if(agency == 0){
                            alert("Agency name is required.");
                            $("#agency").focus();
                            return false;
                    }

            var routeTypeVal= $("input:radio[name='routeTypeVal']:checked").val();
            var routeMarker;
                    if (!routeTypeVal) {
                            alert("Please select a route type.");
                            $("#rt").addClass("highlight");
                            return false;
                    }else{
                            if(routeTypeVal == 1){
                                    $("#routeType").val("3");
                                    $("#routeTypeName").val("Bus");
                                    routeMarker = "PUB" + $("#busMarkerVal").val();

                            }
                            else if(routeTypeVal == 2){
                                    $("#routeType").val("3");
                                    $("#routeTypeName").val("Jeepney");
                                    routeMarker = "PUJ" + $("#jeepMarkerVal").val();
                            }
                            else if(routeTypeVal == 3){
                                    $("#routeType").val("3");
                                    $("#routeTypeName").val("SUV");
                                    routeMarker = "SUV" + $("#suvMarkerVal").val();
                            }
                            else if(routeTypeVal == 4){
                                    $("#routeType").val("2");
                                    $("#routeTypeName").val("Rail");
                                    routeMarker = $("#railMarkerVal").val();
                            }
                    }

            if(routeTypeVal == $("#prevRouteTypeVal").val()){
                    $("#routeId").val($("#prevRouteId").val());
            }
            else{
                    var routeId = agency + "_" + routeMarker;
                    $("#routeId").val(routeId);
            }
            console.log("new route id: " + routeId);
            console.log("prev route id: " + $("#prevRouteId").val());
    });
</script>
```

f. aroundmetro/application/views/route/route_view.php

```
<div style='display: none'>
   <div id='add_route' style='width: 100%'>
      <div class="panel panel-default">
         <p class="header">Add Route</p>
            <div class="fancy">
              <center><h4><strong><?php echo $this->session->userdata('status'); ?></h4></strong></center>
                 <p align="center">
                       <?php echo $this->session->userdata('message'); ?>
                 </p>

                 <?php  if($this->session->userdata('routeId')){ ?>
                       <p align="center" style='font-size: 14px'>
                                <strong>Route ID: </strong>
                                <?php echo $this->session->userdata('routeId')?>
                       </p>
                 <?php
                       $this->session->unset_userdata('status');
                       $this->session->unset_userdata('routeId');
                    } ?>
              <br/>
              <center>
                 <input type=button id="messageOk" class="btn btn-default btn-primary" value="OK">
              </center>
          </div>
      </div>
   </div>
</div>

<?php if($this->session->userdata('message')){ ?>
     <script type="text/javascript">
     $(document).ready(function() {

         $.fancybox({
                'width': '30%',
                'height': '40%',
                'autoScale': true,
                'transitionIn': 'fade',
                'transitionOut': 'fade',
                'type': 'inline',
                'href': '#add_route',
                'modal': true,
                'autoSize': false
        });
    });

</script>
<?php $this->session->unset_userdata('message'); } ?>

<!-- div for edit route -->
<div style='display: none'>
   <div id='editRoute' style='width: 100%'>
      <div class="panel panel-default">
         <p class="header">Edit Route</p>
              <div class="fancy">
                 <center><h4><strong>
                       <?php echo $this->session->userdata('status_update'); ?>
                 </h4></strong></center>
              <p align="center">
                       <?php echo $this->session->userdata('message_update'); ?>
              </p>

              <?php if($this->session->userdata('message_update')){
                       $this->session->unset_userdata('status_update');
              } ?>

                <br />
              <center>
                 <input type=button id="editOk" class="btn btn-default btn-primary" value="OK">
              </center>
         </div>
      </div>
   </div>
</div> <!--end div for edit route-->

 <?php if($this->session->userdata('message_update')){ ?>
<script type="text/javascript">
```

```
    $(document).ready(function() {

        $.fancybox({
               'width': '30%',
               'height': '33%',
               'autoScale': true,
               'transitionIn': 'fade',
               'transitionOut': 'fade',
               'type': 'inline',
               'href': '#editRoute',
               'modal': true,
               'autoSize': false
        });
    });
</script>

<?php $this->session->unset_userdata('message_update'); } ?>

<!-- div for delete route--->
<div style='display: none'>
    <div id='delete_confirm' style='width: 100%'>
             <div class="panel panel-default">
                 <p class="header">Delete Route</p>
                      <div class="fancy">
                         <center><h4><strong>
                            <?php echo $this->session->userdata('status_del'); ?>
                         </h4></strong></center>

                      <br />
                      <p align="center">
                               <?php echo $this->session->userdata('message_del'); ?>
                      </p>
                      <?php    if($this->session->userdata('routeId')){
                                        $this->session->unset_userdata('status_del');
                                        $this->session->unset_userdata('routeId');
                               } ?>

                      <br>
                      <center>
                         <input type=button id="deleteOk" class="btn btn-default btn-primary" value="OK">
                      </center>
                      </div>
             </div>
    </div>
</div>

<?php if($this->session->userdata('message_del')){ ?>

<script type="text/javascript">
    $(document).ready(function() {

        $.fancybox({
               'width': '30%',
               'height': '40%',
               'autoScale': true,
               'transitionIn': 'fade',
               'transitionOut': 'fade',
               'type': 'inline',
               'href': '#delete_confirm',
               'modal': true,
               'autoSize': false
        });
    });

</script>

<?php $this->session->unset_userdata('message_del'); }?>

<body>
   <div class="container">
      <div class="routes-table">
         <div class="panel panel-default">
            <!-- Default panel contents -->
              <div class="panel-heading">
                 <table class="table routes" style='font-size: 15px'>
                       <th style='width: 15%; text-align: center'>Route ID</th>
                       <th style="text-align: center">Route Description</th>
                 </table>
              </div>
```

```
            </div><!-- /panel panel-default -->
            <div class="panel panel-default records originalTable" id="routeTable">
               <table class="table routes" id="routeTable" style='font-size: 14px'>
                  <tbody>
                        <?php if(count($routes)!=0){
                                 foreach ($routes as $route) {
                                          echo "<tr id =".$route['id'].">
                                          <td style='width: 15%; text-align: center'>".$route['id']."</td>
                                          <td >".$route['desc']."</td>
                                          <td style='display:none'>".$route['routeShortName']."</td>
                                          <td style='display:none'>".$route['routeLongName']."</td>
                                          </tr>";
                                    }
                              }
                              else{
                                 echo "<h5 text align = 'center'>No entries to be displayed!</h5>";
                              } ?>
                  </tbody>
               </table>
            </div><!-- /panel panel-default -->

            <div class="panel panel-default records empty" style="display: none;">
               <table class="table users" style='font-size: 14px'>
                  <tbody> <h5 text align = 'center'>No entries to be displayed!</h5> </tbody>
               </table>
            </div>
         </div><!-- /user-table -->
      </div> <!-- /container -->
   </body>

   <script>
       $(document).ready(function(){
                $('#messageOk').click(function () {
                         $.fancybox.close();
                });

                $('#editOk').click(function () {
                         $.fancybox.close();
                });

                $('#deleteOk').click(function () {
                         $.fancybox.close();
                });
       });
   </script>
```

g. aroundmetro/application/views/route/submenu.php

```
<div class="submenu">
    <div class="container">
             <div class="col-lg-6">
                      <div class="input-group">
                         <input type="text" class="form-control" id="search" placeholder="Search">
                         <span class="input-group-btn">
                                <button type="submit" id="searchBtn" class="btn btn-default searchBtn">
                                         <span class="glyphicon glyphicon-search"></span>
                                </button> </span>
                      </div><!-- /input-group -->
             </div><!-- /.col-lg-6 -->

             <a class="btn btn-default" role="button" href=<?php echo site_url('route/add'); ?>>Add</a>
             <a class="btn btn-default" id="edit" role="button" href=<?php echo site_url('route/edit'); ?>
                                                          disabled="disable">Edit</a>
             <a class="btn btn-default" id="view" role="button" href=<?php echo site_url('route/view'); ?>
                                                          disabled="disable">View</a>
             <a class="btn btn-default" id="delete" role="button" disabled="disable">Delete</a>
    </div>
</div><!-- /submenu -->

  <div style='display: none'>
    <div id='delete_route'>

    </div>
  </div>

<script>
    $(document).ready(function() {
             var selected;
```

```
var id;
var url;
var url2;
var url3;
var mainTable;

$("#cancel").click(function() {
	window.location.replace("<?php echo site_url('route'); ?>");
});

$("#routeTable").delegate( "tr", "click", function() {
	var temp= $(this).find('td').map(function() {
		return $(this).text();
	});

	selected = $(this).hasClass("highlight");
	$("#routeTable tr").removeClass("highlight");
	if(!selected){
		id = $(this).closest('tr').attr('id');
		url = ('<?php echo site_url('route/view'); ?>');
		url2 = ('<?php echo site_url('route/edit'); ?>');
		url3 = ('<?php echo site_url('route/delete'); ?>');

		$(this).addClass("highlight");

	if ($(this).hasClass('inactiveUser')) {
		$('#edit').removeAttr('disabled');
			$('#view').removeAttr('disabled');
			$('#delete').removeAttr('disabled');
	} else{
		$('#edit').removeAttr('disabled');
			$('#view').removeAttr('disabled');
			$('#delete').removeAttr('disabled');
	}

} else {
	id=null;
	$('#edit').attr('disabled', 'disabled');
		$('#view').attr('disabled', 'disabled');
		$('#delete').attr('disabled', 'disabled');
	}

	$("a#view").attr("href",url+"/"+id);
	$('a#edit').attr('href', function() {
		    var newURL =  url2 + "/" + id;
		    return newURL;
	});
});

var timer;

$("#delete").on('click', function(){
	$('#delete_route').load('route/delete', {'routeId': id, 'type': 'Route'});
	$.fancybox({
		'autoDimensions': false,
		'width': '40%',
		'height': '40%',
		'autoSize': false,
		'transitionIn': 'fade',
		'transitionOut': 'fade',
		'type': 'inline',
		'modal': true,
		'href': '#delete_route'
	});
});

$("#searchBtn").on('click', function(){
	var str = $.trim( $("#search").val() );

	if( str != "" ) {
		searchItem();
	} else {
		$('#routeTable .searchFalse').show();
		$('#routeTable tr').removeClass('searchFalse');
	}
});
```

```
                $("#search").keyup(function(e){
                        var str = $.trim( $(this).val() );
                        if( str == "" ) {
                                $('#routeTable .searchFalse').show();
                                $('#routeTable tr').removeClass('searchFalse');
                                $('.originalTable').show();
                                $('.empty').hide();
                        }
                        else{
                                if(e.keyCode == 13){  searchItem(); }
                        }
                });

                $('#delete_route').on('click','#deleteOk',function () {
                        $.ajax({
                                type:"POST",
                                url: '<?php echo site_url('route/deleteRoute'); ?>',
                                async: false,
                                data: {'routeId': id},
                                dataType: "TEXT"
                        });
                        window.location.replace("<?php echo site_url('route'); ?>");
                });

                $('#delete_route').on('click','#cancel',function () {
                        $.fancybox.close();
                });
        });

        function searchItem(){
                var str = $.trim( $("#search").val() );
                console.log('string is: ' + str);

                var regx = /[A-Za-z]/;
                if (regx.test(str)) {
                        $('#routeTable tr').each(function() {
                                var texts = $(this).find('td').map(function() {
                                    return $(this).text();
                                });

                                if(texts[1].toLowerCase().indexOf(str.toLowerCase()) < 0){
                                   if(texts[0].toLowerCase().indexOf(str.toLowerCase()) < 0){
                                      if(texts[2].toLowerCase().indexOf(str.toLowerCase()) < 0){
                                         if(texts[3].toLowerCase().indexOf(str.toLowerCase()) < 0){
                                            $(this).addClass('searchFalse');
                                         }
                                      }
                                   }
                                }
                        });

                        $('#routeTable .searchFalse').hide();
                        if($('#routeTable tr:visible').size()==0){
                                console.log('table is empty. hide user table, show empty');
                                $('.originalTable').hide();
                                $('.empty').show();
                        }
                }
        }
</script>
```

h. aroundmetro/application/views/route/viewRoute_view.php

```
<div class="wrapper">
    <div class="container">
            <div class="holder">
                    <div id="type-user">
                       <div class="panel panel-default info">
                             <table class="view" style='width: 100%;'>
                                <tr><th>
                                   <span class="glyphicon glyphicon-tasks"></span>
                                      Route Information
                                </th></tr>
                              </table>

                             <table class="table details" style='font-size: 14px; width: 100%;'>
                                <tr>
                                      <td>Agency</td>
```

```
                                        <td colspan="2"><?php echo $route['agency'] ?></td>
                                    </tr>
                                    <tr>
                                        <td>Route ID</td>
                                        <td colspan="2"><?php echo $route['id'] ?></td>
                                    </tr>
                                    <tr>
                                        <td>Route Type</td>
                                        <td colspan="2"><?php echo $route['routeTypeName'] ?></td>
                                    </tr>
                                    <tr>
                                        <td>Route Short Name</td>
                                        <td colspan="2"><?php echo $route['routeShortName'] ?></td>
                                    </tr>
                                    <tr>
                                        <td>Route Long Name</td>
                                        <td colspan="2"><?php echo $route['routeLongName'] ?></td>
                                    </tr>
                                    <tr>
                                        <td>Route Description</td>
                                        <td colspan="2"><?php echo $route['routeDesc'] ?></td>
                                    </tr>
                                </table>
                            </div>        <br />
                        </div><!--end type:staff-->
                </div><!-- /holder -->
        <div> <!-- /container -->
    </div><!-- /wrapper -->
```

i. aroundmetro/application/views/route/viewSubmenu.php

```
<div class="submenu">
    <div class="container">
            <div align="right" style="margin-bottom: 0;">
                    <a class="btn btn-primary" id="edit" role="button" href=<?php echo
                                site_url('route/edit'); ?> >Edit</a>
                    <a class="btn btn-default" role="button" id="backButton">Back</a>
            </div><!-- /tools -->
    </div>
</div><!-- /submenu -->

<script>
    $(document).ready(function(){
            $("#backButton").click(function() {
                    window.location.replace("<?php echo site_url('route'); ?>");
            });

            var currentUrl = $(location).attr('href');
            var id = currentUrl.substring(currentUrl.lastIndexOf('/') + 1);
            var url2 = ('<?php echo site_url('route/edit'); ?>');
            $('a#edit').attr('href', function() {
                var newURL =  url2 + "/" + id;
                return newURL;
            });
    });
</script>
```

j. aroundmetro/application/views/trip/addTrip_view.php

```
<head>
    <script src="https://maps.googleapis.com/maps/api/js?v=3.exp&sensor=false"></script>
    <script src=<?php echo base_url("/js/maplace.min.js"); ?>></script>
    <script src=<?php echo base_url("/js/maplace.js"); ?>></script>
</head>

<body>
   <div class="container">
     <div class="routes-table">
        <div class="panel panel-default data">
            <div class="leftPanel">
               <table class="details" style='font-size: 14px; width: 100%' >
                    <tr>
                       <th colspan="5"> <span class="glyphicon glyphicon-tasks"></span>
                              Trip Information</th>
                    </tr>
                    <tr><td></td></tr>
                    <tr><td><strong>Add trip details for route:</strong></td></tr>
```

```
                        <tr>
                           <td>Route Name</td>
                           <td><select name="route" class="form-control" id="routeSelect"
                                                                        style="width: 100%;" required>
                                 <option value='0'></option>
                                   <?php foreach($routes as $u){
                                    echo "<option value='".$u['routeId']."'>".$u['routeName']."</option>";
                                   } ?>
                                 </select>
                           </td>
                        </tr>
                        <tr>
                           <td>Route ID</td>
                           <td><input id="routeId" name="routeId" type="text" class="form-control"
                                                                              size="30%" readonly>
                           </td>
                        </tr>
                        <tr><td></td></tr>
                        <tr>
                           <td><strong> Trip Stops:</strong></td>
                        </tr>
                  </table>
                  <table class="stops" id="tableHeaders" style='width: 100%'>
                           <tr>
                              <th style="text-align: center; width: 62%; font-size: 14px;">Stop Name</th>
                              <th style="text-align: center; width: 18%; font-size: 12px;">Arrival Time</th>
                              <th style="text-align: center; width: 20%; font-size: 12px;">Departure Time</th>
                           </tr>
                  </table>

                  <div class="stopsDiv" style="height:50%; overflow-y: auto;">
                     <table id="stopsTable" class="stops" style='font-size: 14px; width: 100%'>
                        <tbody id="stopsTbody">
                           <tr>
                              <td style="display: none"></td>
                              <td style="display: none"></td>
                              <td style="display: none"></td>
                           </tr>
                         </tbody>
                     </table>
                  </div>
               </div><!-- /panel left panel -->
               <div class="rightPanel">

               <div id="map-canvas" style='width: 100%; height:115%; float:right'></div>
                </div>
            </div><!-- /panel panel-default -->
         </div><!-- /user-table -->
      </div>
        </form>
</body>

<script>
     var stops = <?php echo json_encode($stops); ?>;
               console.log(stops[0]);
               var index = 0;
               var markers = [];
               var rows = [];

               var myLatLng = new google.maps.LatLng(14.583333,120.966667);
           var mapOptions = {
                           zoom: 11,
                  mapTypeId: google.maps.MapTypeId.ROADMAP,
                           center: myLatLng
              };

               var map = new google.maps.Map(document.getElementById('map-canvas'), mapOptions);

     $(document).ready(function() {
               var site_url='<?php echo site_url();?>';

               google.maps.event.addListener(map, 'click', function(event) {
                  placeMarker(event.latLng, stops);
               });

               for(var i=0; i<stops.length; i++){
                           var options = {
                                    strokeColor: "#000000",
                                    strokeOpacity: 1,
```

```
                        strokeWeight: 1,
                        fillColor: "#000000",
                        fillOpacity: 1,
                        map: map,
                        center: new google.maps.LatLng(stops[i]['stopLat'], stops[i]['stopLon']),
                        radius: 10
                };
                var newCircle = new google.maps.Circle(options);
                google.maps.event.addListener(newCircle, 'click', function(event) {
                        placeMarker(event.latLng, stops);
                });
        }

        $("#routeSelect").change(function(){
                $("#routeId").val($("#routeSelect").val());
        });

        $('#thisForm').on('submit', function() {
                event.preventDefault();
                var route = $('#routeSelect').val();
                if(route == 0){
                        alert("Route must not be empty!");
                        $("#routeSelect").focus();
                        return false;
                }

                var stp = $("input[name='stopIds[]']").map(function(){
                        return this.value;
                }).get();
                var arv = $("input[name='arrives[]']").map(function(){
                        return this.value;
                }).get();
                var dpt = $("input[name='departs[]']").map(function(){
                        return this.value;
                }).get();

                var routeId = $("#routeSelect").val();

                if(!validateTime(arv) && !validateTime(dpt)){
                        alert("Arrival time and Departure time must follow pattern HH:MM:SS and
                                                                must be in 12-hour format");
                        return false;
                }
                else{
                        var s = $.ajax({
                                type:"POST",
                                url: site_url+"trip/saveTripToDatabase",
                                async: false,
                                data: {'stopIds': JSON.stringify(stp),
                                                'arrives': JSON.stringify(arv),
                                                'departs': JSON.stringify(dpt),
                                                'routeId': routeId},
                                dataType: "JSON"
                        }).responseText;

                        window.location.replace("<?php echo site_url('trip'); ?>");
                        return false;
                }

        });

        function validateTime(arvs){
                for(var i=0; i < arvs.length; i++){
                        if(arvs[i] != '00:00:00'){
                                if(!arvs[i].match(/^(0[0-9]|1[012]):([0-5]\d):([0-5]\d)$/)
                                        || arvs[i]==""){
                                        return false;
                                }
                        }
                }
                return true;
        }
});

function placeMarker(location, stops) {
        var lat = location.lat().toFixed(6);
        var lon = location.lng().toFixed(6);
        console.log(location);
```

```
                for(var i=0; i<stops.length; i++){
                   if(lat <= (stops[i]['stopLat'] + 0.0001) && lat >= (stops[i]['stopLat'] - 0.0001)
                      && lon <= (stops[i]['stopLon'] + 0.0001) && lon >= (stops[i]['stopLon'] - 0.0001)){
                         var stopLoc = new google.maps.LatLng(stops[i]['stopLat'], stops[i]['stopLon']);
                                  var marker = new google.maps.Marker({
                                           position: stopLoc,
                                           map: map,
                                           animation: google.maps.Animation.DROP,
                                           draggable: false
                                  });
                                  marker.id = stops[i]['stopId'];
                                  markers.push(marker);
                                  createRow(stops[i], stops[i]['stopId']);

                                  google.maps.event.addListener(marker, 'click', function(event) {
                                           removeMarker(marker.id, this);
                                  });
                                  break;
                         }
                }
        }

        function removeMarker(markerId, marker){
                marker.setMap(null);
                var row = $("#"+markerId);
                row.remove();
        }

        function createRow(details, idNo){
           var table = $('#stopsTbody');
           var row = $('<tr id="'+idNo+'"></tr>');
           var cell1 = $("<td style='width:60%;' class='css2'>
                         <input type='text' value = '"+ details["stopName"] +"' title='"+ details["stopName"]
                                  +"' style = 'width: 100%; border:none;' readonly/></td>");

           var cell2 = $("<td style='display:none;'>
                         <input type='text' name = 'stopIds[]' value = '"+ details["stopId"] +"'/></td>");
           var cell3 = $("<td style='width:18%'>
                         <input type='text' name='arrives[]' placeholder='HH:MM:SS'" + "maxlength='8'
                                  style = 'width: 100%; text-align:center;' required /></td>");
           var cell4 = $("<td style='width:20%'>
                         <input type='text' name='departs[]' placeholder='HH:MM:SS'" + "maxlength='8'
                                  style = 'width: 100%; text-align:center;' required /></td>");

                row.append(cell1);
                row.append(cell2);
                row.append(cell3);
                row.append(cell4);
                table.append(row);
        }
</script>
```

k. aroundmetro/application/views/trip/addSubmenu.php

```
<div class="submenu">
        <div class="container">
                <div align="right">
                        <form id='thisForm' style="margin-bottom: 0;">
                        <input type=submit class="btn btn-primary"  id="saveButton" value='Save'>
                        <a class="btn btn-default" role="button" id="cancelButton">Cancel</a>
                </div><!-- /tools -->
        </div>
</div><!-- /submenu -->

<script>
        $(document).ready(function(){
                $("#cancelButton").click(function() {
                        window.location.replace("<?php echo site_url('trip'); ?>");
                });
        });
</script>
```

l. aroundmetro/application/views/trip/editTrip_view.php

```
<head>
        <script src="https://maps.googleapis.com/maps/api/js?v=3.exp&sensor=false"></script>
</head>
```

```
<body>
   <div class="container">
      <div class="routes-table">
         <div class="panel panel-default data">
              <div class="leftPanel">
                 <table class="details" style='font-size: 14px; width: 100%' >
                       <tr>
                          <th colspan="5"> <span class="glyphicon glyphicon-tasks"></span>
                                Edit Trip Information for Trip <?php echo $trip['tripId']?></th>
                       </tr>
                       <tr>
                          <td style="width: 20%">Route Name</td>
                          <td><input type="text"  style="width: 100%" class="form-control" name="routeName"
                                                    value="<?php echo $trip['routeLongName']; ?>" readonly/>
                          </td>
                       </tr>
                       <tr>
                          <td style="display: none">
                             <input type="text" id="tripId" value="<?php echo $trip['tripId']; ?>"/>
                             <input type="text" id="stopCount" value="<?php echo $trip['stopCount']; ?>"/>
                          </td>
                       </tr>
                       <tr>
                          <td><strong> Trip Stops:</strong></td>
                       </tr>
                 </table>
                 <table class="stops" id="tableHeaders" style='width: 100%'>
                       <tr>
                          <th style="text-align: center; width: 62%; font-size: 14px;">Stop Name</th>
                          <th style="text-align: center; width: 18%; font-size: 12px;">Arrival Time</th>
                          <th style="text-align: center; width: 20%; font-size: 12px;">Departure Time</th>
                       </tr>
                 </table>
                 <div class="stopsDiv" style="height:70%; overflow-y: auto;">
                       table id="stopsTable" class="stops" style='font-size: 14px; width: 100%'>
                          <tbody id="stopsTbody">
                          <?php foreach ($tripStops as $t) { ?>
                             <tr id="<?php echo $t['stopId'];?>">
                                <td style='width:60%' class="css2"><input type='text'
                                          value = "<?php echo $t['stopName']; ?>"
                                          title="<?php echo $t['stopName'];?>"
                                          style = 'width: 100%; border: none;' readonly/>
                                </td>
                                <td style='display:none;'><input type='text' name = 'stopIds[]'
                                          value = "<?php echo $t['stopId']?>"/></td>
                                <td style='width:18%;'><input type='text' name='arrives[]'
                                          placeholder='HH:MM:SS' value="<?php echo $t['stopArr']?>"
                                          maxlength='8' style = 'width: 100%; text-align: center;'
                                          required/></td>

                                <td style='width:20%;'><input type='text' name='departs[]'
                                          placeholder='HH:MM:SS' value="<?php echo $t['stopDep']?>"
                                          maxlength='8' style = 'width: 100%; text-align: center;'
                                          required/></td>
                                <td style='display:none;'><input type='text' name = 'stopLats[]'
                                          value = "<?php echo $t['stopLat']?>"/></td>
                                <td style='display:none;'><input type='text' name = 'stopLons[]'
                                          value = "<?php echo $t['stopLon']?>"/></td>
                                <td style='display:none;'><input type='text' name = 'stopTimesIds[]'
                                          value = "<?php echo $t['stopTimesId']?>"/></td>
                              </tr>
                             <?php } ?>
                             </tbody>
                       </table>
                    </div>
                 </div><!-- /panel left panel -->
                 <div class="rightPanel">
                 <div id="map-canvas" style='width: 100%; height:115%; float:right'></div>
              </div>
           </div><!-- /panel panel-default -->
      </div><!-- /user-table -->
   </div>
   </form>
</body>

<script>
     var transitStops = <?php echo json_encode($tripStops); ?>;
     var count = 0;
```

```
var markers = [];
var rows = [];
var infowindow = new google.maps.InfoWindow();

var myLatLng = new google.maps.LatLng(14.583333,120.966667);
 var mapOptions = {
                  zoom: 11,
                  mapTypeId: google.maps.MapTypeId.ROADMAP,
                  center: myLatLng
        };

var map = new google.maps.Map(document.getElementById('map-canvas'), mapOptions);

$(document).ready(function() {
         var site_url='<?php echo site_url();?>';
         var bounds = new google.maps.LatLngBounds();

   for(var i=0; i<transitStops.length; i++){
         count++;
         var transStopLoc = new google.maps.LatLng(transitStops[i]['stopLat'],
                                                   transitStops[i]['stopLon']);
         var marker = new google.maps.Marker({
                                 position: transStopLoc,
                                 map: map,
                                 draggable: true
                           });
         marker.id = transitStops[i]['stopId'];
         marker.index = count;
         markers.push(marker);
         bounds.extend(marker.position);

         google.maps.event.addListener(marker, 'click', function(event) {
                  removeMarker(this);
         });

         google.maps.event.addListener(marker, 'dragend', function(event) {
                  updateMarker(this, event.latLng);
         });
   }

   map.fitBounds(bounds);
   $("#stopsTable").delegate("tr", "click", function(){
         var rowId = $(this).find('input').closest('tr').attr('id');
         infowindow.close();
         for(var i=0; i<markers.length; i++){
                  if(markers[i].id == rowId){
                           map.setZoom(17);
                           map.panTo(markers[i].position);
                           infowindow = new google.maps.InfoWindow({
                                    content: '<strong>Stop ' + markers[i].index + </strong>',
                                    map: map,
                           });
                           infowindow.open(map, markers[i]);
                           break;
                  }
         }
   });


$('#thisForm').on('submit', function() {
         event.preventDefault();

         var stp = $("input[name='stopIds[]']").map(function(){ return this.value; }).get();

         var arv = $("input[name='arrives[]']").map(function(){ return this.value; }).get();

         var dpt = $("input[name='departs[]']").map(function(){ return this.value; }).get();

         var lats = $("input[name='stopLats[]']").map(function(){ return this.value; }).get();

         var lons = $("input[name='stopLons[]']").map(function(){ return this.value; }).get();

         var stIds = $("input[name='stopTimesIds[]']").map(function(){ return this.value; }).get();

         var tripId = $("#tripId").val();
         var stopCount = $("#stopCount").val();

         if(!validateTime(arv) && !validateTime(dpt)){
```

```
                        alert("Arrival time and Departure time must follow pattern HH:MM:SS and must be in
                                                12-hour format");
                        return false;
                }else{
                        var s = $.ajax({
                        type:"POST",
                        url: site_url+"trip/updateTripToDatabase",
                        async: false,
                        data: {'stopIds': JSON.stringify(stp),
                                'arrives': JSON.stringify(arv),
                                'departs': JSON.stringify(dpt),
                                'lats': JSON.stringify(lats),
                                'lons': JSON.stringify(lons),
                                'stIds': JSON.stringify(stIds),
                                'tripId': tripId,
                                'stopCount': stopCount},
                                dataType: "JSON"
                        }).responseText;

                        window.location.replace("<?php echo site_url('trip'); ?>");
                        return false;
                        }
                });
        });

        function validateTime(arvs){
           for(var i=0; i < arvs.length; i++){
                if(arvs[i] != '00:00:00'){
                        if(!arvs[i].match(/^(0[0-9]|1[012]):([0-5]\d):([0-5]\d)$/) || arvs[i]==""){
                                return false;
                        }
                }
           }
           return true;
        }

        function updateMarker(marker, latlng){
                var newLat = latlng.lat().toFixed(6);
                var newLon = latlng.lng().toFixed(6);
                var row = $("#"+marker.id);
                row.find('td').eq(4).find(':input').val(newLat);
                row.find('td').eq(5).find(':input').val(newLon);
                console.log(newLat +","+ newLon);
        }

        function removeMarker(marker){
                marker.setMap(null);
                console.log("remove marker: " + marker.id);
                var row = $("#"+marker.id);
        row.remove();
        }

</script>
```

m. aroundmetro/application/views/trip/trip_view.php

```
<!-- div for edit trip -->
<div style='display: none'>
   <div id='edit_trip' style='width: 100%'>
      <div class="panel panel-default">
                <p class="header">Edit Trip</p>
                   <div class="fancy">
                        <center><h4><strong>
                                <?php echo $this->session->userdata('status_edit'); ?>
                        </h4></strong></center>

                        <p align="center"><br />
                                <?php echo $this->session->userdata('message_edit'); ?>
                        </p>
                                <?php if($this->session->userdata('message_edit')){
                                        $this->session->unset_userdata('status_edit');
                                        $this->session->unset_userdata('tripId');
                                }?>

                          <br>
                        <center>
                        <input type=button id="editTripOk" class="btn btn-default btn-primary" value="OK">
                        </center>
```

```
                    </div>
            </div>
    </div>
</div>

<?php if($this->session->userdata('message_edit')){ ?>

<script type="text/javascript">
    $(document).ready(function() {

        $.fancybox({
               'width': '30%',
               'height': '40%',
               'autoScale': true,
               'transitionIn': 'fade',
               'transitionOut': 'fade',
               'type': 'inline',
               'href': '#edit_trip',
               'modal': true,
               'autoSize': false
           });
    });

</script>

<?php $this->session->unset_userdata('message_edit'); } ?>

<!-- div for add trip -->
<div style='display: none'>
    <div id='add_trip' style='width: 100%'>
             <div class="panel panel-default">
                     <p class="header">Add Trip</p>
                     <div class="fancy">
                             <center><h4><strong>
                                     <?php echo $this->session->userdata('status_add'); ?>
                             </h4></strong></center>
                             <p align="center">
                                     <?php echo $this->session->userdata('message_add'); ?>
                             </p>
                                     <?php if($this->session->userdata('message_add')){ ?>
                                     <p align="center" style='font-size: 14px'>
                                             <strong>Trip ID: </strong>
                                             <?php echo $this->session->userdata('tripId')?>
                                     </p>

                                     <?php $this->session->unset_userdata('status_add');
                                             $this->session->unset_userdata('tripId'); } ?>

                       <br>
                     <center>
                     <input type=button id="messageOk" class="btn btn-default btn-primary" value="OK">
                     </center>
              </div>
         </div>
    </div>
</div>

<?php if($this->session->userdata('message_add')){ ?>

<script type="text/javascript">
    $(document).ready(function() {

        $.fancybox({
               'width': '30%',
               'height': '40%',
               'autoScale': true,
               'transitionIn': 'fade',
               'transitionOut': 'fade',
               'type': 'inline',
               'href': '#add_trip',
               'modal': true,
               'autoSize': false
           });
        });
</script>
<?php $this->session->unset_userdata('message_add'); } ?>
```

```
<!-- div for delete trip -->
<div style='display: none'>
     <div id='del_trip' style='width: 100%'>
               <div class="panel panel-default">
                         <p class="header">Delete Trip</p>
                         <div class="fancy">
                                   <center><h4><strong>
                                             <?php echo $this->session->userdata('status_del'); ?>
                                   </h4></strong></center>
                                   <p align="center">
                                             <?php echo $this->session->userdata('message_del'); ?>
                                   </p>
                                             <?php if($this->session->userdata('message_del')){
                                                       $this->session->unset_userdata('status_del');
                                                       $this->session->unset_userdata('tripId');
                                                  } ?>

                         <br>
                         <center>
                         <input type=button id="delTripOk" class="btn btn-default btn-primary" value="OK">
                         </center>
                    </div>
               </div>
        </div>
</div>

<?php if($this->session->userdata('message_del')){ ?>

<script type="text/javascript">
     $(document).ready(function() {

         $.fancybox({
                'width': '30%',
                'height': '35%',
                'autoScale': true,
                'transitionIn': 'fade',
                'transitionOut': 'fade',
                'type': 'inline',
                'href': '#del_trip',
                'modal': true,
                'autoSize': false
            });
         });

</script>

<?php $this->session->unset_userdata('message_del'); }?>

<body>
    <div class="container">
         <div class="routes-table">
              <div class="panel panel-default">
                        <div class="panel-heading">
                           <table class="table routes" style='font-size: 15px'>
                                 <th style='width: 6%; text-align: center'>Trip ID</th>
                                 <th style="width: 12%; text-align: center">Route ID</th>
                                 <th style="width: 10%; text-align: center">No. of Stops</th>
                                 <th style="width: 20%; text-align: center">Route Name</th>
                                 <th style="text-align: center">Route Description</th>
                           </table>
                        </div>

              </div><!-- /panel panel-default -->
              <div class="panel panel-default records originalTable" id="routeTable">
                 <table class="table routes" id="routeTable" style='font-size: 14px'>
                     <tbody>
                     <?php if(count($trips)!=0){
                        foreach ($trips as $trip) {
                           echo "<tr id =".$trip['tripId']."">
                              <td style='width: 6%; text-align: enter'>".$trip['tripId']."</td>
                              <td style='width: 12%; text-align: center'>".$trip['routeId']."</td>
                              <td style='width: 10%; text-align: center'>".$trip['stopCount']."</td>
                              <td style='width: 20%;'>".$trip['routeLongName']."</td>
                              <td >".$trip['routeDesc']."</td>
                              </tr>";
                         }
                        } else{
                              echo "<h5 text align = 'center'>No entries to be displayed!</h5>"; }?>
                     </tbody>
```

```
                  </table>
                </div><!-- /panel panel-default -->

                <div class="panel panel-default records empty" style="display: none;">
                   <table class="table users" style='font-size: 14px'>
                      <tbody>
                         <h5 text align = 'center'>No entries to be displayed!</h5>
                      </tbody>
                   </table>
                </div>
             </div><!-- /user-table -->
     </div> <!-- /container -->
</body>

<script>
     $(document).ready(function(){
              $('#messageOk').click(function () {
                       $.fancybox.close();
              });

              $('#editTripOk').click(function () {
                       console.log("edit ok!");
                       $.fancybox.close();
              });

              $('#delTripOk').click(function () {
                       console.log("edit ok!");
                       $.fancybox.close();
              });
     });
</script>
```

n. aroundmetro/application/views/trip/viewTrip_view.php

```
<head>
     <script src="https://maps.googleapis.com/maps/api/js?v=3.exp&sensor=false"></script>
</head>

<body>
   <div class="wrapper">
      <div class="container">
          <div class="holder">
              <div id="type-user">
                 <div class="leftPanel">
                    <div class="panel panel-default info">
                       <table class="view" style='width: 100%;'>
                          <tr><th style='font-size: 12px;'><span class="glyphicon glyphicon-tasks"></span>
                                 Trip <?php echo $trip['tripId']; ?> Information </th></tr>
                       </table>
                       <form>
                                 <input type="hidden" id="tripId" value="<?php echo $trip['tripId']; ?>">
                       </form>
                       <table class="table details" style='font-size: 12px; width: 100%;'>
                          <tr>
                                 <td>Route ID</td>
                                 <td colspan="2"><?php echo $trip['routeId']; ?></td>
                          </tr>
                          <tr>
                                 <td>Vehicle Type</td>
                                 <td colspan="2"><?php echo $trip['routeTypeName']; ?></td>
                          </tr>
                          <tr>
                                 <td>Route Long Name</td>
                                 <td colspan="2"><?php echo $trip['routeLongName']; ?></td>
                          </tr>
                          <tr>
                                 <td>Route Description</td>
                                 <td colspan="2"><?php echo $trip['routeDesc']; ?></td>
                          </tr>
                       </table>
                    </div>
                  </div> <!--end leftPanel for staff-->
                  <div class="rightPanel">
                         <div id="map-canvas" style='width: 100%; height:100%; float:right'></div>
                   </div>
                   <div class="leftPanel">
                         <div class="panel panel-default info" style="height:47%; ">
                                  <table class="view" style='width: 100%;'>
```

```
                                    <tr><th style="border-bottom:1px solid #ddd; font-size: 12px;">
                                          <span class="glyphicon glyphicon-tasks"></span>
                                          Trip <?php echo $trip['tripId']; ?> Stops </th></tr>
                                 </table>
                           <table class="stops" id="tableHeaders" style='width: 100%'>
                              <tr>
                               <th style="text-align: center; width:5%; font-size: 12px;">No.</th>
                               <th style="text-align:center; width:59%;font-size:12px;">Stop Name</th>
                               <th style="text-align:center;width:18%;font-size:12px;">Arrival Time</th>
                               <th style="text-align: center; width: 20%; font-size: 12px;">Departure Time</th>
                              </tr>
                           </table>
                           <div style="height:75%; overflow-y: auto;">
                                    <table class="stops" id="stopsTable">
                                    <?php
                                       $count = 1;
                                       foreach ($stops as $s) {?>
                                          <tr id="<?php echo $s['stopId'];?>" >
                                          <td style='width:5%; font-size: 13px; text-align: center;'>
                                                                  <?php echo $count; $count++; ?></td>
                                          <td style='width:60%; font-size: 12px;'>
                                                                          <?php echo $s['stopName']; ?></td>
                                          <td style='width:18%; font-size: 13px; text-align: center;'>
                                                                          <?php echo $s['arrTime']; ?></td>
                                          <td style='width:20%; font-size: 13px; text-align: center;'>
                                                                          <?php echo $s['dptTime']; ?></td>
                                          </tr>
                                        <?php } ?>
                                      </table>
                           </div>
                        </div>
                     </div>
                  </div><!--end type:staff-->
               </div><!-- /holder -->
            </div> <!-- /container -->
         </div><!-- /wrapper -->
      </body>

      <script type="text/javascript">
           var transitStops = <?php echo json_encode($stops); ?>;
           var count = 0;
           var markers = [];
           var infowindow = new google.maps.InfoWindow();
           var myLatLng = new google.maps.LatLng(14.583333,120.966667);
           var mapOptions = {
                              zoom: 11,
                              mapTypeId: google.maps.MapTypeId.ROADMAP,
                              center: myLatLng
                     };
           var map = new google.maps.Map(document.getElementById('map-canvas'), mapOptions);

           $(document).ready(function(){
                    var bounds = new google.maps.LatLngBounds();
              for(var i=0; i<transitStops.length; i++){
               count++;
               var transStopLoc = new google.maps.LatLng(transitStops[i]['stopLat'], transitStops[i]['stopLon']);
                    var marker = new google.maps.Marker({
                                       position: transStopLoc,
                                       map: map,
                                       draggable: false
                              });
                    marker.id = transitStops[i]['stopId'];
                    marker.index = count;
                    markers.push(marker);
                    bounds.extend(marker.position);

                    google.maps.event.addListener(marker, 'click', function(event) {
                              infowindow = new google.maps.InfoWindow({
                                       content: '<strong><center>Stop ' + markers[i].index +
                                                '</center></strong><br /><p>' +transitStops[markers[i].index-
                                                                                    1]['stopName'] +'</p>',
                              map: map,
                              });
                              infowindow.open(map, this);
                    });
              }
           map.fitBounds(bounds);

           $("#stopsTable").delegate("tr", "click", function(){
```

```
    var rowId = $(this).closest('tr').attr('id');
            for(var i=0; i<markers.length; i++){
                    if(markers[i].id == rowId){
                            map.setZoom(17);
                            map.panTo(markers[i].position);
                            infowindow = new google.maps.InfoWindow({
                                    content: '<strong><center>Stop ' + markers[i].index +
                                            '</center></strong><br /><p>'
                                            +transitStops[markers[i].index-1]['stopName'] +'</p>',
                                    map: map,
                            });
                            infowindow.open(map, markers[i]);
                            break;
                            }
                    }
            });

    });

</script>
```

o. aroundmetro/application/views/stop/addStop_view.php

```
<head>
    <script src="https://maps.googleapis.com/maps/api/js?v=3.exp&sensor=false"></script>
    <script src=<?php echo base_url("/js/maplace.min.js"); ?>></script>
    <script src=<?php echo base_url("/js/maplace.js"); ?>></script>
</head>
<body>
   <div class="wrapper">
      <div class="container">
         <div class="routes-table">
             <div class="panel panel-default data">
                 <div class="leftPanel">
                    <table class="details" style='font-size: 14px; width: 100%' >
                        <tr>
                          <th colspan="5"> <span class="glyphicon glyphicon-tasks"></span>
                                Personal Information</th>
                        </tr>
                        <tr><td> </td></tr>
                        <tr>
                           <td colspan="4">
                             <input name="stopId" id="stopId" type="hidden" class="form-control" size="26%">
                           </td>
                        </tr>
                        <tr>
                           <td>Stop Name</td>
                           <td colspan="4">
                              <input name="stopName" type="text" class="form-control"  size="60%" required>
                           </td>
                        </tr>
                        <tr>
                           <td>Stop Latitude</td>
                           <td colspan="4">
                              <input name="stopLat" id="stopLat" type="text" class="form-control" size="26%"
                                                                                                  required>
                           </td>
                        </tr>
                        <tr>
                           <td>Stop Longitude</td>
                           <td colspan="4">
                              <input name="stopLon" id="stopLon" type="text" class="form-control" size="26%"
                                                                                                  required>
                           </td>
                        </tr>
                     </table>
                 </div>
                 <div class="rightPanel">
                        <div id="map-canvas" style='width: 100%; height:100%; float:right'></div>
                 </div>
             </div>
           </div><!-- /holder -->
         </div> <!-- /container -->
    </div><!-- /wrapper -->
</form>
</body>
```

```
<script type="text/javascript">

     var myLatLng = new google.maps.LatLng(14.583333,120.966667);
     var mapOptions = {
                        zoom: 11,
                        mapTypeId: google.maps.MapTypeId.ROADMAP,
                        center: myLatLng
        };

     var map = new google.maps.Map(document.getElementById('map-canvas'), mapOptions);

     google.maps.event.addListener(map, 'click', function(event) {
        placeMarker(event.latLng);
     });

     var marker;
     function placeMarker(location) {
              if(marker == null){
                       marker = new google.maps.Marker({
                      position: location,
                      map: map,
                      draggable: true
                   });
              }
              else{
                       marker.setPosition(location);
              }
         $("#stopLat").val(location.lat().toFixed(6));
         $("#stopLon").val(location.lng().toFixed(6));

         google.maps.event.addListener(marker, 'dragend', function(event) {
                       $("#stopLat").val(event.latLng.lat().toFixed(6));
              $("#stopLon").val(event.latLng.lng().toFixed(6));
         });
     }

     $(document).ready(function(){
              var site_url='<?php echo site_url();?>';
              var marker = <?php echo $marker; ?>;

              $("#thisForm").attr("action",site_url+"stop/saveStopToDatabase");
              $("#thisForm").attr("method","POST");

              var stopId = "LTFRB_" + marker;
              console.log("marker: " + stopId);
              $("#stopId").val(stopId);
     });
</script>
```

p. aroundmetro/application/views/stop/editStop_view.php

```
<head>
     <script src="http://maps.google.com/maps/api/js?sensor=false&libraries=geometry&v=3.7"></script>
     <script src=<?php echo base_url("/js/maplace.min.js"); ?>></script>
     <script src=<?php echo base_url("/js/maplace.js"); ?>></script>
</head>

<div class="wrapper">
   <div class="container">
      <div class="routes-table">
          <div class="leftPanel">
              <div class="panel panel-default data">
                 <table class="details" style='width: 100%;'>
                       <tr><th><span class="glyphicon glyphicon-tasks"></span>  
                                Edit Stop Information </th></tr>
                 </table>
                 <table class="details infodetails" style='font-size: 14px; width: 100%;'>
                       <tr>
                          <td>Stop ID</td>
                          <td colspan="4">
                             <input name="secretId" type="hidden" value="<?php echo $stop['stopId']; ?>">
                             <input name="stopId" type="text" class="form-control"
                                      value="<?php echo $stop['stopId']; ?>" size="26%" disabled>
                          </td>
                       </tr>
                       <tr>
                          <td>Stop Name</td>
                          <td colspan="4">
```

```
                              <input name="stopName" type="text" class="form-control"
                                        value="<?php echo $stop['stopName']; ?>" size="60%" required>
                           </td>
                        </tr>
                        <tr>
                           <td>Stop Latitude</td>
                           <td colspan="4">
                              <input name="stopLat" id="stopLat" type="text" class="form-control"
                                        value="<?php echo $stop['stopLat']; ?>" size="26%" required>
                           </td>
                        </tr>
                        <tr>
                           <td>Stop Longitude</td>
                           <td colspan="4">
                                 <input name="stopLon" id="stopLon" type="text" class="form-control"
                                        value="<?php echo $stop['stopLon']; ?>" size="26%" required>
                           </td>
                        </tr>
                     </table>
                  </div>        <br />
            </div>   <!--end leftPanel for staff-->
            <div class="rightPanel">
                     <div id = "gmap"></div>
            </div>
         </div><!-- /holder -->
      </div> <!-- /container -->
   </div><!-- /wrapper -->
</form>

<script type="text/javascript">

   $(document).ready(function(){
            var site_url='<?php echo site_url();?>';
            $("#thisForm").attr("action",site_url+"stop/update");
            $("#thisForm").attr("method","POST");

            var stopLat = <?php echo $stop['stopLat']; ?>;
            var stopLon = <?php echo $stop['stopLon']; ?>;

            var mMap = new Maplace({
                     locations: [{
                              lat: stopLat,
                              lon: stopLon,
                              show_infowindow: true,
                              visible: true,
                              draggable: true
                     }],
                     map_options: {
                              set_center: [stopLat, stopLon],
                              zoom: 15
                     }
            });

            mMap.o.afterCreateMarker = function (index, location, marker) {
                     google.maps.event.addListener(marker, 'dragend', function(e) {
                              $("#stopLat").val(e.latLng.lat().toFixed(6));
                              $("#stopLon").val(e.latLng.lng().toFixed(6));
                     });
            };

            mMap.Load();
   });

</script>
```

q. aroundmetro/application/views/stop/stop_view.php

```
<!-- div for add stop -->
<div style='display: none'>
   <div id='add_stop' style='width: 100%'>
            <div class="panel panel-default">
                     <p class="header">Add Stop</p>
                     <div class="fancy">
                              <center><h4><strong>
                                       <?php echo $this->session->userdata('status'); ?>
                              </h4></strong></center>
                              <p align="center">
                                       <?php echo $this->session->userdata('message'); ?>
```

```
                                </p>
                                        <?php if($this->session->userdata('message')){ ?>
                                        <p align="center" style='font-size: 14px'>
                                        <strong>Stop ID: </strong>
                                                <?php echo $this->session->userdata('stopId')?>
                                        </p>

                                        <?php $this->session->unset_userdata('status');
                                              $this->session->unset_userdata('stopId'); }?>

                                <br>
                                <center>
                                <input type=button id="messageOk"
                                        class="btn btn-default btn-primary" value="OK"></center>
                        </div>
                </div>
        </div>
</div>

<?php if($this->session->userdata('message')){ ?>

<script type="text/javascript">
     $(document).ready(function() {

         $.fancybox({
                'width': '30%',
                'height': '40%',
                'autoScale': true,
                'transitionIn': 'fade',
                'transitionOut': 'fade',
                'type': 'inline',
                'href': '#add_stop',
                'modal': true,
                'autoSize': false
            });
         });

</script>

<?php $this->session->unset_userdata('message'); } ?>

<!-- div for edit stop -->
  <div style='display: none'>
     <div id='editStop' style='width: 100%'>
          <div class="panel panel-default">
               <p class="header">Edit Stop</p>
                        <div class="fancy">
                                <center><h4><strong>
                                        <?php echo $this->session->userdata('status_update'); ?>
                                </h4></strong></center>
               <p align="center">
                        <?php echo $this->session->userdata('message_update'); ?>
               </p>
               <?php if($this->session->userdata('message_update')){
                        $this->session->unset_userdata('status_update'); }?>
               <br />
               <center>
                        <input type=button id="editOk" class="btn btn-default btn-primary" value="OK">
               </center>
           </div>
         </div>
       </div>
  </div>
  <!--end div for edit stop-->

 <?php if($this->session->userdata('message_update')){ ?>

<script type="text/javascript">
     $(document).ready(function() {

         $.fancybox({
                'width': '30%',
                'height': '33%',
                'autoScale': true,
                'transitionIn': 'fade',
                'transitionOut': 'fade',
                'type': 'inline',
                'href': '#editStop',
                'modal': true,
```

```
                'autoSize': false
        });
    });

</script>

<?php $this->session->unset_userdata('message_update'); }?>

<!-- div for delete stop -->
  <div style='display: none'>
     <div id='deleteStop' style='width: 100%'>
              <div class="panel panel-default">
                       <p class="header">Edit Stop</p>
                       <div class="fancy">
                                <center><h4><strong>
                                         <?php echo $this->session->userdata('status_del'); ?>
                                </h4></strong></center>
                       <p align="center">
                                <?php echo $this->session->userdata('message_del'); ?>
                       </p>

                       <?php    if($this->session->userdata('message_del')){
                                         $this->session->unset_userdata('status_del'); }?>
                       <br />
                       <center>
                          <input type=button id="deleteOk" class="btn btn-default btn-primary" value="OK">
                       </center>
              </div>
         </div>
      </div>
  </div>
  <!--end div for edit stop-->

 <?php if($this->session->userdata('message_del')){?>

<script type="text/javascript">

     $(document).ready(function() {

         $.fancybox({
                'width': '30%',
                'height': '33%',
                'autoScale': true,
                'transitionIn': 'fade',
                'transitionOut': 'fade',
                'type': 'inline',
                'href': '#deleteStop',
                'modal': true,
                'autoSize': false
        });
    });

</script>

<?php $this->session->unset_userdata('message_del'); }?>


<body>
    <div class="container">
         <div class="routes-table">
              <div class="panel panel-default">
                       <div class="panel-heading">
                       <table class="table routes" style='font-size: 15px'>
                                <th style='width: 20%; text-align: center'>Stop ID</th>
                                <th style="width: 50%; text-align: center">Stop Name</th>
                                <th style="width: 15%; text-align: center">Latitude</th>
                                <th style="width: 25%; text-align: center">Longitude</th>
                       </table>
                       </div>

              </div><!-- /panel panel-default -->
              <div class="panel panel-default records originalTable" id="routeTable">
                       <table class="table routes" id="routeTable" style='font-size: 14px'>
                                <tbody>
                                <?php if(count($stops)!=0){
                                foreach ($stops as $stop) {
                                         echo "<tr id =".$stop['stopId']."">
                                         <td style='width: 20%; padding-left:6%;'>".$stop['stopId']."</td>
                                         <td style='width: 50%;'>".$stop['stopName']."</td>
```

```
                                    <td style='width: 15%; text-align: center'>".$stop['stopLat']."</td>
                                    <td style='width: 25%;text-align: center'>".$stop['stopLon']."</td>
                                </tr>";
                               }
                            } else{
                                    echo "<h5 text align = 'center'>No entries to be displayed!</h5>"; }?>
                            </tbody>
                        </table>
                        </div><!-- /panel panel-default -->

                        <div class="panel panel-default records empty" style="display: none;">
                            <table class="table users" style='font-size: 14px'>
                                    <tbody>
                                             <h5 text align = 'center'>No entries to be displayed!</h5>
                                    </tbody>
                            </table>
                        </div>
                </div><!-- /user-table -->
        </div> <!-- /container -->
</body>

<script>
        $(document).ready(function(){
                $('#messageOk').click(function () {
                        $.fancybox.close();
                });

                $('#editOk').click(function () {
                        $.fancybox.close();
                });

                $('#deleteOk').click(function () {
                        $.fancybox.close();
                });
        });
</script>
```

r. aroundmetro/application/views/stop/viewStop_view.php

```
<head>
    <script src="http://maps.google.com/maps/api/js?sensor=false&libraries=geometry&v=3.7"></script>
    <script src=<?php echo base_url("/js/maplace.min.js"); ?>></script>
    <script src=<?php echo base_url("/js/maplace.js"); ?>></script>
</head>

<body>
   <div class="wrapper">
      <div class="container">
         <div class="holder">
              <div id="type-user">
                 <div class="leftPanel">
                    <div class="panel panel-default info">
                       <table class="view" style='width: 100%;'>
                          <tr><th><span class="glyphicon glyphicon-tasks"></span>
                                Stop Information </th></tr>
                       </table>
                       <table class="table details" style='font-size: 14px; width: 100%;'>
                          <tr>
                             <td>Stop ID</td>
                             <td colspan="2"><?php echo $stop['stopId']; ?></td>
                          </tr>
                          <tr>
                             <td>Stop Name</td>
                             <td colspan="2"><?php echo $stop['stopName']; ?></td>
                          </tr>
                          <tr>
                             <td>Stop Latitude</td>
                             <td colspan="2"><?php echo $stop['stopLat']; ?></td>
                          </tr>
                          <tr>
                             <td>Stop Longitude</td>
                             <td colspan="2"><?php echo $stop['stopLon']; ?></td>
                          </tr>
                       </table>
                    </div>      <br />
                 </div>         <!--end leftPanel for staff-->
                 <div class="rightPanel">
                      <div id = "gmap"></div>
```

```
                    </div>
                </div><!--end type:staff-->
            </div><!-- /holder -->
        </div> <!-- /container -->
    </div><!-- /wrapper -->
</body>

<script type="text/javascript">

    $(document).ready(function(){
            var stopLat = <?php echo $stop['stopLat']; ?>;
            var stopLon = <?php echo $stop['stopLon']; ?>;
            var stopId = '<?php echo $stop['stopId']; ?>';


            $(function() {
                    new Maplace({
                            locations: [{
                                    lat: stopLat,
                                    lon: stopLon,
                                    html: '<h4>' + stopId + '</h4>',
                                    show_infowindow: true,
                                    visible: true
                            }],
                            map_options: {
                                    set_center: [stopLat, stopLon],
                                    zoom: 15
                            }
                    }).Load();
            });
    });

</script>
```

s. aroundmetro/application/views/graph/graph_view.php

```
<body>
    <div class="container">
            <div class="wrapper">
            <div class="panel panel-default data" style="text-align: justify">
            <table class="details" style='font-size: 14px; width: 100%' >
                    <tr>
                            <th colspan="5"> <span class="glyphicon glyphicon-tasks"></span>
                                    Graph Update Instructions</th>
                    </tr>
                    <tr><td></td></tr>
            </table>
            <p style="padding: 2%">
                    <strong>IMPORTANT! </strong> Before updating the graph, shutdown Tomcat server. Make
                    sure that all routes, trips, and stops are properly checked. Once the <strong>Update
                    Graph</strong> button is clicked, the system will create the General Transit Feed
                    Specification (GTFS) file and automatically build the graph. The update process
                    cannot be interrupted and cannot be cancelled. After successful graph update,
                    restart the server. This process will take a while and you may want to take a coffee
                    break first.
            </p>
            <br />
                    </div><!-- /panel panel-default -->
        </div>
    </div>
</body>

<div style='display: none'>
    <div id="updateGraph">
            <div class="panel panel-default" id="onSuccess" style="display: none">
                    <p class="header"> </p>
                    <div class="fancy">
                            <center><h4><strong>SUCCESS!</h4></strong></center>
                            <p align="center"><br />
                                    Graph successfully updated.
                            </p><br>
                            <center><input type=button id="updateOk" class="btn btn-default btn-
primary" value="OK"></center>
                    </div>
            </div>
            <div class="panel panel-default" id="onFail" style="display: none">
                    <p class="header"> </p>
                    <div class="fancy">
```

```
                                <center><h4><strong>ERROR!</h4></strong></center>
                                <p align="center"><br />
                                        Error in updating the graph.
                                </p><br>
                        <center>
                            <input type=button id="failOk" class="btn btn-default btn-primary" value="OK">
                        </center>
                    </div>
                </div>
        </div>
</div>

<script>
    $(document).ready(function() {
            var site_url='<?php echo site_url();?>';
            $("#gtfsButton").click(function(){
                    $(".fancybox").fancybox({
                            'modal': true,
                            helpers : {
                                    overlay : {
                                             css: { 'background': 'rgba(0, 0, 255, 0.5)' }
                                    }
                            }
                    });

                    $.fancybox.helpers.overlay.open();
                    $.fancybox.showLoading();
                    $("#gtfsButton").attr("disabled", "disabled");

                    $.ajax({
                            type: "POST",
                            url: site_url+"graph/updateGraph",
                            async: true,
                            data: {   'data': 'alohomora'},
                            dataType: "TEXT",
                            success: function(data){
                                    console.log(data);
                                    var s = JSON.parse(data);
                                    console.log(s['result']);
                                    if(s['result'] === true){
                                            $.fancybox.helpers.overlay.close();
                                            $.fancybox.hideLoading();
                                            showfancy();
                                            $("#onSuccess").show();
                                            $("#onFail").hide();
                                    }
                                    else{
                                            $.fancybox.helpers.overlay.close();
                                            $.fancybox.hideLoading();
                                            showfancy();
                                            $("#onSuccess").hide();
                                            $("#onFail").show();
                                    }
                                    $("#gtfsButton").attr("disabled", false);
                            }
                    });
            });

            $('#updateOk').click(function () { $.fancybox.close(); });

            $('#failOk').click(function () { $.fancybox.close(); });

            function showfancy(){
                    $.fancybox({
                            'autoDimensions': false,
                            'width': '20%',
                            'height': '35%',
                            'autoSize': false,
                            'transitionIn': 'fade',
                            'transitionOut': 'fade',
                            'type': 'inline',
                            'modal': true,
                            'href': '#updateGraph'
                    });
            }

    });

</script>
```

t. aroundmetro/application/views/fare/fare_view.php

```
<div class="wrapper">
    <div class="container">
            <div class="holder">
                    <div class="leftFarePanel" id="lrt1"></div>
                    <div class="rightFarePanel" id="bus" style=" margin-left: 2%"></div>
                    <div class="leftFarePanel" id="lrt2" style="margin-top: 1%;"></div>
                    <div class="rightFarePanel" id="jeep" style="margin-top: 2%; margin-left: 2%"></div>
                    <div class="leftFarePanel" id="mrt3" style="margin-top: 1%;"></div>

            </div>
    </div>
</div>

<script>
    var xml;

    $(document).ready(function() {
            var path = '<?php echo base_url(); ?>';

            xml= $.ajax({
                            type: "GET",
                    async: false,
                    url: path+"xml/railfare.xml",
                    dataType: "xml"
            }).responseText;

            xml= $.parseXML(xml);

            var lrt1 = [];
            var lrt2 = [];
            var mrt3 = [];
            var bus = new Object;
            var jeep = new Object;


            $(xml).find("vehicle").each(function(){
                    if($(this).attr("name") =="LRT 1"){
                            $(this).children().each(function(){
                                    var object =  new Object;
                                    object.range = $(this).find('startRange').text() + " - " +
                                                    $(this).find('endRange').text();
                                    object.fare =  $(this).find('fare').text();
                                    lrt1.push(object);
                            });
                    }
                    else if($(this).attr("name") =="LRT 2"){
                            $(this).children().each(function(){
                                    var object =  new Object;
                                    object.range = $(this).find('startRange').text() + " - " +
                                                    $(this).find('endRange').text();
                                    object.fare =  $(this).find('fare').text();
                                    lrt2.push(object);
                            });
                    }
                    else if($(this).attr("name") =="MRT 3"){
                            $(this).children().each(function(){
                                    var object =  new Object;
                                    object.range = $(this).find('startRange').text() + " - " +
                                                    $(this).find('endRange').text();
                                    object.fare =  $(this).find('fare').text();
                                    mrt3.push(object);
                            });
                    }
                    else if($(this).attr("name") =="Bus"){
                            $(this).children().each(function(){
                                    bus.baseFare = $(this).find('baseFare').text();
                                    bus.firstKm = $(this).find('firstKm').text();
                                    bus.nextKm = $(this).find('nextKm').text();
                            });
                    }
                    else if($(this).attr("name") =="Jeepney"){
                            $(this).children().each(function(){
                                    jeep.baseFare = $(this).find('baseFare').text();
                                    jeep.firstKm = $(this).find('firstKm').text();
                                    jeep.nextKm = $(this).find('nextKm').text();
                            });
```

```
                }
        });

        console.log(lrt1);

        $('#lrt1').load('fare/rail', {'name': 'LRT 1', 'data': lrt1});
        $('#jeep').load('fare/pubj', {'name': 'Jeep', 'baseFare': jeep.baseFare,
                                                      'firstKm': jeep.firstKm,
                                                      'nextKm': jeep.nextKm});
        $('#lrt2').load('fare/rail', {'name': 'LRT 2', 'data': lrt2});
        $('#bus').load('fare/pubj', {'name': 'Bus', 'baseFare': bus.baseFare,
                                                      'firstKm': bus.firstKm,
                                                      'nextKm': bus.nextKm});
        $('#mrt3').load('fare/rail', {'name': 'MRT 3', 'data': mrt3});
    });
</script>
```

u. aroundmetro/application/views/delete_view.php

```
<!-- div for delete route -->
<div>
    <div id='delete_trip' style='width: 100%'>
            <div class="panel panel-default">
                    <p class="header">Delete <?php echo $type; ?></p>
                    <div class="fancy">
                            <center><h4><strong>WARNING!</h4></strong></center>
                            <p align="center"><br />
                                    Are you sure you want to delete <?php echo $type; ?>
                                            <?php echo $id; ?>?
                            </p>
                    <br /> <br />
               <center>
                  <input type=button id="deleteOk" class="btn btn-default btn-primary" value="Delete">
             nbsp; <input type=button id="cancel" class="btn btn-default btn-primary" value="Cancel">
                </center>
                    </div>
            </div>
    </div>
</div>
```

v. aroundmetro/application/views/busJeep_view.php

```
<div class="panel panel-default info">
    <table class="view" style='width: 100%;'>
            <tr><th><span class="glyphicon glyphicon-tasks"></span>
                    <?php echo $name;?> Fare Information </th></tr>
    </table>

    <table class="table details" style='font-size: 14px; width: 100%;'>
            <tr>
                    <td rowspan="3" >
                            <?php if($name == "Bus"){?>
                                    <img style="width: 100px; height: 100px;"
                                            src=<?php echo base_url("images/mode_bus.png")?>>
                            <?php } else {?>
                                    <img style="width: 100px; height: 100px;"
                                            src=<?php echo base_url("images/mode_jeep.png")?>>
                            <?php }?>
                    </td>
                    <td>Base Fare</td>
                    <td colspan="2"><?php echo $baseFare; ?></td>
            </tr>
            <tr>
                    <td>Initial Distance</td>
                    <td colspan="2"><?php echo $firstKm; ?></td>
            </tr>
            <tr>
                    <td>Next km Fare</td>
                    <td colspan="2"><?php echo $nextKm; ?></td>
            </tr>
    </table>
</div>
```

w. aroundmetro/application/views/rail_view.php

```
<div class="panel panel-default info">
    <table class="view" style='width: 100%;'>
```

```
                <tr><th><span class="glyphicon glyphicon-tasks"></span>   
                        <?php echo $name; ?> Fare Information </th></tr>
        </table>
        <table class="table details" style='font-size: 14px; width: 100%;'>
                <tr>
                        <td rowspan="3" >
                                <img style="width: 80px; height: 80px;"
                                        src=<?php echo base_url("images/mode_train.png")?>>
                        </td>
                        <td style="text-align: center">Station range</td>
                                <?php
                                        foreach($data as $d){
                                                echo "<td style='text-align:
                                                center;'><strong>".$d['range']."</strong></td>";
                                        }
                                ?>
                </tr>
                <tr>
                        <td style="text-align: center">Fare</td>
                                <?php
                                        foreach($data as $d){
                                                echo "<td style='text-align:
                                                center;'>".$d['fare']."</td>";
                                        }
                                ?>
                </tr>
        </table>
</div>
```

x. aroundmetro/application/controllers/trip.php

```
<?php if ( ! defined('BASEPATH')) exit('No direct script access allowed');

class Trip extends CI_Controller {

    function Trip()
    {
       parent::__construct();
            $this->load->model('trip_model');
            $this->session->set_userdata('page', 'trip');
    }

    /*
     * index() - always fired first in controller files
     * Inside this function, the main content (view file) will be assigned and called.
     */

    public function index() {
            $this->session->unset_userdata('action');
            $data['main_content']='trip/trip_view';
            $data['title']='Trip';
            $data['trips']=$this->trip_model->getTrips();
            $this->load->view('template', $data);
    }

    public function view($id){
            $this->session->set_userdata('action', 'view');
            $data['main_content']='trip/viewTrip_view';
            $data['title']='View Trip';
            $data['tripId'] = $id;
            $data['trip'] = $this->trip_model->getTripInfo($id);
            $data['stops'] = $this->trip_model->getTripStops($id);
            $this->load->view('template', $data);
    }

    public function add(){
            $this->session->set_userdata('action', 'add');
            $data['main_content']='trip/addTrip_view';
            $data['title']='Add Trip';
            $data['routes']=$this->trip_model->getRoutes();
            $data['stops']=$this->trip_model->getStops();
            $this->load->view('template', $data);
    }

    public function saveTripToDatabase(){

            $data = array(
                            'route_id' => trim($this->input->post('routeId')),
```

```
                        'trip_id' =>  $this->trip_model->getMarker(6),
                        'service_id' => 724594
                );

                $trip = $this->trip_model->saveTripToDatabase($data);
                $tripId = $data['trip_id'];
                $tripMarkerVal = $tripId + 1;

                if(!$trip){
                        $this->session->set_userdata('status','An error occurred!');
                        $this->session->set_userdata('message', 'Cannot save trip to database.');
                }
                else{
                        $stopData = json_decode($this->input->post('stopIds'));
                        $arriveData =json_decode($this->input->post('arrives'));
                        $departData = json_decode($this->input->post('departs'));

                        $tripData = array();

                        for($x = 0; $x < count($stopData); $x++){
                                $tripData[] = array(
                                                'trip_id' => $tripId,
                                                'stop_id' => $stopData[$x],
                                                'arrival_time' => $arriveData[$x],
                                                'departure_time' => $departData[$x]
                                );
                        }

                        $tripSuccess = $this->trip_model->saveTripStopsToDatabase($tripData);

                        if(!$tripSuccess){
                                $this->session->set_userdata('status_add','An error occurred!');
                                $this->session->set_userdata('message_add', 'Cannot save trip stops to
                                                                                        database.');
                        }
                        else{
                                $markerData = array('marker_value' => $tripMarkerVal );
                                $bool = $this->trip_model->updateMarkerToDatabase(6, $markerData);

                                if($bool){
                                        $this->session->set_userdata('status_add','Success!');
                                        $this->session->set_userdata('tripId',$tripId);
                                        $this->session->set_userdata('message_add','A new trip was
                                                                                        added.');
                                }
                        }
                }
        $this->output->set_content_type('application/json');
        $this->output->set_output(json_encode(array('result'=>$bool)));


        }

        public function edit($id){
                $this->session->set_userdata('action', 'edit');
                $data['main_content']='trip/editTrip_view';
                $data['title']='Edit Trip';
                $data['tripId'] = $id;
                $data['trip'] = $this->trip_model->getTripInfo($id);
                $data['tripStops'] = $this->trip_model->getStopsForTrip($id);
                $this->load->view('template', $data);
        }

        public function delete(){
                $data['id'] = $this->input->post('tripId');
                $data['type'] = $this->input->post('type');
                $this->load->view('delete_view', $data);
        }

        public function deleteTrip(){
                $tripId = $this->input->post('tripId');
                $trip = $this->trip_model->deleteTripToDatabase($tripId);

                if(!$trip){
                        $this->session->set_userdata('status_del','An error occurred!');
                        $this->session->set_userdata('message_del', 'Cannot delete Trip '.$tripId.' from
                                                                                database.');
                }
                else{
```

```
                        $this->session->set_userdata('status_del','Success!');
                        $this->session->set_userdata('tripId',$tripId);
                        $this->session->set_userdata('message_del','Trip '.$tripId.' has been successfully
                                                                                        deleted.');
                }
        }

        public function updateTripToDatabase(){
                $tripId = trim($this->input->post('tripId'));
                $stopCount = trim($this->input->post('stopCount'));

                $stopData = json_decode($this->input->post('stopIds'));
                $arriveData =json_decode($this->input->post('arrives'));
                $departData = json_decode($this->input->post('departs'));
                $lats = json_decode($this->input->post('lats'));
                $lons = json_decode($this->input->post('lons'));
                $stIds = json_decode($this->input->post('stIds'));

                $latLngData = array();
                for($x = 0; $x < count($stopData); $x++){
                        $latLngData[] = array(
                                                'stop_id' => $stopData[$x],
                                                'stop_lat' => $lats[$x],
                                                'stop_lon' => $lons[$x]
                        );
                }

                $transitStops = array();
                for($x = 0; $x < count($stopData); $x++){
                        $transitStops[$x] = array(
                                                'stop_times_id' => $stIds[$x],
                                                'trip_id' => $tripId,
                                                'stop_id' => $stopData[$x],
                                                'arrival_time' => $arriveData[$x],
                                                'departure_time' => $departData[$x]
                        );
                }

                $updateSuccess = $this->trip_model->updateStops($latLngData);

                if(!$updateSuccess){
                        $this->session->set_userdata('status','An error occurred!');
                        $this->session->set_userdata('message', 'Cannot update trip and stops to
                                                                                        database.');
                }
                else{
                        if($stopCount == count($stopData)){
                                if($this->trip_model->updateTimes($transitStops)){
                                        $this->session->set_userdata('status_edit','Success!');
                                        $this->session->set_userdata('tripId',$tripId);
                                        $this->session->set_userdata('message_edit','Trip '.$tripId.' was
                                                                                successfully updated.');
                                }
                        }
                        else{
                                $success = $this->trip_model-> deleteInsertTrip($transitStops, $tripId);
                                if($success){
                                        $this->session->set_userdata('status_edit','Success!');
                                        $this->session->set_userdata('tripId',$tripId);
                                        $this->session->set_userdata('message_edit','Trip '.$tripId.' was
                                                                                successfully updated.');
                                }
                        }
                }

                $this->output->set_content_type('application/json');
                $this->output->set_output(json_encode(array('result'=>$updateSuccess)));
        }
}
```

y. aroundmetro/application/controllers/stop.php

```
<?php if ( ! defined('BASEPATH')) exit('No direct script access allowed');

class Stop extends CI_Controller {

    function Stop()
    {
```

```
   parent::__construct();
         $this->load->model('stop_model');
         $this->session->set_userdata('page', 'stop');
}

/*
 * index() - always fired first in controller files
 * Inside this function, the main content (view file) will be assigned and called.
 */

public function index() {
         $this->session->unset_userdata('action');
         $data['main_content']='stop/stop_view';
         $data['title']='Stop';
         $data['stops']=$this->stop_model->getStops();
         $this->load->view('template', $data);
}

public function add(){
         $this->session->set_userdata('action', 'add');
         $data['title']='Add Stop';
         $data['main_content']='stop/addStop_view';
         $data['marker']=$this->stop_model->getMarker(5);
         $this->load->view('template', $data);
}

public function saveStopToDatabase(){
         $data = array(
                           'stop_id' => trim($this->input->post('stopId')),
                           'stop_name' => trim(ucwords($this->input->post('stopName'))),
                           'stop_lat' => trim($this->input->post('stopLat')),
                           'stop_lon' => trim($this->input->post('stopLon'))
         );
         $success = $this->stop_model->saveStopToDatabase($data);

         if(!$success){
                  $this->session->set_userdata('status','An error occurred!');
                  $this->session->set_userdata('message', 'Cannot save stop to database.');
         }
         else{
                  $markerVal = $this->stop_model->getMarker(5);
                  $markerVal = $markerVal + 1;
                  $markerData = array('marker_value' => $markerVal );
                  $bool = $this->stop_model->updateMarkerToDatabase(5, $markerData);

                  if($bool){
                           $this->session->set_userdata('status','Success!');
                           $this->session->set_userdata('stopId',$data['stop_id']);
                           $this->session->set_userdata('message','A new stop was added.');
                  }
         }

         redirect('stop');
}

public function view($id){
         $this->session->set_userdata('action', 'view');
         $data['main_content']='stop/viewStop_view';
         $data['title']='View Stop '.$id;
         $data['stopId'] = $id;
         $data['stop'] = $this->stop_model->getStopInfo($id);
         $this->load->view('template', $data);
}

public function edit($id){
         $this->session->set_userdata('action', 'edit');
         $data['main_content']='stop/editStop_view';
         $data['title']='Edit Stop '.$id;
         $data['stopId'] = $id;
         $data['stop'] = $this->stop_model->getStopInfo($id);
         $this->load->view('template', $data);
}

public function update(){
         $data = array(
                           'stop_name' => trim(ucwords($this->input->post('stopName'))),
                           'stop_lat' => trim($this->input->post('stopLat')),
                           'stop_lon' => trim($this->input->post('stopLon'))
         );
```

```
                $id = $this->input->post('secretId');
                $success = $this->stop_model->updateStopToDatabase($id, $data);

                if($success){
                        $this->session->set_userdata('status_update','Success!');
                        $this->session->set_userdata('message_update','Stop '.$id.' updated.');
                }
                else{
                        $this->session->set_userdata('status_update','An error ocurred!');
                        $this->session->set_userdata('message_update','Cannot update Stop '.$id);
                }

                redirect('stop');
        }

        public function delete(){
                $data['id'] = $this->input->post('stopId');
                $data['type'] = $this->input->post('type');
                $this->load->view('delete_view', $data);
        }

        public function deleteStop(){
                $stopId = $this->input->post('stopId');
                $stop = $this->stop_model->deleteStopToDatabase($stopId);

                if(!$stop){
                        $this->session->set_userdata('status_del','An error occurred!');
                        $this->session->set_userdata('message_del', 'Cannot delete Stop '.$stop.' from
                                                                                        database.');
                }
                else{
                        $this->session->set_userdata('status_del','Success!');
                        $this->session->set_userdata('stopId',$stopId);
                        $this->session->set_userdata('message_del','Stop '.$stopId.' has been successfully
                                                                                        deleted.');
                }
        }
}
```

z. aroundmetro/application/controllers/route.php

```
<?php if ( ! defined('BASEPATH')) exit('No direct script access allowed');

class Route extends CI_Controller {

        function Route()
        {
           parent::__construct();
                $this->load->model('route_model');
                $this->session->set_userdata('page', 'route');
        }

        /*
         * index() - always fired first in controller files
         * Inside this function, the main content (view file) will be assigned and called.
         */

        public function index() {
                $this->session->unset_userdata('action');
                $data['main_content']='route/route_view';
                $data['title']='Route';
                $data['routes']=$this->route_model->getRoutes();
                $this->load->view('template', $data);
        }

        public function add(){
                $this->session->set_userdata('action', 'add');
                $data['main_content']='route/addRoute_view';
                $data['title']='Add Route';
                $data['agency'] = $this->route_model->getAgencies();
                $data['marker'] = $this->route_model->getMarkers();
                $this->load->view('template', $data);
        }

        public function saveRouteToDatabase(){
                $data = array(
                                'agency_id' => trim($this->input->post('agency')),
```

```
                        'route_short_name' => trim(ucwords($this->input->post('routeShortName'))),
                        'route_long_name' => trim(ucwords($this->input->post('routeLongName'))),
                        'route_desc' => trim(ucwords($this->input->post('routeDescription'))),
                        'route_url' => trim($this->input->post('routeUrl')),
                        'route_type' => trim($this->input->post('routeType')),
                        'route_type_name' => trim($this->input->post('routeTypeName')),
                        'route_type_value' => trim($this->input->post('routeTypeVal')),
                        'route_id' => trim($this->input->post('routeId'))
                );

                $route = $this->route_model->saveRouteToDatabase($data);

                if(!$route){
                        $this->session->set_userdata('status','An error occurred!');
                        $this->session->set_userdata('message', 'Cannot save route to database.');
                }
                else{
                        $routeVehicleId = $this->input->post('routeTypeVal');
                        $markerVal = $this->route_model->getMarker($routeVehicleId);
                        $markerVal = $markerVal + 1;
                        $markerData = array('marker_value' => $markerVal );
                        $bool = $this->route_model->updateMarkerToDatabase($routeVehicleId, $markerData);

                        if($bool){
                                $this->session->set_userdata('status','Success!');
                                $this->session->set_userdata('routeId',$data['route_id']);
                                $this->session->set_userdata('message','A new route was added.');
                        }
                }
                redirect('route');
        }

        public function view($id){
                $this->session->set_userdata('action', 'view');
                $data['main_content']='route/viewRoute_view';
                $data['title']='View Route';
                $data['routeId'] = $id;
                $data['route'] = $this->route_model->getRouteInfo($id);
                $this->load->view('template', $data);
        }

        public function edit($id){
                $this->session->set_userdata('action', 'edit');
                $data['main_content']='route/editRoute_view';
                $data['title']='Edit Route';
                $data['routeId'] = $id;
                $data['route'] = $this->route_model->getRouteInfo($id);
                $data['agency'] = $this->route_model->getAgencies();
                $data['marker'] = $this->route_model->getMarkers();
                $this->load->view('template', $data);
        }

        public function delete(){
                $data['id'] = $this->input->post('routeId');
                $data['type'] = $this->input->post('type');
                $this->load->view('delete_view', $data);
        }

        public function deleteRoute(){
                $routeId = $this->input->post('routeId');
                $route = $this->route_model->deleteRouteToDatabase($routeId);

                if(!$route){
                        $this->session->set_userdata('status_del','An error occurred!');
                        $this->session->set_userdata('message_del', 'Cannot delete Route '.$routeId.' from
                                                                                        database.');
                }
                else{
                        $this->session->set_userdata('status_del','Success!');
                        $this->session->set_userdata('routeId',$routeId);
                        $this->session->set_userdata('message_del','Route '.$routeId.' has been successfully
                                                                                        deleted.');
                }
        }

        public function update(){
                $prevId = trim($this->input->post('prevRouteId'));
                $currId = trim($this->input->post('routeId'));
```

```
                $data = array(
                                'agency_id' => trim($this->input->post('agency')),
                                'route_short_name' => trim(ucwords($this->input->post('routeShortName'))),
                                'route_long_name' => trim(ucwords($this->input->post('routeLongName'))),
                                'route_desc' => trim(ucwords($this->input->post('routeDescription'))),
                                'route_url' => trim($this->input->post('routeUrl')),
                                'route_type' => trim($this->input->post('routeType')),
                                'route_type_name' => trim($this->input->post('routeTypeName')),
                                'route_type_value' => trim($this->input->post('routeTypeVal')),
                                'route_id' => trim($this->input->post('routeId'))
                );

                $success = FALSE;
                if($prevId == $currId){
                        $success = $this->route_model->updateRouteToDatabase($prevId, $data);
                }
                else{
                        $success = $this->route_model->saveRouteToDatabase($data);
                }

                if($success){
                        $this->session->set_userdata('status_update','Success!');
                        $this->session->set_userdata('message_update','Route '.$prevId.' updated.');
                }
                else{
                        $this->session->set_userdata('status_update','An error ocurred!');
                        $this->session->set_userdata('message_update','Cannot update Route '.$prevId);
                }

                redirect('route');
        }
}
```

aa. aroundmetro/application/controllers/login.php

```
<?php if ( ! defined('BASEPATH')) exit('No direct script access allowed');

class Login extends CI_Controller {

    /*
     * index() - always fired first in controller files
     * Inside this function, the main content (view file) will be assigned and called.
     */

    public function index() {
             $this->session->set_userdata("logged_in", FALSE);
             $this->session->set_userdata('page', 'login');  //start guest session
             $data['main_content']='user/login_view';
             $data['title']='Login';
             $this->load->view('template', $data);
    }


    /*
     * checkFields() -called from the view file upon submitting the login form (see 'login_view.php' file)
     * This function will validate the user input. Once validated, the user will be redirected to the
     * corresponding home page. Otherwise, the user will be redirected back to the login page.
     */
    public function checkFields() {
             $config = array(
             array(
                    'field'   => 'username',
                    'label'   => 'Username',
                    'rules'   => 'required'
                ),
              array(
                    'field'   => 'password',
                    'label'   => 'Password',
                    'rules'   => 'required|callback_verifyLogin'
                )
            );

             $this->form_validation->set_rules($config);


             if($this->form_validation->run() == false){
                      $data['main_content']='user/login_view';
                      $data['title']='Login';
```

```
                        $this->load->view('template', $data);
                }
                else{
                        redirect('route');
        }

        }
        /*end checking*/


        /*
         * verifyLogin() -called from the checkFields function
         * This fucntion will send the data input to the model for varification.
         */
        public function verifyLogin(){
                $username = $this->input->post('username');
                $password = $this->input->post('password');
                $this->load->model('login_model');

                $login=$this->login_model->login($username, $password);
                if($login) {
                        $this->session->set_userdata("username", $username);
                        $this->session->set_userdata("logged_in", TRUE);
                        return true;
                }

                else {
                        $this->form_validation->set_message('verifyLogin', 'The username/password is
                                                                                incorrect.');
                        return false;
                }
        }


        public function signout(){
                $this->session->unset_userdata("user_type");
                $this->session->unset_userdata("username");
                $this->session->unset_userdata("user");
                $this->session->set_userdata("logged_in", FALSE);
                return;
        }
}
```

bb. aroundmetro/application/controllers/graph.php

```
<?php if ( ! defined('BASEPATH')) exit('No direct script access allowed');

class Graph extends CI_Controller {

     function Graph()
     {
        parent::__construct();
               $this->load->model('graph_model');
               $this->session->set_userdata('page', 'graph');
     }


     /*
      * index() - always fired first in controller files
      * Inside this function, the main content (view file) will be assigned and called.
      */

     public function index() {
               $this->session->unset_userdata('action');
               $data['main_content']='graph/graph_view';
               $data['title']='Graph';
               //$data['routes']=$this->route_model->getRoutes();
               $this->load->view('template', $data);
     }

     public function updateGraph(){
               if (function_exists("set_time_limit") == TRUE AND @ini_get("safe_mode") == 0){
                   @set_time_limit(30000);
               }
               $files_to_zip = array(
                                 'routes.txt',
                                 'stops.txt',
                                 'stop_times.txt',
```

```
                        'trips.txt',
                        'agency.txt',
                        'calendar.txt',
                );

        $data = trim($this->input->post('data'));

        $routes = $this -> graph_model -> getRoutes();
        $this -> writeToText($routes, 'routes.txt');

        $trips = $this -> graph_model -> getTrips();
        $this -> writeToText($trips, 'trips.txt');

        $stops = $this -> graph_model -> getStops();
        $this -> writeToText($stops, 'stops.txt');

        $stopTimes = $this -> graph_model -> getTripStops();
        $this -> writeToText($stopTimes, 'stop_times.txt');

        create_zip($files_to_zip,'/home/aknarvaza/public_html/otp/gtfs/gtfs.zip');
        chdir('/home/aknarvaza/public_html/otp/gtfs/');
        exec('zip /home/aknarvaza/public_html/otp/gtfs/gtfs.zip '.$files_to_zip[0]);
        exec('zip /home/aknarvaza/public_html/otp/gtfs/gtfs.zip '.$files_to_zip[1]);
        exec('zip /home/aknarvaza/public_html/otp/gtfs/gtfs.zip '.$files_to_zip[2]);
        exec('zip /home/aknarvaza/public_html/otp/gtfs/gtfs.zip '.$files_to_zip[3]);
        exec('zip /home/aknarvaza/public_html/otp/gtfs/gtfs.zip '.$files_to_zip[4]);
        exec('zip /home/aknarvaza/public_html/otp/gtfs/gtfs.zip '.$files_to_zip[5]);


        exec("java -Xmx1024M -jar graph-builder.jar graph-config.xml", $results);
        $res = false;
        for($x = count($results)-1; $x >= 200; $x--){
                if(preg_match("/Graph written/", $results[$x])){
                        $res = true;
                        break;
                }
        }
        $this->output->set_content_type('application/json');
        $this->output->set_output(json_encode(array('result'=>$res)));
    }

    public function writeToText($data, $filename){
        $file = "/home/aknarvaza/public_html/otp/gtfs/".$filename;
        file_put_contents($file, implode(PHP_EOL, $data));
    }

    function create_zip($files = array(),$destination = '',$overwrite = true) {
        if(file_exists($destination) && !$overwrite) { return false; }
        $valid_files = array();
        if(is_array($files)) {
                foreach($files as $file) {
                        if(file_exists($file)) {
                                $valid_files[] = $file;
                        }
                }
        }
        if(count($valid_files)) {
                $zip = new ZipArchive();
                if($zip->open($destination,$overwrite ? ZIPARCHIVE::OVERWRITE :
                                ZIPARCHIVE::CREATE) !== true) {
                        return false;
                }
                foreach($valid_files as $file) {
                        if(file_exists($file) || is_readable($file)){
                                $parts  = explode('/', $file);
                                $zip->addFile($file,$parts[3]);
                        }
                }

                $zip->close();

                return file_exists($destination);
        }
        else
        {
                return false;
        }
    }
}
```

cc. aroundmetro/application/controllers/fare.php

```
<?php if ( ! defined('BASEPATH')) exit('No direct script access allowed');

class Fare extends CI_Controller {

     function Fare()
     {
        parent::__construct();
              $this->load->model('fare_model');
              $this->session->set_userdata('page', 'fare');
     }

     public function index() {
              $this->session->unset_userdata('action', 'view');
              $data['main_content']='fare/fare_view';
              $data['title']='Fare';
              $data['fares']=$this->fare_model->getFare();
              $this->load->view('template', $data);
     }

     public function rail(){
              $data['name'] = $this->input->post('name');
              $data['data'] = $this->input->post('data');
              $this->load->view('rail_view', $data);
     }

     public function pubj(){
              $data['name'] = $this->input->post('name');
              $data['baseFare'] = $this->input->post('baseFare');
              $data['firstKm'] = $this->input->post('firstKm');
              $data['nextKm'] = $this->input->post('nextKm');
              $this->load->view('busJeep_view', $data);
     }

}
```

dd. aroundmetro/application/models/trip_model.php

```
<?php
     /*
      * trip_model - the one which will connect to the database when data needs to be fetched and saved
      * This file is called from the trip.php controller
      */
     class trip_model extends CI_Model {

              public function getTrips(){
                       $sql = "Select t.route_id, t.trip_id, r.route_desc, r.route_long_name
                                          from trips as t, routes as r
                                          where r.route_id=t.route_id order by t.route_id";

                       $query=$this->db->query($sql);  //run the query
                       if ($query->num_rows() == 0){
                       return null;
                   }
                       $i=0;

                       foreach ($query->result() as $row){
                       $results[$i]['routeId']=$row->route_id;
                       $results[$i]['tripId']=$row->trip_id;
                                $results[$i]['routeDesc']=$row->route_desc;
                                $results[$i]['routeLongName']=$row->route_long_name;
                                $results[$i]['stopCount']=$this->trip_model->getStopCount($row->trip_id);
                                $i++;
                       }
                       return $results;
              }

              public function getStopCount($id){
                       $sql = "Select count(*) as stopCount from stop_times where trip_id = $id";
                       $query=$this->db->query($sql);
                       $row = $query->row();
                       return $row->stopCount;
              }
```

```
public function getTripStops($id){
         $sql = "Select x.stop_id, y.stop_lat, y.stop_lon, y.stop_name, x.arrival_time,
                   x.departure_time from stop_times as x, stops as y where x.trip_id = '$id'
                   and x.stop_id = y.stop_id order by x.stop_times_id";

         $query=$this->db->query($sql);
         if ($query->num_rows() == 0){
         return null;
    }
         $i=0;
         foreach ($query->result() as $row){
                   $results[$i]['stopId']=$row->stop_id;
         $results[$i]['stopName']=$row->stop_name;
         $results[$i]['stopLat']=$row->stop_lat;
                   $results[$i]['stopLon']=$row->stop_lon;
                   $results[$i]['arrTime']=$row->arrival_time;
         $results[$i]['dptTime']=$row->departure_time;
                   $i++;
         }
         return $results;
}

public function getTripInfo($id){
         $sql = "Select t.route_id, t.trip_id, r.route_desc, r.route_long_name,
                   r.route_type_name  from trips as t, routes as r
                   where t.trip_id = $id and r.route_id=t.route_id";

         $query=$this->db->query($sql);  //run the query
         if ($query->num_rows() == 0){
         return null;
    }

         foreach ($query->result() as $row){
         $results['routeId']=$row->route_id;
         $results['tripId']=$row->trip_id;
                   $results['routeDesc']=$row->route_desc;
                   $results['routeLongName']=$row->route_long_name;
                   $results['routeTypeName']=$row->route_type_name;
                   $results['stopCount']=$this->trip_model->getStopCount($row->trip_id);
         }
         return $results;
}

public function getRoutes(){
         $sql = "Select route_id, route_long_name
                             from routes
                             where route_id not in (select route_id from trips)";

         $query=$this->db->query($sql);  //run the query
         if ($query->num_rows() == 0){
         return null;
    }
         $i=0;
         foreach ($query->result() as $row){
         $results[$i]['routeId']=$row->route_id;
         $results[$i]['routeName']=$row->route_long_name;
                   $i++;
         }
         return $results;
}

public function getStops(){
         $sql = "Select stop_id, stop_lat, stop_lon, stop_name from stops";

         $query=$this->db->query($sql);  //run the query
         if ($query->num_rows() == 0){
         return null;
    }
         $i=0;
         foreach ($query->result() as $row){
                   $results[$i]['stopId']=$row->stop_id;
         $results[$i]['stopName']=$row->stop_name;
         $results[$i]['stopLat']=$row->stop_lat;
                   $results[$i]['stopLon']=$row->stop_lon;
                   $i++;
         }
         return $results;
}
```

```
public function getStopsForTrip($id){
         $sql = "Select y.stop_times_id, x.stop_id, x.stop_lat, x.stop_lon, x.stop_name,
                   y.arrival_time, y.departure_time from stops as x, stop_times as y
                   where y.trip_id=$id and x.stop_id = y.stop_id order by y.stop_times_id";

         $query=$this->db->query($sql);  //run the query
         if ($query->num_rows() == 0){
         return null;
    }
         $i=0;
         foreach ($query->result() as $row){
                   $results[$i]['stopTimesId']=$row->stop_times_id;
                   $results[$i]['stopId']=$row->stop_id;
         $results[$i]['stopName']=$row->stop_name;
         $results[$i]['stopLat']=$row->stop_lat;
                   $results[$i]['stopLon']=$row->stop_lon;
                   $results[$i]['stopArr']=$row->arrival_time;
                   $results[$i]['stopDep']=$row->departure_time;
                   $i++;
         }
         return $results;
}

public function getMarker($id){
         $sql="SELECT marker_value from marker WHERE marker_id='$id'";
         $query=$this->db->query($sql);
         $row = $query->row();
         return $row->marker_value;
}

public function saveTripToDatabase($data){
         if($this->db->insert('trips',$data)){
                   return true;
         }
         else
                   return false;
}

public function saveTripStopsToDatabase($data){
         if($this->db->insert_batch('stop_times',$data)){
                   return true;
         }
         else
                   return false;
}

public function updateMarkerToDatabase($id, $data){
         if($this->db->update('marker', $data, "`marker_id` = '".$id."'")){
                   return true;
         }
         return false;
}

public function updateStops($data){
         $this->db->trans_start();
         $this->db->update_batch('stops', $data, 'stop_id');
         $this->db->trans_complete();

         if ($this->db->trans_status() === FALSE){
              return false;
         }
         else{
             return true;
         }
}

public function updateTimes($data){
         $x =0;
         foreach ($data as $d) {
                   $this->db->where('stop_times_id', $d['stop_times_id']);
                   $this->db->update('stop_times', $d);
                   $x++;
         }

         if($x == count($data))
                   return true;
         else return false;

}
```

```
	public function deleteTripToDatabase($id){
		return $this->db->delete('trips', array('trip_id' => $id));
	}

	public function deleteInsertTrip($data, $id){
		$this->db->where('trip_id', $id);
		if($this->db->delete('stop_times')){
			if($this->db->insert_batch('stop_times',$data))
				return true;
			else
				return false;
		}
		else {
			return false;
		}
	}
    }
?>
```

ee. aroundmetro/application/models/stop_model.php

```
<?php
    /*
     * trip_model - the one which will connect to the database when data needs to be fetched and saved
     * This file is called from the trip.php controller
     */
    class stop_model extends CI_Model {

	public function getStops(){
		$sql = "Select *  from stops order by stop_id";
		$query=$this->db->query($sql);  //run the query
		if ($query->num_rows() == 0){
		return null;
		}
		$i=0;

		foreach ($query->result() as $row){
		$results[$i]['stopId']=$row->stop_id;
			$results[$i]['stopName']=$row->stop_name;
			$results[$i]['stopLat']=$row->stop_lat;
			$results[$i]['stopLon']=$row->stop_lon;
			$i++;
		}
		return $results;
	}

	public function getStopInfo($id){
		$sql = "Select *  from stops where stop_id='$id'";
		$query=$this->db->query($sql);  //run the query
		if ($query->num_rows() == 0){
		return null;
		}

		foreach ($query->result() as $row){
		$results['stopId']=$row->stop_id;
			$results['stopName']=$row->stop_name;
			$results['stopLat']=$row->stop_lat;
			$results['stopLon']=$row->stop_lon;
		}
		return $results;
	}

	public function updateStopToDatabase($id, $data){
		if($this->db->update('stops', $data, "`stop_id` = '".$id."'")){
			return true;
		}
		return false;
	}

	public function updateMarkerToDatabase($id, $data){
		if($this->db->update('marker', $data, "`marker_id` = '".$id."'")){
			return true;
		}
		return false;
	}

	public function saveStopToDatabase($data){
```

```
                if($this->db->insert('stops', $data)){
                        return true;
                }
                return false;
        }

        public function deleteStopToDatabase($id){
                return $this->db->delete('stops', array('stop_id' => $id));
        }

        public function getMarker($id){
                $sql="SELECT marker_value from marker WHERE marker_id='$id'";
                $query=$this->db->query($sql);
                $row = $query->row();
                return $row->marker_value;
        }
    }
?>
```

ff. aroundmetro/application/models/route_model.php

```
<?php
    /*
     * route_model - the one which will connect to the database when data needs to be fetched and saved
     * This file is called from the route.php controller
     */
    class route_model extends CI_Model {

        public function getRoutes(){
                $sql="SELECT route_id, route_desc, route_short_name, route_long_name from routes";
                $query=$this->db->query($sql);  //run the query
                if ($query->num_rows() == 0){
                return null;
            }
                $i=0;

                foreach ($query->result() as $row){
                $results[$i]['id']=$row->route_id;
                $results[$i]['desc']=$row->route_desc;
                        $results[$i]['routeShortName']=$row->route_short_name;
                        $results[$i]['routeLongName']=$row->route_long_name;
                        $i++;
                }
                return $results;
        }

        public function getAgencies(){
                $sql="SELECT agency_id, agency_name from agency";
                $query=$this->db->query($sql);  //run the query
                if ($query->num_rows() == 0){
                return null;
            }
                $i=0;

                foreach ($query->result() as $row){
                $results[$i]['id']=$row->agency_id;
                $results[$i]['name']=$row->agency_name;
                        $i++;
                }
                return $results;
        }

        public function getMarkers(){
                $sql="SELECT * from marker";
                $query=$this->db->query($sql);  //run the query
                if ($query->num_rows() == 0){
                return null;
            }
                $i=1;

                foreach ($query->result() as $row){
                $results[$i]['id']=$row->marker_id;
                $results[$i]['name']=$row->marker_name;
                        $results[$i]['value']=$row->marker_value+1;
                        $i++;
                }
                return $results;
        }
```

```
		public function getMarker($id){
			$sql="SELECT marker_value from marker WHERE marker_id='$id'";
			$query=$this->db->query($sql);
			$row = $query->row();
			return $row->marker_value;
		}

		public function getAgencyName($id){
			$sql="SELECT agency_name from agency WHERE agency_id='$id'";
			$query=$this->db->query($sql);
			$row = $query->row();
			return $row->agency_name;
		}

		public function saveRouteToDatabase($data){
			$result=$this->db->insert('routes',$data);
			if($result){
				return true;
			}
			else
				return false;
		}

		public function updateMarkerToDatabase($id, $data){
			if($this->db->update('marker', $data, "`marker_id` = '".$id."'")){
				return true;
			}
			return false;
		}

		public function updateRouteToDatabase($id, $data){
			if($this->db->update('routes', $data, "`route_id` = '".$id."'")){
				return true;
			}
			return false;
		}

		public function deleteRouteToDatabase($id){
			return $this->db->delete('routes', array('route_id' => $id));
		}

		public function getRouteInfo($id){
			$sql="SELECT * from routes where route_id='$id'";
			$query=$this->db->query($sql);  //run the query
			if ($query->num_rows() == 0){
			return null;
		    }

				foreach ($query->result() as $row){
				$results['id']=$row->route_id;
				$results['routeShortName']=$row->route_short_name;
					$results['routeLongName']=$row->route_long_name;
					$results['routeDesc']=$row->route_desc;
					$results['routeTypeValue']=$row->route_type_value;
					$results['routeTypeName']=$row->route_type_name;
					$results['routeUrl']=$row->route_url;
					$results['agency']=$this->route_model->getAgencyName($row->agency_id);
				}
				return $results;
		}
	}
?>
```

gg. aroundmetro/application/models/login_model.php

```
<?php
    /*
     * login_model - the one which will connect to the database when data needs to be fetched
     * This file is called from the login.php controller
     */
    class login_model extends CI_Model {

		/*
		 * login() - called from the verifyLogin(), login.php controller
		 */
		public function login($username, $password){
			$sql="SELECT * from users where user_name='$username' and password='$password'";
```

```
                        $query=$this->db->query($sql);
                        if($query) {
                                 return $query->row();
                        }else {
                                return NULL;
                        }
                }
        }

?>
```

hh. aroundmetro/application/models/graph_model.php

```
<?php
    /*
     * route_model - the one which will connect to the database when data needs to be fetched and saved
     * This file is called from the route.php controller
     */
    class graph_model extends CI_Model {

            public function getRoutes(){
                    $sql="SELECT route_id, route_desc, route_short_name, route_long_name,
                            route_type, agency_id from routes";
                    $query=$this->db->query($sql);  //run the query
                    if ($query->num_rows() == 0){
                    return null;
                }


    $results[0]='agency_id,route_id,route_short_name,route_long_name,route_desc,route_type';
                    $i=1;
                    foreach ($query->result() as $row){
                    $results[$i]=$row->agency_id.','.$row->route_id.','.
                            $row->route_short_name.',"'.$row->route_long_name.'","'.
                            $row->route_desc.'",'.$row->route_type;
                            $i++;
                    }
                    return $results;
            }

            public function getTrips(){
                    $sql="SELECT route_id, service_id, trip_id from trips";
                    $query=$this->db->query($sql);  //run the query
                    if ($query->num_rows() == 0){
                    return null;
                }

                    $results[0]='route_id,service_id,trip_id';
                    $i=1;
                    foreach ($query->result() as $row){
                    $results[$i]=$row->route_id.','.$row->service_id.','.$row->trip_id;
                            $i++;
                    }
                    return $results;
            }

            public function getStops(){
                    $sql="SELECT stop_id, stop_name, stop_lat, stop_lon from stops";
                    $query=$this->db->query($sql);  //run the query
                    if ($query->num_rows() == 0){
                    return null;
                }

                    $results[0]='stop_id,stop_name,stop_lat,stop_lon';
                    $i=1;
                    foreach ($query->result() as $row){
                    $results[$i]=$row->stop_id.',"'.$row->stop_name.'",'.
                                    $row->stop_lat.','.$row->stop_lon;
                            $i++;
                    }
                    return $results;
            }

            public function getTripStops(){
                    $sql="SELECT trip_id from trips";
                    $query=$this->db->query($sql);  //run the query
                    if ($query->num_rows() == 0){
                    return null;
```

```
                }
                        $results[0]='trip_id,stop_sequence,stop_id,arrival_time,departure_time';
                        $i=1;
                        foreach ($query->result() as $row){
                                $index = 1;
                        $trips = $this-> getStopTimes($row->trip_id);

                                foreach($trips as $t){
                                        $results[$i]=$t['trip_id'].','.$index.','.$t['stop_id'].','.
                                                        $t['arrival_time'].','.$t['departure_time'];
                                        $index++;
                                        $i++;
                                }
                        }
                        return $results;
                }

                public function getStopTimes($id){
                        $sql="SELECT trip_id, stop_id, arrival_time, departure_time from stop_times where
                        trip_id = '".$id."' order by stop_times_id";
                        $query=$this->db->query($sql);  //run the query
                        if ($query->num_rows() == 0){
                        return null;
                    }

                        $i=0;
                        foreach ($query->result() as $row){
                        $results[$i]['trip_id']=$row->trip_id;
                        $results[$i]['stop_id']=$row->stop_id;
                                $results[$i]['arrival_time']=$row->arrival_time;
                                $results[$i]['departure_time']=$row->departure_time;
                                $i++;
                        }
                        return $results;
                }
        }
?>
```

ii. aroundmetro/css/style.css

```
body {
  padding-top: 50px;
}
.navbar{
  margin-bottom:0;
}

/*submenu*/
.submenu{
  padding: 10px 0 10px 0;
  background-color: #e7e7e7;
}
.col-lg-6{
  width: 25%;
  padding-left: 0;
}
.searchBtn{
     padding: 9px;
}
.inactive-users{
     display:inline;
     margin-left: 10px;
}

/*route's table*/
.panel{
     margin-bottom:0;
}
.panel-heading{
     padding:0;
     border-bottom:0;
}
.form-control{
     display: inline;
     width: auto;
}
.table{
     table-layout:fixed;
     margin-bottom:0;
}
.infodetails td{
     border-top:none;
}
.infodetails tr{
     padding:50px;
}

.routes-table{
  width: 100%;
  float: left;
  margin-top: 8px;
}
.data{
  padding: 8px;
  position: relative;
  max-height: 85%;
  overflow-y:auto;
  height: 85%;
}

/*for information*/
.details td{
     padding: 10px;
}
.details th{
     border-bottom:1px solid #ddd;
     font-size: 15px;
     padding: 6px;
}
.stops th{
     font-size: 15px;
     padding: 6px;
}
```

```
.stops tr:hover td{
    background-color: #E7E7E7;
}
/*
.routes td, .routes th{
    width:24.5%;
    word-wrap: break-word;
}*/

.routes tr:hover td{
   background-color: #E7E7E7; /* or #000 */
}

/*properties of table*/
.panel{
    margin-bottom:0;
}
.panel-heading{
    padding:0;
    border-bottom:0;
}
.table{
    table-layout:fixed;
    margin-bottom:0;
}
.records{
    max-height: 78%;
    overflow-y:auto;
    height: 85%;
}
.highlight{
    background: #E7E7E7;
}
.wrapper{
  margin-top: 8px;
  max-height: 88%;
  overflow-y:auto;
}
.info{
    padding: 8px;
}
.view th{
    font-size: 15px;
    padding: 6px;
}

/*route details view*/
.details td{
    padding: 5px;
}
.details th{
    border-bottom:1px solid #ddd;
    font-size: 15px;
    padding: 6px;
}
/*route trip view*/
.leftPanel{
  width: 49%;
  float: left;
  padding-left: 4px;
}
.rightPanel{
  width: 49%;
  height: 85%;
  float: right;
  padding-left: 4px;
}

/*fare view*/
.leftFarePanel{
  width: 49%;
  float: left;
}
.rightFarePanel{
  width: 49%;
  float: right;
}
/*header in fancybox*/

.header{
    background: #3276b1;
    text-align: center;
    color: #FFFFFF;
    padding: 10px;
    margin-top: 0;
    font-size: 17px;
    font-style: bold;
}
.fancy{
    padding: 10px;
}
.fancy h4{
    margin-top: 0;
    margin-bottom: 5px;
}
/*other css*/
.css1{
    width:10%;
}

.css2 input:hover{
    background-color: #E7E7E7;
}

.css3 tr:hover{
    background-color: #E7E7E7;
}

/*scrollbar*/
::-webkit-scrollbar {
    width: 8px;
    height: 8px;
}
::-webkit-scrollbar-track {
    -webkit-box-shadow: inset 0 0 6px
rgba(0,0,0,0.3);
    -webkit-border-radius: 10px;
    border-radius: 10px;
}
::-webkit-scrollbar-thumb {
    -webkit-border-radius: 10px;
    border-radius: 10px;
    background: rgba(221, 221, 221, 1.0);
    -webkit-box-shadow: inset 0 0 6px
rgba(0,0,0,0.5);
}
::-webkit-scrollbar-thumb:window-inactive {
    background: rgba(221, 221, 221, 1.0);
}
```

4. Hypothetical Traffic Monitoring System

a. mmda/editTraffic.php

```
<?php
     mysql_connect("localhost", "root", "root") or die(mysql_error());
     mysql_select_db("aroundMetro") or die(mysql_error());

     $data = mysql_query("SELECT * FROM traffic") or die(mysql_error());
?>
<html>
     <head>
        <title>MMDA Traffic</title>
        <script src="https://maps.googleapis.com/maps/api/js?v=3.exp&sensor=false"></script>
        <script src="jquery-1.10.2.min.js"></script>
        <script src="jquery-ui.js"></script>
        <script type="text/javascript" src="fancybox/jquery.fancybox.js?v=2.1.5"); ?>></script>
        <link rel="stylesheet" type="text/css" href="fancybox/jquery.fancybox.css?v=2.1.5"media="screen" />
        <link href="btn.css" rel="stylesheet" />
     </head>

     <body>
               <div>
               <form id="thisForm" method="post" action="update.php">
               <div style="width: 100%; height: 10%; background-color: #E7E7E7">
                        <strong><p style="position: absolute; margin-left: 1%;  margin-bottom: 20px; font-
                                 size: 20px; font-family: 'Segoe UI Light','Segoe WPC','Segoe UI'; color:
                                 #0066CC">Pseudo MMDA Traffic Monitoring System</p></strong>

                        <input class="myButton" style="float: right; margin-right: 2%; margin-top: 10px;"
                                 type="submit" value="UPDATE" />
               </div>
               <div class="leftPanel" style="height:100%; overflow-y: auto; display: none;">
                        <table id="area" class="areaTable" style="display: none;">
                                 <?php
                                          $water = array(81, 82, 91, 92, 101, 102, 111, 112, 121, 122, 131,
                                                   132,141, 142, 151, 152, 161, 162, 171, 160,
                                                   169, 170, 179, 180, 189, 190, 199, 200);
                                          while($rows=mysql_fetch_array($data)){
                                 ?>
                                 <tr>
                                          <?php
                                                   if(in_array($rows['area_id'], $water))
                                                            echo "<td id = 'inWater'></td>";
                                                   else
                                                            echo "<td id = 'outWater'></td>";
                                          ?>
                                 <td id='<?php echo $rows['area_code']; ?>' class='areaTDview'><?php echo
                                          $rows['area_name'];?></td>
                                 <td class='info' style='width: 75%; text-align: center;'><?php echo
                                          $rows['traffic_info']; ?></td>
                                 <td><input type="hidden" name="info[]" id="<?php echo "id".$rows['area_id'];?>"
                                          value="<?php echo $rows['traffic_info']?>">

                                          </tr>
                                          <?php } ?>
                                          </form>
                                 </table>
               </div>
                                 <div id="map-canvas" style='width: 100%; height:90%; float:right'>         </div>
               </div>
               <div id="changeTraffic" title="Change Traffic Information" style="display: none">
                        <p align="center">
                                          Change Traffic state of these sectors
                        </p>
                        <br />
                        <center>
                                 <input type=button id="lightBtn" value="LIGHT">
                                 <input type=button id="modBtn" value="MODERATE">
                                 <input type=button id="heavyBtn" value="HEAVY">
                        </center>
               </div>
     </body>
     <script>
               var ulBound = new google.maps.LatLng(14.665761,120.950546);
               var xDif = 0.0138359;
               var yDif = 0.0104315;

               var myLatLng = new google.maps.LatLng(14.568195,121.034660);
```

```
var mapOptions = {
                  disableDoubleClickZoom: true,
                  zoom: 12,
                  mapTypeId: google.maps.MapTypeId.ROADMAP,
                  center: myLatLng
       };

var map = new google.maps.Map(document.getElementById('map-canvas'), mapOptions);
var ctrlUp = false;
var rects = [];
var boxes = [];

$(document).ready(function(){
         $("#changeTraffic").hide();
         $('#area tr').each(function() {
                  var show = $(this).closest('tr').children('td').eq(0).attr('id');
                  var temp = $(this).closest('tr').children('td').eq(1).attr('id');
        var text = $(this).closest('tr').children('td').eq(2).html();

        var color, color2;
        if(text == "LIGHT TRAFFIC"){
          color = "#00CC00";
          color2 = "#009900";
        }
        else if(text == "MODERATE TRAFFIC"){
          color2 = "#FFD119";
          color = "#FFFF19";
        }
        else{
          color = "#FF0000";
          color2 = "#A30000";
        }

                  var t = temp.split('_');
                  if(show != "inWater")
                           catcher(t[0] + ":" + t[1], color, color2, text);
         });

         $('#lightBtn').click(function () {
                  changeColor('#00CC00','#009900', 'LIGHT TRAFFIC');
                  $.fancybox.close();
         });
         $('#modBtn').click(function () {
                  changeColor('#FFFF19','#FFD119', 'MODERATE TRAFFIC');
                  $.fancybox.close();
         });
         $('#heavyBtn').click(function () {
                  changeColor('#FF0000', '#A30000', 'HEAVY TRAFFIC');
                  $.fancybox.close();
         });
});

function catcher(text, fill, stroke, str){
         var temp = text.split(':');
         var row = parseInt(temp[0]);
         var col = parseInt(temp[1]);
         var index = ((row -1)*10) + col;

         var topLng = ulBound.lng() + (xDif*(col-1));
         var topLat = ulBound.lat() - (yDif*(row-1));

         var botLng = ulBound.lng() + (xDif*col);
         var botLat = ulBound.lat() - (yDif*row);

         var rectBounds = new google.maps.LatLngBounds(
               new google.maps.LatLng(topLat, topLng),
               new google.maps.LatLng(botLat, botLng));

         var rectangle = new google.maps.Rectangle({
             strokeColor: stroke,
             strokeOpacity: 0.8,
             strokeWeight: 2,
             fillColor: fill,
             fillOpacity: 0.35,
             map: map,
             bounds: rectBounds
         });

         rectangle.id = index;
```

```
		boxes.push(rectangle);

		google.maps.event.addListener(rectangle,"click",function(event){
			if(ctrlUp == true){
				this.setOptions({
							fillColor: '#E7E7E7',
							strokeColor: '#E7E7E7'
					});
				rects.push(rectangle);
			}
			else{
				console.log("click");
				var info;
				if(this.fillColor == '#00CC00'){
					info = "MODERATE TRAFFIC";
					this.setOptions({
							fillColor: '#FFFF19',
							strokeColor: '#FFD119'
					});
				}
				else if(this.fillColor == '#FFFF19'){
					info = "HEAVY TRAFFIC";
					this.setOptions({
							fillColor: '#FF0000',
							strokeColor: '#A30000'
					});
				}
				else{
					info = "LIGHT TRAFFIC";
					this.setOptions({
							fillColor: '#00CC00',
							strokeColor: '#009900'
					});
				}
			}

			var elem = $("#id"+this.id);
			elem.val(info);
		});
}

var down = [];
$(document).keydown(function(e){
		if(e.ctrlKey){
			ctrlUp = true;
		}
		down[e.keyCode] = true;
}).keyup(function(e){
		if(e.keyCode == 17){
			$("#changeTraffic").show();
			showDiv();
			ctrlUp = false;
		}
		if(down[16] && down[76]){
			changeAll('#00CC00','#009900', 'LIGHT TRAFFIC');
		}

		if(down[16] && down[72]){
			changeAll('#FF0000', '#A30000', 'HEAVY TRAFFIC');
		}

		if(down[16] && down[77]){
			console.log("moderate all");
			changeAll('#FFFF19','#FFD119', 'MODERATE TRAFFIC');
		}
		down.length = 0;
});

function showDiv(){
		$.fancybox({
	'width': '20%',
	'height': '20%',
	'autoScale': true,
	'transitionIn': 'fade',
	'transitionOut': 'fade',
	'type': 'inline',
	'href': '#changeTraffic',
	'modal': false,
	'autoSize': false
```

```
            });
              }

              function changeColor(fill, stroke, text){
                       for(var x = 0; x < rects.length; x++){
                                for(var y =0; y < boxes.length; y++){
                                         if(rects[x].id == boxes[y].id){
                                                  boxes[y].setOptions({
                                                           fillColor: fill,
                                                           strokeColor: stroke
                                                  });
                                                  var elem = $("#id"+boxes[y].id);
                                                  elem.val(text);
                                         }
                                }
                       }
                       rects.length = 0;
              }

              function changeAll(fill, stroke, text){
                       for(var y =0; y < boxes.length; y++){
                                boxes[y].setOptions({
                                         fillColor: fill,
                                         strokeColor: stroke
                                });
                                var elem = $("#id"+boxes[y].id);
                                elem.val(text);
                       }
              }
     </script>
</html>
```

b. mmda/update.php

```
<?php
     mysql_connect("localhost", "root", "root") or die(mysql_error());
     mysql_select_db("aroundMetro") or die(mysql_error());

     $data = mysql_query("SELECT * FROM traffic") or die(mysql_error());

     $a = array("LIGHT TRAFFIC", "LIGHT TRAFFIC", "LIGHT TRAFFIC");
     $b = array("LIGHT TRAFFIC", "MODERATE TRAFFIC", "HEAVY TRAFFIC");

     $info = $_POST['info'];
     $data = array();
     $rowCode =1;
     $colCode = 1;
     for($x =1; $x<=200; $x++){
              if($x % 10 ==1 && $x > 10) $rowCode++;
              if($colCode > 10) $colCode = 1;
              $data[$x] = array(
                                        'area_name' => 'Area '.$x,
                                        'traffic_info' => $info[$x-1],
                                        'area_code' => $rowCode."_".$colCode
                                         );
              $colCode++;
     }
     $index =1;
     foreach ($data as $d){
              mysql_query("UPDATE traffic SET `traffic_info` ='".$d['traffic_info']."'
                                                           WHERE `area_id` = '$index'");
              $index++;
     }

     mysql_close();
     echo "<script>
                       window.location.replace('editTraffic.php');
                       </script>";
?>
```

XII. Acknowledgement

Everything really happens for a reason. Andami nating what-ifs at mga "paano kaya kung…" moments na minsan nakakalimutan natin i-appreciate ang kung ano meron tayo. Nalalaman na lang natin ang halaga ng isang bagay kapag wala na. Kaya ngayon, ibubuhos ko lahat ng pasasalamat ko dito. Read at your own risk. OK, eto na….

First and foremost, sa **Panginoong Diyos**. Kay Papa God. Kay Bro. Siguro iisipin mong ang corny nitong acknowledgement kasi ang typical masyado. Pero wala eh. Hindi mo kasi ramdam. Alam mo yun, yung sa panahong tinatapos ko ang SP ko, sa panahong nagbi-breakdown ako (mga twice or thrice ata yun), sa oras ng defense ko, sa paghahabol ko ng revisions, at sa pagtapos ng paper ko, isa lang ang parati kong sinasabi, "Lord, last push na ituuu. Patulak naman!"

Pero seriously, di ko alam pano ko Ikaw mapapasalamatan, Lord. You never let go kahit alam ko naman na ako ang may kasalanan kung bakit ako naghahabol ng bongga. Pero you stayed with me. OMG. I cannot thank you enough. And for that, I will always be your vessel. Use me as You will. Forever and ever. Amen.

Siyempre nakasunod sa listahan ang mga taong tumulong at gumabay sa akin sa pagtapos nito. Unang-una na doon si **Sir Greg Baes**. Sa lahat po ng inputs niyo, lahat ng comments, suggestions, advise, dagdag na trabaho, comments ulit, dagdag na trabaho, dagdag na trabaho, advise, at siyempre pa ang matamis niyong "GO" signal. Alam ko naman po na lahat ng 'yun ay para sa ikagaganda po ng SP ko, kaya po Super Thank You po talaga. Sa pagtitiis po sa akin sa panahong ako'y lutang at sabog. Salamat po sa patience at effort niyo. Hinding-hindi ko po kayo makakalimutan. Lalo na po yung tinawag niyo po ako by my first name. Siguro yun na yung first and last. Pero still, memorable. Gusto ko

din po malaman niyo, na proud ako and forever grateful kasi isa ako sa mga advisees niyo. Thank you po, Sir.

Isusunod ko na ang lahat ng mga Profs na aking na-perwisyo at mga Profs na pinagaan ang aking loob. Kay **Sir Bryann Chua**, na aking naperwisyo ng bongga dahil sa server, Sir, sorry po talaga. Last na po 'yun. :)) Kay **Sir Geoffrey Solano**, at sa kanyang JPEG jokes, Sir thank you po sa lahat po ng encouragement at support. Kay **Maam Ave Carpio**, na super pretty and super bait, Maam mamimiss ko po kayo kahit di niyo po ako ma-mimiss. Thank you Maam. At siyempre sa Panel ko nung defense, hindi ko talaga ineexpect na ganun ang aking defense. Maraming salamat. Kay **Doc Peter Magboo,** at kay **Sir Marvin Ignacio,** thank you po. At special mention kay **Maam Eden,** na hulog ng langit. Thank you Maam sa pagsagot sa aking mga academic needs. Isa kang anghel sa lupa.

But opkors, makakalimutan ko ba ang aking mga **PRENDS**?! Haru! Sa mga panahong ako'y lugmok at depressed at walang patutunguhan (though halos parehas naman tayong lost souls), salamat dahil parati kayong andiyan para ipagluto ako, patulugin sa kama niyo, at sa pagpagamit ng malakas na Intenet. Thank you talaga ng marami.

Kay **Pebbe Bunoan,** dahil andun ka nung nagpapanic ang aking kaluluwa at ikaw ang taga-neutralize, salamat ng bonggabels. Hindi mo ako iniwan hanggang defense ko omg it's so touching I wanna cry ;((Thank you talaga ng marami Pebs. I don't know what I would do without you.

Kay **Janella Marey,** aka **Mother Dragon,** salamat sa iyong masarap na nilulutong puds, sa mga Words of Wisdom mo, sa mga jokes at mga kwento, sa pagtitiwala sa akin, thank you so much, Bitch! Ikaw na talaga ang Reyna at ang Diva! *I woke up like this**I woke up like this**FLAWLESS!*

Kay **JM Vitor,** (char sa JM hahaha), sa pagsama sakin magpuyat o sa pagpuyat sa akin *HUUUYYYY!* Joke. I mean sa pagsama sa akin magpuyat at sa mga kwento mo para lang di ako makatulog, Thanks Mau! Ikaw ang aking naging wonder counselor ko lalo na pagdating sa aking mga dilemma sa UI, sa system, at sa aking buhay. Charaught! Huwag kang mag-alala, pag ikaw naman nangangailangan, isang text lang ako. Wala nga lang load. ;) Thanks, Mau. *NP: Underpressure*

At siyempre sa mga taong binati ako nung defense ko at nagsabi sakin ng "Good Luck," Thanks, guys! I really need all the luck in the world nung oras na yun. Kay **Patricia, Jamie, Ate Gela, Jennifer, Talen,** salamat talaga. Dahil kahit hindi naman tayo masyado nagkakusap, naalala niyo pa din ako. Labyu, guys. Mwahugs XOXO

And siyempre, sa 3 Elements of life: si Bagis, si Upaw, kag si Medyas, kamo gid ang inspiration ko habang ginatapos ko ang SP. Kay ginaisip ko nga malagaw ta dayon after grad. LOLS **Caru** and **Titel**, thanks for the friendship nga asta subong still alive and kicking. Sorry kung wala gid ko kayo kapabatyag during SP days, busy lang gid gals. Mabawi gid ko sa inyo. Promise!

And of course, sa akon mga ginikanan, **Mang, Pang**, Thank you gid kay wala gid kamo nadulaan salig sa akon kag sa padayon niyo nga pag-suporta sa akon. I cannot thank you enough. Sa tanan ta nga naagyan and stuff, wala gid kamo nagbigay. Sorry lang gid kung medyo nadugayan ining part nga ni (SP). Sorry kay kinpabungol-bungolan ko lang sadto. Sorry gid. Thank you gid sa tanan niyo nga efforts sa amon ni B-girl. I promise nga hindi makadto sa wala tanan niyo nga pag-antos kag paningkamot. Salamat gid sa tanan. I love you Mang. I love you Pang.

Kag kay **Jonah Mae,** sorry kay daw halos wala ko pulos nga brother sa imo, pasensiya lang gid. Salamat sa pag-atipan sa akon sa panahon nga ngarag ko kag wala sa buot. Salamat

sa pagluto ka sud-an kag sa panghugas ka plato. Mayad lang may manghod ko nga kaintitndi sang akon mga ginaagyan. Hindi ko man mahambal ni in person, pero I am so proud nga may manghod ko like you. Kag Magna Cum Laude pa ha. Odiba? Flying colors. I promise that I will be by your side forever. And that you can always rely on me. Love you, Sissy! :*

Also, sa aking Self. Dahil nakayanan mo lahat ng emotional, physical, mental, spiritual, and all traumatic challenges. Thank you, Self,6 kasi hindi ka bumigay kahit napapabayaan na kita. I promise I will take care of you after nito. Maliligo na ako at magto-toothbrush. Charing. I love you, Self. :))

Muntik ko na makalimutan. Sa aking mga inspirasyon, sa aking mga crushes, OMG ang gwapo niyo pa din kahit di ko na kayo nakikita in person sa CAS. May Facebook pa naman kayo, okay na yun. Thank you, simply because nag-exist kayo para magbigay inspirasyon sa lahat (alam kong di ako nag-iisa) at kahit na alam naman namin na wala talagang pag-asa sa inyo, OK lang yun. Basta mahalaga buhay kayo at maayos ang kalagayan. Salamat dahil ginawa niyong makulay ang College life ko. OMG ang boring ng buhay ko – hanggang crush? WEAK. Pero anyway, mag-ingat kayo lagi. At sana makita ko pa kayo one of these days. *wenk**wenk* Kasama ka dun, Gerald, wag kang mag-aalala.

Okay, that is all. Career ba masyado? Wala eh. Last na 'to. Kaya Career kung career. Ok sige, maliligo na ako. Itutuloy ko na lang to sa aking self-titled novel. Sa lahat, once again, SUPER DUPER MEGA THANK YOU! :)