University of the Philippines Manila
College of Arts and Science
Department of Physical Sciences and Mathematics

A. CO: Android for Customer Organization

An Android Application for Customer Queue Management Based on J. Co's Business Processes

A Special Problem in Partial Fulfillment
of the Requirements for the Degree of
Bachelor of Science in Computer Science

Submitted by:

Beverly Cay L. Festejo
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 "A.CO: Android for Customer Organization, an Android Application for Customer Queue Management Based on J.Co's Business Processes prepared and submitted by Beverly Cay L. Festejo in partial fulfillment of the requirements for the degree of Bachelor of Science in Computer Science has been examined and is recommended for acceptance.

_____

Aldrich Colin K. Co, M.Sc. ***(candidate)***
Adviser

EXAMINERS:

	Approved	**Disapproved**
1. Gregorio B. Baes, Ph.D. *(candidate)*	_____	_____
2. Avegail D. Carpio, M.Sc.	_____	_____
3. Richard Bryann L. Chua Ph.D. *(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 fulfillment 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

"A.CO: Android for Customer Organization, an Android Application for Customer Queue Management Based on J.Co's Business Processes" is an advanced queue management system employing mobile and QR code technology in providing a virtual queuing environment for the long lines of customers of J.Co Donuts and Coffee. The system uses Android as its platform, with the utilization of a QR code image processing library as a remote login security measure.

Keywords: queue management, QR code, Android application, mobile computing

Table of Contents

List of Figures

List of Tables

I. Introduction

A. Background of the Study

Everybody eats snacks in some occasion. And since J.Co Donuts and Coffee's establishment in 2005, doughnut as snacks has become all the more famous [1]. Its unexpected popularity, considering it was a simple local brand from Indonesia, has attracted more customers across Asian countries. It has presently lead into incredible queues of people spending 30 minutes or even two hours only to be able to taste a J.Co donut [2]. This queuing is an expected procedure when it comes to delivering a service to customers wherein the satisfaction for the service is inversely proportional to the amount of time spent in waiting.

All of us, according to one eminent analyst of the queue, American sociologist Dr. Barry Schwartz of University of Pennsylvania, 'throw away the small-change time—five minutes here, a quarter hour there—that accumulates in any ordinary day' [3]. Various estimates have been made of the time spent queuing. One much quoted statistic was produced during the 1980s by a consulting firm, Priority Management Pittsburgh, to the effect that average Americans spend five years of their lives waiting in lines and six months sitting at traffic lights (compared, incidentally, with four years doing housework and six years eating) [4]. Even if the time spent in queues has been more heavily curtailed in recent years by the prevalence of online retail transactions, the statistic retains a certain force [5].

The solution to this problem of queuing is actually obvious—in J.Co's case, they just have to provide more servers to attend to all the people. While it is easier to visualize this solution for the situation, it is not economically recommendable. It is due to the fact that J.Co, like most service or product deliverers, has limited resources and/or is within the constraints of having to keep a certain amount of profit to sustain the continuity of its whole production [6]. For instance, employing a lot of tills in their store will essentially give a faster service to all the people coming and going on weekends. However, this will entail that most of the time a big number of the tills will be idle during weekdays, on nonpeak hours, wherein the number of customers is at its lowest.

As a way to cope with the inevitable situation, different queuing systems have been developed and are being utilized. Among these, Single Server Systems and Multiserver Systems were recognized as the most basic ones [7]. These two systems are the foundation of all the other queuing systems that were eventually further refined. They honed and incorporated queue disciplines in its execution, among which First-In-First-Out (FIFO) discipline grew to be the most implemented one. This is due to the assumed customer fairness which is primarily based on the most effort exhibited through waiting for service [8].

These systems that were developed, manipulated, and/or adjusted, however efficient they may seem, still don't fully answer the problems of a waiting customer. These waiting customer problems include:

a. Balking of queue which occurs when an arriving customer feels discouraged to join a queue due to the long line and/or long wait for the service,
b. Reneging of queue which occurs when a customer who initially joined a queue does not withstand the long waiting time and/or the slow flow of service delivery,
c. Jockeying of queue which occurs when a customer leaves his/her original queue to join another in the hopes of getting served faster [9], and
d. Skipping the queue which occurs when a customer tries to dishonestly skip a queue position to go to another position in the hopes to get served at an earlier time.

The first two are more due to the customers' impatience brought about by the observation of the length of queue, lack of time, insufficient waiting space and/or improper care while waiting. The last two, however, are simply a matter of a customers' adaptation and way of surviving the test of waiting which then results to their "strategizing".

In a recent study conducted in the Harvard Business School, a solution to the customer impatience is proposed. The idea is to let the people, the recipient of the service waiting in a queue, to see what is taking the task so long. In their study, they found out that people find waiting more tolerable when they can see the work being done on their behalf. When these people see the progress, they tend to value the service more. This also holds true even when what's shown is merely the appearance of effort—termed as the "labor illusion"—which is basically a demonstration of the

progress, whether literal or not, expended to meet the customer's needs. In addition to this, it is concluded that many customers who endure waits but see a running tally of tasks end up happier than those who don't have to wait at all [10]. This task is easily remedied by a simple transparency of work progress or a projection of currently running tasks. Just like how other restaurants exhibit transparency in their accomplishing of tasks with the construction of a see-through kitchen where people can watch the chef's progress on the customers' orders. This essentially accomplishes the first alternative solution to the customer impatience. This lets the people see the strenuous task the chefs have to complete to be able to serve their customers the best food. This, in turn, indirectly answers the customers' inquiries regarding the time their food would take before it would be ready which will then make them feel more at ease while waiting.

Another solution to the aforementioned customer impatience is to let the people in queue have the liberty to indulge their selves into something else besides simply standing in line. In another article reported in Northeastern University, it was discussed that in a situation wherein waiting is unavoidable, majority of the people spend their time reading, chatting, working (on a computer, tablet, phone, etc.), and/or walking around the area [11]. This has already been achieved by some other restaurants that make use of paging systems. This lets the people roam around the area or sit comfortably, instead of anxiously standing in a slow moving line all the while fidgeting on the order's time-consuming completion. This makes waiting more tolerable

and more relaxed but less boring because these people were then made to be at ease in doing more worthwhile activities.

Conveniently, all these activities can be now done with the use of the features of a single smartphone in hand. This works with more versatility especially when connected to the internet [12] and/or when loaded with the correct applications. One promising innovation is the application that scans and interprets a QR code. This QR code, which is also known as “Quick Response” code, is an advanced form of a barcode that can store over 1800 characters of text information. These QR codes have grown in popularity for its convenience in transferring information to a cell phone like a website URL, contact information, a phone number, a geographic location, an SMS message to be sent to a phone number, a plain text message, and a calendar event [13]. This QR code can be integrated in queuing systems. As an example, it can be used to encode customer information, unique physical description, and time of arrival to log in into the queuing system and serve as a ticket to join the virtual queue. With over a thousand characters of text information fit to be stored in a single QR code, as well as the proposed system’s own security input integrated to the would be generated QR code, a customer’s order and position in the queue is assured upon claiming of order. Also, it will grant automatic and seamless entry to people remotely accessing the system through an internet connected mobile device with an installed QR code reading application. This is similar to what this project integrates into a queue management system.

B. Statement of the Problem

Despite the numerous queuing systems and strategies proposed, for some reason or another, J.Co Donuts and Coffee is yet to implement an efficient queuing system. One typical problematic scenario in their challenging queue processing is when their incoming customers become improperly directed, which is primarily due to their being huge in number and J.Co's lack of line directors. This then make the customers approach the store in an unorganized and unrelenting manner. This eventually evolves into an unsupervised assumption of a position in a traditional queue but without the actual assurance of getting their orders in a regulated manner. Another problem in this inefficient, customer-initiated "system" is the more competitive but less honest customers who try to "reserve a position in the queue" and the sneaky customers who skip the line. The former is done through sending one representative for at least two sets of orders which, in turn, makes the line grow within it rather than on its tail based on the chronological manner of who came in first; while the latter is done simply by going ahead without being concerned about others who were there first. These incidences trigger aggravation on the side of other customers whose position in the assumed line is compromised [14].

Conversely on J.Co's other branches, they already implement the traditional or unautomated queuing system equipped with a supervising line person (who also acts as the store security personnel). However, it is far from optimal in the way where it doesn't address the concerns of an impatiently waiting customer. These branches had

been inclined to believe that the satisfaction of their customers relies solely on their product. Little to no attention has been given into the investment of offering additional services like a staff dedicated solely in attentively updating the service progress to the customer, answering customer's inquiries, or simply attending to their other needs to appease them in their long wait. In reality, this flaw in their system can, at most, affect the customer's likeliness to return or seek their service or, at the very least, can simply affect the customer's satisfaction and opinion of the service. It is proven in a similar study, where it was found that 78% of customers have gone to a competitor's services due to poor performance at peak times from their original product deliverer; while 88% became less likely to return after a poor experience [15]. This reluctance of J.Co to invest in offering additional services can be attributed to the risk of costliness. This is witnessed in numerous scenarios where customers simply refuse to avail of their products due to unwillingness to be in a tedious line with no one to properly direct them and with no option to move around, to roam outside the queuing area, or even to simply go to the bathroom while waiting [16].

From the previously stated innovation and in relation to the advancement in technology, specifically in the mobile computing field, the integration to the ever-developing mobile applications of the two aforementioned alternative solutions for customer's impatience (the transparency of progress and the liberty to do something else) has now been viewed as a potential better solution to this queuing problem.

C. Objectives of the Study

This study developed a queue management system that can be easily integrated to J.Co's business processes which doesn't have an efficient automated queuing system yet. This application potentially jump-starts J.Co into having an effective virtual queue system where the system accepts a customer in an organized virtual queue, which eliminates the need of employing a line person. The primary key here is the utilization of the QR code that securely stores customer details which then automatically logs in the said customer into the queuing system and allows him/her to join in the virtual queue. This addresses the need of a customer to be entertained immediately and afterwards be able to freely do what he/she would rather do on his/her waiting time.

This QR code also stores information of a customer's would be waiting time. Therefore, while the said customer is out and about waiting for his/her order(s), a mobile display of a remaining time waiting provided by the information stored in the QR code can be viewed. A proactive interval reminding of the time remaining is also a part of the developed system. This then addresses the need of a customer to see the progress of his/her wait relative to the J.Co's production pace in completing its orders. In addition to this, the QR code produced does not need to be printed; only captured through a smartphone camera. This addresses the costly and wasteful distribution of ticketing materials, making A.CO more budget-friendly and secured all the while also addressing an impatient customer's need to "be cared for" while in queue. After the development of the primary functionalities, this study then provided a better queue

update projected to a Heads Up screen display of queue movement which then addresses the problem of impatient customers through progress transparency. The ticketing scheme is also a digital view-only secured process with a unique identifier input that doesn't need any printed material. This again supports the non-use of costly disposable resources.

On the part of J.Co, this study gives the potential to provide a more organized environment for the staff through a singular Android tablet-manipulated system that monitors the traffic of incoming customers as well as the load of their requests even at peak times. By giving them the power to organize customer queues and take care of them through the use of progress transparency and updates, long term profitably can now be considered as a more probable result. This holds true for it is when a customer is satisfied and is retained that other prospective customers are attracted to the organization. Customers' satisfaction essentially leads to customer loyalty which according to recent studies is 'crucial to long term profitability. Loyal customers spend more and refer new clients to the service deliverer' [17].

Summing it up, the general objective of this study is to help J.Co organize orders and/or requests of people without having the need to manually manage long lines of differently-tempered individuals. The application developed and tested in this proposal also adds flexibility to the customer's way of spending their waiting times while waiting for the completion of orders.

Specific Objectives:

1. Create an Android for Customer Organization (A.CO) system to be managed by a Counter Staff through an Android tablet device in his/her area that has the following features:
    a. Allows the Counter Staff to arrange all Customers in a virtual queue where their order of arrival will be monitored and organized.
    b. Allows the Counter Staff to manage Customer orders and the said order's completion time estimation in the tablet's Customer Orders Page.
    c. Allows the machine, on behalf of the Counter Staff, to notify the Customers with the application of their orders' completion.
    d. Allows the Counter Staff to update the Customer Queue Progress display.
    e. Allows the Counter Staff to see the items on the Menu List Display.
2. Create an Interactive Page in a tablet near the entrance of the establishment to be utilized by all types of Customers that has the following features:
    a. Allows the Customer who has no Android smartphone nor has the application installed to input their information and join the virtual queue.
    b. Allows the Customer who has no Android smartphone nor has the application installed to log in into the system.
    c. Allows the Customer who has no Android smartphone nor has the application installed to browse the "Menu List" of the day, select the foods he/she would like to avail of, and input his/her other order details.
    d. Allows the Customer who has no Android smartphone nor has the application installed to have a queue number.

e. Allows the Customer who has the application installed to have a QR code to be captured by the said Customer's phone camera.

3. Create a mobile application running on Android platform to be utilized by Customers with Android smartphones that has the following features:
   a. Allows the Customer to capture a QR code and join the virtual queue.
   b. Allows the Customer to log in into the queuing system remotely.
   c. Allows the Customer to browse the "Menu List" of the day, select the foods he/she would like to avail of, and input his/her other order details.
   d. Allows the Customer to remotely view the queue progress and his/her position in the queue.
   e. Allows the Customer to see his/her remaining waiting time which updates synchronously as the system time.
   f. Allows the Customer to receive a notification 10 minutes, 5 minutes and 1 minute before his/her order is completed.
4. Create a Check Out page utilized by the Cashier that has the following feature:
   a. Allows the Cashier to issue receipts for the exchange of the ordered item and the payment between the establishment and the Customer.
5. Create a non-interactive page which will be reflected in a Heads Up Display in the waiting area that has the following feature:
   a. Allows the Customers to view the queue progress.

D. Significance of the Project

The Android for Customer Organization's installation and implementation reduces the manual labor of managing the bulk of different customers, who have different patience levels, in a queue. Replacing the line personnel with this technology is essentially more budget-friendly (compared to costly systems used by banks). With the utilization of a virtual queue on an honest and organized first-come-first serve basis, jockeying of queues and skipping the lines can now be avoided. Also, with less physical presence of people in the waiting area, an illusion of a short queue will be created thus improving the problem of customer balking and/or reneging of queue.

Customers can now also enjoy the convenience of moving around, roaming outside the waiting area still with the assurance of having a secured position in the virtual queue after logging into the system through the QR code in their mobile devices (for customers with the application) or through the tablet device on the entrance of the area (for customers without the application). Furthermore, customers who are more concerned on the transparency can now enjoy the real-time update of remaining waiting time. Customers who are more concerned on being notified by the progress of the queue processing can now look into the actual numbers already being served by the server. Both of which are covered by the functionalities offered by the Heads Up Display in the waiting area or by the customer's mobile device.

E. Scope and Limitations

1. The application is designed to run on specific Android versions namely OS versions 2.3 (Gingerbread) and higher. This is based on the data collected from the new Google Play Store app during the 7-day period ending on March 3, 2014 which summarizes that 98.8% of Android devices are running on version 2.3.3 and higher [18].
2. The application does not take into consideration a customer's other schedule for the day or time of waiting unless it is specified in the order form he/she will send through the application. Any changes of time of pick-up or time of availability of the customer is the sole responsibility of the customer to inform the Counter Staff. Consequently, any misunderstandings or clearings up of details are the responsibility of a knowledgeable Counter Staff.
3. The application is synchronized via the web server but requires a consistent internet connection. Synchronous data in the system is equivalent to the real-time waiting time update by the Counter Staff. On the event that there will be a bad connection on the part of the Counter Staff, a local backup of the system can be accessed and manual operation (working with the last data update but without the benefits of continuous real-time update or the capability to accept new and incoming customers remotely) will commence. If the bad connection is experienced on the customer side, manual inputting of the order into the facility's tablet near the entrance will have to be done (by those who have not used their QR codes nor submitted their orders yet) or a

simple reliance to the last time update will have to be watchfully considered (by those who have already submitted their orders and are just roaming around while waiting for the notification).

4. The application is designed primarily to accommodate Android users. Non-Android app wielding customers may opt for the service of manual inputting of details and orders in the tablet near the entrance of the facility.
5. The application is designed to give a time estimate of an order's completion based on J.Co's nature of having its orders completed in a roughly fixed time for most cases.
6. The application is designed primarily for the customers in the queue waiting for their order's completion. It will also potentially reduce the probability of having a disorganized customers in the queue (if there's any) waiting for their turn to input their orders as well as the queue (if there's any) waiting for their turn to pay; but this is will be limited by the number of additional tablets to be made available for usage on the said queues.
7. The application is designed primarily for J.Co's business processes that employs a First-Come-First-Serve policy but can also be used on other establishments who have long lines like banks, government offices, healthcare clinics, etc. who wants to manage their customers in a more organized manner. It can be easily customized but this entails that their whole workflow is to be conveyed beforehand to the system creator.

F. Assumptions

1. The mobile device has a camera to capture the QR Code.
2. The Customer who has the application installed realizes that several attempts may be needed to capture the QR code if tried capturing by unsteady hands even though QR codes were developed to withstand the wear and tear of a factory and can still be used even if they are dirtied or damaged (because it employs Reed Solomon code which is a type of mathematical error correction code that can handle up to 30% of correction level) [19]. It is advantageous if the mobile device owner has a phone camera with fast shutter speed which reduces blurs on pictures.
3. The Counter Staff is knowledgeable of the average processing time the processor(s) will take to complete the order(s). This is crucial in inputting a more accurate estimation of waiting time. The system suggests an approximate time but it is still up to the Counter Staff to manipulate/override the system suggestions.
4. The orders are already prepared and are ready to be given by the time customers come in the establishment to order it. If not, it will be easily made on the spot.
5. All Customers will input orders which are understandable, specifically on the additional remarks/preferences form section.
6. The Customer who has the application installed has the liberty to be in the queue even without having sent his/her order(s) yet. However, this applies

only until before his/her order's turn to be processed comes or else his/her priority number will be reduced to give way to other customers who have their orders already sent to the Counter Staff.

7. The Customers will respond to their order alerts on time or else they will have their queue priorities reduced or their order(s) served at a later time.
8. The Customer who has the application installed has a consistent internet connection for time waiting is consistently updated for the whole duration of the wait. Customers who don't have a consistent internet connection can simply rely on the last update he/she has received before his/her connection has been cut off. On the other hand, Customers who don't have an Android smartphone with the application installed can simply rely on the last update he/she has seen on the Android tablet near the entrance upon his/her completion of his/her order input. The change in the waiting time estimation can be assumed to be within a considerable range.

II. Review of Related Literature

Every domain, like Financial, Health Care, Public and Retail Sectors, has a method of handling queues. This method is called Queue Management. However, a need for more sophisticated and more advanced systems for managing queues has been stressed in more situations than one. These situations require Queue Management Systems (QMS). Queue Management Systems can be reactive or proactive. Reactive systems simply organize the existing queue while Proactive systems consist of a queue management statistics gathering system so that trends can be identified and anticipated. These Queue Management Systems work by streamlining front-end operations into centralized contact points which enable managers to monitor and set performance thresholds [20].

In a similar study in Auburn University, they integrated a small scale Queue Management System, called QueueAdmin in a barber shop. They provided centralized contact points where a customer can add himself/herself to the queue, where a customer is to be called by the barber to the barber chair, and where the customer is finished with his or her haircut. In this study, they were able to prove that through the use of automated queuing system, waiting list of customers became more accurately managed. Also, the system provided less confusion on who is next to being served in line. Topping it off, the study concluded that the system increased the overall productivity and quality of the barbershop service [20].

Another example of the utilization of a queue management system is the Onlinet. Onlinet is equipped with the advanced combinations of queue management system and customer control flow making the use of a ticketing system. It aims to improve the customer waiting experience, efficiency of workforce, and sales. It also boasts its own digital signage solution with information and self-service kiosks that can reach out to customers more, improve existing services, and produce more income [21]. These features can also be observed on BPI's BEA machine [22].

The proposed and developed Android for Customer Organization (A.CO), on the other hand, addresses the issues of customer behavior by utilizing a queue management system which is more organized than conventional ones, and which at the same time gives customers the freedom to spend their time more wisely with other options to be provided to them. It integrates the following key concepts and approaches observed on other Queuing Systems and Customer Service Deliverers:

A. Real-Time Update of Remaining Time Waiting

The study conducted by Lu, Musalem, Olivares, and Schilkurt came up with an approach which empirically determines the most important factors in a queuing system that influence customer behavior. They found out that customers appear to focus primarily on the length of the line when deciding to join a queue; whereas the number of servers attending the queue, which determines the speed at which the queue advances, has a much smaller impact on customers' decisions. This finding has

important implications for the design of a queuing system. Some precautions should be taken in moving towards the traditional queue system and it may be critical to provide information about the expected waiting time so that customers are not drawn away by longer queues [23].

An application called QMinder is a free smartphone application for remote queuing that makes use of the aforementioned concept of providing information about the expected waiting time. This application takes one's place in line and notifies the person using it if his/her turn is approaching. This application offers precise information, reminding of turn with the option for the user to roam around freely [24]. However, it has received low content maturity based on the users' feedback. It also has no definite association to the office/organization it was supposed to establish an alternate queue for.

B. QR Code Technology Integration

The QR code in the application developed will be used on securely encoding Customer Information and automatic login details to join the virtual queue [25]. This QR code does not have to be printed, only be captured by the customer to serve as a link to the queuing system. A QR code is a great innovation to bridge the gap between the tactile and the virtual world [26]. This supports one of the goals of the Special Problem which is utilizing innovative technology in encoding information.

C. Queue Positioning Based on Time Arrival

The concept of time adjustment that is in the application developed was done through the use of manipulation on the part of the Counter Staff. This concept has already been demonstrated by other applications like Queuelo, and by Marco Venier's thesis entitled "*An Android Application for Queues Management"* only with a slightly different approach.

Queuelo is a Virtual Queue Management app that allows customers to organize and control their appointments in real-time, taking the burden of system arrangements from business owners while letting them focus on what they are good at. By using its social capabilities, Queuelo puts virtual queue members in touch: allowing them to communicate, exchange places in queue, leave reviews for the service experience and even propose deals to the service deliverer [27].

On the other hand, Marco Venier, under the supervision of Prof. Bernhard Rinner, DI Bernhard Dieber (both from Alpen-Adria University of Klagenfurt), and Prof. Gian Luca Foresti (from University of Udine), wrote a master's thesis on "*An Android Application for Queues Management"*. In his thesis, the application utilizes a queue number adjustment capability allowing the customer to roam around first while fully knowing that his/her turn is not yet approaching. It also integrates the use of QR codes on tickets in joining queues as well as a map navigator [28].

D. Ease of Use through Android Smartphones

Laptop and PC use (but mainly PC use) has been steadily declining with the release of smartphones whose usage has only increased exponentially over time. Almost everything being done in a computing machine is now also being done on a smartphone. Contrary to what the situation was some ten years ago, owning a smartphone now is no longer a luxury but already a necessity [29]. Almost everybody from all age demographics own at least one mobile phone on which a huge percentage of these population owns at least one smartphone as well [30]. In America alone, an early 2010 report said that over 91% of Americans have a cell phone [31]. On the other hand, Nielsen reports that 21% of Americans have smartphones [32]. To top it off, by the third quarter of 2011, that number was projected to increase to 50% as shown in Figure 1. This puts the number of smartphone users in the U.S. at around 142.8 million [31]. From this, it can be concluded that smartphones are indeed the future. The vast majority will continue to grow and continue to utilize the versatile and easy to carry gadget for all the tasks and activities they are to do.

Furthermore, the Android OS is officially the #1 smartphone operating system in the US, and possibly for good reason: Android OS is owned by Google, and is an open-sourced freeware under the GPL license. It's under the nonexclusive rights called copyleft (as opposed to copyright) which means the product is free and anyone can edit it—so long as their modified version is *also* free [33]. This only implies that if

one makes use of an Android smartphone and vigorously fiddles with it, one can edit the operating system to create various applications at one's own perusal.

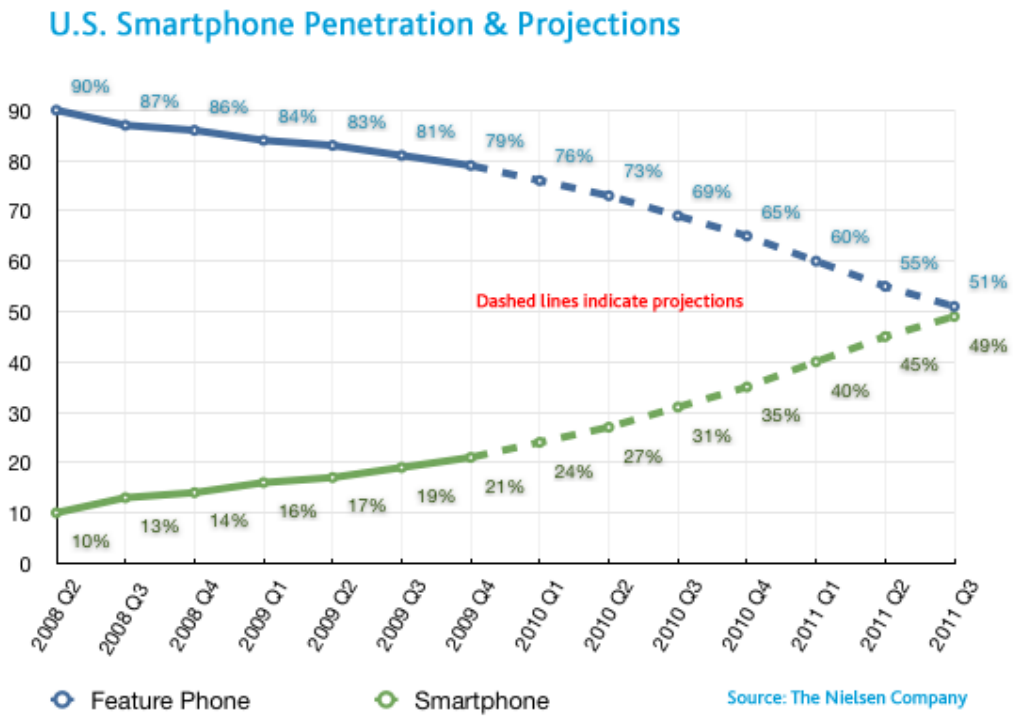

Figure 1: Smartphone Adoption in the US

The significance of the Android smartphone in this application created is its ease of modification and development, ease of use and carrying, ease of time monitoring and management. A.CO for J.Co involves the Android capabilities in a smartphone device that makes tracking of queue waiting time, registration to the system, and remotely done food ordering possible even the customer or the user is outside the queue premises.

III. Theoretical Framework

The Android for Customer Organization (A.CO) is mobile in nature and runs on an Android phone and/or tablet. It includes the following attributes:

A. Types of User

The users of A.CO have access to different functionalities within the whole system. Users include the Counter Staff, the Cashier, and the Customers. The Counter Staff and the Cashier have a tablet while the customers have a choice between owning a mobile phone which has an Android operating system (version 2.3 and above) and has the A.CO installed, or owning any mobile device that can capture the image of the queue number to be assigned to them. This application is recommended for customers who prefer to manage and spend their times doing something else or going somewhere else while still having a secured position in a queue.

B. Menu List

The Menu List is a unique formulation of A.CO. It is an extensive everyday list of services or products offered by J.Co. This list provides flexibility to the Counter Staff to manage items on the list to show what is to be offered on certain days. For example, on weekends, this list can feature weekend specialties for families like bigger servings of meals, or perhaps added dessert choices for the kids; while on weekdays, this list can

feature more quick and to-go specialties like coffees and sandwiches for rushing office people. Another example, on special seasons, this list can feature additional food items like donuts shaped like bouquets of flowers, hearts, and/or teddy bears during Valentine's season; donuts shaped like ghosts, pumpkins and witch hats during Halloween season; or donuts shaped like candy canes, Christmas trees, and/or Christmas balls during the Christmas season. This list includes pictures and/or short descriptions of services or products offered.

C. QR Code

QR codes are utilized by the system that allows customers who have the application to securely maintain their positions in the queue even if they took their liberties to roam around outside the premises of the J.Co waiting area. The generated QR code comes from an open-source, multi-format 1D/2D barcode image processing library implemented in Java [34]. It integrates the technology to read and interpret a captured image of a QR code to the A.CO application. This QR code contains a secure and unique identifier which comes from both the customer's input and the system's auto-generated combination. This then redirects the customer to the order inputting page of the system. Once finished with the inputting of the order(s) and filling in of the customer's personal information as well as the mandatory input of the said customer's distinctive physical description (for identification upon claiming of order), this code serves as the automatic entry to the queue.

D. Securing the Queue Position Even After Leaving for a While

Android for Customer Organization accommodates all types of customers. Therefore, the system can handle customers who are in a hurry, customers who have other appointments and are in doubt of their time of return, customers who have a prior knowledge of their specific time of arrival, customers who simply wanted to stay in their places and wait in the area provided for them, and customers who are not able to respond to the notifications sent by the Counter Staff, and/or have decided not to return.

Customers who are in a hurry can make the most of their times through installing the application. They can simply capture the QR code generated for their position in the virtual queue and they will be allowed to simply do the inputting of orders while they go and proceed on doing their other tasks. They can simply pick their orders after the specified waiting time is up or they can just inform the Counter Staff that their schedule is flexible and include the details of their circumstance in the order form. On special cases, like emergency situations for urgent matters but quick processes, the Counter Staff has the capability to adjust this rushing customer's priority and serve him or her sooner, manually overriding the system's ordering. This, however, shall be moderated so as not to be abused and be a cause of the instability of the proposed customer First-Come-First-Served fairness.

Customers who have other appointments and are in doubt of their time of return can include in their order forms their earliest estimation of time for their return. On the event that they become able to identify the chances of not being able to return soon, they are given the option to simply take a lower priority than their original queue number. This makes way for other customers to be processed by the system. At the same time, this still gives the assurance that once the aforementioned customers return, they will still be considered again and will not need to join the tail of the queue for a second time.

Customers who have a prior knowledge of their specific time of arrival, especially if the said time is later than the supposed waiting time that comes with the queue number, can simply be repositioned on another queue number which will be served at a later time. This is to match the real waiting time of the customer and the ideal waiting time that he/she has included in his/her order form. This approach caters for a customer's need to make an appointment to other offices and have a certain time slot reserved for him/her before he/she may come in.

Customers who simply wanted to stay in their places and wait in the area provided for them will enjoy the connectivity he/she will achieve through the queue progress update projected by the Heads Up Display. This will provide the customer an insight on how J.Co operates, how they accommodate customers, or simply how they function as a whole.

Customers who are not able to respond to the notifications sent by the Counter Staff, and/or have decided not to return will automatically have their priority numbers reduced. The same also applies to customers who are not able input their orders before the suggested time for them is over. Their priority numbers will continually be reduced until they reach the tail of the queue, until they respond to the notification(s) sent by the Counter Staff, or until J.Co closes.

E. Queuing System Analysis

Queuing systems are characterized by three components: arrival process, service mechanism, and queue discipline. While the arrival process of the customers can not be controlled and while the service mechanism of a service or product deliverer can be very limited to the point of no more further adjustment disparaging to the budget constraint can be implemented, the queue discipline can somehow be analyzed and perhaps be improved. The queue discipline of a queuing system is the rule that a service or product deliverer uses to choose the next customer (if any) when the server completes the process for the current customer. The commonly used queue disciplines are:

a. First-in-first-out customer service (FIFO) which serves first the longest waiting customer in queue;
b. Last-in-first-out customer service (LIFO) which serves first the most recent customer in queue; and

c. Priority service which serves customers in order of their importance on the basis of their service requirements [8].

Among the three mentioned disciplines, FIFO is the most implemented one because of the assumed customer fairness which is basically based on the most effort exhibited through waiting. From this discipline, further refinement of the queuing system has been developed to cope with the bulk of customer arrivals as well as the limits of a service delivery. This then resulted to the two most basic customer queuing systems which are:

a. The Basic Single Server System wherein there is only one server and one queue that processes customer orders/requests. This can further be expanded through operating numerous single server single queues.
b. The Multiple-Server Exponential Time Model which utilizes multiple servers within the single queue wherein the next customer in line may proceed to the next available server without having the need to wait for the customer before him/her to finish first [7].

To make decisions about what type of resources are needed to provide a service, in this case which appropriate customer queuing system is what's being asked, the results of the queuing theory applied on the queuing systems should be analyzed. This queuing theory is the mathematical study of waiting lines or queues wherein a model is constructed so the queue length and waiting times can be predicted [35].

This exact prediction tool, an analytic model of queues based on queuing theory, is to be expressed in a set of equations. These equations can be solved to yield the desired parameters (like the length of time the system takes to finish a process, the amount of customers served in a whole duration of the system activation, etc). However, a disadvantageous part of this queuing theory exists, that is, a number of simplifying assumptions on system requirements must be made to derive equations for the parameters of interest [36]. But even so, these queuing models still provide the analyst or designer of the queuing system with a powerful tool of advantage in evaluating the performance of the designed queuing system [37].

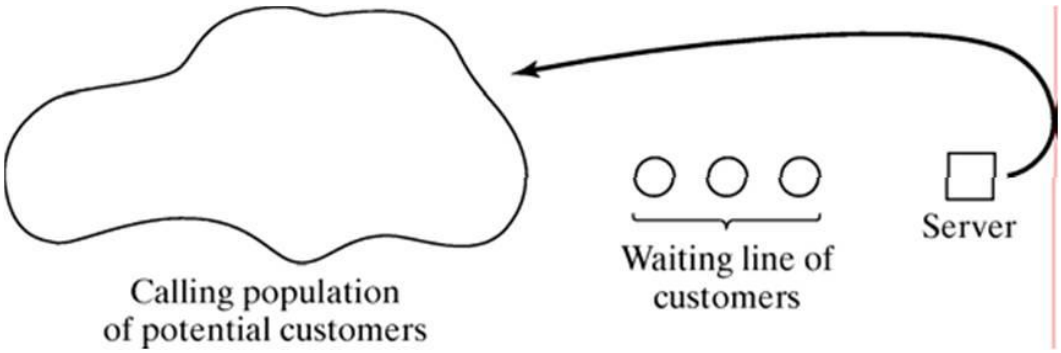

Figure 2: A simple but typical Queuing Model

The following are more detailed descriptions of the queuing systems models that can make use of the proposed Android for Customer Organization:

1. Singe Server Queue

This is the simplest queuing system. The central element of this system is the *Server*. A *Server* provides service to *Items*, which, on this paper, are the

customers in a queue. These *Items,* from some *Population of Items* outside the premises of the system, arrive at the system to be served. If the *Server* is idle, an *Item* is served immediately. Otherwise, an arriving *Item* joins a waiting line or *Queue*. When the *Server* has completed serving an *Item*, that *Item* departs. If there are *Items* still waiting in the queue, one is immediately dispatched to the *Server*. The *Server* in this model can represent anything that performs some function or service for a collection of *Items* [36].

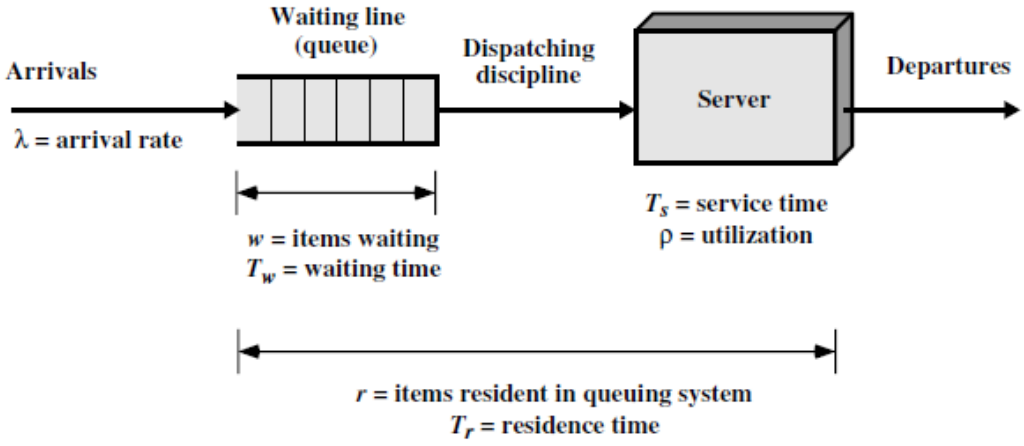

Figure 3: Queuing System Structure and Parameters for Single-Server Queue

2. Multiserver Queue

Figure 4 shows a generalization of the simple model we have been discussing for multiple servers, all sharing a common queue. If an *Item* arrives and at least one *Server* is available, then the said *Item* is immediately dispatched to that *Server*. It is assumed that all *Servers* are identical in the manner they operate or which type of services they can provide; thus, if more than one *Server*

is available, it makes no difference which *Server* is chosen for the *Item*. If all *Servers* are busy, then a *Queue* will begin to form. As soon as one *Server* becomes free, an *Item* is dispatched from the *Queue* using the queue discipline in force (FIFO, LIFO, Priority, etc.). With the exception of utilization ρ, all of the parameters illustrated in Figure 3 carry over to the Multiserver Queuing System with the same interpretation [36].

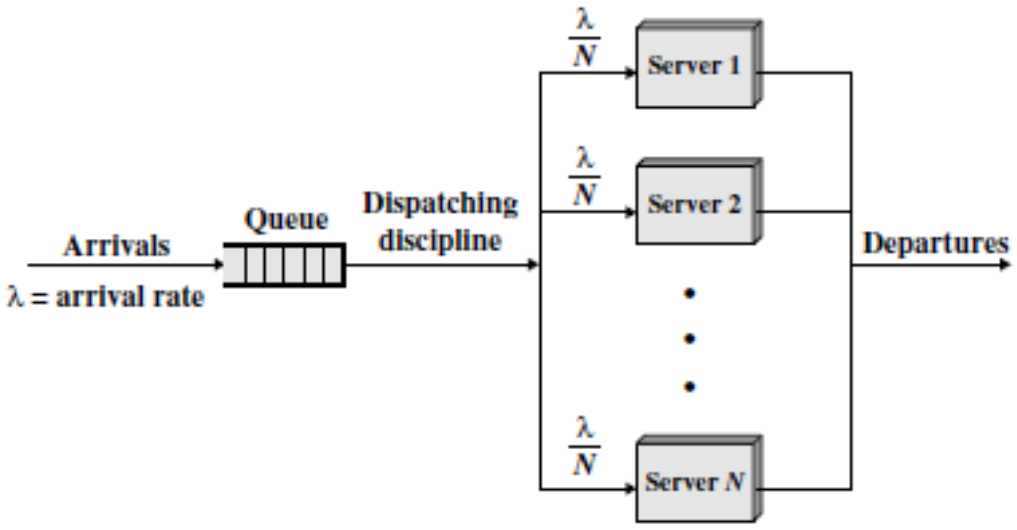

Figure 4: Queuing System Structure and Parameters for Multiserver Queue

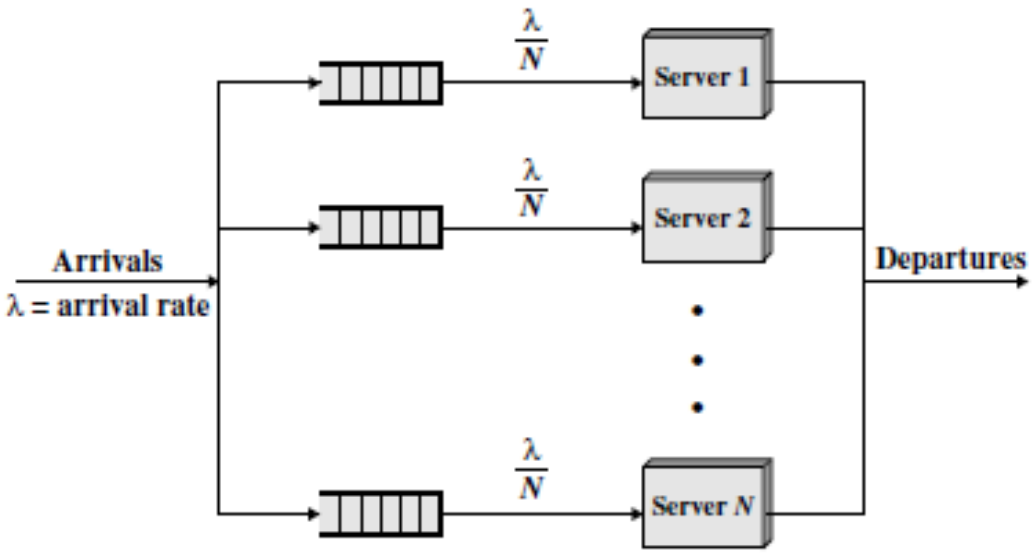

Figure 5: Queuing System Structure and Parameters for Multiples Single Server Queue

To summarize, from this Queuing System Analysis integrated into the proposed Android for Customer Organization (A.CO), it can be figured out what the maximum input rate that can be handled by the system is. Hereafter, appropriate control can be applied to the whole system to prevent it from having an overload of customers to attend. This significantly reduces the disappointments of customers who made an effort of approaching J.Co, be told that he/she will be served, only to be told again to simply come back next time after overestimating the capacity of the system.

λ	=	arrival rate; mean number of arrivals per second
T_s	=	mean service time for each arrival; amount of time being served, not counting time waiting in the queue
σ_{Ts}	=	standard deviation of service time
ρ	=	utilization; fraction of time facility (server or servers) is busy
u	=	traffic intensity
r	=	mean number of items in system, waiting and being served (residence time)
R	=	number of items in system, waiting and being served
T_r	=	mean time an item spends in system (residence time)
T_R	=	time an item spends in system (residence time)
σ_r	=	standard deviation of r
σ_{Tr}	=	standard deviation of T_r
w	=	mean number of items waiting to be served
σ_w	=	standard deviation of w
T_w	=	mean waiting time (including items that have to wait and items with waiting time = 0)
T_d	=	mean waiting time for items that have to wait
N	=	number of servers
$m_X(y)$	=	the yth percentile; that value of y below which x occurs y percent of the time

Table 1: Parameters for Queuing Systems

Table 1 summarizes the notation of parameters used in Figure 3, 4 and 5. It also introduces some other parameters that are useful.

Assumptions:

1. Poisson arrival rate.
2. Dispatching discipline does not give preference to items based on service times.
3. Formulas for standard deviation assume first-in, first-out dispatching.
4. No items are discarded from the queue.

(a) General Service Times (M/G/1)

$$A = \frac{1}{2}\left[1 + \left(\frac{\sigma_{T_s}}{T_s}\right)^2\right]$$

$$r = \rho + \frac{\rho^2 A}{1-\rho}$$

$$w = \frac{\rho^2 A}{1-\rho}$$

$$T_r = T_s + \frac{\rho T_s A}{1-\rho}$$

$$T_w = \frac{\rho T_s A}{1-\rho}$$

(b) Exponential Service Times (M/M/1)

$$r = \frac{\rho}{1-\rho} \qquad w = \frac{\rho^2}{1-\rho}$$

$$T_r = \frac{T_s}{1-\rho} \qquad T_w = \frac{\rho T_s}{1-\rho}$$

$$\sigma_r = \frac{\sqrt{\rho}}{1-\rho} \qquad \sigma_{T_r} = \frac{T_s}{1-\rho}$$

$$\Pr[R = N] = (1-\rho)\rho^N$$

$$\Pr[R \le N] = \sum_{i=0}^{N} (1-\rho)\rho^i$$

$$\Pr[T_R \le T] = 1 - e^{-(1-\rho)\frac{1}{T_s}}$$

$$m_{T_r}(y) = T_r \times \ln\left(\frac{100}{100-y}\right)$$

$$m_{T_w}(y) = \frac{T_w}{\rho} \times \ln\left(\frac{100\rho}{100-y}\right)$$

(c) Constant Service Times (M/D/1)

$$r = \frac{\rho^2}{2(1-\rho)} + \rho$$

$$w = \frac{\rho^2}{2(1-\rho)}$$

$$T_r = \frac{T_s(2-\rho)}{2(1-\rho)}$$

$$T_w = \frac{\rho T_s}{2(1-\rho)}$$

$$\sigma_r = \frac{1}{1-\rho}\sqrt{\rho - \frac{3\rho^2}{2} + \frac{5\rho^3}{6} - \frac{\rho^4}{12}}$$

$$\sigma_{T_r} = \frac{T_s}{1-\rho}\sqrt{\frac{\rho}{3} - \frac{\rho^2}{12}}$$

Table 2: Formulas for Single Server Queues

Table 2a provides some equations for single server queues that follow the M/G/1 model. That is, the arrival rate is Poisson (M), the service time is general (G), and the number of queues is one (1). The key factor in the scaling parameter, A, is the ratio of the standard deviation of service time to the mean. When the standard deviation is equal to the mean, the service time distribution changes into exponential (M/M/1). This is the simplest case and the easiest one for calculating results. Table 2b shows the simplified versions of equations for the standard deviation of r and T_r, plus some other parameters of interest. The other case is a standard deviation of service time equal to zero, that is, a constant service time (M/D/1). The corresponding equations are shown in Table 2c.

Assumptions:

1. Poisson arrival rate.
2. Exponential service times
3. All servers equally loaded
4. All servers have same mean service time
5. First-in, first-out dispatching
6. No items are discarded from the queue

$$K = \frac{\sum_{I=0}^{N-1} \frac{(N\rho)^I}{I!}}{\sum_{I=0}^{N} \frac{(N\rho)^I}{I!}} \qquad \text{Poisson ratio function}$$

$$\text{Erlang-C function} = \text{Probability that all servers are busy} = C = \frac{1-K}{1-\rho K}$$

$$r = C\frac{\rho}{1-\rho} + N\rho \qquad w = C\frac{\rho}{1-\rho}$$

$$T_r = \frac{C}{N}\frac{T_s}{1-\rho} + T_s \qquad T_w = \frac{C}{N}\frac{T_s}{1-\rho}$$

$$\sigma_{T_r} = \frac{T_s}{N(1-\rho)}\sqrt{C(2-C) + N^2(1-\rho)^2}$$

$$\sigma_w = \frac{1}{1-\rho}\sqrt{C\rho(1+\rho-C\rho)}$$

$$\Pr[T_w > t] = Ce^{-N(1-\rho)t/T_s}$$

$$m_{T_w}(y) = \frac{T_s}{N(1-\rho)}\ln\left(\frac{100C}{100-y}\right)$$

$$T_d = \frac{T_s}{N(1-\rho)}$$

Table 3: Formulas for Multiserver Queues (M/M/N)

Table 3 lists the formulas for some key parameters for the multiserver queue. Useful congestion statistics for this model have been obtained only for the case of M/M/N, where the exponential service times are identical for the *N* servers.

IV. Design and Implementation

A. Diagrams

A. Context Diagram

This system creates two applications. One is for the Counter Staff, the Cashier, and the Customer without the application installed, where A.CO will be run on a tablet device; and the other is for the Customer with the application installed, where A.CO will be run on a mobile device. The context diagram is shown in Figure 6.

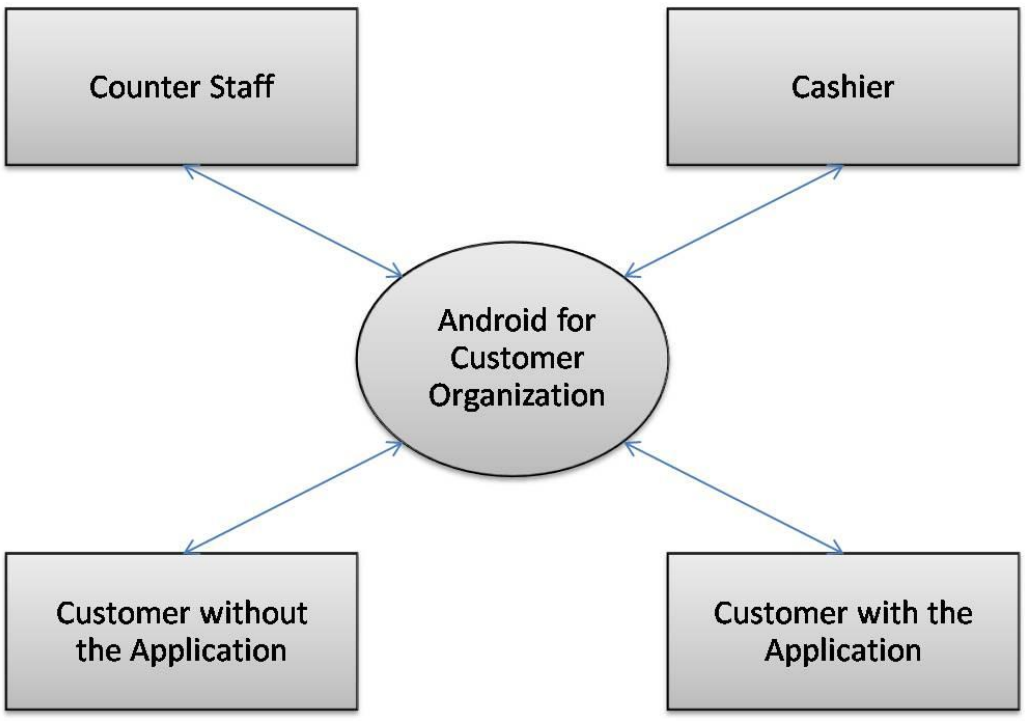

Figure 6: Level 0 Context Diagram of A.CO

B. Use-Case Diagram

The system features four types of users. Only the Customer without the Application has no need to install the application on his/her device to benefit from the system. Instead, he/she can just manually input his/her information into the system's tablet upon his/her arrival and join the virtual queue. He/she can then log in and input his/her order(s) and have a queue number. However, no notifications will be directly sent to him/her except for the display on the Heads Up Screen which is also a part of the system.

On the other hand, the Customer with the Application has the ability to capture a QR code to join the queue, which also serves as his/her login to the system's database. Upon logging in, he/she can proceed to writing his/her order(s). In addition to these functionalities, he/she can then check his/her position in the queue as well as have a real-time update of the remaining waiting time. On the event that his/her turn is already approaching, he/she will receive the notification sent by the Counter Staff.

As for the Counter Staff, he/she has the ability to accept customers in the virtual queue, manage the said customers' orders, and its completion time estimations. He/she can also notify the customers of their approaching completion of orders. In addition, he/she is also able to oversee the Queue Progress as well as the Menu List via the Android tablet.

And for the Cashier, he/she has the ability to release receipts and accept payment upon a customer's pick-up of orders. The use-case diagram is shown in Figure 7.

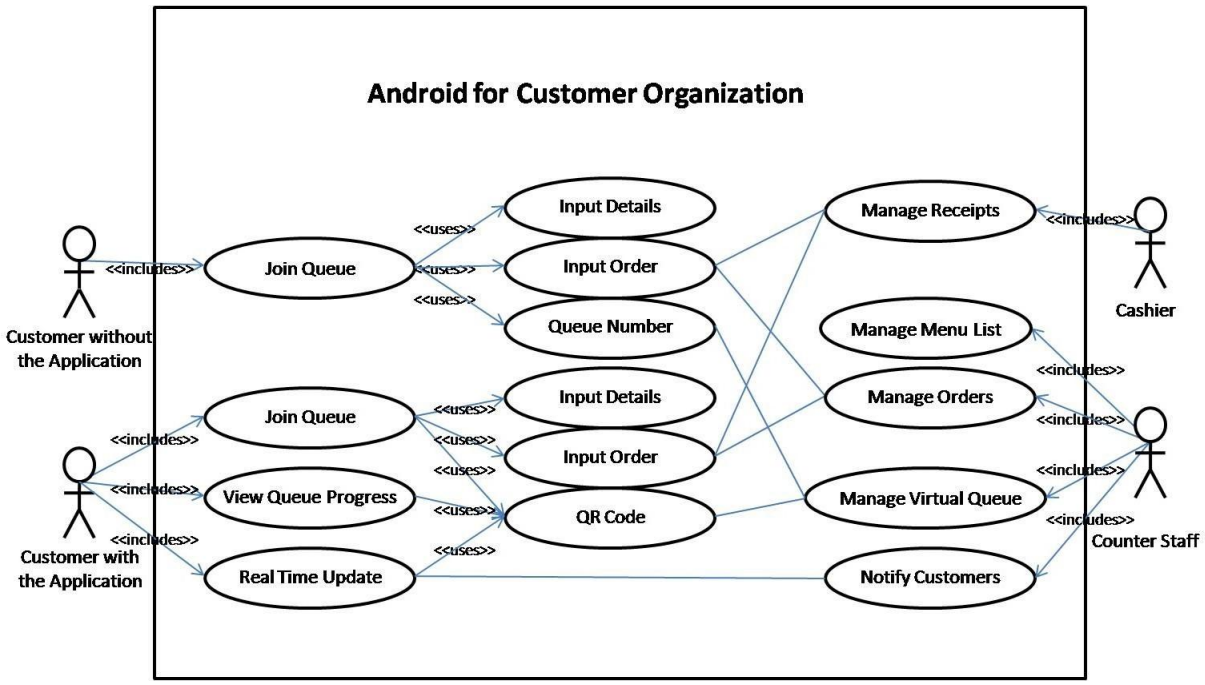

Figure 7: Use-Case Diagram of A.CO

C. Sequence Diagrams

The sequence diagram below in Figure 8 illustrates the sequence of processes of J.Co using the Android for Customer Organization designed for a Customer who has the application installed. It starts from the Customer's arrival in the area to request to join the queue in the area and ends with the Customer's picking up of his/her order(s).

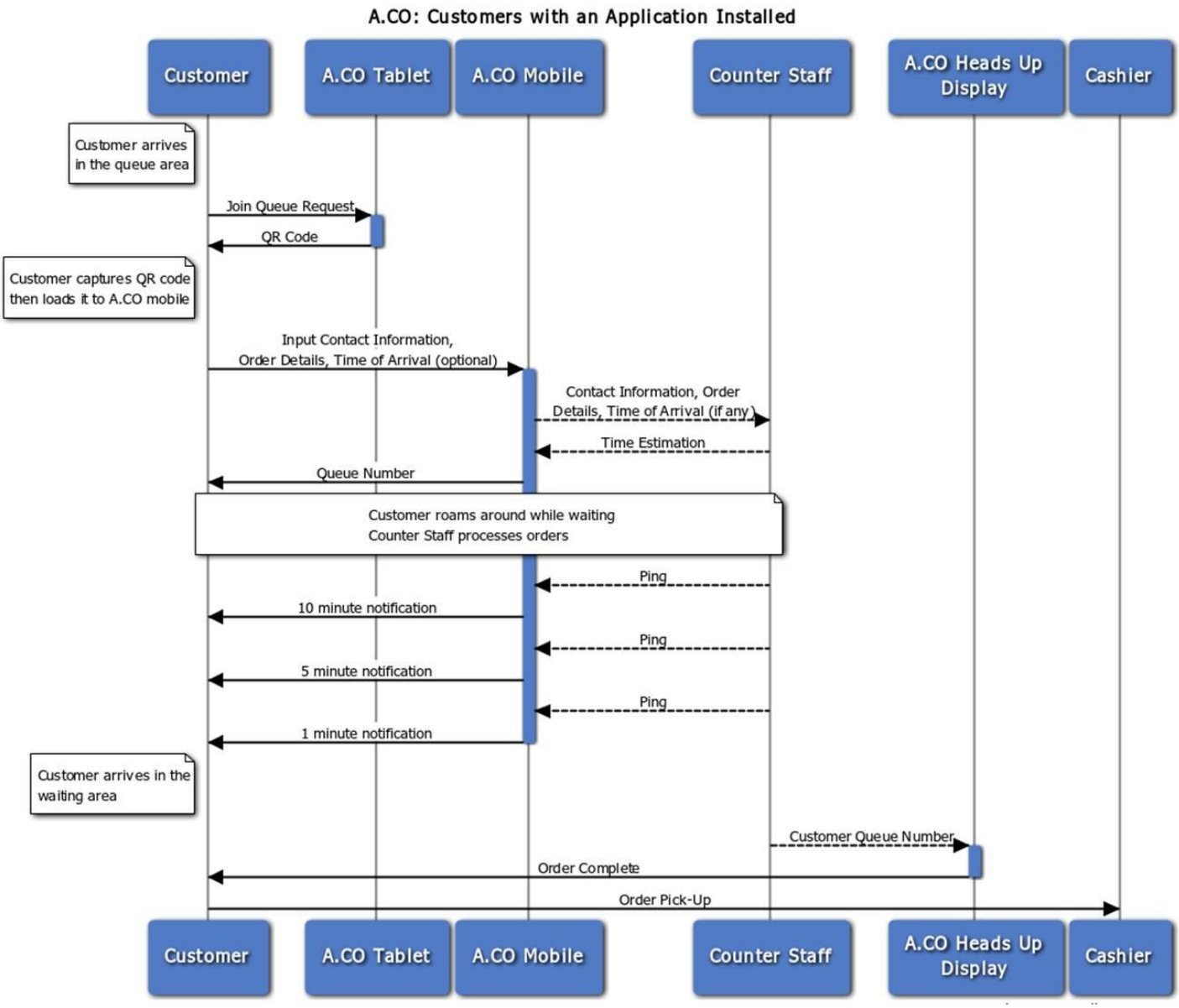

Figure 8: Sequence Diagram of A.CO for Customers with an Application Installed

The sequence diagram below in Figure 9, on the other hand, illustrates the sequence of processes J.Co using the Android for Customer Organization designed for a Customer who has not installed the application yet. It starts from the Customer's arrival in the area to join the queue and ends with the Customer's picking up of his/her order(s).

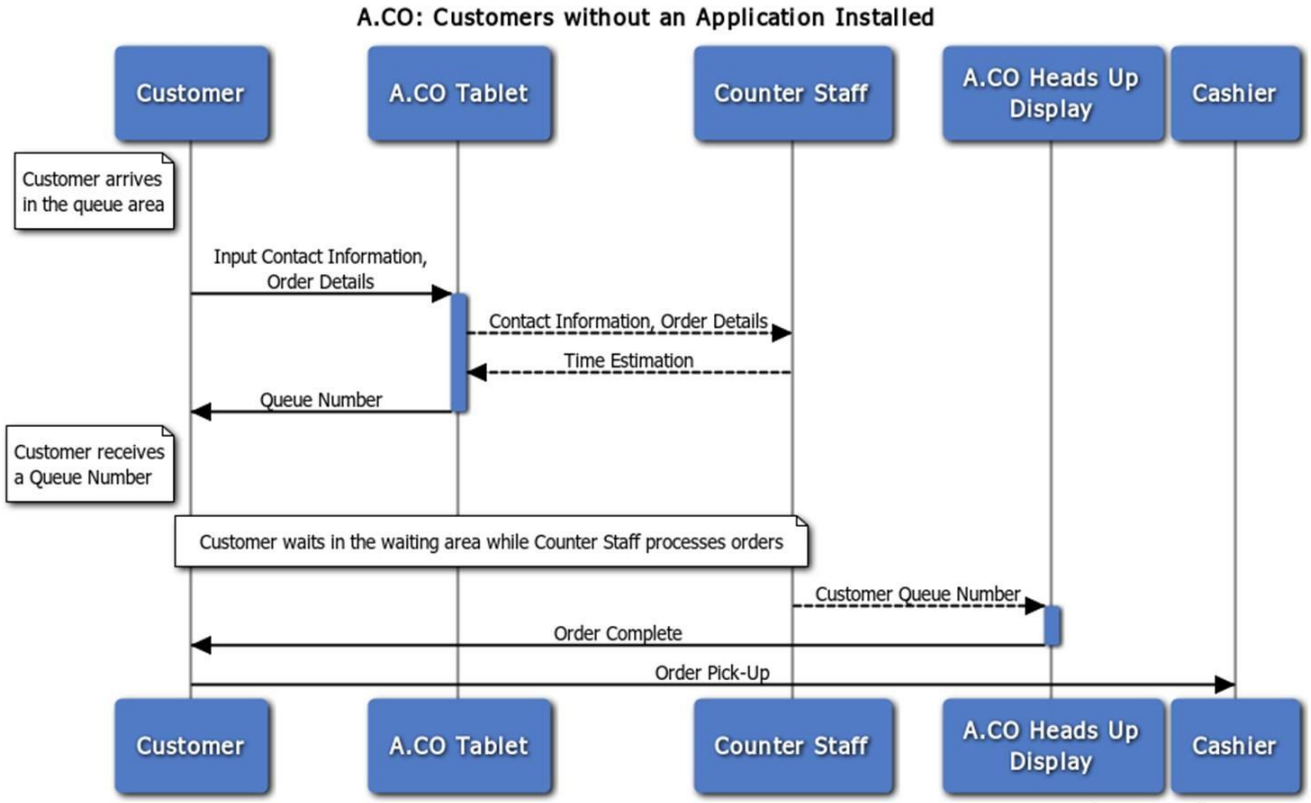

Figure 9: Sequence Diagram of A.CO for Customers without an Application Installed

D. Entity Relationship Diagram

A database is necessary for the implementation of multiple Customers. It provides a way for the Counter Staff to view each and everyone's orders as well as for the Cashier to view each and everyone's receipts. The entity relationship diagram of A.CO is shown in Figure 10 and 11.

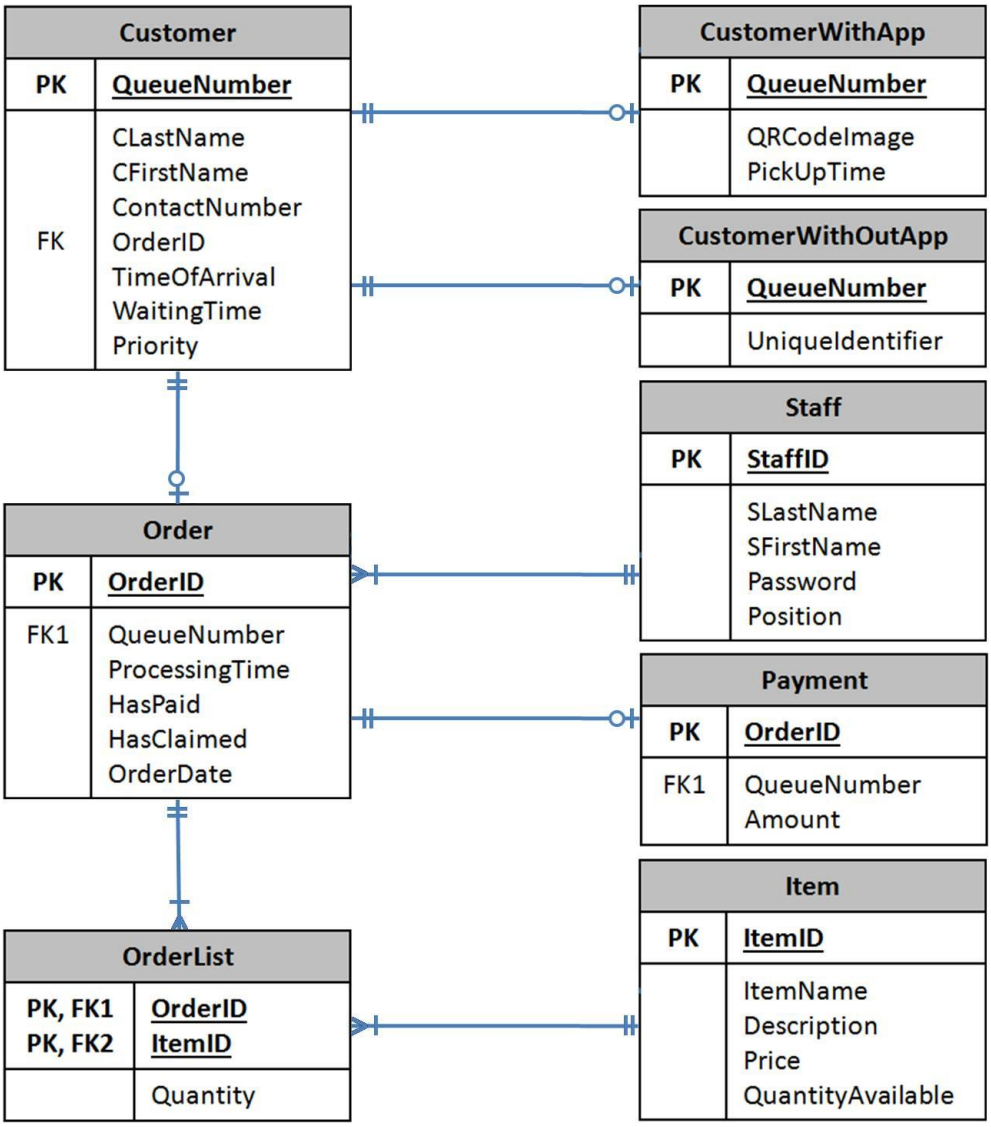

Figure 10: Entity Relationship Diagram of A.CO

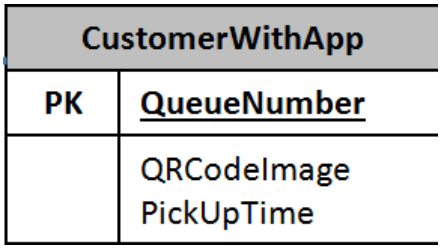

Figure 11: Entity Relationship Diagram of Customer's Local Android Device

E. Data Dictionary

Listed below are the tables in A.CO and their respective data fields.

Customer – contains the basic information of the Customer.

Field	Type	Description
QueueNumber	INTEGER	Queue number of the customer
CLastName	VARCHAR(50)	Last name of customer
CFirstName	VARCHAR(50)	First name of customer
ContactNumber	VARCHAR(15)	Contact number of customer
OrderID	INTEGER	Order number of the transaction
TimeOfArrival	TIMESTAMP	Time arrival of customer
WaitingTime	TIME	Waiting time before customer's turn
Priority	INTEGER	Priority number of customer

Table 4: Customer Table of A.CO

Order – contains the order details of the customer's transaction.

Field	Type	Description
OrderID	INTEGER	Order number of the transaction
QueueNumber	INTEGER	Queue number of the customer
ProcessingTime	TIME	Total processing time for all the orders
HasPaid	Set('Yes', 'No')	Indicates if the order has been paid
HasClaimed	Set('Yes', 'No')	Indicates if the order has been claimed
OrderDate	TIMESTAMP	Date of transaction

Table 5: Order Table of A.CO

OrderList – contains the quantity of items ordered in the transaction.

Field	Type	Description
OrderID	INTEGER	Order number of the transaction
ItemID	INTEGER	Item number of the food item
Quantity	INTEGER	Quantity of items

Table 6: OrderList Table of A.CO

Item – contains the list of items to be availed of by the customer

Field	Type	Description
ItemID	INTEGER	Item number of the food item
ItemName	VARCHAR(50)	Name of the food item
Description	VARCHAR(255)	Description of the food item
Price	INTEGER	Price of the food item
QuantityAvailable	INTEGER	Number of food items available

Table 7: Item Table of A.CO

Payment – contains the billing summary of the transaction.

Field	Type	Description
OrderID	INTEGER	Order number of the transaction
QueueNumber	INTEGER	Queue number of the customer
Amount	INTEGER	Total amount of the items ordered

Table 8: Payment Table of A.CO

Staff – contains the basic information of the staff.

Field	Type	Description
StaffID	INTEGER	ID number of the staff
SLastName	VARCHAR(50)	Last name of staff
SFirstName	VARCHAR(50)	First name of staff
Password	VARCHAR(20)	Log in password of the staff
Position	Set('Cashier', 'Counter')	Position of the staff

Table 9: Staff Table of A.CO

CustomerWithOutApp – contains the description of customer without the app.

Field	Type	Description
QueueNumber	INTEGER	Queue number of the customer
UniqueIdentifier	VARCHAR(150)	Physical description of the customer

Table 10: CustomerWithOutApp Table of A.CO

CustomerWithApp – contains the unique attributes of a customer with the app.

Field	Type	Description
QueueNumber	INTEGER	Queue number of the customer
QRCodeImage	BLOB	QR code image given to the customer
PickUpTime	TIME	Time of order pick-up of customer

Table 11: CustomerWithApp Table of A.CO

B. Technical Architecture

The web server does not have a web client. Thus, the mobile and tablet applications are reliant on themselves to do the alterations to the database without resorting to the usage of an administrative tool. The Database's minimum requirements are:

- MySQL 5.1.41
- Continuous internet connection
- Running on Ubuntu Linux 10.04
- 10GB or more of free space
- 2.4 MHz single core processor
- 512MB RAM

The Android for Customer Organization is developed in a Microsoft Windows 7 Ultimate machine. The mobile application was tested on an LG Optimus 2x SU660 mobile device. The minimum requirements, however, were tested on a Samsung Galaxy Y S5360 for the minimum specifications.

The mobile device requirements for the Customer are:

- At least 86MB of RAM
- At least 800MHz processor
- 100MB free space
- Wi-Fi 802.11 connectivity

- Android 2.3 (Gingerbread)
- 3MP primary camera

The tablet application was tested on a Samsung Galaxy Tab 3 10.1 P5220 tablet device. The minimum requirements, however, were tested on a 10.1" WXGA Tablet for the minimum specifications.

The tablet device requirements for the Counter Staff as well as the Cashier Staff are:

- At least 86MB of RAM
- At least 800MHz processor
- 100MB free space
- Wi-Fi 802.11 connectivity
- Android 3.0(Honeycomb)

For a minimal set-up, a total of five devices is needed for the system: one Heads Up Display for the waiting people in the area, one mobile device to carry for Customers with the application, one tablet device near the entrance of the facility for Customers without the application and for QR code releasing for Customers with the application, one tablet device for the Cashier, and one tablet device for the Counter Staff. For an optimal set-up, more tablet devices near the entrance of the facility, more tablets for more Cashiers and more tablets for more Counter Staffs (for bigger establishments) can be utilized.

V. Results

The Android for Customer Organization (A.CO) provides a system accessible to different types of J.Co store customers as well as the different types of J.Co store personnel. This allows the said people to transact with each other conveniently with the use of their Android devices.

There are two major application packages developed to be utilized by four kinds of users of the A.CO system—the Counter Staff, the Cashier, the Customer who has the A.CO installed on their Android mobile devices, and the Customer who has no Android mobile device with the A.CO installed. Each of these users has their own areas accessible only to their specific user type. However, some of their functionalities behave similarly or equivalently to like how the other users' functionalities do.

A. Application Package for the Store Personnel and the Heads Up Display

The Registration Screen, as shown in Figure 12, allows the Counter Personnel and the Cashier Personnel, especially the newly appointed ones, to register their selves into the A.CO system.

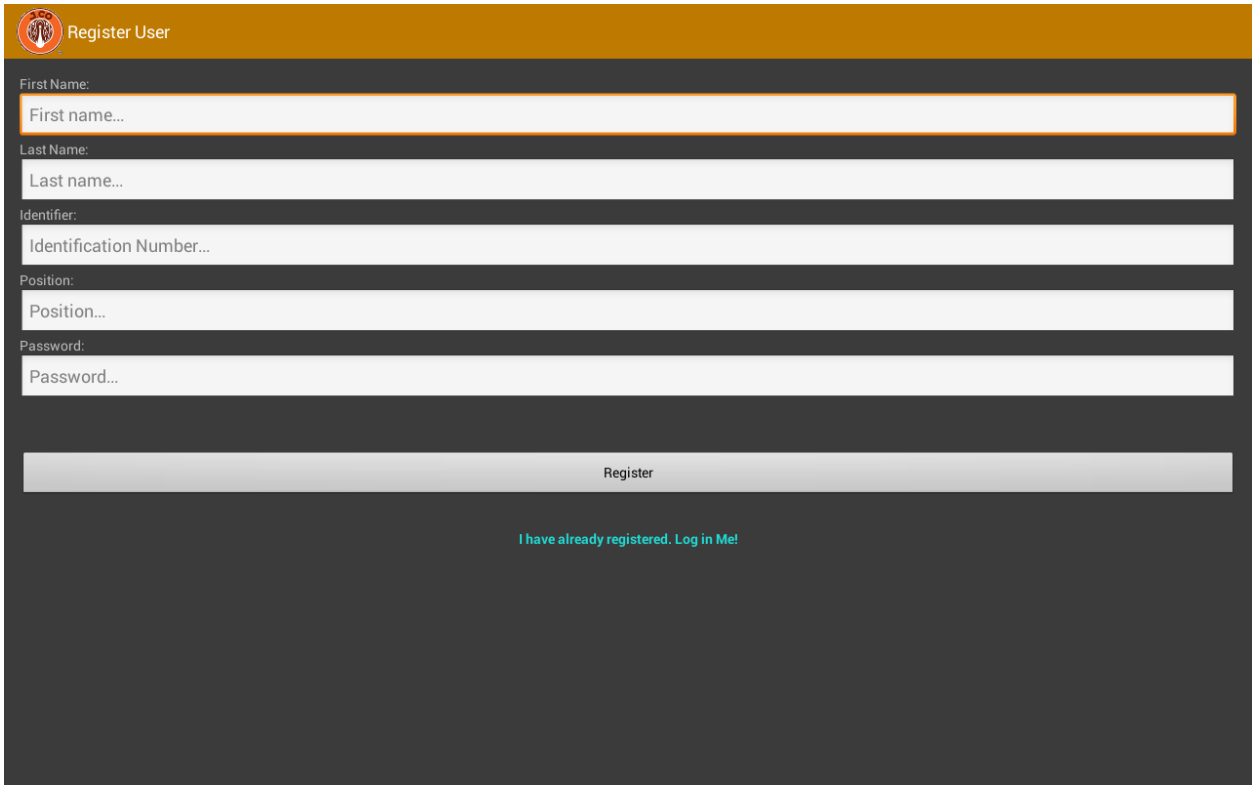

Figure 12: Store Personnel Registration Screen

The store personnel are limited to inputting the proper system role that they will be playing. Failing to input the proper position will result to an error message alert, as shown in Figure 13. This is to add a layer of protection to the system for those who are not knowledgeable of the approved user roles for A.CO.

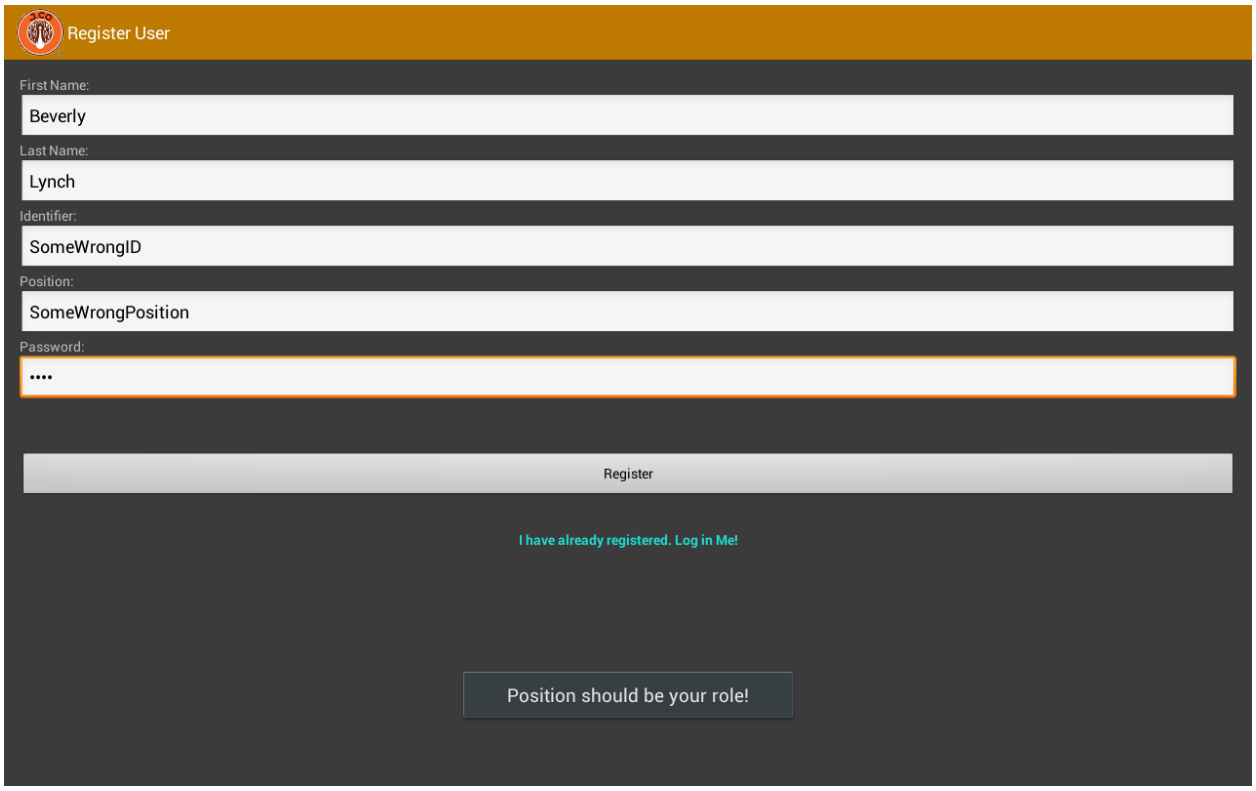

Figure 13: Store Personnel Registration Error in System Role Alert

The store personnel are bound to complete filling up all the input fields to be able to successfully register into the system. Failing to fill up all the necessary fields will result to an error message alert, as shown in Figure 14.

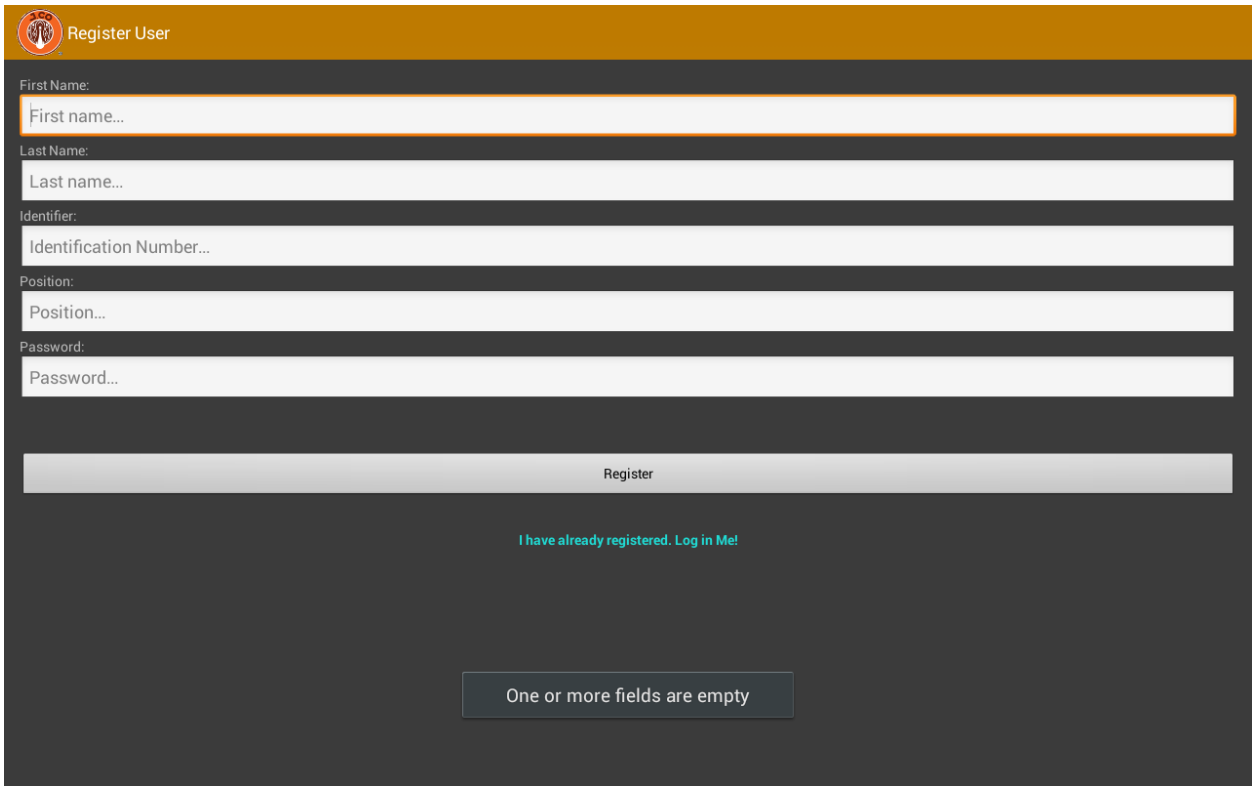

Figure 14: Store Personnel Registration Error in Input Completion Alert

The tablet device has to have an internet connection for it to be able to register a store personnel. Failure to comply with this will result to an error in network connection alert, as shown in Figure 15.

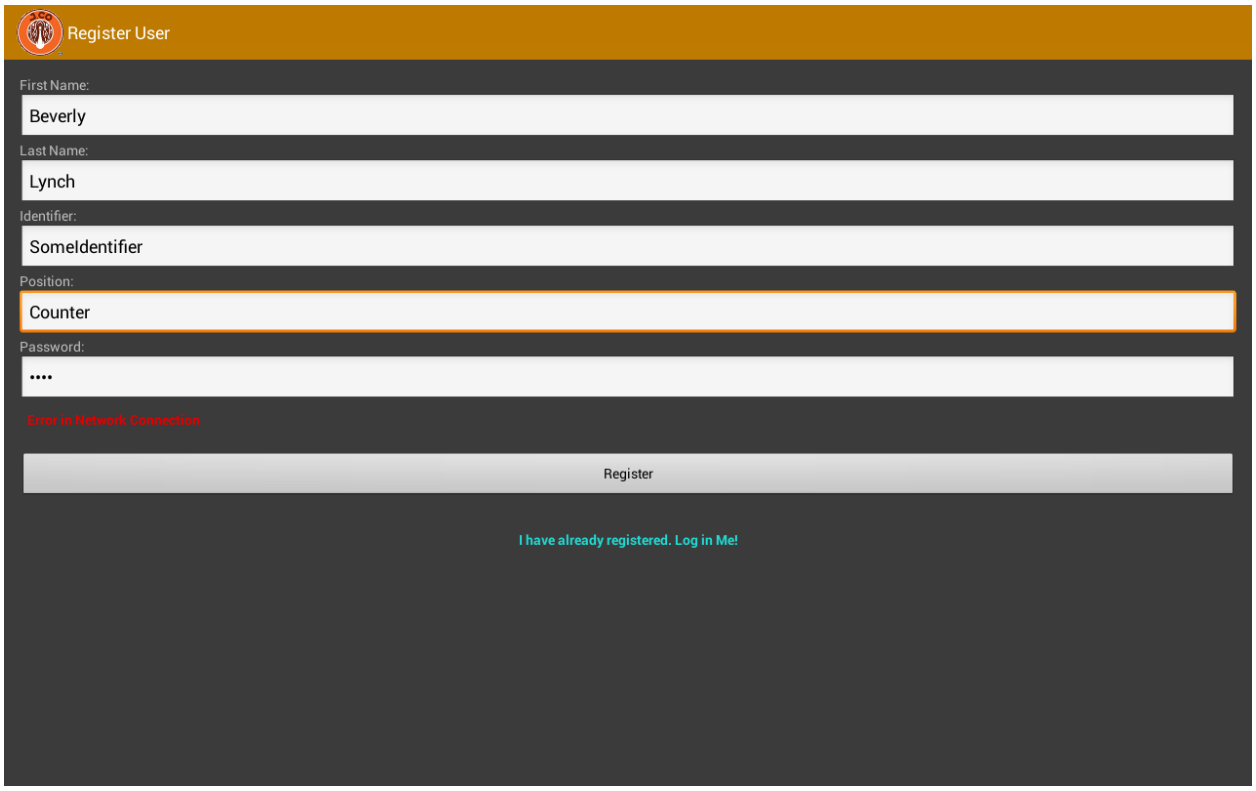

Figure 15: Store Personnel Registration Error in Network Connection Alert

On the other hand, the Login Screen, as shown in Figure 16, allows the Counter Personnel and the Cashier Personnel to access the A.CO system. This keeps unauthorized personnel from entering the system.

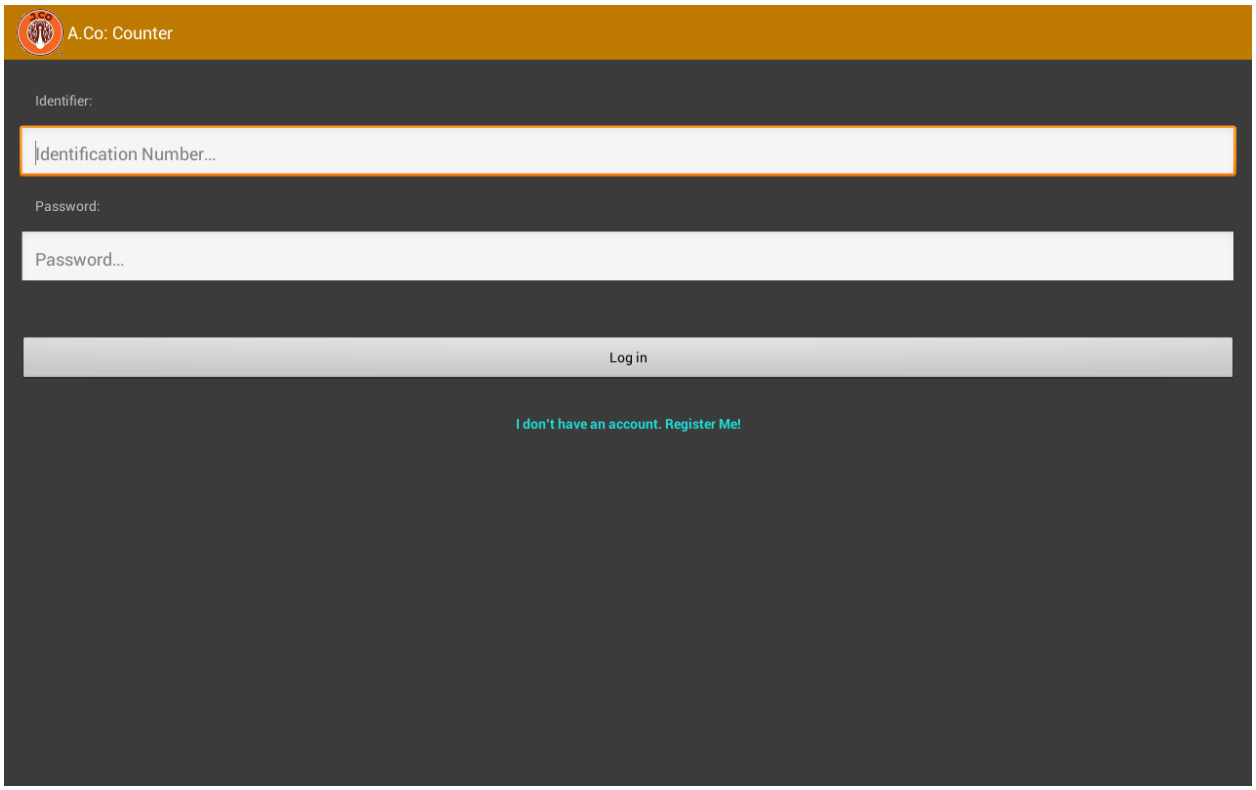

Figure 16: Store Personnel Login Screen

The store personnel are bound to complete filling up all the input fields to be able to successfully log in into the system. Failing to fill up any of the necessary fields will result to an error message alert, as shown in Figure 17, Figure 18, and Figure 19.

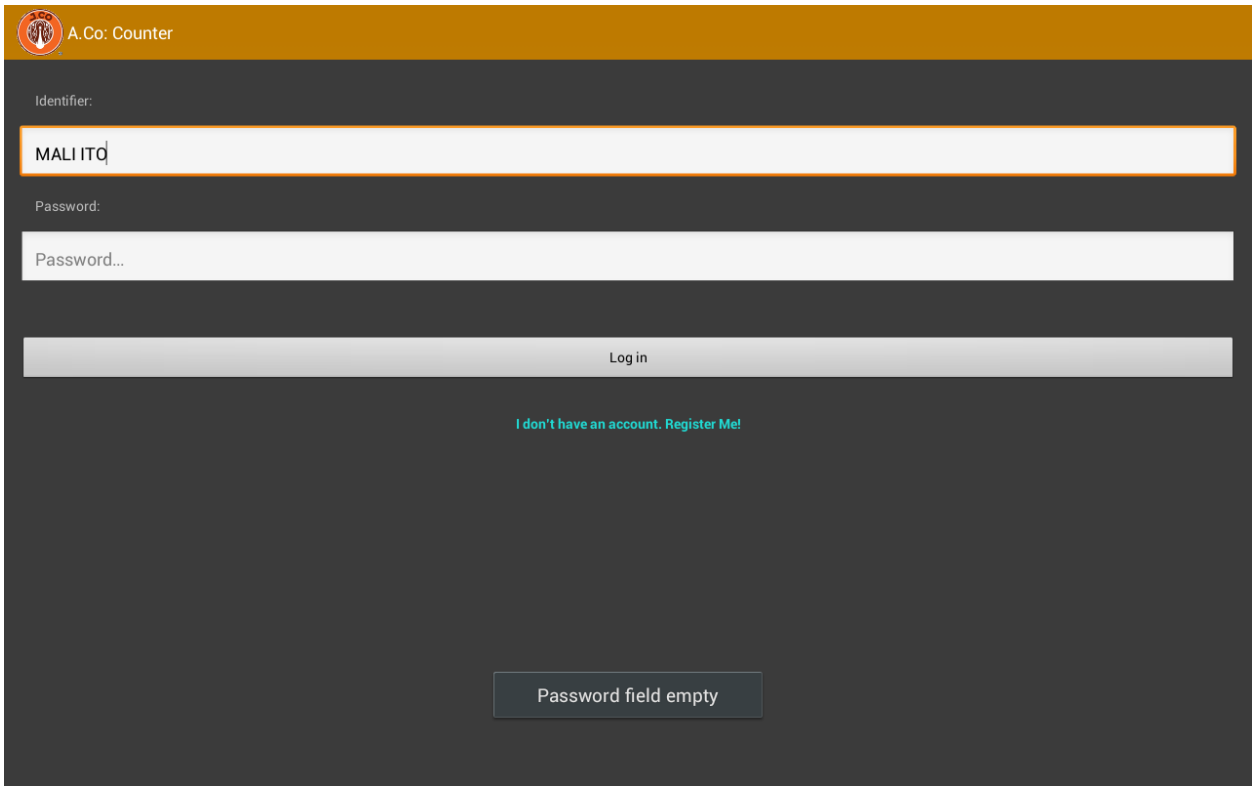

Figure 17: Store Personnel Login Error in Input Password Alert

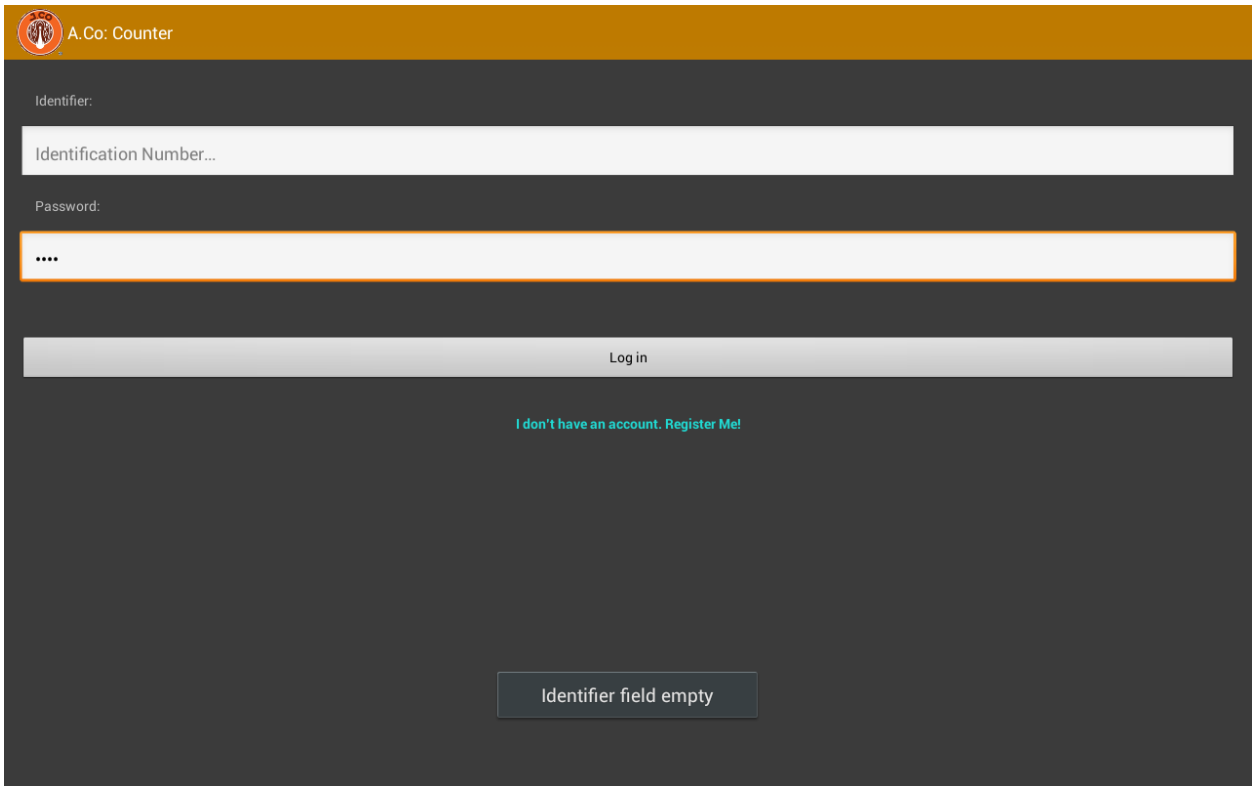

Figure 18: Store Personnel Login Error in Input Identifier Alert

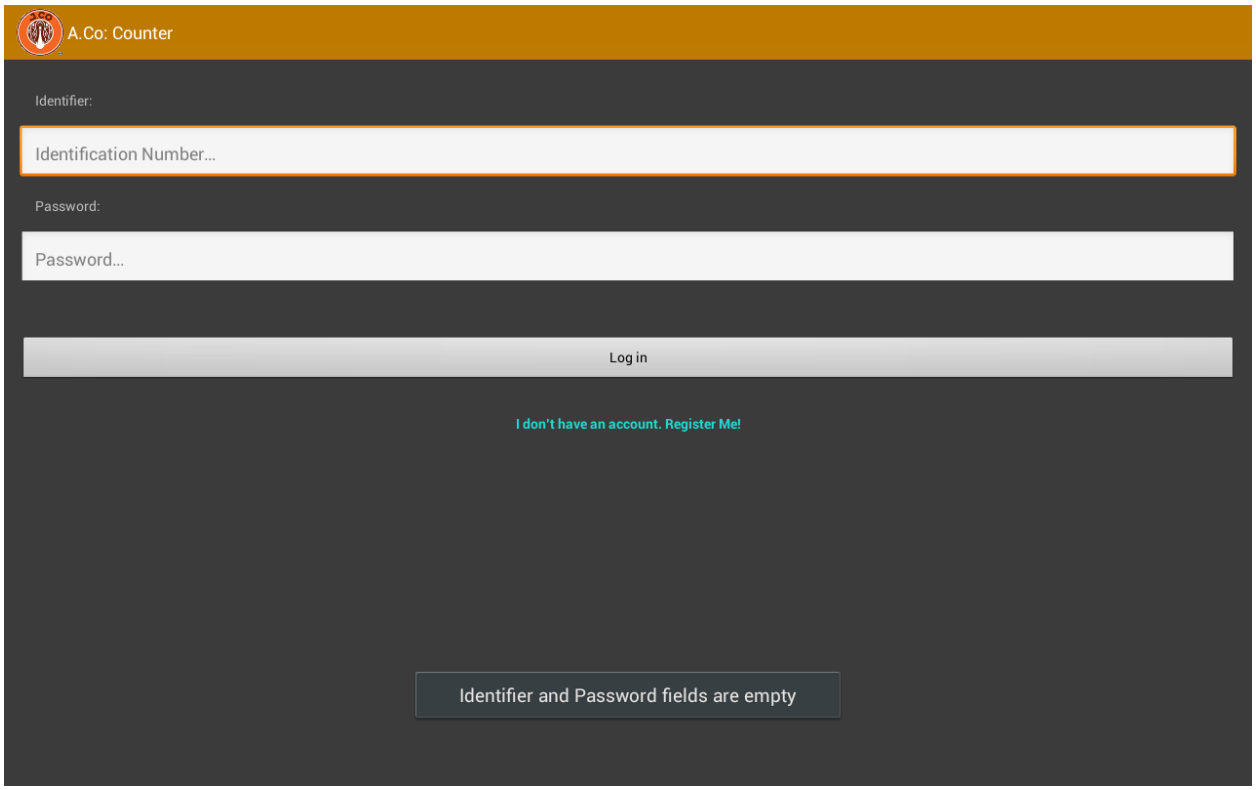

Figure 19: Store Personnel Login Error in Input Fields Alert

The tablet device has to have an internet connection for it to be able to log in a store personnel. Failure to comply with this will result to an error in network connection alert, as shown in Figure 20.

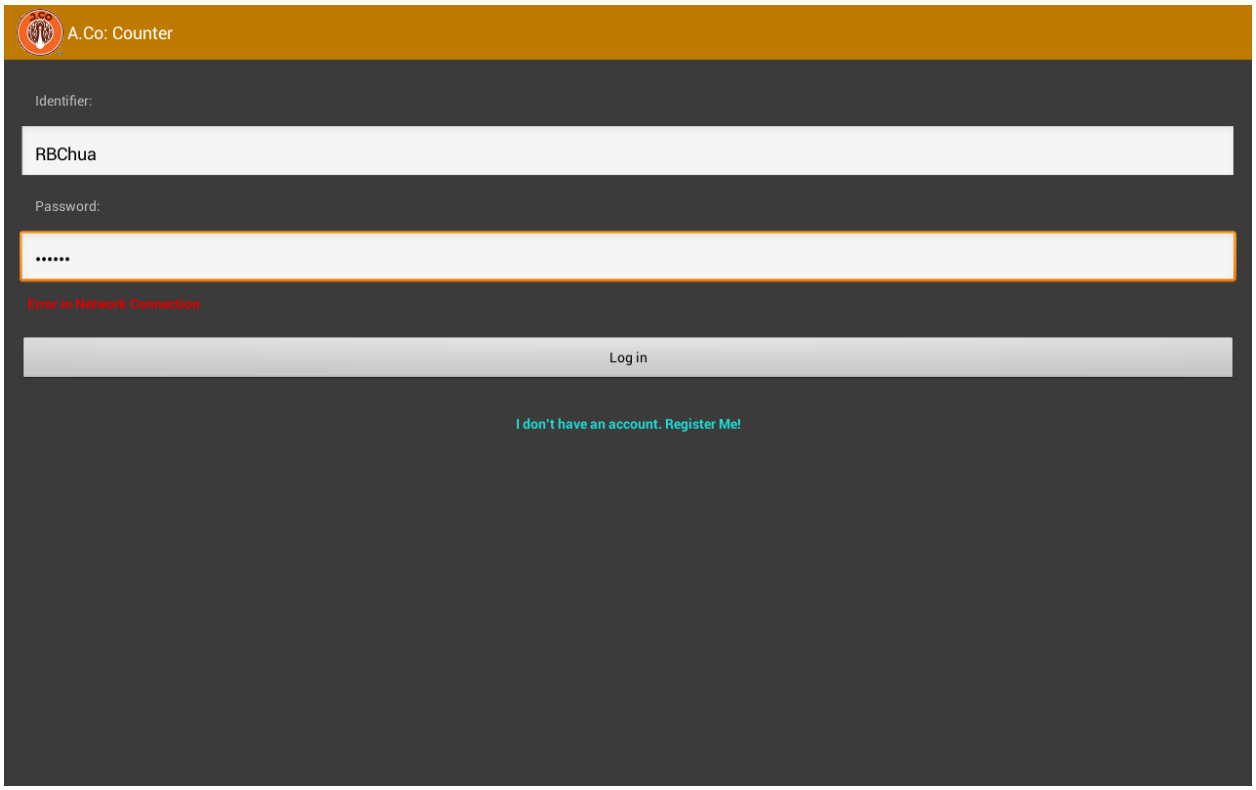

Figure 20: Store Personnel Login Error in Network Connection Alert

After a successful log in or registration attempt, the store personnel will then be directed to the Personnel Dashboard Screen, as shown in Figure 21. He/she will have two options: first, by clicking the "*Let me serve!*" button, he/she will be able to perform his/her duties after being directed to the Main Screen; and second, by clicking the *"Log me out!"* button, he/she will be able to log out of the system.

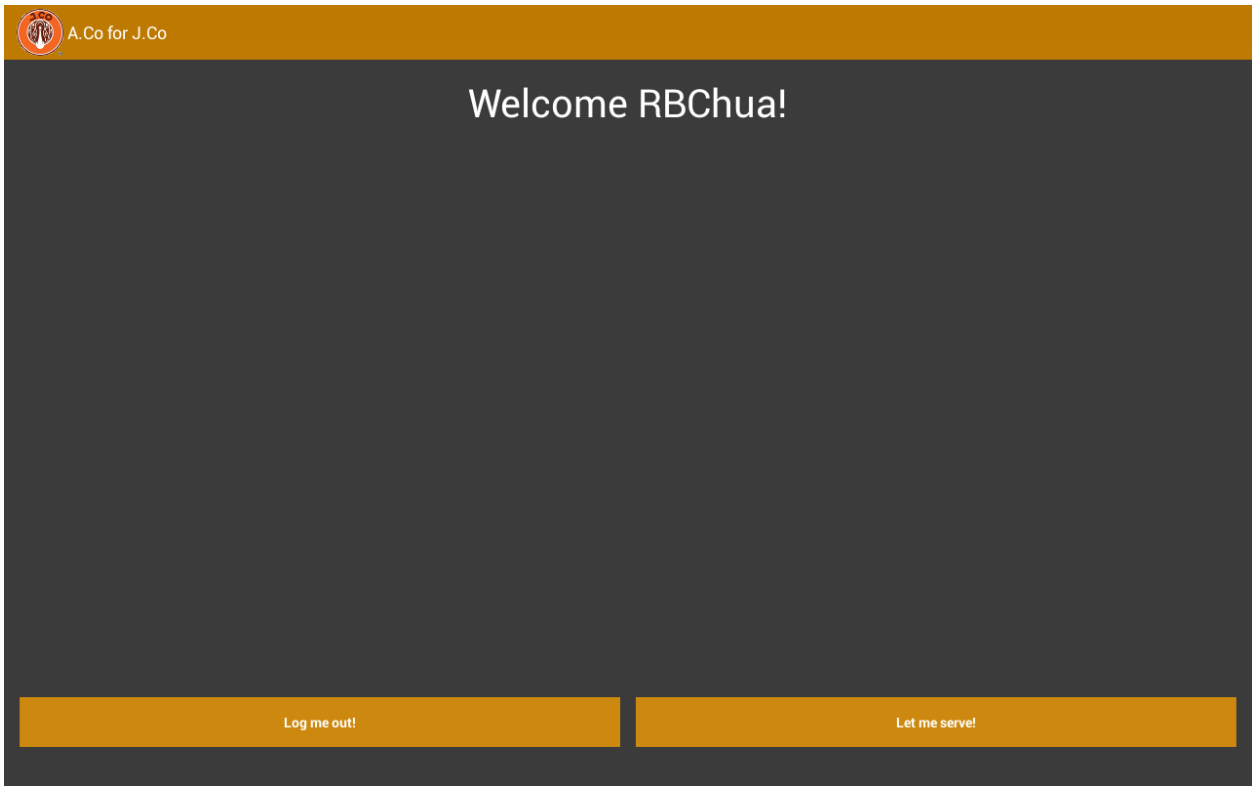

Figure 21: Store Personnel Dashboard Screen

Upon choosing the *Let Me Serve* option in the Dashboard screen, the store personnel will be directed to the Main Screen of the system, as shown in Figure 22. This screen is divided into three parts: the navigation tabs, which lets the store personnel in to the other functionalities of the system; the header, which states the current navigation position of the store personnel with respect to the functionality pages of the system; and the body, which changes accordingly as the store personnel navigates his/her self through the functionalities of the system. The *Home* navigation tab navigates the store personnel to this Main Screen.

Figure 22: Store Personnel Main Screen

The *Menu* navigation tab navigates the store personnel to the Menu Screen, as shown in Figure 23. This Menu Screen contains the five different categories of products offered by J.Co Donuts and Coffee. This functionality page provides options for the store personnel to pick which category he/she will perform changes on.

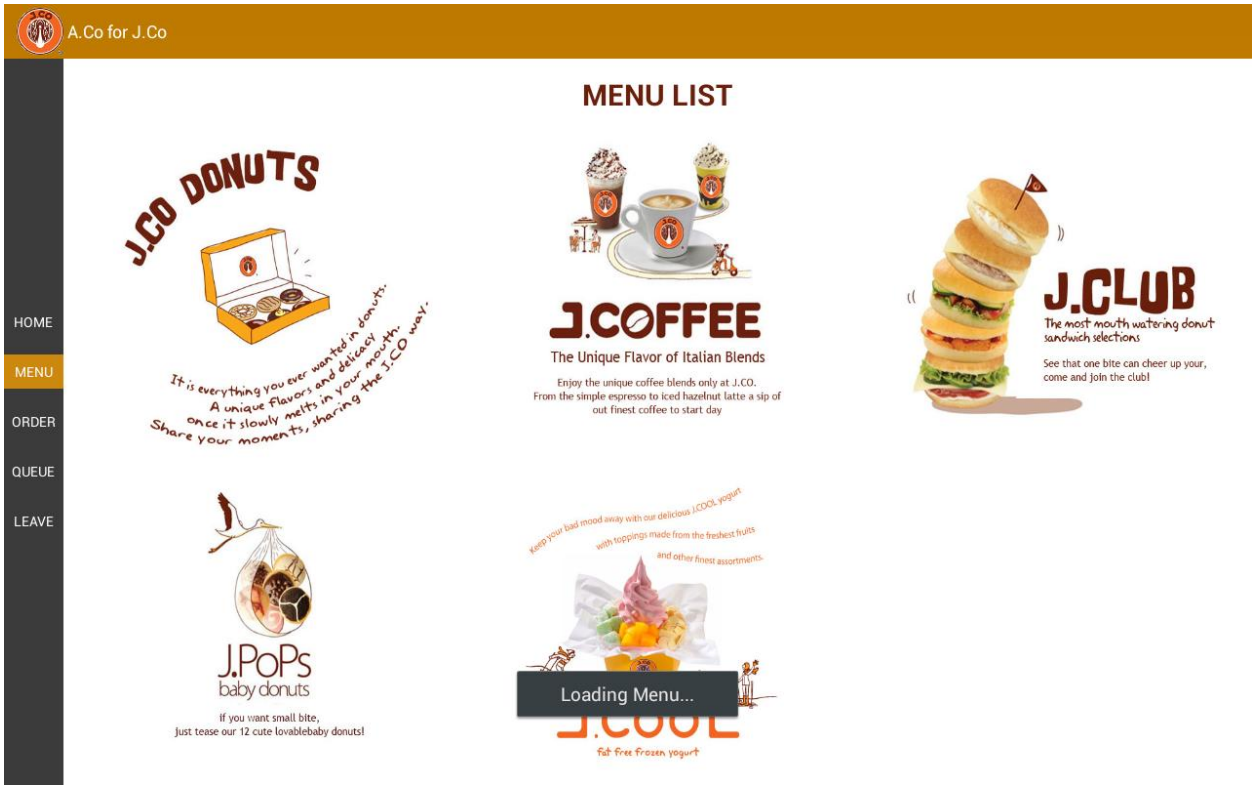

Figure 23: Store Personnel Menu Screen

The Menu List Screen, as shown in Figure 24, contains the food catalog of J.CO Donuts and Coffee with respect to the selected food category from the Menu Screen. This functionality page allows the store personnel to manage the items to be displayed on the Customer Menu List Screen.

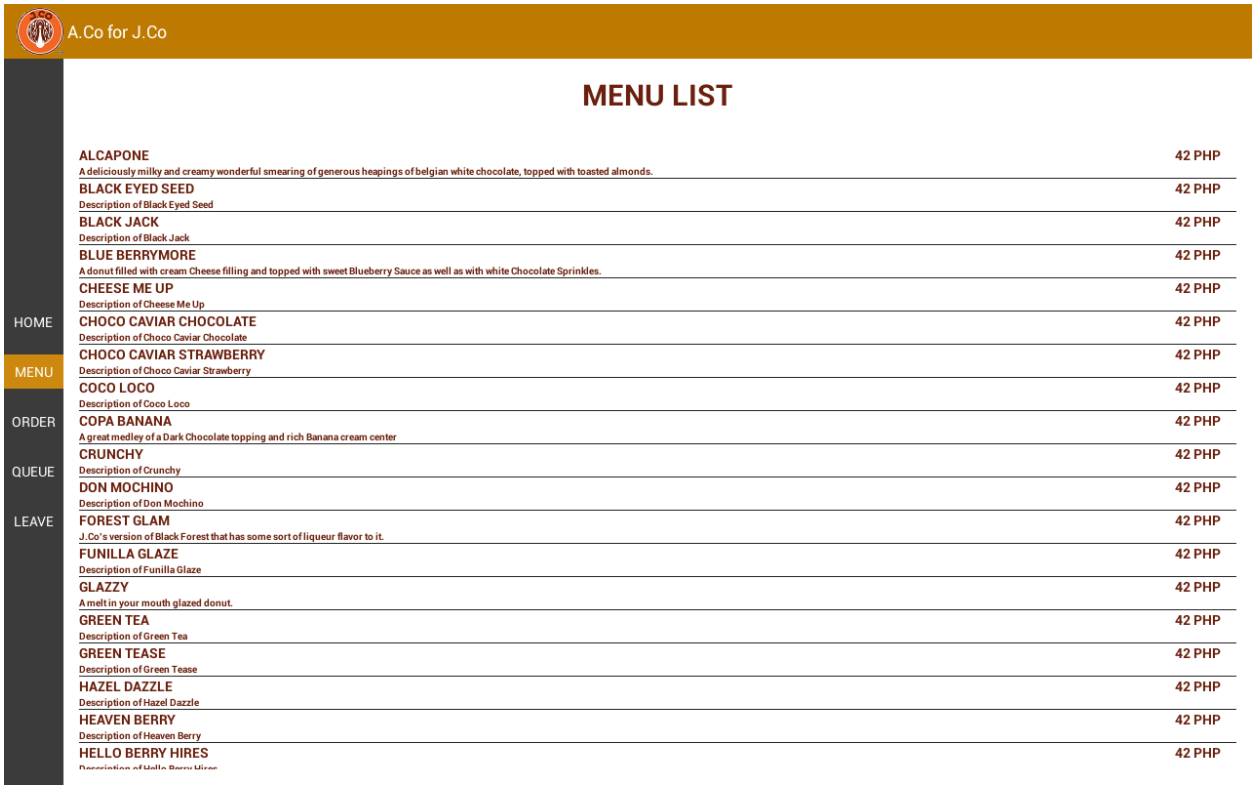

Figure 24: Store Personnel Menu List Screen

The *Order* navigation tab navigates the store personnel to the Customer Screen, as shown in Figure 25. This Customer Screen contains the list of the customers currently in queue. This functionality page allows a Counter Personnel to pick a customer in the virtual queue, view the orders of the said customer, and relay the order status of the said customer to the Cashier Personnel.

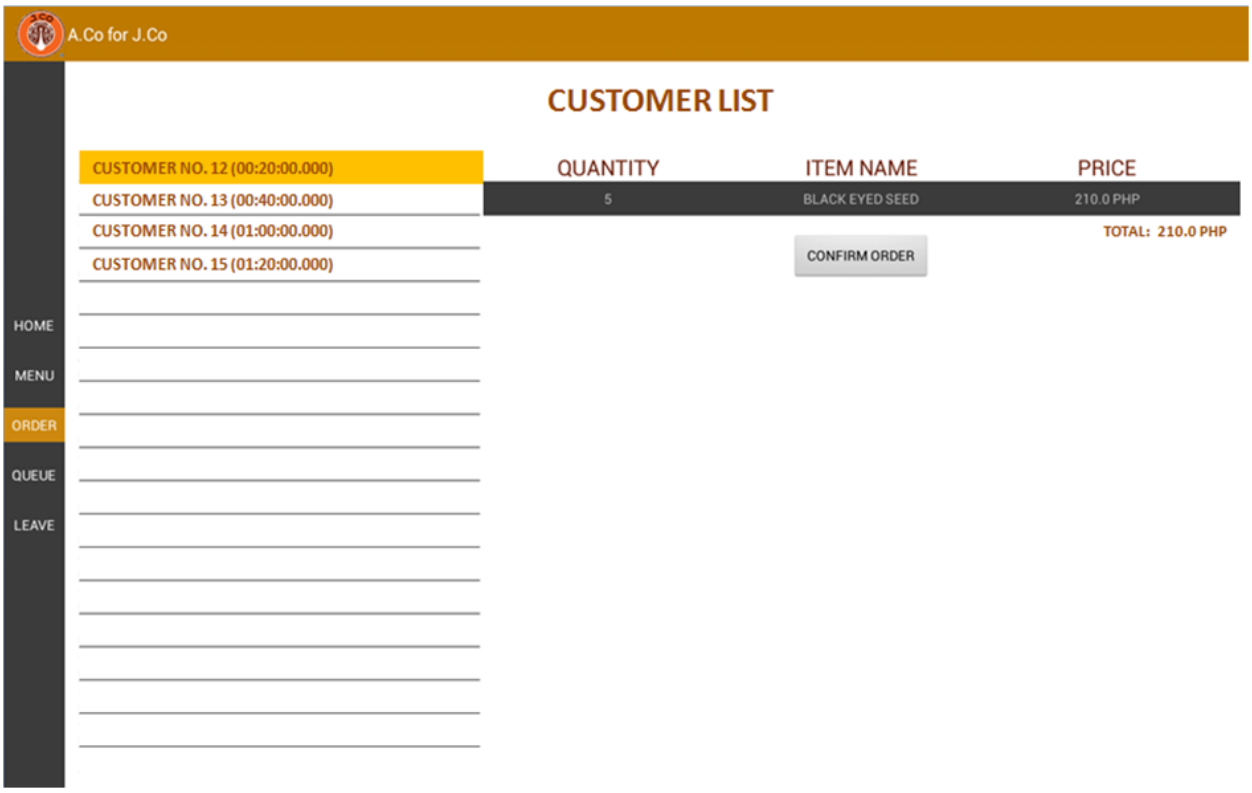

Figure 25: Counter Personnel's Customer Screen

Upon confirming the order status of the currently being served customer on the Counter Personnel's Customer Screen, the Cashier Personnel's Customer Screen, as shown in Figure 26, will now allow the Cashier Personnel to pick the said customer in the customer list and call him/her for order payment and pick-up. This concludes the transaction process between the said Customer and the Store.

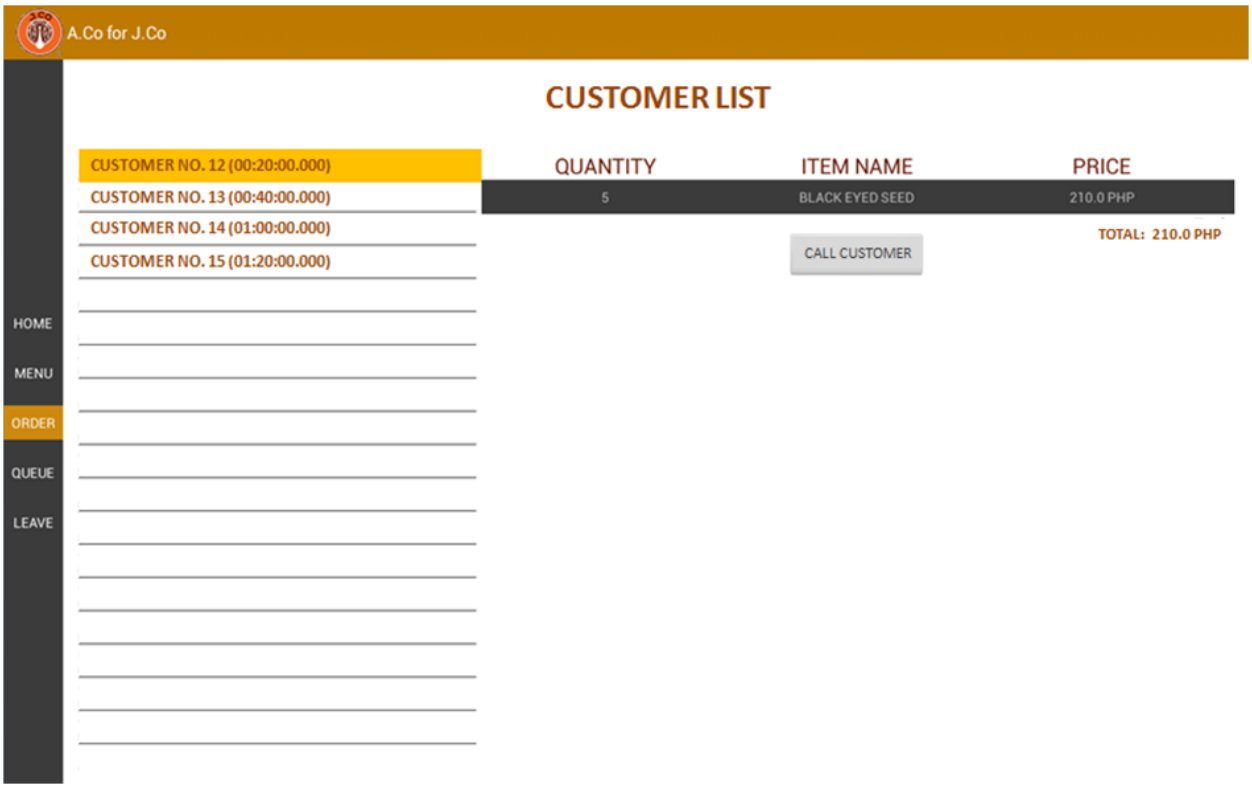

Figure 26: Cashier Personnel's Customer Screen

Upon confirming the order status of the Customer on the Counter Personnel's Customer Screen, and upon concluding the transaction between the said Customer and the Store on the Cashier Personnel's Customer Screen, the Queue Progress Screen, as shown in Figure 27, will now display the updated *Currently Being Served* customer number and *Next On Queue* numbers. The *Queue* navigation tab navigates the store personnel to this Queue Progress Screen.

Figure 27: Store Personnel Queue Progress Screen

The *Leave* navigation tab navigates the store personnel out of the system, as shown in Figure 28. This functionality button provides an option for the store personnel to end the current transaction session.

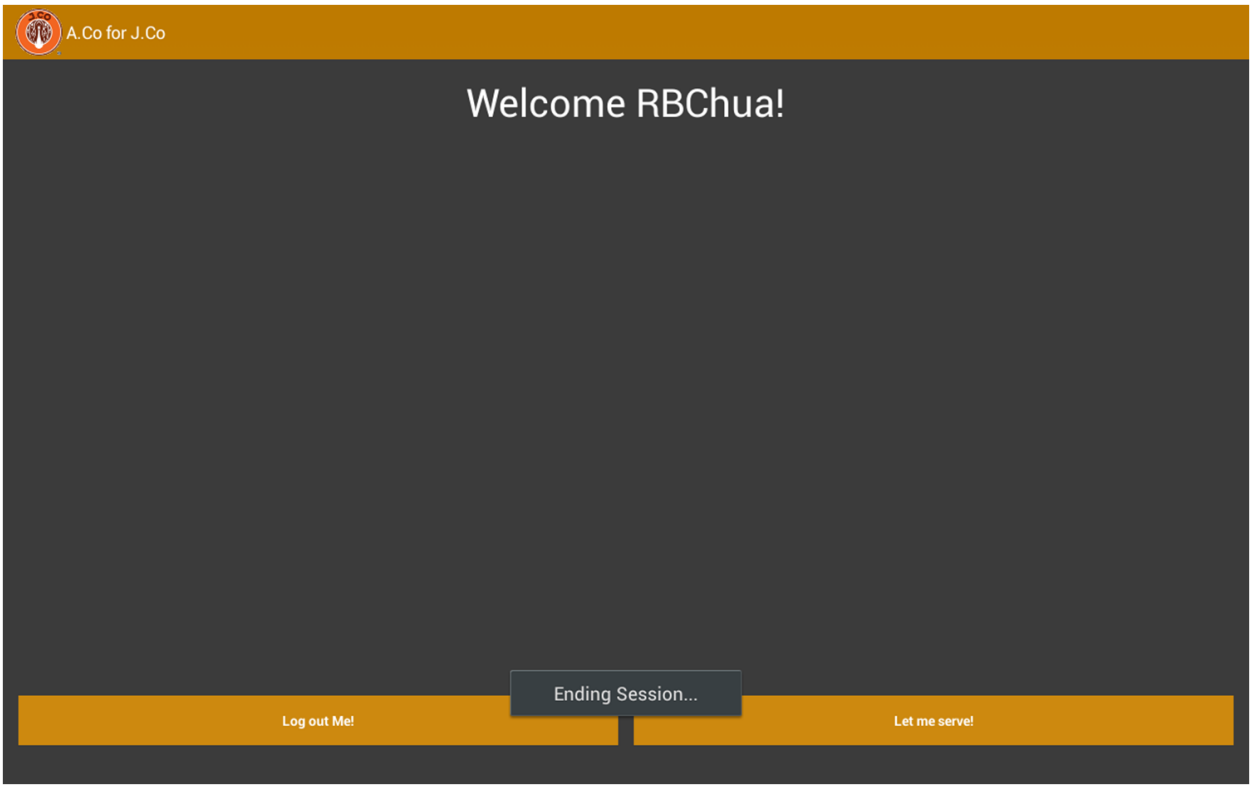

Figure 28: Store Personnel Ending Session

After leaving the session, the store personnel will be directed once again to the Personnel Dashboard Screen, as shown in Figure 29.

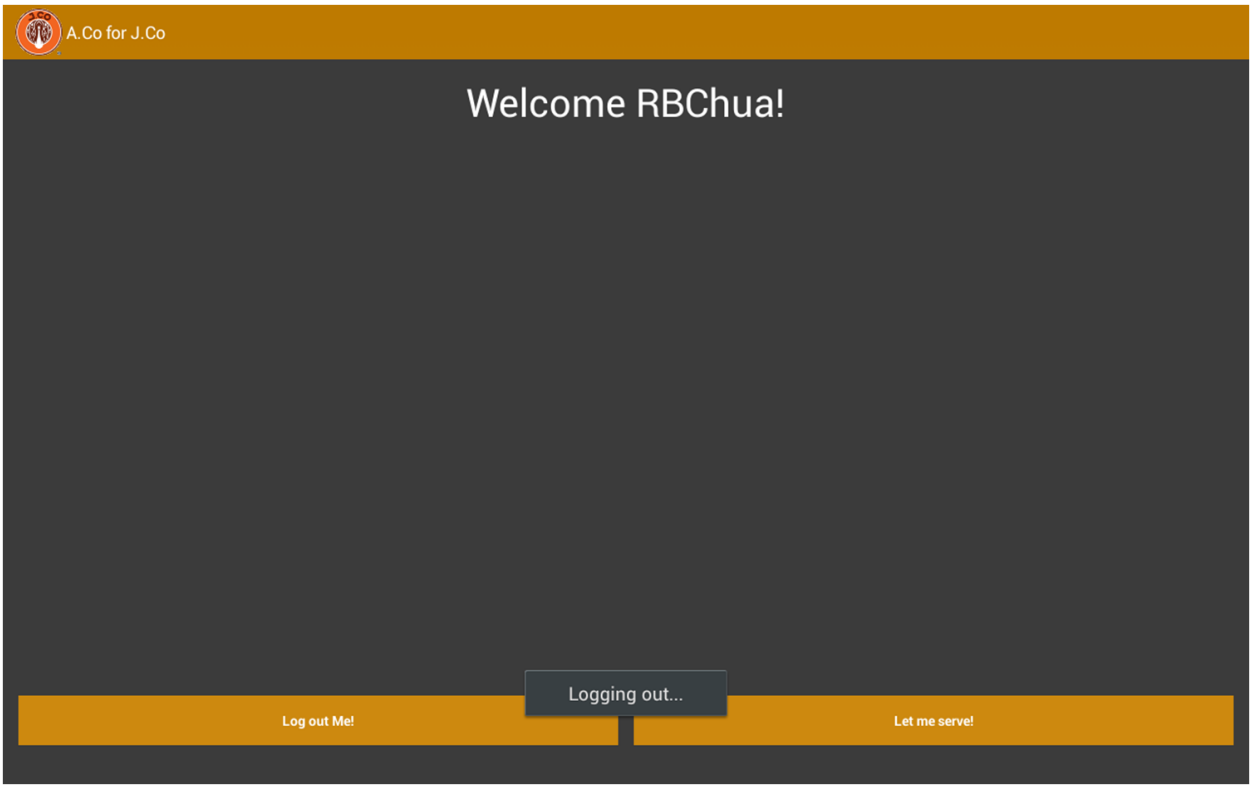

Figure 29: Store Personnel Leaving Dashboard Screen

B. Application Package for the Store Entrance

The Welcome Screen, as shown in Figure 30, provides two options for the Customers: first, a Customer who has no Android device that has A.CO installed on it may tap on the *While You Wait* option; and second, a Customer who has an A.CO installed on his/her device may tap on the *As You Roam* option.

Figure 30: Customer Welcome Screen

Upon choosing the *While You Wait* option in the Welcome Screen, the Customer will be directed to the Customer Registration Screen, as shown in Figure 31. This Customer Registration Screen allows a Customer to register their selves into the A.CO system. The customer is limited to inputting his/her full name and visible physical details for the store personnel to be able to verify his/her identity upon claiming of orders.

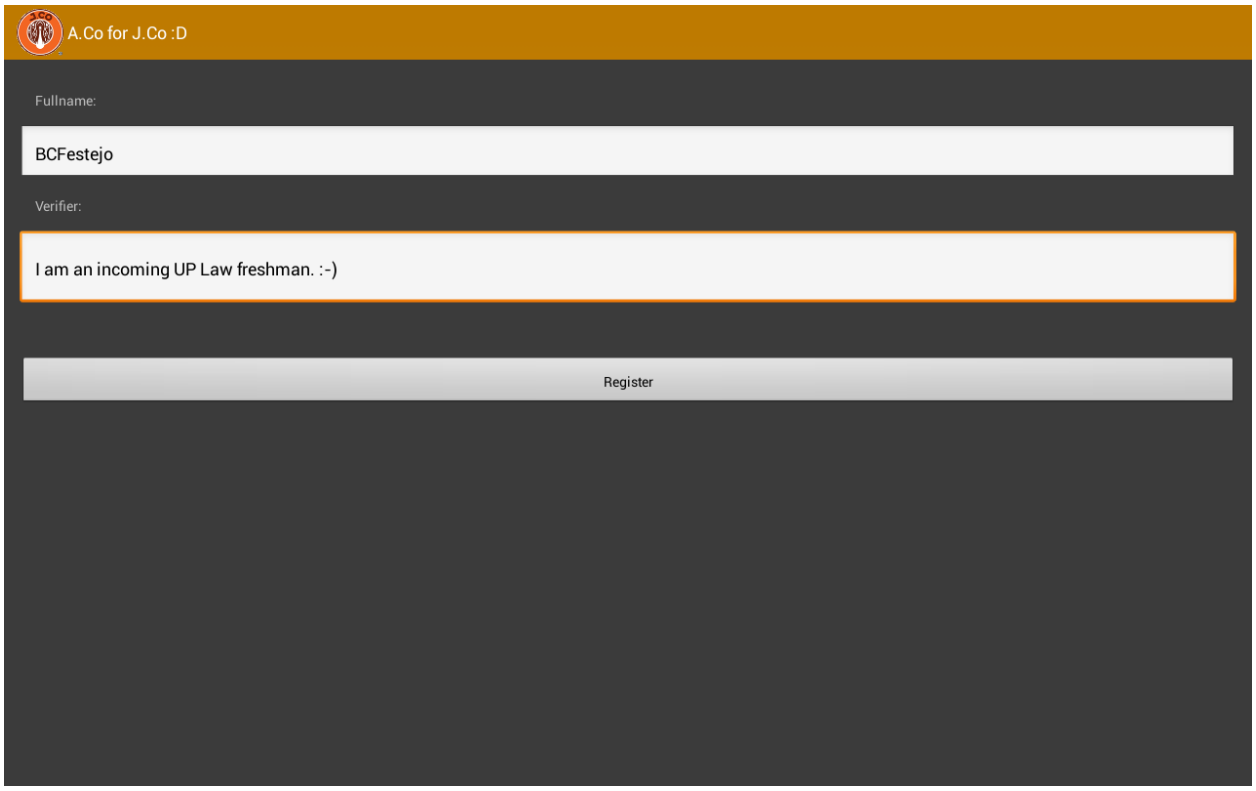

Figure 31: Walk In Customer 'While You Wait' Registration Screen

The customer is bound to complete filling up all the input fields to be able to successfully register into the system. Failing to fill up all the necessary fields will result to an error message alert, as shown in Figure 32, Figure 33, and Figure 34.

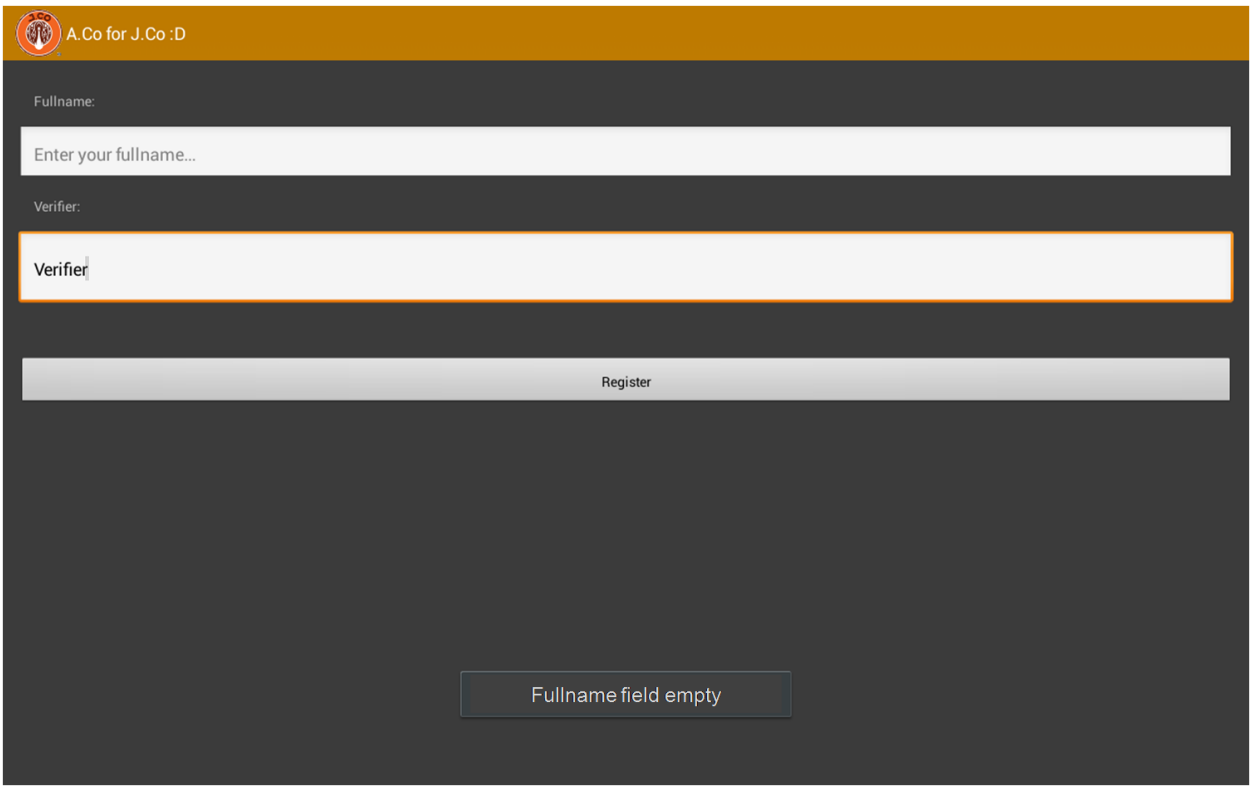

Figure 32: Walk In Customer Registration Error in Input Full Name Alert

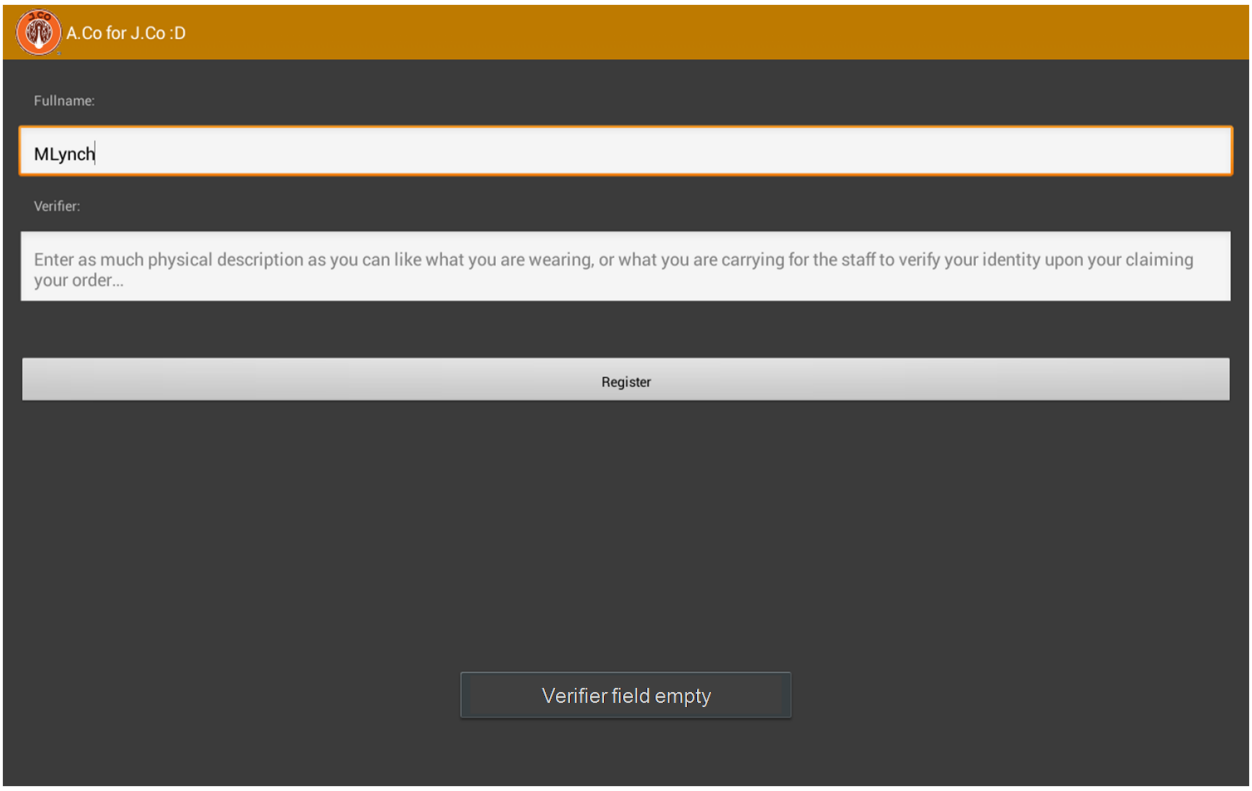

Figure 33: Walk In Customer Registration Error in Input Verifier Alert

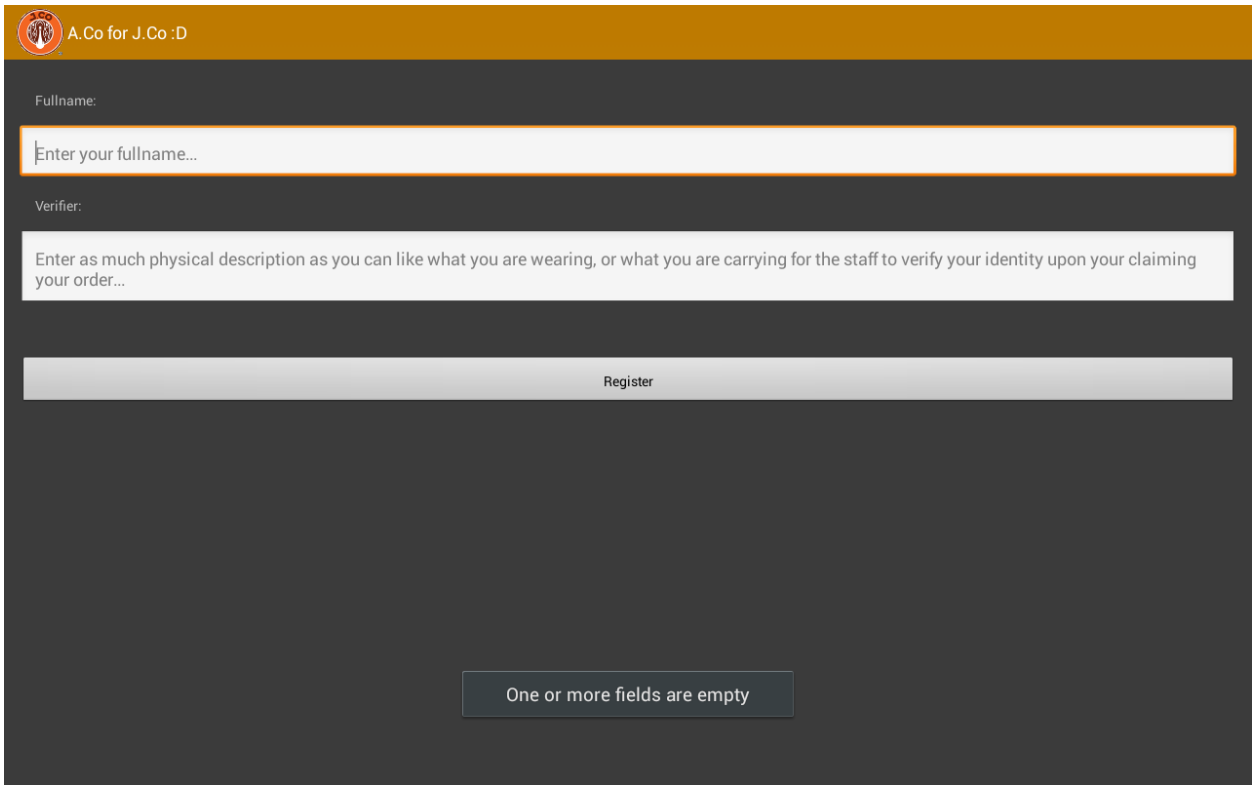

Figure 34: Walk In Customer Registration Error in Input Fields Alert

The tablet device has to have an internet connection for it to be able to register a customer. Failure to comply with this will result to an error in network connection alert, as shown in Figure 35.

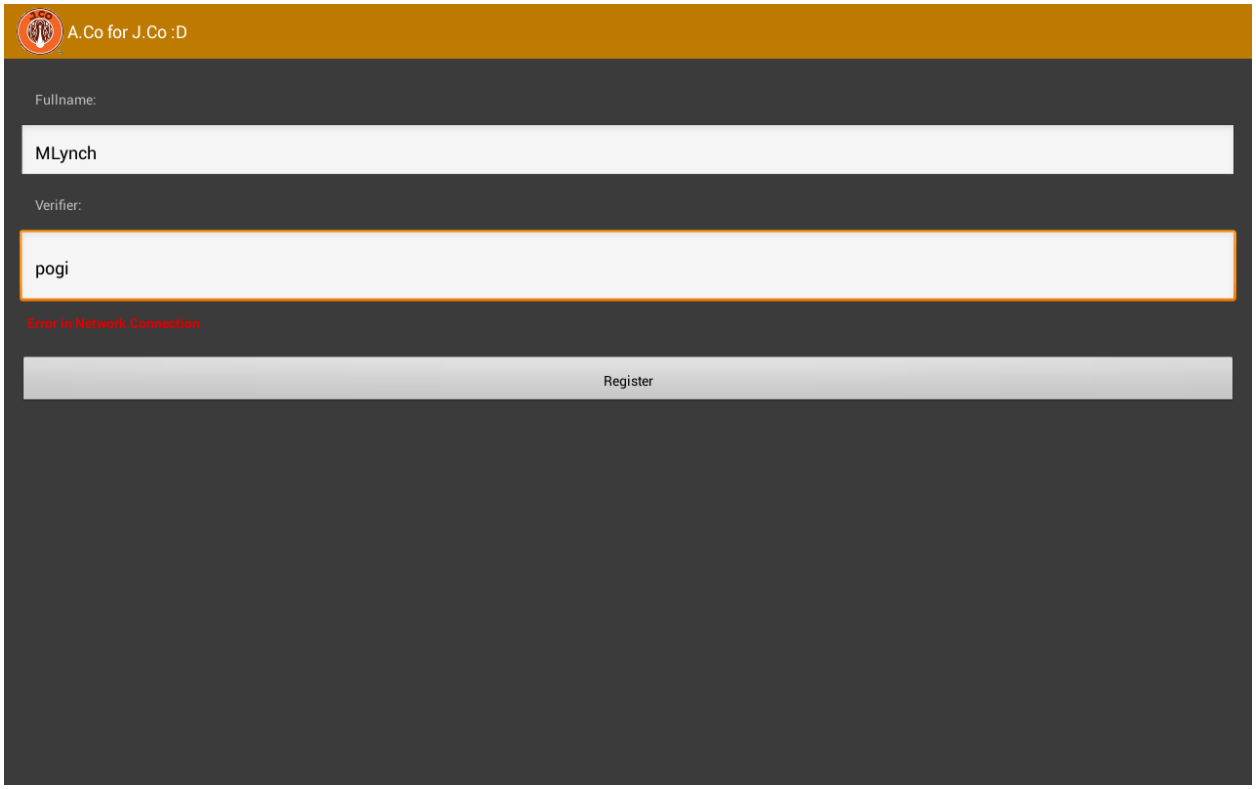

Figure 35: Walk In Customer Registration Error in Network Connection Alert

After a successful registration attempt, the Customer will then be directed to a Customer Dashboard Screen, as shown in Figure 36. He/she will be greeted by his/her queue number and will have two options: first, by clicking the "*Let me order!*" button, he/she will be able to send order requests after being directed to the Customer Home Screen; and second, by clicking the "*Loge me out!*" button, he/she will be able to simply log out of the system.

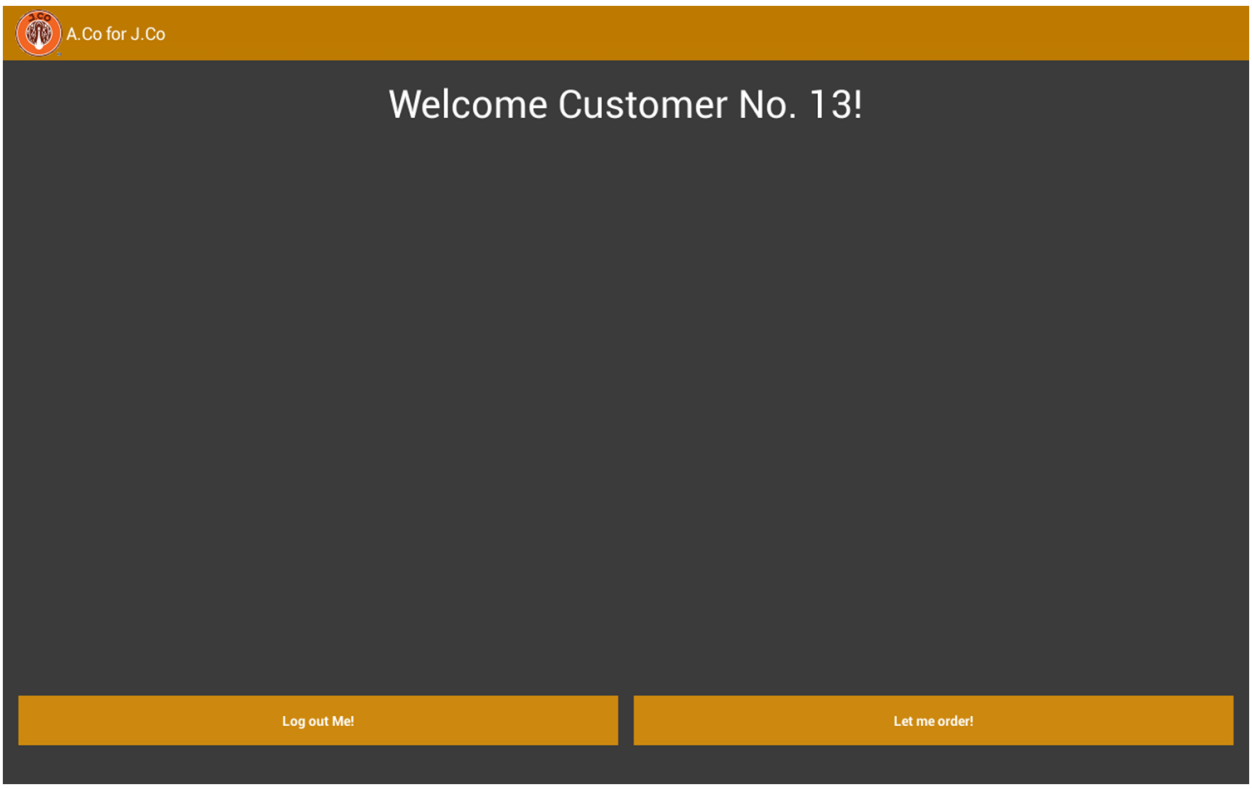

Figure 36: Walk In Customer Dashboard Screen

Upon choosing the *Let Me Order* option in the Dashboard screen, the customer will be directed to the Home Screen of the system, as shown in Figure 37. This screen is similar in structure to that of the Store Personnel's which is divided into three parts: the navigation tabs, which lets the customer in to the other ordering functionalities of the system; the header, which states the current navigation position of the customer with respect to the ordering functionality pages of the system; and the body, which changes accordingly as the customer navigates his/her self through the ordering functionalities of the system. The *Home* navigation tab navigates the customer to this Home Screen.

Figure 37: Walk In Customer Home Screen

The *Menu* navigation tab navigates the customer to the Customer Menu Screen, as shown in Figure 38. This Customer Menu Screen contains the five different categories of products offered by J.Co Donuts and Coffee. This functionality page provides options for the customer to pick which category he/she will browse and order from.

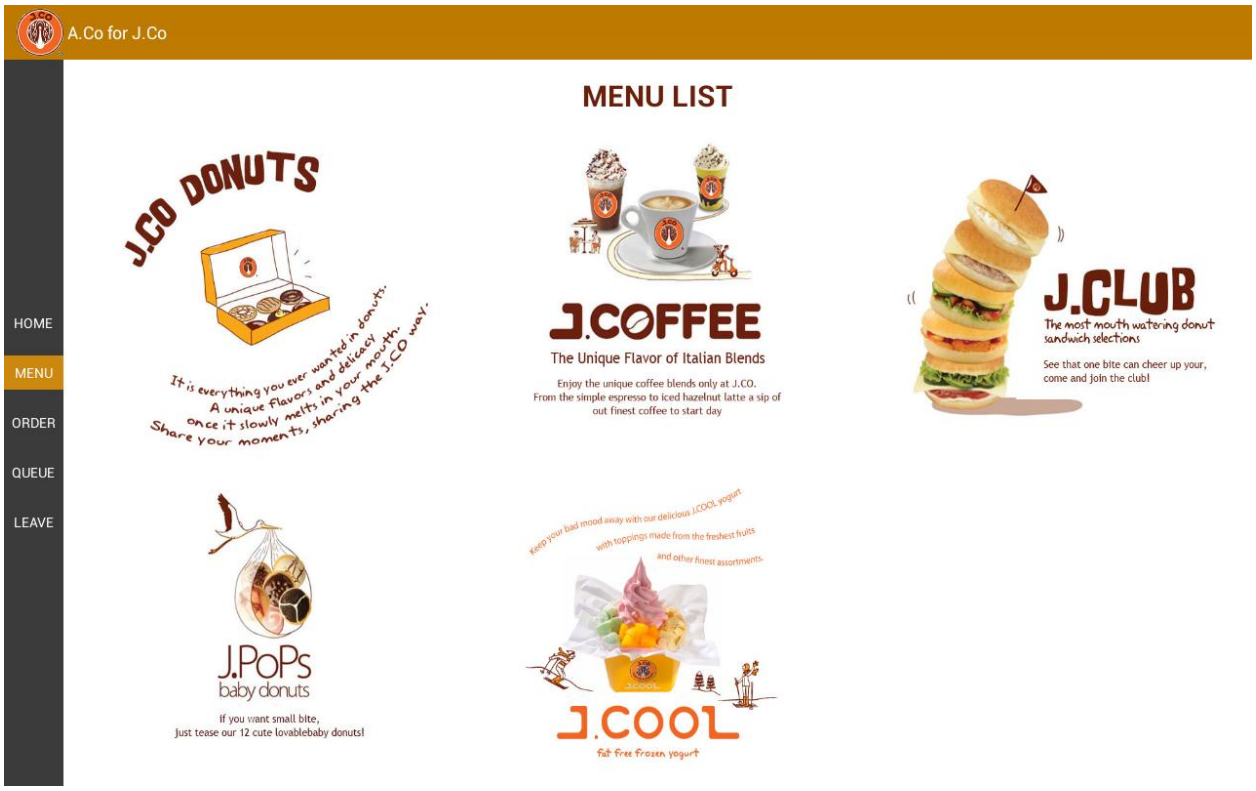

Figure 38: Walk In Customer Menu Screen

The Customer Menu List Screen, as shown in Figure 39, contains the food catalog of J.Co Donuts and Coffee with respect to the selected food category from the Customer Menu Screen. This functionality page allows the customer to browse the items under the category he/she has chosen. He/she is allowed to pick an item to view its product description and to put it on his/her order list.

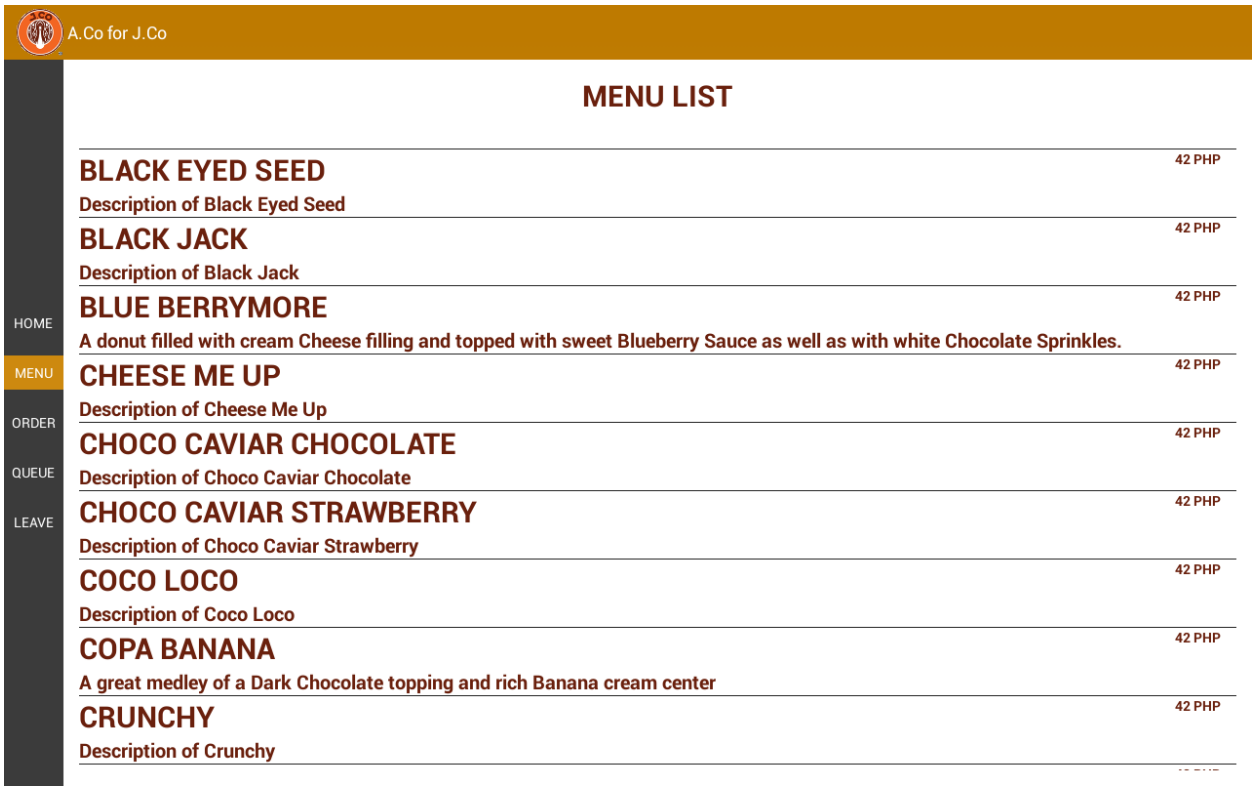

Figure 39: Walk In Customer Menu List Screen

Upon picking an item on the Customer Menu List Screen, the Customer will be directed to the Product Description Screen, as shown in Figure 40. This screen allows the customer to view the detailed information of the said item and to specify the quantity of the said item he/she would like to order. This order list will then be reflected to the Customer's Order Screen after finishing off the order inputting.

Figure 40: Product Description Screen

After a successful ordering attempt, the customer will then be directed back to the Customer Menu Screen, along with an addition to the order list alert, as shown in Figure 41.

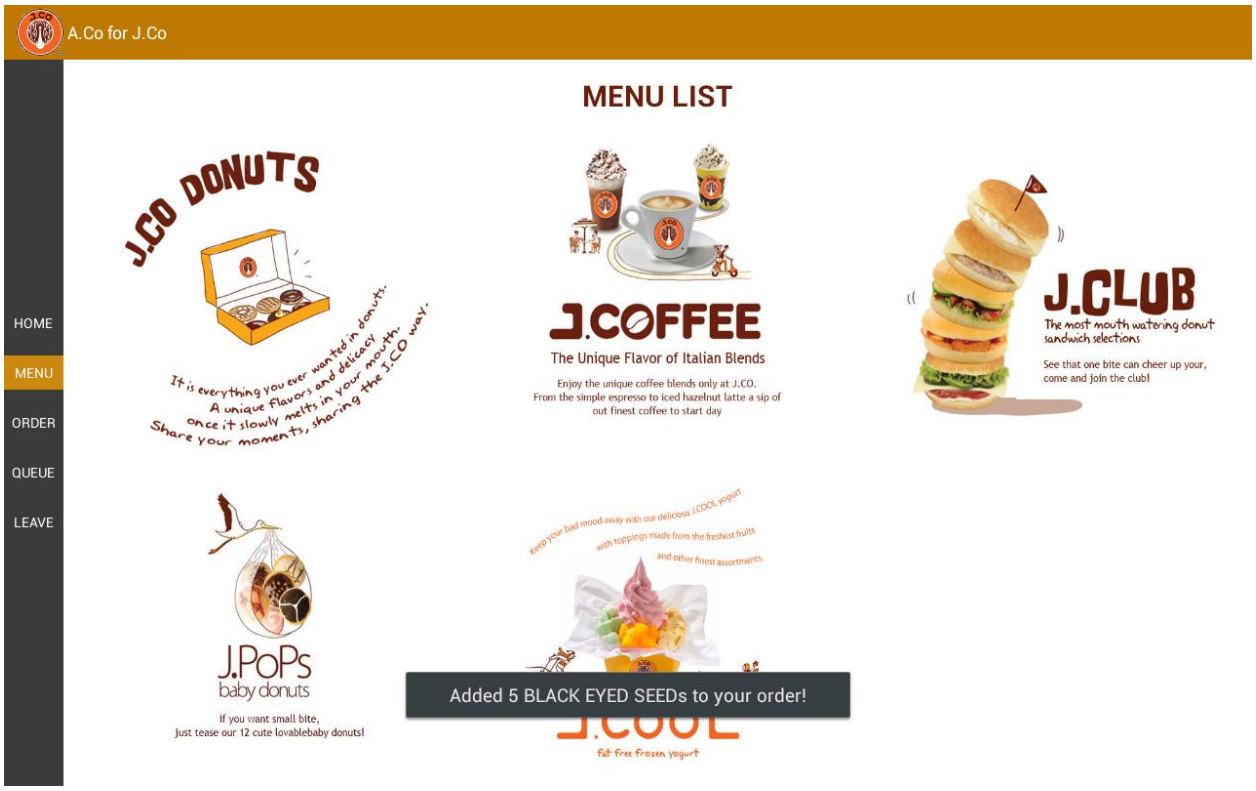

Figure 41: Walk In Customer Order Addition Alert

The *Order* navigation tab navigates the customer to the Order Screen, as shown in Figure 42. This Customer's Order Screen contains the summary of the items picked by the customer to be put in his/her order list. This ordering functionality page gives the Customer two options: first, by clicking the *Confirm Order* button, the list of items ordered by the customer will be sent to the Store Personnel for accomplishment; and second, by simply going back, the customer will directed to the Menu Screen and will be allowed to order some more.

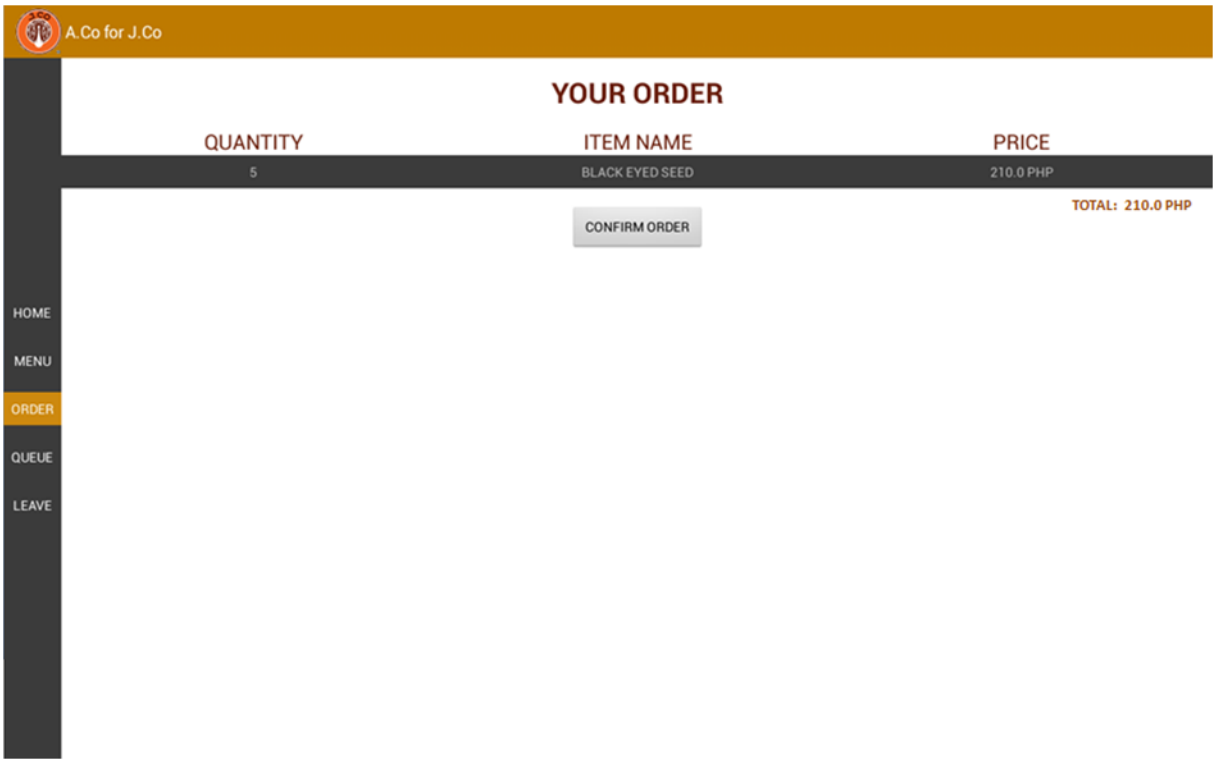

Figure 42: Walk In Customer's Order Screen

A Customer is allowed to Add, Edit, or Delete an Order item, as shown in Figure 43, Figure 44, and Figure 45.

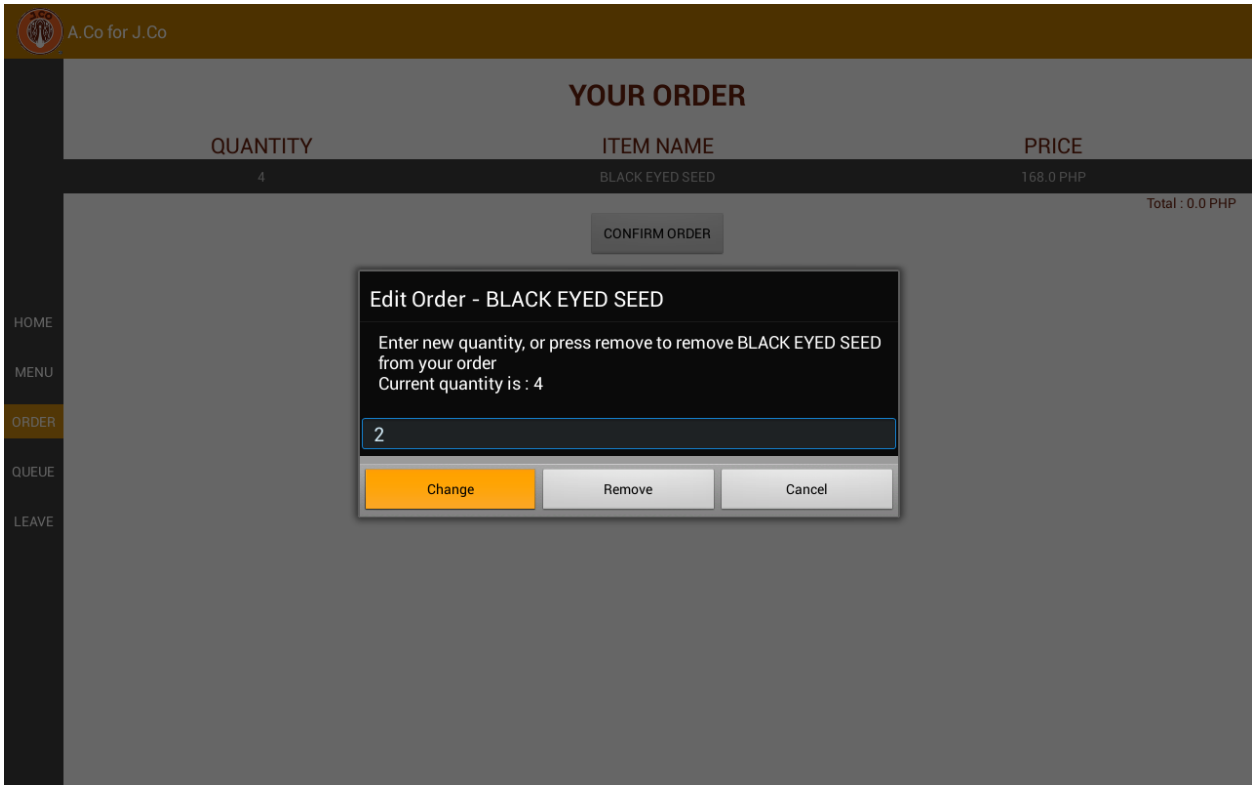

Figure 43: Walk In Customer Change Quantity of an Order Item

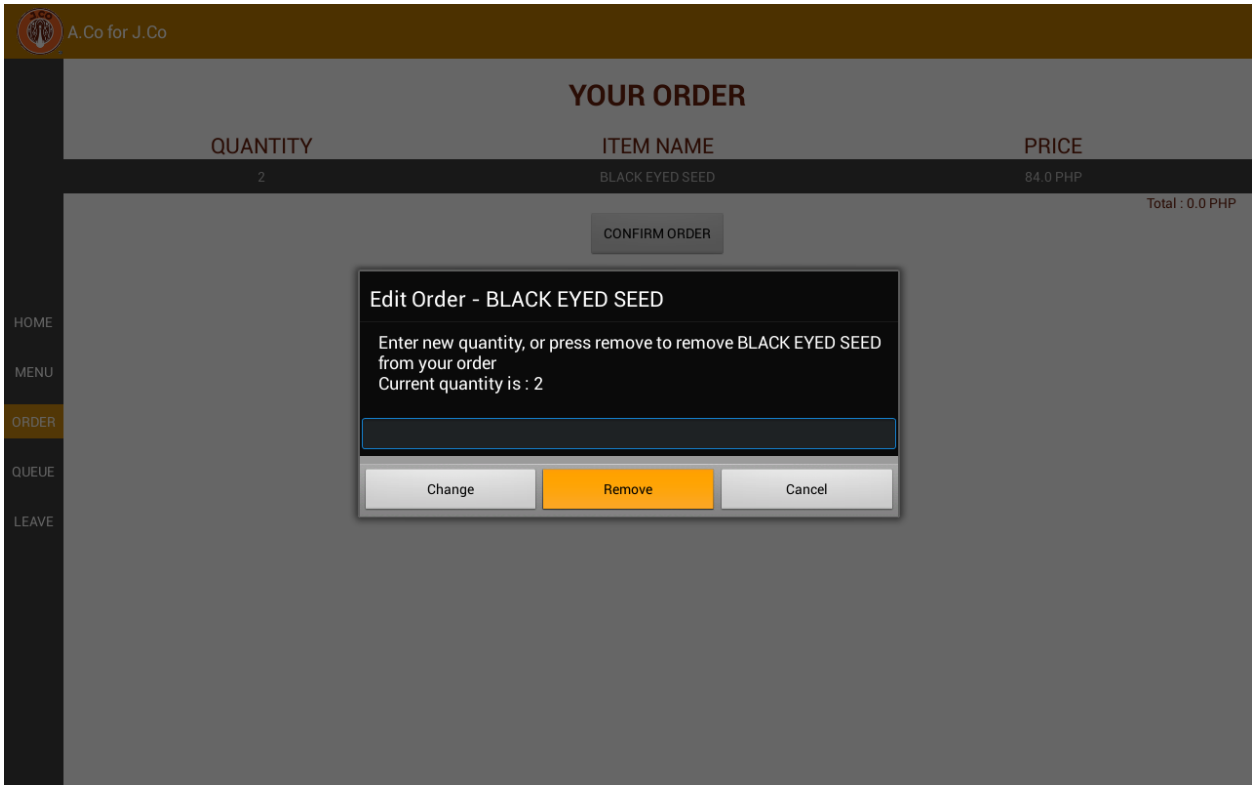

Figure 44: Walk In Customer Remove Quantity an Order Item

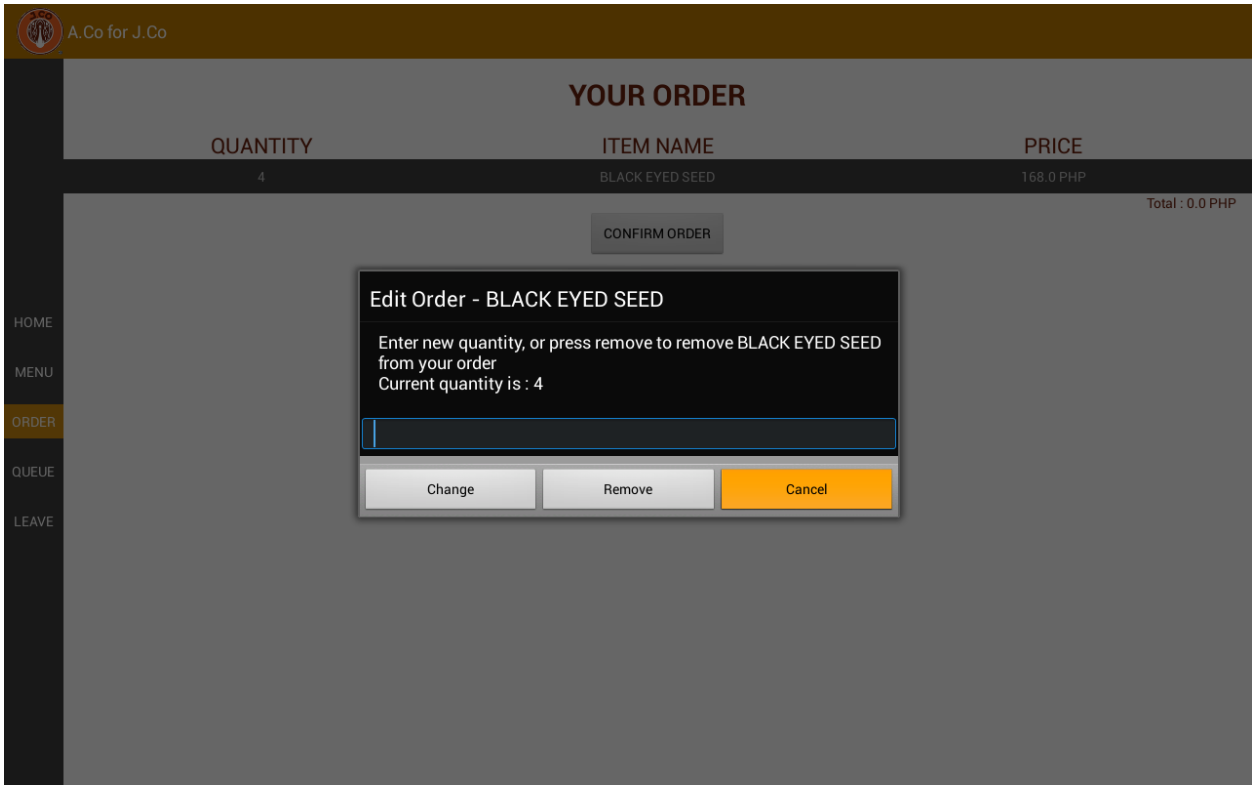

Figure 45: Walk In Customer Cancel Change to an Order Item

An Empty Order Screen, as shown in Figure 46, will not allow a Customer to Confirm his/her order. Trying to confirm an empty order screen will result to an alert.

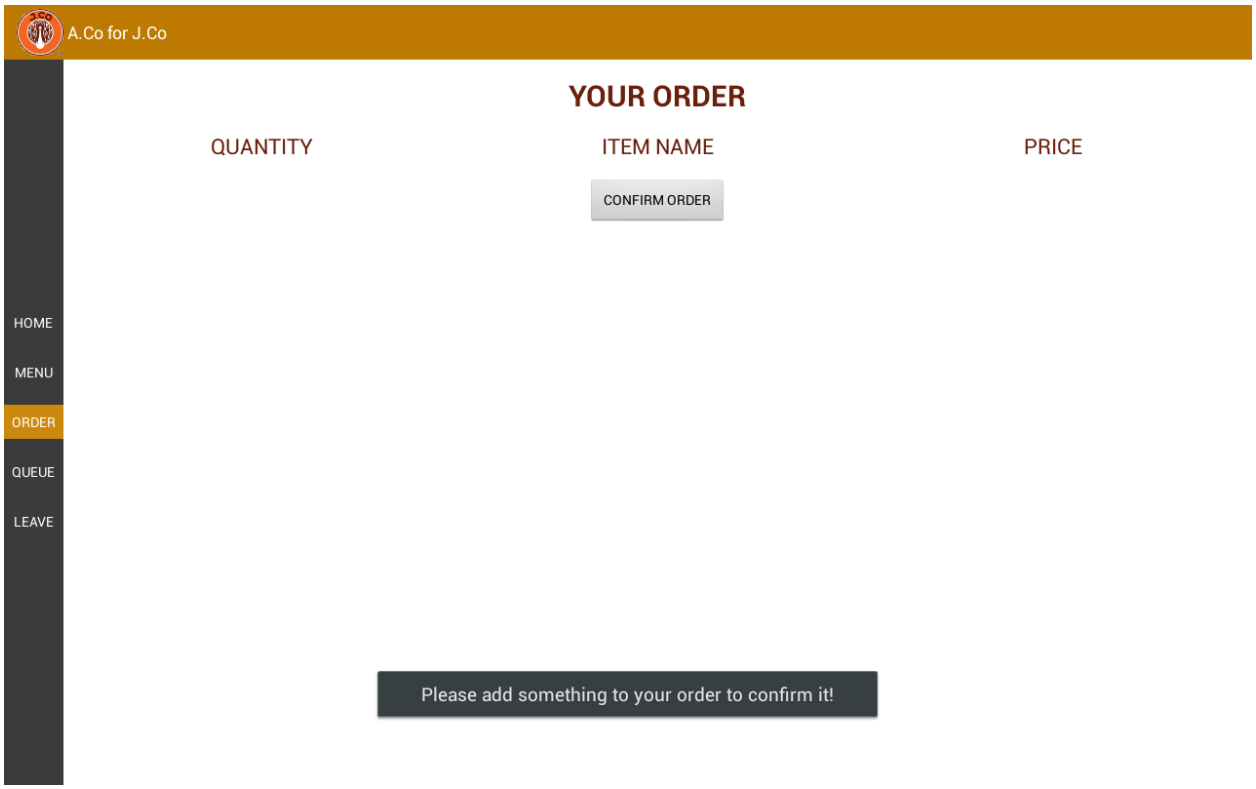

Figure 46: Walk In Customer Order Empty Alert

Upon confirming the orders list on the Orders Screen, the Customer will now be not allowed to modify his/her orders list anymore. He/she will be directed to a Receipt Screen which contains the summary of the items ordered, the quantities of each of the said items, the total prices of each of the said item, and the total amount incurred for the whole transaction, as shown in Figure 47. He/she will then be allowed to opt for the *End Session* button of the Customer's Order Screen to conclude the ordering process between the said Customer and the Store, as shown in Figure 48. This allows the Customer to relax on the J.Co customer waiting area while waiting for the completion of his/her orders. The *Leave* navigation tab navigates the customer out of the current ordering transaction too.

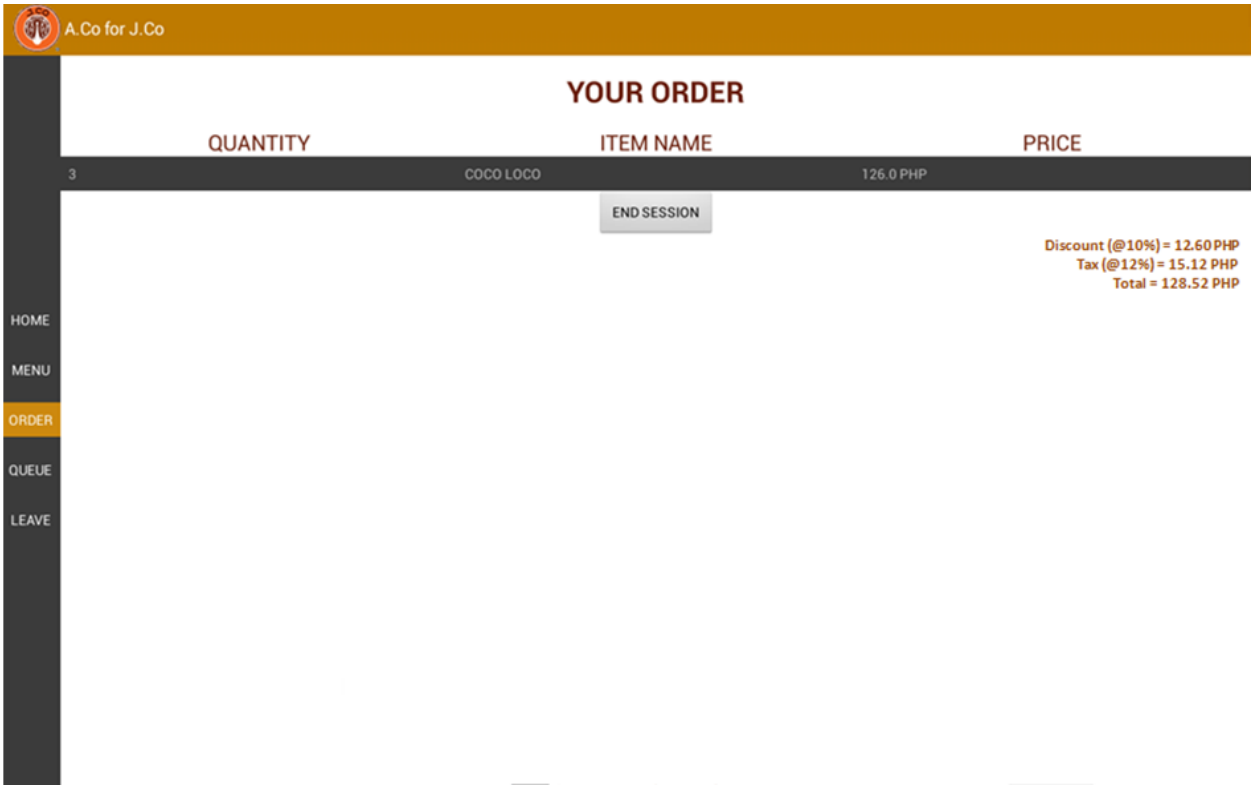

Figure 47: Walk In Customer Order Receipt Screen

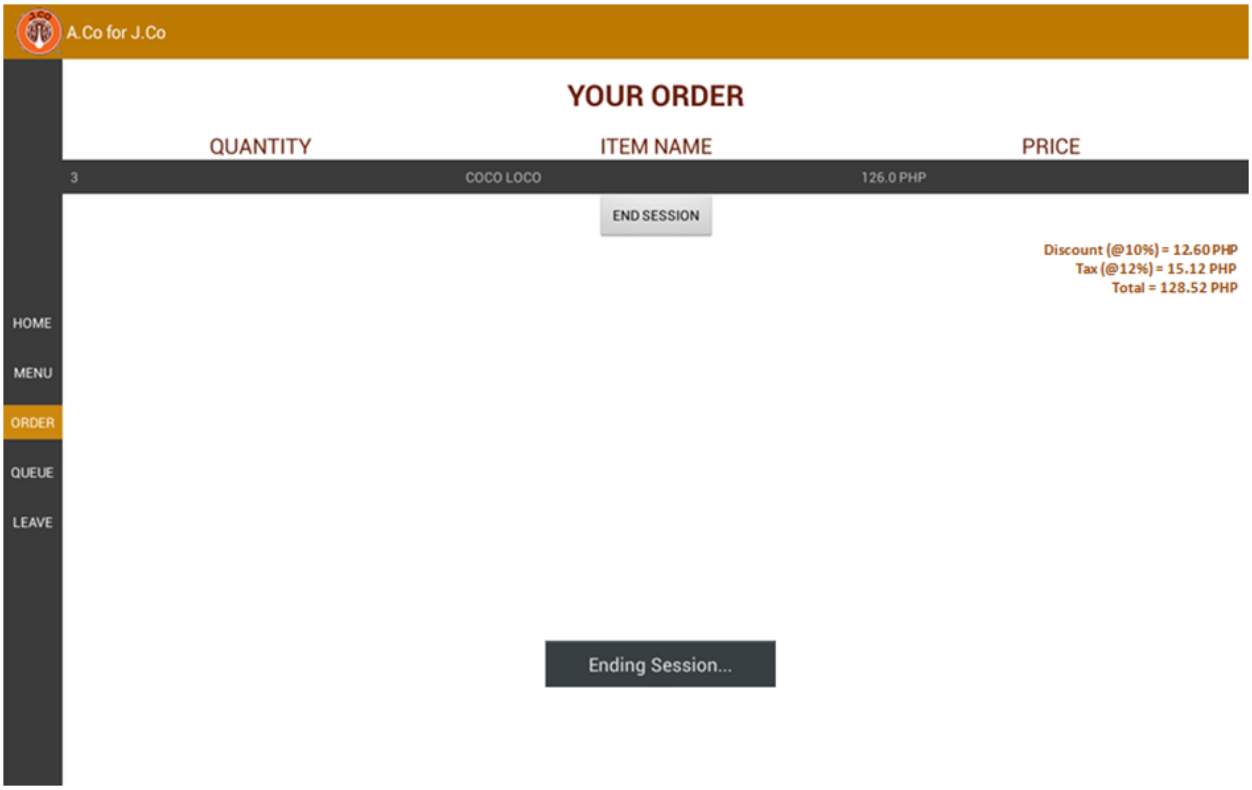

Figure 48: Walk In Customer End Session Button and Leave Navigation Tab

Upon summarizing the whole order list of the Customer on the Receipt Screen, the Queue Progress Screen, as shown in Figure 49, will now display the *Remaining Time Waiting* of the ongoing customer transaction and the updated *Currently Being Served* customer number. The *Queue* navigation tab navigates the customer to this Queue Progress Screen.

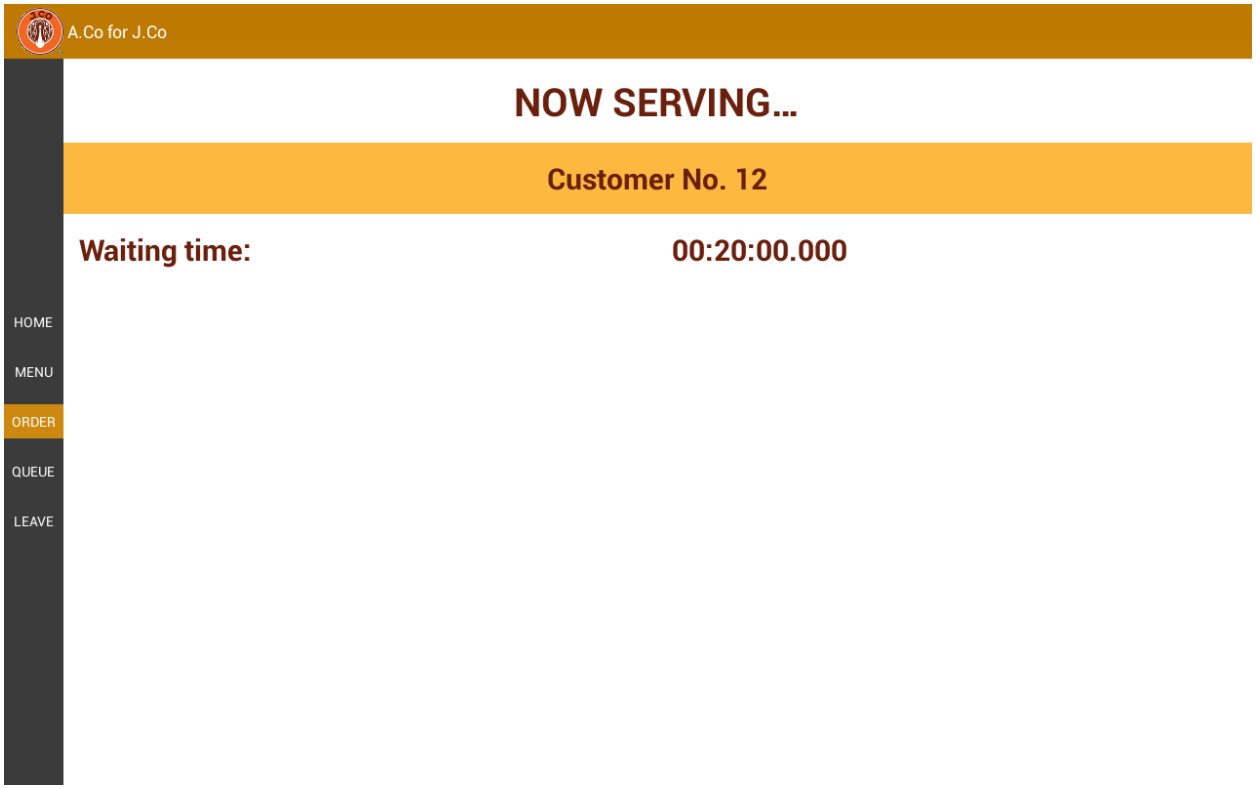

Figure 49: Walk In Customer Queue Progress Screen

Upon opting for the *End Session* option on the Receipt Screen, the tablet device will now display the Welcome Screen again. This concludes the ordering transaction of the last customer to operate on the tablet device. A new Customer will once again have the option of tapping on the *While You Wait* button and the *As You Roam* option. Opting for the *As You Roam* option will direct the Connected Customer, who has an Android device that has A.CO install on it, to a QR Code Screen, as shown in Figure 50. This QR Code Screen allows a Customer to capture a QR Code and be able to automatically register his/her self into the A.CO system, and to join the virtual queue.

Figure 50: QR Code Screen

C. Application Package for the Connected Customers

The Connected Customers Screen, as shown in Figure 51, provides a welcome and instruction screen for the Connected Customer who has an Android mobile device that has A.CO installed on it. The Connected Customer may tap on it to proceed.

Figure 51: Connected Customer Welcome Screen

Upon tapping on the display of Connected Customer Welcome Screen, the Connected Customer will be directed to the Capture Screen, as shown in Figure 52. This Capture Screen allows the Connected Customer to point the camera of his/her mobile device to the QR code displayed on the previously discussed Store Entrance Tablet, as shown in Figure 53.

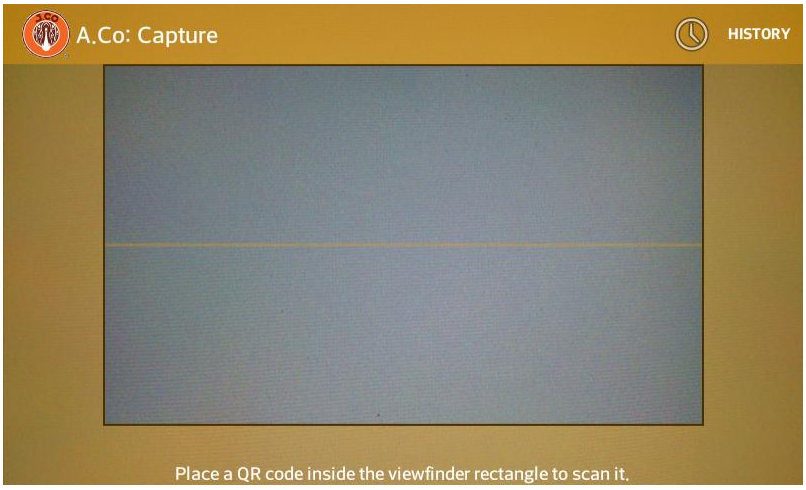

Figure 52: Connected Customer Capture Screen

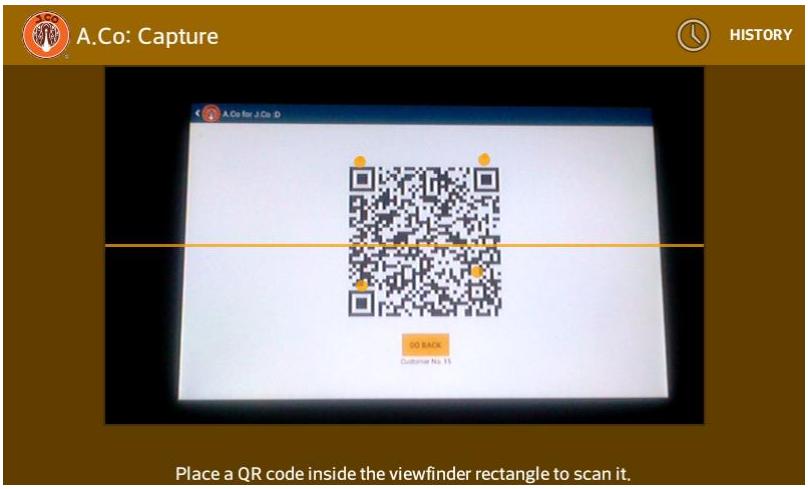

Figure 53: Connected Customer Capture Screen Capturing a QR Code

Upon capturing of the QR Code displayed on the store entrance tablet device, the Queue Number of the Connected Customer will be displayed and stored on the mobile device, as shown in Figure 54. This provides two options for the Connected Customer: first, by clicking the "*Order Now"* button, he/she will already be able to log in into the A.CO system and send order requests after being directed to the Customer Home Screen; and second, by navigating out of the Capture Screen, he/she will be able to let the code be stored in the mobile device and simply order at a later time.

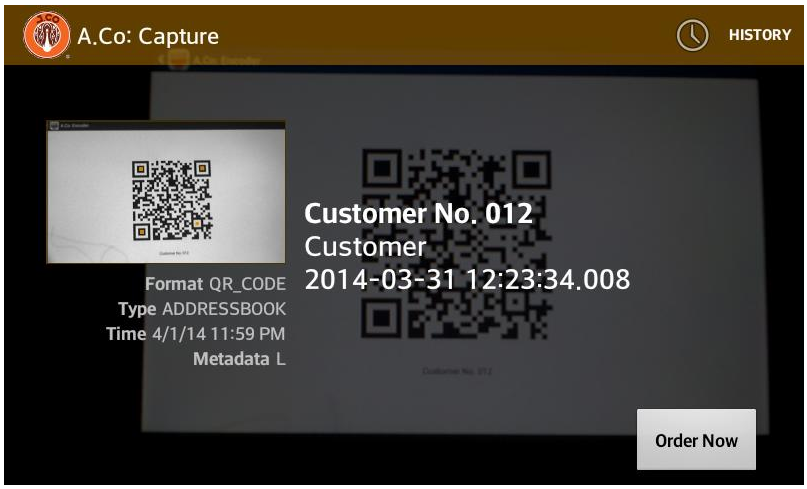

Figure 54: Connected Customer Captured QR Code

The mobile device has to have an internet connection for it to be able to log in a Connected Customer. Failure to comply with this will result to an error in network connection alert, as shown in Figure 55.

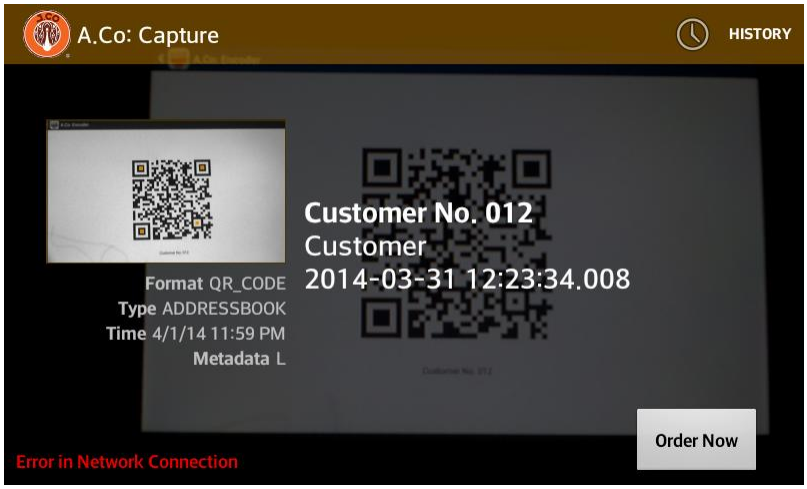

Figure 55: Connected Customer Captured QR Code Error in Network Connection Alert

Upon deciding to send an order request in the device, the Connected Customer can simply navigate again through the application's Connected Customer Welcome

Screen and through the Capture Screen. On the upper right corner of the screen, as shown in Figure 56, an option to view the Capture History can be tapped.

Figure 56: History Button

This History Screen, as shown in Figure 57, contains the list of the captured QR Codes of the Connected Customer. This functionality page provides an option for the customer to pick the captured QR Code where he/she will get his/her automatic log in connection from.

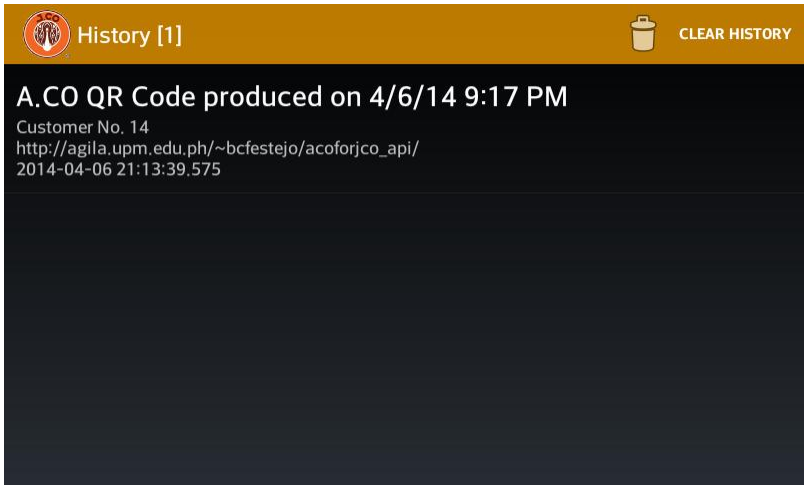

Figure 57: History Screen

A Connected Customer is allowed to delete a single QR Code History entry by long pressing the said History item and confirming its deletion, as shown in Figure 58 and Figure 59; or to delete all of the QR Code History entries by pressing the upper right

corner button and confirming its deletion on the dialog box, as shown in Figure 60 and Figure 61.

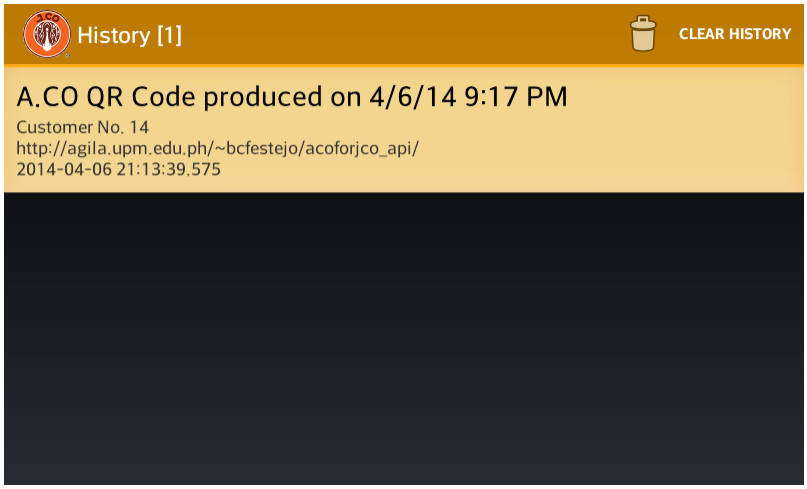

Figure 58: Delete Single History Item

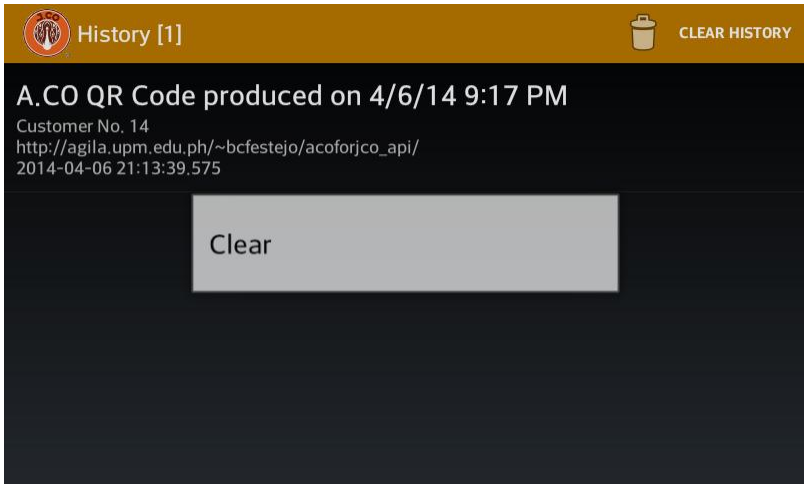

Figure 59: Confirm Deletion of Single History Item

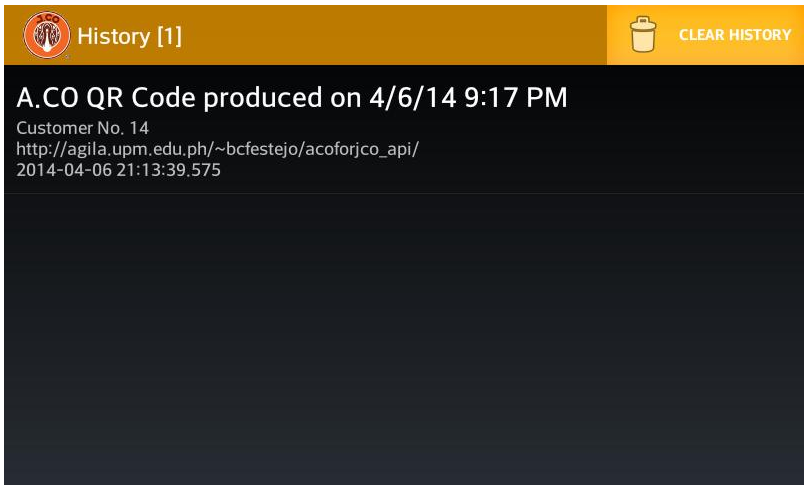

Figure 60: Delete All History Items

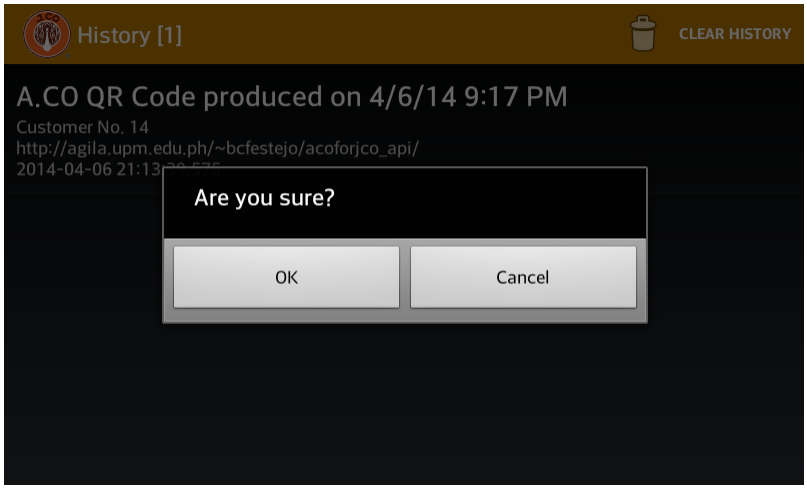

Figure 61: Confirm Deletion of All History Items

An Empty History Screen, as shown in Figure 62, will display an empty record of a QR code capture.

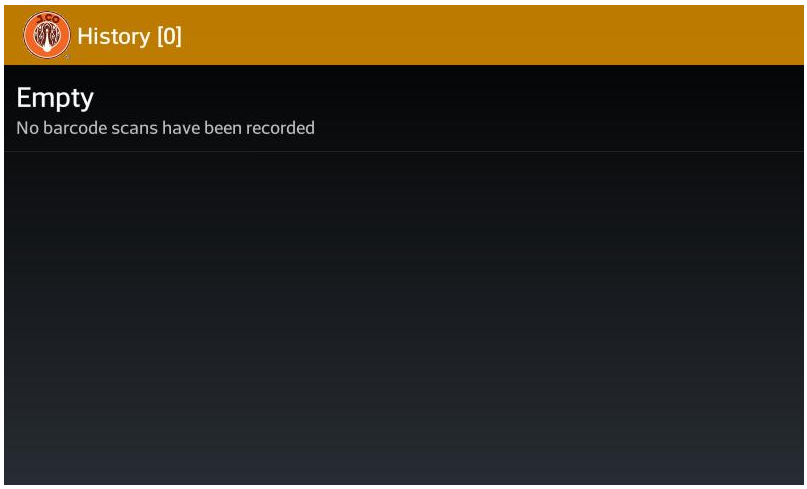

Figure 62: Empty History Screen

Upon clicking an existing record of a QR Code capture on the History Screen, the Connected Customer will be directed back to the captured QR Code screen, as shown in Figure 63. This screen provides an option to click the "*Order Now*" button and already be able to log in into the A.CO system.

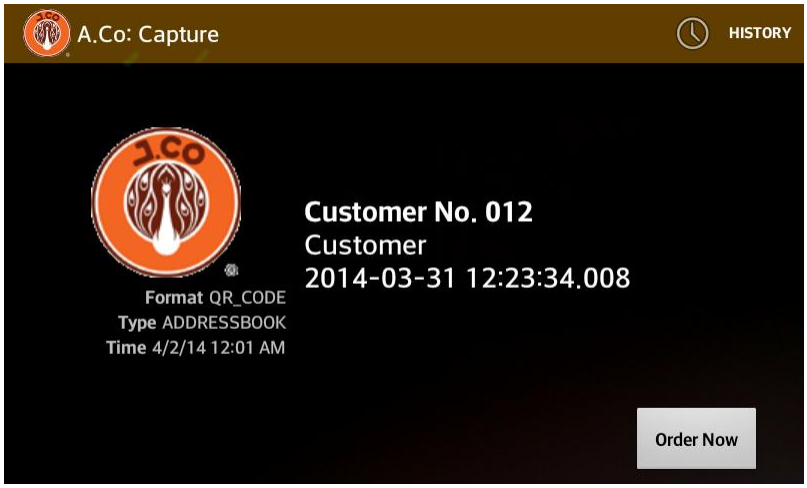

Figure 63: Connected Customer Viewing a Stored QR Code

The mobile device will simulate the registration of the Connected Customer, as shown in Figure 64, to provide a seamless login.

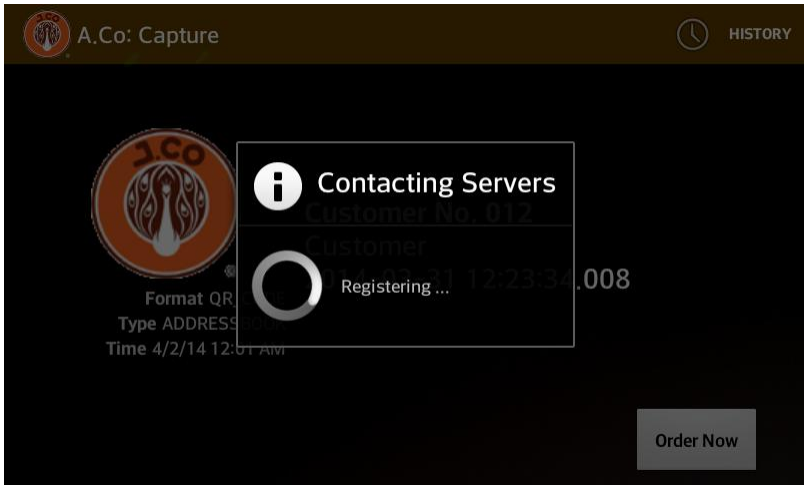

Figure 64: Connected Customer Simulated Registration

After a successful registration, the Connected Customer will then be directed to a Connected Customer Dashboard Screen, as shown in Figure 65. He/she will be greeted by his/her queue number and will have two options: first, by clicking the "*Let me order!"* button, he/she will be able to send order requests after being directed to the Customer Home Screen; and second, by clicking the "*Loge me out!"* button, he/she will be able to simply log out of the system.

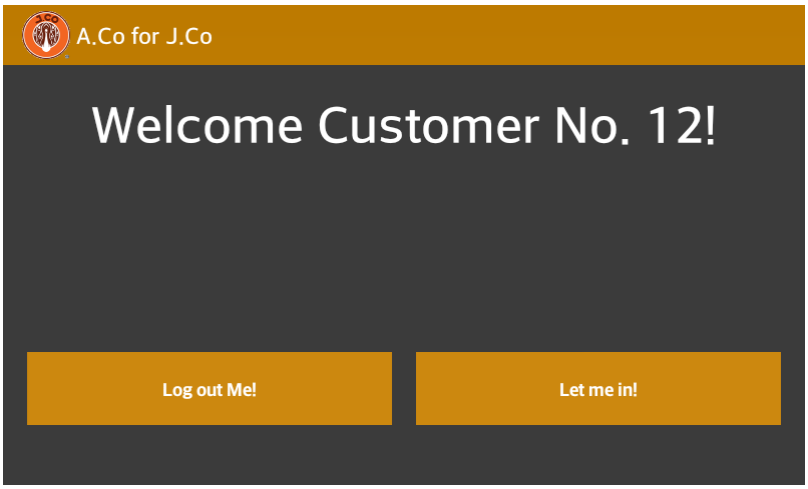

Figure 65: Connected Customer Dashboard Screen

Upon choosing the *Let Me Order* option in the Dashboard screen, the Connected Customer will be directed to the Home Screen of the system, as shown in Figure 66. This screen is similar in structure to that of all the other users' which is divided into three parts: the navigation tabs, which lets the Connected Customer in to the other ordering functionalities of the system; the header, which states the current navigation position of the Connected Customer with respect to the ordering functionality pages of the system; and the body, which changes accordingly as the Connected Customer navigates his/her self through the ordering functionalities of the system. The *Home* navigation tab navigates the customer to this Home Screen.

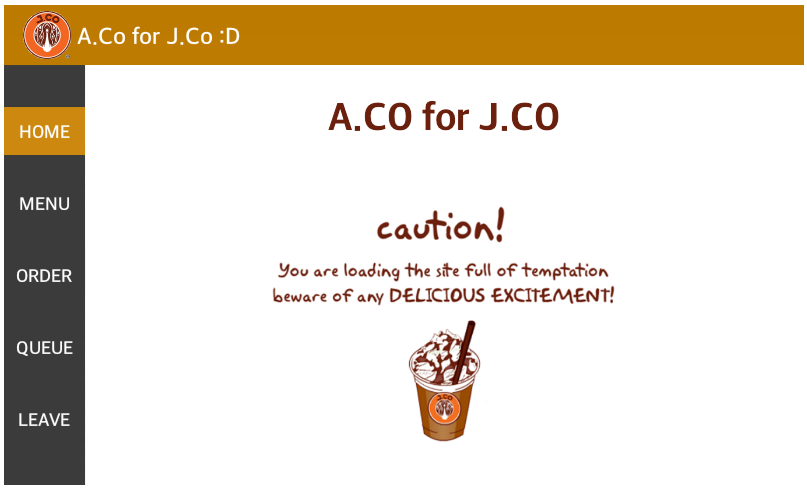

Figure 66: Connected Customer Home Screen

The *Menu* navigation tab navigates the Connected Customer to the Connected Customer Menu Screen, as shown in Figure 67. This Connected Customer Menu Screen contains the same five different categories of products offered by J.Co Donuts and Coffee to Walk-In Customers who opted for the entrance tablet for order inputting. This functionality page provides options for the Connected Customer to pick which category he/she will browse and order from.

Figure 67: Connected Customer Menu Screen

The Connected Customer Menu List Screen, as shown in Figure 68, contains the food catalog of J.Co Donuts and Coffee with respect to the selected food category from the Connected Customer Menu Screen. This functionality page allows the Connected Customer the same privileges as that of the Walk-In customers which is to browse the items under the category he/she has chosen. He/she is allowed to pick an item to view its product description and to put it on his/her order list.

Figure 68: Connected Customer Menu List Screen

Upon picking an item on the Connected Customer Menu List Screen, the Connected Customer will be directed to the Product Description Screen, as shown in Figure 69. This screen allows the Connected Customer to view the detailed information of the said item and to specify the quantity of the said item he/she would like to order. This order list will then be reflected to the Connected Customer's Order Screen after finishing off the order inputting.

Figure 69: Connected Customer Product Description Screen

After a successful ordering attempt, the Connected Customer will then be directed back to the Connected Customer Menu Screen, along with an addition to the order list alert, as shown in Figure 70.

Figure 70: Connected Customer Order Addition Alert

The *Order* navigation tab navigates the Connected Customer to the Order Screen, as shown in Figure 71. This Connected Customer's Order Screen contains the summary of the items picked by the connected customer to be put in his/her order list. This ordering functionality page gives the Connected Customer two options: first, by clicking the *Confirm Order* button, the list of items ordered by the connected customer will be sent to the Store Personnel for accomplishment; and second, by simply going back, the customer will directed to the Menu Screen and will be allowed to order some more.

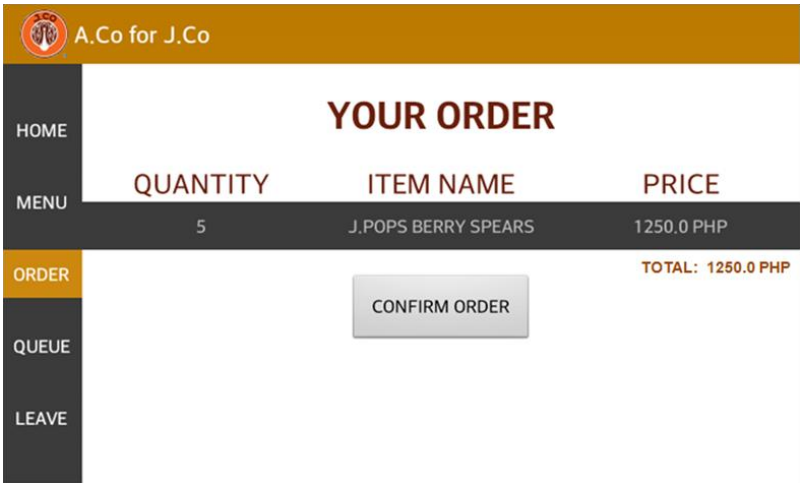

Figure 71: Connected Customer's Order Screen

A Connected Customer is allowed to Add, Edit, or Delete an Order item, as shown in Figure 72, Figure 73 and Figure 74.

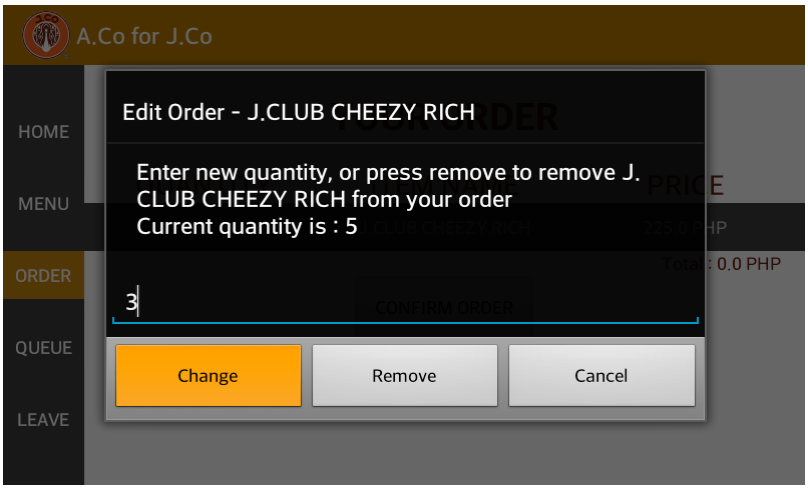

Figure 72: Connected Customer Change Quantity of an Order item

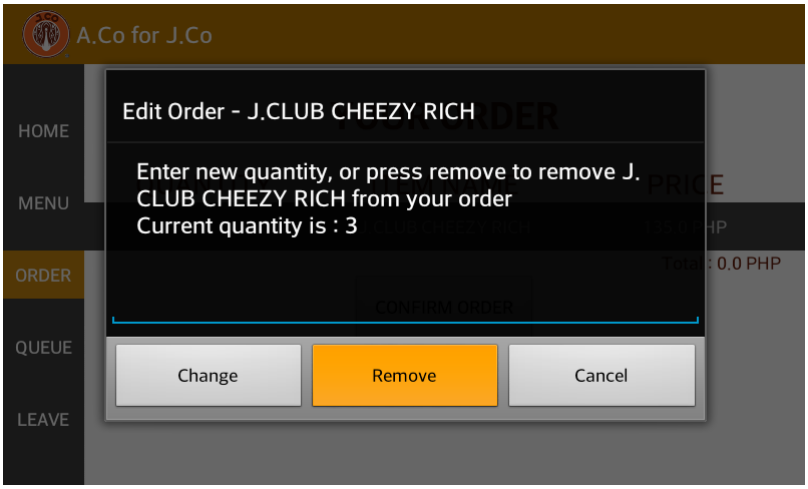

Figure 73: Connected Customer Remove Quantity of an Order item

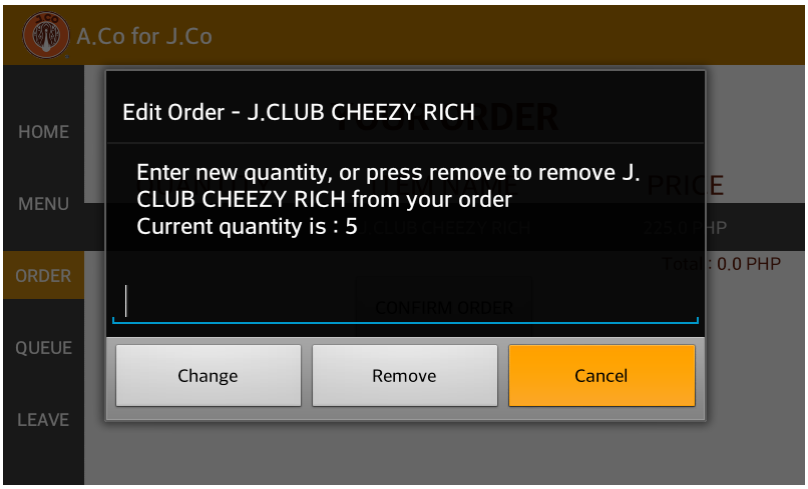

Figure 74: Connected Customer Cancel Change to an Order item

An Empty Order Screen, as shown in Figure 75, will not allow a Connected Customer to Confirm his/her order. Trying to confirm an empty order screen will result to an alert.

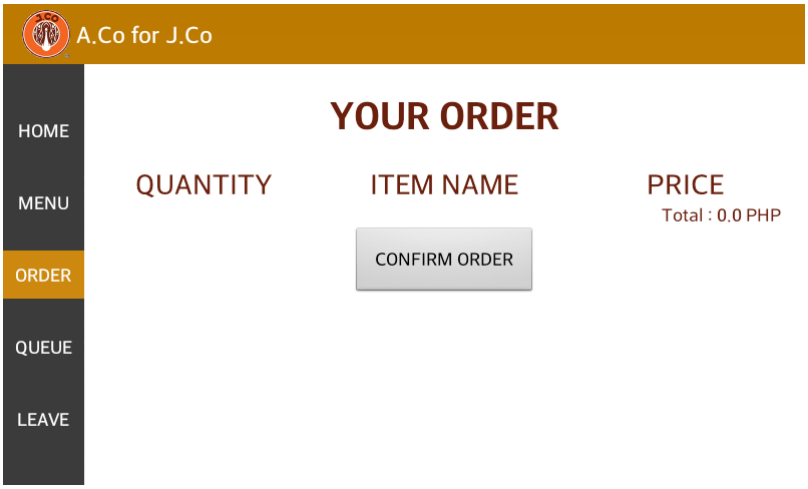

Figure 75: Connected Customer Order Empty Alert

Upon confirming the orders list on the Orders Screen, the Connected Customer will now be not allowed to modify his/her orders list anymore. He/she will be directed

to a Receipt Screen which contains the summary of the items ordered, the quantities of each of the said items, the total prices of each of the said item, and the total amount incurred for the whole transaction, as shown in Figure 76. He/she will then be allowed to opt for the *End Session* button of the Connected Customer's Order Screen to conclude the ordering process between the said Connected Customer and the Store, as shown in Figure 77. This allows the Connected Customer to relax on the J.Co customer waiting area or to roam around outside the J.Co premises while waiting for the notification on his/her mobile device of the completion of his/her orders. The *Leave* navigation tab navigates the customer out of the current ordering transaction too.

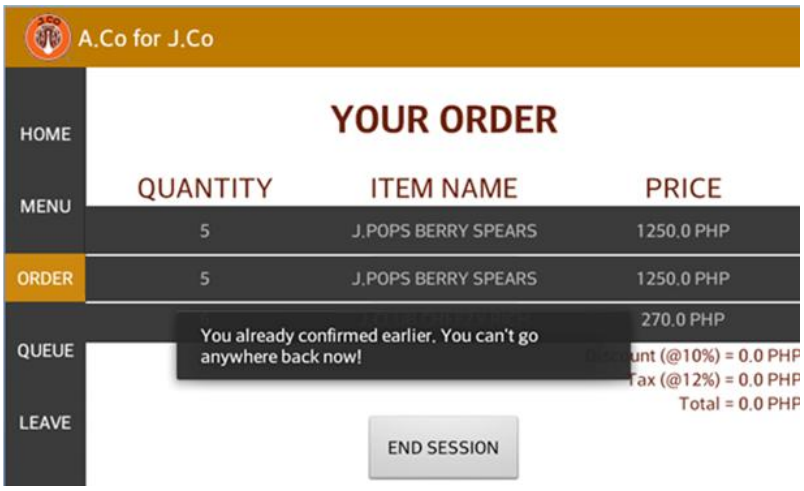

Figure 76: Connected Customer Order Receipt Screen

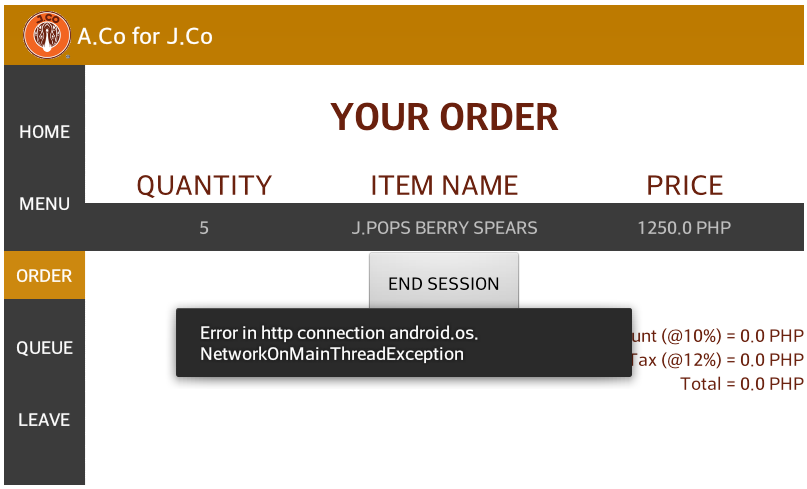

Figure 77: Connected Customer End Session Button and Leave Navigation Tab

Upon summarizing the whole order list of the Connected Customer on the Receipt Screen, the Queue Progress Screen, as shown in Figure 78, will now display the *Remaining Time Waiting* of the Connected Customer the updated *Currently Being Served* customer number. The *Queue* navigation tab navigates the customer to this Queue Progress Screen.

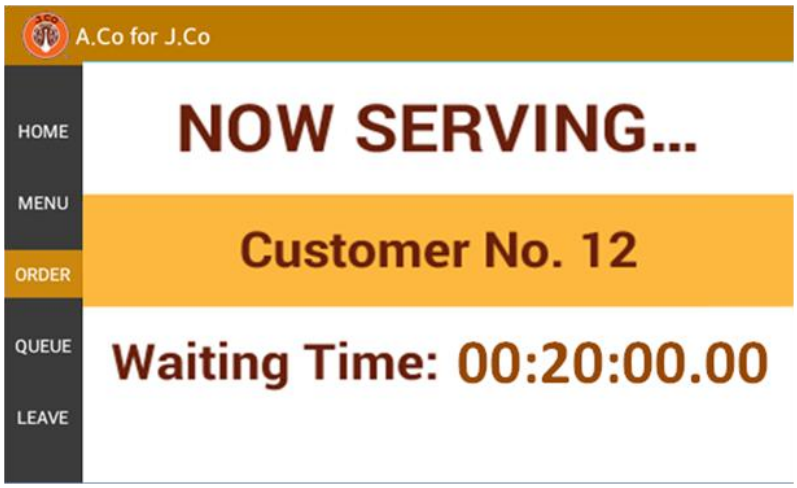

Figure 78: Connected Customer Queue Progress Screen

Upon opting for the *End Session* option on the Receipt Screen, the mobile device will now display the *Waiting Time Remaining Update*, as shown in Figure 79. This concludes the ordering transaction of the connected customer.

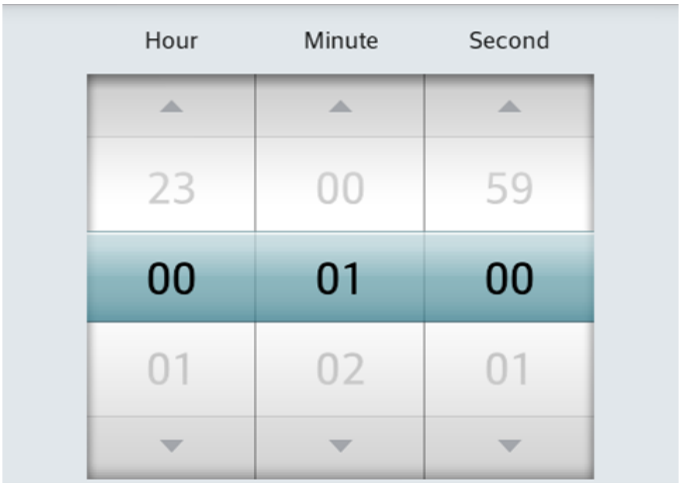

Figure 79: Connected Customer Waiting Time Remaining Update

Upon ending of the Connected Customer's time wait, the mobile device will now notify the said Customer of the order's completion, as shown in Figure 80. This allows the Connected Customer to pay and pick up his/her orders.

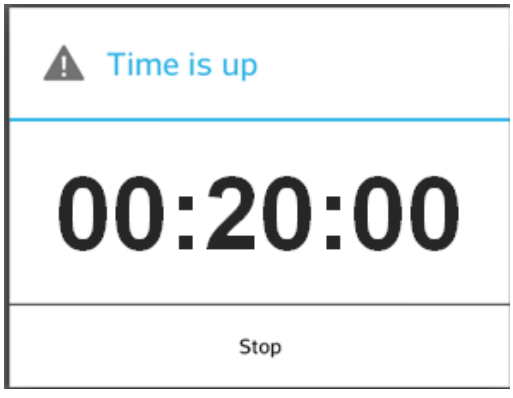

Figure 80: Connected Customer Order Completion Notification

VI. Discussions

The Android for Customer Organization (A.CO) is a Queue Management System (QMS) that effectively manages customers in a virtual queue. It has two major application packages that cater to four types of users.

The first major application package, where most of the functionalities were defined, is the one to be installed on an Android tablet device. This is made to be at the direct disposal of Counter, Cashier, and Customers without the installed application. This allows the Counter and Cashier personnel to register themselves in the system and to login and perform their operational duties for J.Co. This application package also directly serves as the login medium for the Customers who don't have A.CO on their own mobile devices in the way that it allows these customers to simply manually login and input their orders. As for the Customers who has the application already installed in their own mobile devices, this A.CO application package serves as their login access in the way that it releases a system generated QR code to be captured by the aforementioned customers. This gives customers on-the-go a more convenient option for ordering and waiting for their orders to be completed.

The other major application package, where most of the remote access activities were implemented, is the one to be installed on an Android mobile device. This was developed with a Customer-on-the-go's needs in mind. After capturing the QR code generated by A.CO's tablet installation package, these customers can then proceed on doing

their other tasks (like roaming around and going to other places). They have the option to simply send their order requests on a more convenient time for them. Also, from this A.CO installation, they have the privilege of directly knowing of their order's completion regardless of where they are.

For the user types, the system has the Counter Staff who is in charge of monitoring the virtual queue of incoming and ordering J.Co customers. He/she is bound to supervise the queue progress as well as the order requests of the aforementioned customers. He/she have a wider scope of access on the system. On the other hand, the Cashier, who has the same system privileges as that of the Counter Staff's, is more dedicated on checking out customer orders and managing the products requested in exchange of an equivalent amount of payment. These two types of user work together on the store side to deliver the product on time to a waiting customer and to notify the said customer of the completion of their orders.

Conversely, the system has the Customers who either has the application on their own utilization or has the A.CO tablet installation to accommodate them. Both have the privileges of seamlessly registering and logging in into A.CO, being put into its virtual queue, sending orders virtually, and being notified of their order's completion conveniently.

VII. Conclusions

The Android for Customer Organization (A.CO) is a tool for customer queue management that is specifically designed to be integrated into J.Co Donuts and Coffee's business processes. It promotes effective customer virtual queue organization by utilizing the capabilities of the rapidly growing market of smartphones, especially the ones running on the Android operating system versions. With this system, the customers need not be engaged in an anarchic customer queuing situation with other misdirected customers eager to be served as fast as they can. A.CO has provided efficient handling of customer bulks as well as their orders thus providing easy and quick production and completion of transactions. Moreover, by providing central points of access and accommodation for the different users, the system then answers the problem of the impatient customer's queuing behaviors all the while catering the easing up of the counter personnel's tasks.

VIII. Recommendations

The Android for Customer Organization (A.CO) is an Android application which enables J.Co's store personnel to monitor the bulk of customer orders. It also enables J.Co's customers to join a virtual queue and to send their orders in a more organized manner. Its modules can be extended to accommodate more scenarios. Its queue display module can be improved by providing the other functionalities of a Heads Up Display such as projection of advertisements and infomercials to inform people of upcoming store events, of news of new product launches, or of announcements of new product promos. Its order confirmation module can be extended into including a customer feedback form to allow customers to ask questions about the products or the J.Co establishment itself, to give suggestions for further improvement of their experience, or to simply share their experience on J.Co's products and services.

Furthermore, the system can also be expanded to share its Android smartphone users' functionalities to other smartphones that have a different mobile operating system.

IX. Bibliography

[1] Horaga, Andrea Elok, Otniel Ozora, and Stiefanie. *The Factors of Brand Image Which Influence Customer Loyalty of J.CO*. Paper: Retail Business and Academic Writing. President University, 2013. *Academia.edu*. Web. 16 Mar. 2014. <http://www.academia.edu/5904356/Factors_of_Brand_Image_Which_Influence_Customer_Loyalty_of_J.Co>.

[2] Lopez, Tony. "J.Co: The Caviar of Donuts." *Manilatimes.net*. The Manila Times, 28 Aug. 2013. Web. 08 Mar. 2014. <http://manilatimes.net/j-co-the-caviar-of-donuts/34263/>.

[3] Schwartz, Barry. "Waiting, Exchange, and Power: The Distribution of Time in Social Systems."*American Journal of Sociology* 79. No. 4 (1974): 841-70. *JSTOR*. Web. 13 Jan. 2014. <http://www.jstor.org/discover/10.2307/2776346?uid=3738824&uid=2134&uid=2&uid=70&uid=4&sid=21103543675933>.

[4] Berner, Jeff. *The Joy of Working from Home: Making a Life While Making a Living*. San Francisco: Berrett-Koehler, 1994. *Google Books*. Web. 13 Jan. 2014. <http://books.google.com.ph/books?id=Xu__e7mtM8sC&printsec=frontcover&source=gbs_ge_summary_r&cad=0#v=onepage&q&f=false>.

[5] Gray, Kevin, PhD, LLD, DCL, FBA. "The Legal Order of the Queue." 2007. TS Q7RTF. The London School of Economics and Political Science, University of Cambridge. *The London School of Economics and Political Science Department of Law*. Web. 13 Jan. 2014. <http://www.lse.ac.uk/collections/law/projects/techniquesofownership/tech-gray.pdf>.

[6] Beasley, John. "OR-Notes." *Brunel University West London*. Web. 13 Jan. 2014. <people.brunel.ac.uk/~mastjjb/jeb/or/queue.html>.

[7] Eshete, Addissu, and Yuming Jiang. *On the Flow Fairness of Aggregate Queues*. Thesis. Norwegian University of Science and Technology, 2011. *IEEE*. Web. 8 Mar. 2014. <http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.419.6601&rep=rep1&type=pdf>.

[8] Browne, James C. "MEN170: Systems Modeling and Simulation." *Introduction to Queuing Models Courtesy of QUT*. 14 Jan. 2003. Web. 14 Jan. 2014. <http://www.cs.utexas.edu/~browne/cs380ns2003/Papers/SimpleQueuingModelspdf.pdf>.

[9] Hurst, Bethany, Jodi-Kay Edwards, and Nicolas Bross. "Queuing Theory/Waiting Line Analysis." Reading. Production/Operations Management. 14 Jan. 2014. *St. Norbert College*. Web. 14 Jan. 2014. <home.snc.edu/eliotelfner/333/Team%204%20Queuing%20Analysis.ppt>.

[10] Buell, Ryan W., and Michael I. Norton. "Think Customers Hate Waiting? Not So Fast..."*Harvard Business Review Idea Watch* 89. No. 5 (2011). *Harvard Business School Faculty and Research*. Web. 13 Jan. 2014. <http://www.people.hbs.edu/mnorton/buell%20norton%20hbr.pdf>.

[11] Utami, Dina. *What Do People Do in the Waiting Room?* Rep. Northeastern University College of Computer and Information Science. Web. 13 Jan. 2014. <http://www.ccs.neu.edu/home/dinau/hw3.pdf>.

[12] Hargittai, Eszter, and Su Jung Kim. "The Prevalence of Smartphone Use Among a Wired Group of Young Adults." 2010. WP-11-01. Rpt. in *Northwestern University Institute for*

Policy Research Working Paper Series. IL: Northwestern University, 2010. *Northwestern University Institute for Policy Research Working Papers*. Web. 13 Jan. 2014. <http://www.ipr.northwestern.edu/publications/docs/workingpapers/2011/IPR-WP-11-01.pdf>.

[13] "Barcodes and QR Codes." *Stanford School of Medicine Information Resources & Technology*. Web. 14 Jan. 2014. <http://med.stanford.edu/irt/teaching/barcode.html>.

[14] Bautista, Tisha C.. "The Sweet Life." *Philstar.com*. 9 Apr. 2013. Web. 8 Mar. 2014. <http://www.philstar.com/health-and-family/2013/04/09/928375/sweet-life>.

[15] Tudenhöfner, Eduard. *Integration of Performance Management into the Application Lifecycle*. Thesis. Hochschule Für Technik (HFT Stuttgart), Stuttgard, Deutschland, 2011. Hamburg: Diplomica (R) Verlag GmbH, 2011. ISBN: 978-3-8428-2047-0. *Books.google.com*. Google Books, 2011. Web. 8 Mar. 2014. <http://books.google.com.ph/books?id=8mJgAQAAQBAJ&printsec=frontcover&source=gbs_ge_summary_r&cad=0#v=onepage&q&f=false>.

[16] DevonshireElite. "J.Co Sometimes Good Sometimes Bad." *Sett.com*. Apr. 2013. Web. 8. Mar. 2014. <http://sett.com/bertiebahru>.

[17] Franca, Iroegbu Ngozi, and Okoro Agu. "Queuing Model for Effective Customer Service Delivery in the Banking Industry: A Study of Union Bank PLC in Enugu Metropolis." *International Journal of Management & Information Technology* 3rd ser. 7 (2013): 1142-154. *Council for Innovative Research*. Dec. 2013. Web. 15 Jan. 2014. <http://cirworld.com/index.php/ijmit/article/viewFile/733118/2176>.

[18] "Dashboards on Platform Versions." *Developers.android.com*. Google Play Store, 3 Mar. 2014. Web. 8 Mar. 2014. <https://developer.android.com/about/dashboards/index.html>.

[19] "How Do QR Codes Work?" *CreateQRCodes.org*. Web. 16 Jan. 2014. <http://www.createqrcodes.org/how-do-qr-codes-work.html>.

[20] Gosha, Kinnis K. "QueueAdmin: The Effects of an Advance Queue Management System on Barbershop Administration." Thesis. Auburn University Alabama, 2007. *Auburn University Theses and Dissertations* (2007). *AUETD*. Web. 17 Jan. 2014. <http://etd.auburn.edu/etd/bitstream/handle/10415/74/GOSHA_KINNIS_25.pdf>.

[21] Onlinet. "Onlinet - Queue Management, Queuing System." Web. 12 Jan. 2014. <http://www.onlinet.co.uk/hu/subContent/customerManagementSystems>.

[22] "BPI Express Assist Online." *BPI*. Web. 14 Jan. 2014. <https://www.mybpimag.com/index.php?option=com_content&view=article&id=1016&Itemid=1180>.

[23] Lu, Yina, Andrés Musalem, Marcelo Olivares, and Ariel Schilkurt. "Measuring the Effect of Queues on Customer Purchases." Thesis. Stanford University Graduate School of Business, 2012. 27 Sept. 2012. Web. 16 Jan. 2014. <www.gsb.stanford.edu/sites/default/files/documents/YLu_Paper.pdf>.

[24] "Qminder Reviews." *Qminder*. Web. 19 Feb. 2014. <https://play.google.com/store/apps/details?id=com.qminderapp.mobile&hl=en>.

[25] Aguinaga, Salvador, and Christian Poellabauer. *Method for Privacy-Protecting Display and Exchange of Emergency Information on Mobile Devices*. Poster Paper. Department of

Computer Science University of Notre Dame, May 2012. Web. 17 Jan. 2014. <http://www3.nd.edu/~saguinag/baja/qrcscanposterpaper.pdf>.

[26] Probst, Ali. "The Expectations of Quick Response (QR) Codes in Print Media: An Empirical Data Research Anthology." *UW-La Crosse Journal of Undergraduate Research* XV (2012): 1-13. *University of Wisconsin - La Crosse*. Dr. Stephen Brokaw. Web. 17 Jan. 2014. <http://www.uwlax.edu/urc/JUR-online/PDF/2012/probst.ali.pdf>.

[27] Queuelo. "Queuelo - Virtual Queue Management App." Web. 15 Jan. 2014. <http://queueloapp.appspot.com/>.

[28] Marco Venier. "Double Degree IT: Android Application for Queues Management.wmv." Web. 15 Jan. 2014. <http://www.uni-klu.ac.at/tewi/ict/nes/pc/10197.htm>.

[29] Perez, Sarah. "For The Young, Smartphones No Longer A Luxury Item." *TechCrunch*. *TechCrunch Mobile*. TechCrunch, 20 Feb. 2012. Web. 17 Jan. 2014. <http://techcrunch.com/2012/02/20/for-the-young-smartphones-no-longer-a-luxury-item/>.

[30] Hargittai, Eszter. and Kim, Su Jung. The Prevalence of Smartphone Use Among a Wired Group of Young Adults. Institute for Policy Research Northwestern University Working Paper Series. WP-11-01 December 2010 <http://www.ipr.northwestern.edu/publications/workingpapers/2011/IPR-WP-11-01.pdf>

[31] "USA to Add 80 Million New Smartphone Users by 2011." *TwitTown: The Apps and Widgets Community*. TwitTown, 26 Mar. 2010. Web. 17 Jan. 2014. <http://twittown.com/mobile/mobile-blog/usa-add-80-million-new-smartphone-users-2011>.

[32] Etner, Roger. *Smartphones to Overtake Feature Phones in U.S. by 2011*. Rep. The Nielsen Company, 26 Mar. 2010. Web. 17 Jan. 2014. <http://www.nielsen.com/us/en/newswire/2010/smartphones-to-overtake-feature-phones-in-u-s-by-2011.html>.

[33] G, Nick. "Why Android Is Where It's at for CS Students." Web log post. *Get Real About Your Engineering or Computer Science Future in Oregon*. Oregonstate.edu, 4 Feb. 2011. Web. 17 Jan. 2014. <http://blogs.oregonstate.edu/getreal/2011/02/android-where-its-at-for-cs-students>.

[34] *Official ZXing (Zebra Crossing) Project Home*. *Github.com*. Web. 20 Feb. 2014. <https://github.com/zxing/zxing>.

[35] Sundarapandian, Vaidyanathan. "7. Queueing Theory." *Probability, Statistics and Queueing Theory*. ISBN 8120338448. PHI Learning. 2009. *Google Books*. Web. 19 Jan. 2014. <http://books.google.com.ph/books?id=9oUS6BBJkCYC&printsec=frontcover#v=onepage&q&f=false>.

[36] Stallings, William. "Queuing Analysis." Lecture. Tele 302. University of Otago Dunedin, New Zealand. *Telecommunications Programme*. Web. 18 Jan. 2014. <http://www.telecom.otago.ac.nz/tele302/ref/Stallings_QT.pdf>.

[37] Li, Xueping, PhD. "Queueing Models." Lecture. Knoxville, Tennessee. 30 Nov. 2008. *Courses IE 406 S07*. Web. 17 Jan. 2014. <http://iiesl.utk.edu/Courses/IE406%20S07/Slides/Queueing%20Models.pdf>.

X. Appendix

acoforjco_api\include\config.php

```
<?php

/**
 * Database config variables
 */

define('DB_HOST', "localhost");
define('DB_USER', "QueueIntendance");
define('DB_PASSWORD', "sirchuathegreat");
define('DB_DATABASE', "QueueIntendance");

?>
```

acoforjco_api\include\DB_Connect.php

```
<?php

class DB_Connect {

    // constructor
    function __construct() {

    }

    // destructor
    function __destruct() {
        // $this->close();
    }

    // Connecting to database
    public function connect() {
        require_once __DIR__ . '/config.php';
        // connecting to mysql
        $con = mysql_connect(DB_HOST, DB_USER, DB_PASSWORD);
        // selecting database
        mysql_select_db(DB_DATABASE);

        // return database handler
        return $con;
    }

    // Closing database connection
    public function close() {
        mysql_close();
    }
}

?>
```

acoforjco_api\include\DB_Functions.php

```
<?php

class DB_Functions {

    private $db;

    //put your code here
    // constructor
    function __construct() {
        require_once __DIR__ . '/DB_Connect.php';
        // connecting to database
        $this->db = new DB_Connect();
        $this->db->connect();
    }
```

```
    // destructor
    function __destruct() {

    }

    /**
     * Adding new user to mysql database
     * returns user details
     */
    public function storeUser($fname, $lname, $identifier, $description, $password) {
        $uuid = uniqid('', true);
        $hash = $this->hashSSHA($password);
        $encrypted_password = $hash["encrypted"]; // encrypted password
        $salt = $hash["salt"]; // salt
        $result = mysql_query("INSERT INTO users(unique_id, firstname, lastname, identifier, description,
encrypted_password, salt, created_at) VALUES('$uuid', '$fname', '$lname', '$identifier', '$description',
'$encrypted_password', '$salt', NOW())");
        // check for successful store
        if ($result) {
             // get user details
             $uid = mysql_insert_id(); // last inserted id
            $result = mysql_query("SELECT * FROM users WHERE uid = $uid");
            // return user details
            return mysql_fetch_array($result);
        } else {
            return false;
        }
    }

    /**
     * Verifies user by identifier and password
     */
    public function getUserByIdentifierAndPassword($identifier, $password) {
        $result = mysql_query("SELECT * FROM users WHERE identifier = '$identifier'") or die(mysql_error());
        // check for result
        $no_of_rows = mysql_num_rows($result);
        if ($no_of_rows > 0) {
            $result = mysql_fetch_array($result);
            $salt = $result['salt'];
            $encrypted_password = $result['encrypted_password'];
            $hash = $this->checkhashSSHA($salt, $password);
            // check for password equality
            if ($encrypted_password == $hash) {
                // user authentication details are correct
                return $result;
            }
        } else {
            // user not found
            return false;
        }
    }

    /**
     * Check user is existed or not
     */
    public function isUserExisted($identifier) {
        $result = mysql_query("SELECT identifier from users WHERE identifier = '$identifier'");
        $no_of_rows = mysql_num_rows($result);
        if ($no_of_rows > 0) {
            // user existed
            return true;
        } else {
            // user not existed
            return false;
        }
    }

    /**
     * Encrypting password
     * @param password
     * returns salt and encrypted password
     */
    public function hashSSHA($password) {

        $salt = sha1(rand());
        $salt = substr($salt, 0, 10);
        $encrypted = base64_encode(sha1($password . $salt, true) . $salt);
        $hash = array("salt" => $salt, "encrypted" => $encrypted);
        return $hash;
    }
```

```
    /**
     * Decrypting password
     * @param salt, password
     * returns hash string
     */
    public function checkhashSSHA($salt, $password) {

        $hash = base64_encode(sha1($password . $salt, true) . $salt);

        return $hash;
    }

}

?>
```

acoforjco_api\index.php

```
<?php
/**
 * File to handle all API requests
 * Accepts GET and POST
 *
 * Each request will be identified by TAG
 * Response will be JSON data

/**
 * check for POST request
 */
if (isset($_POST['tag']) && $_POST['tag'] != '') {
    // get tag
    $tag = $_POST['tag'];

    // include db handler
    require_once __DIR__ . '/include/DB_Functions.php';
    $db = new DB_Functions();

    // response Array
    $response = array("tag" => $tag, "success" => 0, "error" => 0);

    // check for tag type
    if ($tag == 'login') {
        // Request type is check Login
        $identifier = $_POST['identifier'];
        $password = $_POST['password'];

        // check for user
        $user = $db->getUserByIdentifierAndPassword($identifier, $password);
        if ($user != false) {
            // user found
            // echo json with success = 1
            $response["success"] = 1;
            $response["user"]["fname"] = $user["firstname"];
            $response["user"]["lname"] = $user["lastname"];
             $response["user"]["identifier"] = $user["identifier"];
            $response["user"]["description"] = $user["description"];
            $response["user"]["uid"] = $user["unique_id"];
            $response["user"]["created_at"] = $user["created_at"];
            $response["user"]["updated_at"] = $user["updated_at"];

            echo json_encode($response);
        } else {
            // user not found
            // echo json with error = 1
            $response["error"] = 1;
            $response["error_msg"] = "Incorrect identifier or password!";
            echo json_encode($response);
        }
    } else if ($tag == 'register') {
        // Request type is Register new user
        $fname = $_POST['fname'];
         $lname = $_POST['lname'];
        $identifier = $_POST['identifier'];
        $description = $_POST['description'];
        $password = $_POST['password'];

         $subject = "Registration";
        $message = "Hello $fname,\n\nYou have sucessfully registered to our service.\n\nRegards,\nAdmin.";
```

```
        $from = "Beverly Festejo";
        $headers = "From:" . $from;

        // check if user is already existed
        if ($db->isUserExisted($identifier)) {
            // user is already existed - error response
            $response["error"] = 2;
            $response["error_msg"] = "User already existed";
            echo json_encode($response);
        } else {
            // store user
            $user = $db->storeUser($fname, $lname, $identifier, $description, $password);
            if ($user) {
                // user stored successfully
                $response["success"] = 1;
                $response["user"]["fname"] = $user["firstname"];
                 $response["user"]["lname"] = $user["lastname"];
                 $response["user"]["identifier"] = $user["identifier"];
                $response["user"]["description"] = $user["description"];
                $response["user"]["uid"] = $user["unique_id"];
                $response["user"]["created_at"] = $user["created_at"];
                $response["user"]["updated_at"] = $user["updated_at"];
                echo json_encode($response);
            } else {
                // user failed to store
                $response["error"] = 1;
                $response["error_msg"] = "JSON Error occured in Registration";
                echo json_encode($response);
            }
        }
    } else {
         $response["error"] = 3;
        $response["error_msg"] = "JSON ERROR";
        echo json_encode($response);
        echo "Invalid Request";
    }
} else {
    echo "A.Co for J.Co Login API";
}
?>
```

acoforjco_api\test.php

```
<?php

$con = mysql_connect("localhost","QueueIntendance","sirchuathegreat");

if (!$con) {
    die('Could not connect: ' . mysql_error());
}
//echo "1 record added";
mysql_select_db("QueueIntendance", $con);
//echo "1 record added";

$sql = "INSERT INTO items (name, quantity, price) VALUES('$_POST[name]','$_POST[quantity]', '$_POST[price]')";

if (!mysql_query($sql,$con)) {
    die('Error: ' . mysql_error());
}

mysql_close($con);
?>
```

A.Co_for_J.Co\src\com\specialproblem\acoforjco\library\DatabaseHandler.java

```
package com.specialproblem.acoforjco.library;

import java.util.HashMap;

import android.content.ContentValues;
import android.content.Context;
import android.database.Cursor;
import android.database.sqlite.SQLiteDatabase;
import android.database.sqlite.SQLiteOpenHelper;

public class DatabaseHandler extends SQLiteOpenHelper {
```

```
    // All Static variables
    // Database Version
    private static final int DATABASE_VERSION = 6;

    // Database Name
    private static final String DATABASE_NAME = "acoforjco_api";
    // private static final String DATABASE_NAME = "QueueIntendance";

    // Login table name
    private static final String TABLE_LOGIN = "login";

    // Login Table Columns names
    private static final String KEY_ID = "id";
    private static final String KEY_FIRSTNAME = "fname";
        private static final String KEY_LASTNAME = "lname";
    private static final String KEY_IDENTIFIER = "identifier";
    private static final String KEY_DESCRIPTION = "description";
    private static final String KEY_UID = "uid";
    private static final String KEY_CREATED_AT = "created_at";

    public DatabaseHandler(Context context) {
        super(context, DATABASE_NAME, null, DATABASE_VERSION);
    }

    // Creating Tables
    @Override
    public void onCreate(SQLiteDatabase sqLiteDatabase) {
        sqLiteDatabase.execSQL("CREATE TABLE " + TABLE_LOGIN + "(" +
                KEY_ID + " INTEGER PRIMARY KEY," +
                KEY_FIRSTNAME + " TEXT," +
                KEY_LASTNAME + " TEXT," +
                KEY_IDENTIFIER + " TEXT UNIQUE," +
                KEY_DESCRIPTION + " TEXT," +
                KEY_UID + " TEXT," +
                KEY_CREATED_AT + " TEXT" + ");");
    }

    // Upgrading database
    @Override
    public void onUpgrade(SQLiteDatabase sqLiteDatabase, int oldVersion, int newVersion) {
        // Drop older table if existed
        sqLiteDatabase.execSQL("DROP TABLE IF EXISTS " + TABLE_LOGIN);

        // Create tables again
        onCreate(sqLiteDatabase);
    }

    /**
     * Storing user details in database
     * */
    public void addUser(String fname, String lname, String identifier, String description, String uid, String
created_at) {
        SQLiteDatabase sqLiteDatabase = this.getWritableDatabase();

        ContentValues values = new ContentValues();
        values.put(KEY_FIRSTNAME, fname); // FirstName
        values.put(KEY_LASTNAME, lname); // LastName
                values.put(KEY_IDENTIFIER, identifier); // Identifier
        values.put(KEY_DESCRIPTION, description); // Description
        values.put(KEY_UID, uid);
        values.put(KEY_CREATED_AT, created_at); // Created At

        // Inserting Row
        sqLiteDatabase.insert(TABLE_LOGIN, null, values);
        sqLiteDatabase.close(); // Closing database connection
    }

    /**
     * Getting user data from database
     * */
    public HashMap<String, String> getUserDetails(){
        HashMap<String,String> user = new HashMap<String,String>();
        String selectQuery = "SELECT  * FROM " + TABLE_LOGIN;

        SQLiteDatabase db = this.getReadableDatabase();
        Cursor cursor = db.rawQuery(selectQuery, null);
        // Move to first row
        cursor.moveToFirst();
```

```
            if(cursor.getCount() > 0){
                user.put("fname", cursor.getString(1));
                    user.put("lname", cursor.getString(2));
                    user.put("identifier", cursor.getString(3));
                user.put("description", cursor.getString(4));
                user.put("uid", cursor.getString(5));
                user.put("created_at", cursor.getString(6));
            }
            cursor.close();
            db.close();
            // return user
            return user;
        }

        /**
         * Getting user login status
         * return true if rows are there in table
         * */
        public int getRowCount() {
            String countQuery = "SELECT  * FROM " + TABLE_LOGIN;
            SQLiteDatabase sqLiteDatabase = this.getReadableDatabase();
            Cursor cursor = sqLiteDatabase.rawQuery(countQuery, null);
            int rowCount = cursor.getCount();
            sqLiteDatabase.close();
            cursor.close();

            // return row count
            return rowCount;
        }

        /**
         * Re create database
         * Delete all tables and create them again
         * */
        public void resetTables(){
            SQLiteDatabase db = this.getWritableDatabase();
            // Delete All Rows
            db.delete(TABLE_LOGIN, null, null);
            db.close();
        }

}
```

A.Co_for_J.Co\src\com\specialproblem\acoforjco\library\JSONParser.java

```
package com.specialproblem.acoforjco.library;

import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStream;
import java.io.InputStreamReader;
import java.io.UnsupportedEncodingException;
import java.util.List;

import org.apache.http.HttpEntity;
import org.apache.http.HttpResponse;
import org.apache.http.NameValuePair;
import org.apache.http.client.ClientProtocolException;
import org.apache.http.client.entity.UrlEncodedFormEntity;
import org.apache.http.client.methods.HttpPost;
import org.apache.http.impl.client.DefaultHttpClient;
import org.json.JSONException;
import org.json.JSONObject;

import android.util.Log;

public class JSONParser {

    static InputStream is = null;
    static JSONObject jObj = null;
    static String json = "";

    // constructor
    public JSONParser() {
```

```
    }

    public JSONObject getJSONFromUrl(String url, List<NameValuePair> params) {

        // Making HTTP request
        try {
            // defaultHttpClient
            DefaultHttpClient httpClient = new DefaultHttpClient();
            HttpPost httpPost = new HttpPost(url);
            httpPost.setEntity(new UrlEncodedFormEntity(params));

            HttpResponse httpResponse = httpClient.execute(httpPost);
            HttpEntity httpEntity = httpResponse.getEntity();
            is = httpEntity.getContent();

        } catch (UnsupportedEncodingException e) {
            e.printStackTrace();
        } catch (ClientProtocolException e) {
            e.printStackTrace();
        } catch (IOException e) {
            e.printStackTrace();
        }

        try {
            BufferedReader reader = new BufferedReader(new InputStreamReader(
                    is, "iso-8859-1"), 8);
            StringBuilder sb = new StringBuilder();
            String line = null;
            while ((line = reader.readLine()) != null) {
                sb.append(line + "n");
            }
            is.close();
            json = sb.toString();
            Log.e("JSON", json);
        } catch (Exception e) {
            Log.e("Buffer Error", "Error converting result " + e.toString());
        }

        // try parse the string to a JSON object
        try {
            jObj = new JSONObject(json);

            Log.e("JSONParser", "ha");
        } catch (JSONException e) {
            Log.e("JSON Parser", "Error parsing data " + e.toString());
        }

        // return JSON String
        return jObj;

    }

}
```

A.Co_for_J.Co\src\com\specialproblem\acoforjco\library\JSONParser2.java

```
package com.specialproblem.acoforjco.library;

import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStream;
import java.io.InputStreamReader;
import java.io.UnsupportedEncodingException;
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 com.specialproblem.acoforjco.menulist.MenuList;
```

```
import android.util.Log;

public class JSONParser2 {
     static InputStream is2 = null;
     static JSONObject jObj2 = null;
     static String json2 = "";

     public ArrayList<MenuList> items  = new ArrayList<MenuList>();
     public MenuList mL = new MenuList();

     public JSONParser2() {

     }

     public ArrayList<MenuList> getJSONFromUrl2(String url) {

          try {
               HttpClient httpClient = new DefaultHttpClient();
               HttpGet httpGet = new HttpGet(url);
               HttpResponse httpResponse = httpClient.execute(httpGet);
               HttpEntity httpEntity = httpResponse.getEntity();

               if (httpEntity != null)
                    is2 = httpEntity.getContent();
          } catch (UnsupportedEncodingException e) {
               e.printStackTrace();
          } catch (ClientProtocolException e) {
               e.printStackTrace();
          } catch (IOException e) {
               e.printStackTrace();
          }

          try {
               BufferedReader reader = new BufferedReader(new InputStreamReader(is2));
               StringBuilder sb = new StringBuilder();
               String line = null;
               while ((line = reader.readLine()) != null) {
                    sb.append(line);
               }
               is2.close();
               json2 = sb.toString();
               Log.e("JSON", json2);
          } catch (Exception e) {
               Log.e("Buffer Error", "Error converting result " + e.toString());
          }

          try {
               JSONArray jArr = new JSONArray(json2);

               for (int i = 0; i < jArr.length(); i++){
                    jObj2 = jArr.getJSONObject(i);
                    String name = jObj2.getString("name");
                    String description = jObj2.getString("description");
                    String price = jObj2.getString("price");
                    String preptime = jObj2.getString("preptime");

                    MenuList mL = new MenuList();
                    mL.setName(name);
                    mL.setDescription(description);
                    mL.setPrice(price + " PHP");

                    items.add(mL);
               }
               Log.e("JSONParser2", items.toString());
          } catch (JSONException e) {
               Log.e("JSON Parser2", "Error parsing data " + e.toString());
          }
          return items;
     }
}
```

A.Co_for_J.Co\src\com\specialproblem\acoforjco\library\JSONParser3.java

```
package com.specialproblem.acoforjco.library;
```

```
import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStream;
import java.io.InputStreamReader;
import java.io.UnsupportedEncodingException;
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 com.specialproblem.acoforjco.menulist.MenuList;
import com.specialproblem.acoforjco.menulist.OrderList;

import android.util.Log;

public class JSONParser3 {
     static InputStream is3 = null;
     static JSONObject jObj3 = null;
     static String json3 = "";

     public ArrayList<OrderList> items = new ArrayList<OrderList>();
     public OrderList oL = new OrderList();
     public int price;
     public static final double total = 0;

     public JSONParser3() {

     }

     public ArrayList<OrderList> getJSONFromUrl3(String name, int quantity, double total, String url) {

          try {
               HttpClient httpClient = new DefaultHttpClient();
               HttpGet httpGet = new HttpGet(url);

               HttpResponse httpResponse = httpClient.execute(httpGet);
               HttpEntity httpEntity = httpResponse.getEntity();
               if (httpEntity != null)
                    is3 = httpEntity.getContent();
          } catch (UnsupportedEncodingException e) {
               e.printStackTrace();
          } catch (ClientProtocolException e) {
               e.printStackTrace();
          } catch (IOException e) {
               e.printStackTrace();
          }

          try {
               BufferedReader reader = new BufferedReader(new InputStreamReader(is3));
               StringBuilder sb = new StringBuilder();
               String line = null;
               while ((line = reader.readLine()) != null) {
                    sb.append(line);
               }
               is3.close();
               json3 = sb.toString();
               Log.e("JSON", json3);
          } catch (Exception e) {
               Log.e("Buffer Error", "Error converting result " + e.toString());
          }

          try {
               JSONArray jArr = new JSONArray(json3);

               for (int i = 0; i < jArr.length(); i++){
                    jObj3 = jArr.getJSONObject(i);
                    String name1 = jObj3.getString("name");

                    if(name.equals(name1)){
                         price = Integer.parseInt(jObj3.getString("price"));
                         double amount = price * quantity;
                         total = total + amount;
```

```
                    System.out.println(total);

                    OrderList oL = new OrderList();
                    oL.setQuantity(Integer.toString(quantity));
                    oL.setName(name);
                    oL.setAmount(Double.toString(amount) + " PHP");
                    items.add(oL);
                    break;
                }
            }
            Log.e("JSONParser3", "ha");
        } catch (JSONException e) {
            Log.e("JSON Parser3", "Error parsing data " + e.toString());
        }

        return items;
    }
}
```

A.Co_for_J.Co\src\com\specialproblem\acoforjco\library\JSONParser4.java

A.Co_for_J.Co\src\com\specialproblem\acoforjco\library\UserFunctions.java

```
package com.specialproblem.acoforjco.library;

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

import org.apache.http.NameValuePair;
import org.apache.http.message.BasicNameValuePair;
import org.json.JSONObject;

import com.specialproblem.acoforjco.URLStorage;

import android.content.Context;

public class UserFunctions {

    private JSONParser jsonParser;

    private static String loginURL = URLStorage.APIFolder;
    private static String registerURL = URLStorage.APIFolder;

    private static String LOGIN_TAG = "login";
    private static String REGISTER_TAG = "register";

    // constructor
    public UserFunctions(){
        jsonParser = new JSONParser();
    }

    /**
     * function make Login Request
     * @param identifier
     * @param password
     * */
    public JSONObject loginUser(String identifier, String password){
        // Building Parameters
        List<NameValuePair> params = new ArrayList<NameValuePair>();
        params.add(new BasicNameValuePair("tag", LOGIN_TAG));
        params.add(new BasicNameValuePair("identifier", identifier));
        params.add(new BasicNameValuePair("password", password));
        JSONObject json = jsonParser.getJSONFromUrl(loginURL, params);
        // return json
        // Log.e("JSON", json.toString());
        return json;
    }

    /**
     * function to register Request
     * @param identifier
     * @param description
```

```
     * @param password
     * */
    public JSONObject registerUser(String fname, String lname, String identifier, String description, String password){
        // Building Parameters
        List<NameValuePair> params = new ArrayList<NameValuePair>();
        params.add(new BasicNameValuePair("tag", REGISTER_TAG));
        params.add(new BasicNameValuePair("fname", fname));
        params.add(new BasicNameValuePair("lname", lname));
            params.add(new BasicNameValuePair("identifier", identifier));
        params.add(new BasicNameValuePair("description", description));
        params.add(new BasicNameValuePair("password", password));

        // getting JSON Object
        JSONObject json = jsonParser.getJSONFromUrl(registerURL, params);
        // return json
        return json;
    }

    /**
     * Function get Login status
     * */
    public boolean isUserLoggedIn(Context context){
        DatabaseHandler db = new DatabaseHandler(context);
        int count = db.getRowCount();
        if(count > 0){
            // user logged in
            return true;
        }
        return false;
    }

    /**
     * Function to logout user
     * Reset the SQLite Database
     * */
    public boolean logoutUser(Context context){
        DatabaseHandler db = new DatabaseHandler(context);
        db.resetTables();
        return true;
    }

}
```

A.Co_for_J.Co\src\com\specialproblem\acoforjco\menulist\Club.java

```
package com.specialproblem.acoforjco.menulist;

import java.util.ArrayList;

import com.specialproblem.acoforjco.DashboardActivity;
import com.specialproblem.acoforjco.R;
import com.specialproblem.acoforjco.URLStorage;
import com.specialproblem.acoforjco.library.JSONParser2;

import android.app.Activity;
import android.app.ProgressDialog;
import android.content.Context;
import android.content.Intent;
import android.os.AsyncTask;
import android.os.Bundle;
import android.view.View;
import android.view.View.OnClickListener;
import android.widget.AdapterView;
import android.widget.Button;
import android.widget.ListView;
import android.widget.Toast;
import android.widget.AdapterView.OnItemClickListener;

public class Club extends Activity {

    private JSONParser2 jsonParser2;
    private Context mContext;

    Intent homeIntent;
    Intent menuIntent;
```

```
Intent orderIntent;
Intent queueIntent;
Intent logoutIntent;
Intent detailsIntent;

Bundle fromAll;
ListView listOrder;
ArrayList<MenuList> items;

@Override
public void onCreate(Bundle savedInstanceState) {
     super.onCreate(savedInstanceState);
     setContentView(R.layout.activity_food_items);

     fromAll = getIntent().getExtras();

     Button homeTabButton = (Button)findViewById(R.id.button01);
     homeIntent = new Intent(getBaseContext(), HomeActivity.class);
               homeTabButton.setOnClickListener((OnClickListener) new homeTabListener());

               Button menuTabButton = (Button)findViewById(R.id.button02);
               menuIntent = new Intent(getBaseContext(), MenuActivity.class);
          menuTabButton.setOnClickListener((OnClickListener) new menuTabListener());

     Button customerOrderTabButton = (Button)findViewById(R.id.button03);
          orderIntent = new Intent(getBaseContext(), CustomerOrderActivity.class);
          customerOrderTabButton.setOnClickListener((OnClickListener) new customerOrderTabListener());

     Button queueTabButton = (Button)findViewById(R.id.button04);
          queueIntent = new Intent(getBaseContext(), QueueProgressActivity.class);
          queueTabButton.setOnClickListener((OnClickListener) new queueTabListener());

          Button logoutTabButton = (Button)findViewById(R.id.button05);
          logoutIntent = new Intent(getBaseContext(),DashboardActivity.class);
          logoutTabButton.setOnClickListener((OnClickListener) new logoutTabListener());

     detailsIntent  = new Intent(getBaseContext(), FoodDetails.class);

     listOrder = (ListView) findViewById (R.id.list_order);

     mContext = getApplicationContext();
     new ClubMenu().execute();
}

private class ClubMenu extends AsyncTask<String, String, ArrayList<MenuList>>{
     private ProgressDialog pDialog;

     @Override
     protected void onPreExecute() {
          super.onPreExecute();

          pDialog = new ProgressDialog(Club.this);
          pDialog.setTitle("Contacting Servers");
          pDialog.setMessage("J.Club ...");
          pDialog.setIndeterminate(false);
          pDialog.setCancelable(true);
          pDialog.show();
     }

     @Override
     protected ArrayList<MenuList> doInBackground(String... arg0) {
          jsonParser2 = new JSONParser2();
          items = jsonParser2.getJSONFromUrl2(URLStorage.ClubURL);

          pDialog.dismiss();
          return items;
     }

     @Override
     protected void onPostExecute (ArrayList<MenuList> items){
          listOrder.setAdapter(new MenuAdapter(mContext, items));
          listOrder.setOnItemClickListener((OnItemClickListener) new itemListener());
     }
}

@Override
public void onBackPressed() {
     menuIntent.putExtras(fromAll);
     startActivity(menuIntent);
```

```
        return;
    }

    public class itemListener implements OnItemClickListener {

        public void onItemLongClick1(AdapterView<?> parent, View view, int position, long id) {
        }

        @Override
        public void onItemClick(AdapterView<?> arg0, View arg1, int arg2, long arg3) {
            MenuList mL = items.get(arg2);
            String nn = mL.getName();

            int category = 3;

            detailsIntent.putExtras(fromAll);
            detailsIntent.putExtra("category", category);
            detailsIntent.putExtra("jObj2", nn);

            startActivity(detailsIntent);
        }
    }

    private class homeTabListener implements OnClickListener {

        @Override
            public void onClick(View arg0) {

            Toast.makeText(getApplicationContext(), "Loading Home Screen...", Toast.LENGTH_SHORT).show();
            homeIntent.putExtras(fromAll);
            startActivity(homeIntent);
        }
        }

    private class menuTabListener implements OnClickListener {

        @Override
        public void onClick(View arg0) {

            Toast.makeText(getApplicationContext(), "Loading Menu...", Toast.LENGTH_SHORT).show();
            menuIntent.putExtras(fromAll);
            startActivity(menuIntent);
        }
    }

        private class customerOrderTabListener implements OnClickListener {

        @Override
        public void onClick(View arg0) {

            Toast.makeText(getApplicationContext(), "Loading your current Order...", Toast.LENGTH_SHORT).show();
            orderIntent.putExtras(fromAll);
            startActivity(orderIntent);
        }
    }

    private class queueTabListener implements OnClickListener {

        @Override
        public void onClick(View arg0) {

            Toast.makeText(getApplicationContext(), "Loading Queue progress display...", Toast.LENGTH_SHORT).show();
            queueIntent.putExtras(fromAll);
            startActivity(queueIntent);
        }
    }

    private class logoutTabListener implements OnClickListener {

        @Override
        public void onClick(View arg0) {

            Toast.makeText(getApplicationContext(), "Logging out...", Toast.LENGTH_SHORT).show();
            startActivity(logoutIntent);
        }
    }
}
```

A.Co_for_J.Co\src\com\specialproblem\acoforjco\menulist\Coffee.java

```
package com.specialproblem.acoforjco.menulist;

import java.util.ArrayList;

import com.specialproblem.acoforjco.DashboardActivity;
import com.specialproblem.acoforjco.R;
import com.specialproblem.acoforjco.URLStorage;
import com.specialproblem.acoforjco.library.JSONParser2;

import android.app.Activity;
import android.app.ProgressDialog;
import android.content.Context;
import android.content.Intent;
import android.os.AsyncTask;
import android.os.Bundle;
import android.view.View;
import android.view.View.OnClickListener;
import android.widget.AdapterView;
import android.widget.Button;
import android.widget.ListView;
import android.widget.Toast;
import android.widget.AdapterView.OnItemClickListener;

public class Coffee extends Activity {

     private JSONParser2 jsonParser2;
     private Context mContext;

     Intent homeIntent;
     Intent menuIntent;
     Intent orderIntent;
     Intent queueIntent;
     Intent logoutIntent;
     Intent detailsIntent;

     Bundle fromAll;
     ListView listOrder;
     ArrayList<MenuList> items;

     @Override
     public void onCreate(Bundle savedInstanceState) {
          super.onCreate(savedInstanceState);
          setContentView(R.layout.activity_food_items);

          fromAll = getIntent().getExtras();

          Button homeTabButton = (Button)findViewById(R.id.button01);
          homeIntent = new Intent(getBaseContext(), HomeActivity.class);
          homeTabButton.setOnClickListener((OnClickListener) new homeTabListener());

          Button menuTabButton = (Button)findViewById(R.id.button02);
          menuIntent = new Intent(getBaseContext(), MenuActivity.class);
          menuTabButton.setOnClickListener((OnClickListener) new menuTabListener());

          Button customerOrderTabButton = (Button)findViewById(R.id.button03);
          orderIntent = new Intent(getBaseContext(), CustomerOrderActivity.class);
          customerOrderTabButton.setOnClickListener((OnClickListener) new customerOrderTabListener());

          Button queueTabButton = (Button)findViewById(R.id.button04);
          queueIntent = new Intent(getBaseContext(), QueueProgressActivity.class);
          queueTabButton.setOnClickListener((OnClickListener) new queueTabListener());

          Button logoutTabButton = (Button)findViewById(R.id.button05);
          logoutIntent = new Intent(getBaseContext(),DashboardActivity.class);
          logoutTabButton.setOnClickListener((OnClickListener) new logoutTabListener());

          detailsIntent  = new Intent(getBaseContext(), FoodDetails.class);

          listOrder = (ListView) findViewById (R.id.list_order);

          mContext = getApplicationContext();
          new CoffeeMenu().execute();
     }

     private class CoffeeMenu extends AsyncTask <String, String, ArrayList<MenuList>> {
```

```
    private ProgressDialog pDialog;

    @Override
    protected void onPreExecute() {
        super.onPreExecute();

        pDialog = new ProgressDialog(Coffee.this);
        pDialog.setTitle("Contacting Servers");
        pDialog.setMessage("J.Coffee ...");
        pDialog.setIndeterminate(false);
        pDialog.setCancelable(true);
        pDialog.show();
    }

    @Override
    protected ArrayList<MenuList> doInBackground(String... arg0) {
        jsonParser2 = new JSONParser2();
        items = jsonParser2.getJSONFromUrl2(URLStorage.CoffeeURL);

        pDialog.dismiss();
        return items;
    }

    @Override
    protected void onPostExecute (ArrayList<MenuList> items){
        listOrder.setAdapter(new MenuAdapter(mContext, items));
        listOrder.setOnItemClickListener((OnItemClickListener) new itemListener());
    }
}

@Override
public void onBackPressed() {
    menuIntent.putExtras(fromAll);
    startActivity(menuIntent);
    return;
}

public class itemListener implements OnItemClickListener {

    public void onItemLongClick1(AdapterView<?> parent, View view, int position, long id) {
    }

    @Override
    public void onItemClick(AdapterView<?> arg0, View arg1, int arg2, long arg3) {
        MenuList mL = items.get(arg2);
        String nn = mL.getName();

        int category = 2;

        detailsIntent.putExtras(fromAll);
        detailsIntent.putExtra("category", category);
        detailsIntent.putExtra("jObj2", nn);

        startActivity(detailsIntent);
    }
}

private class homeTabListener implements OnClickListener {

    @Override
    public void onClick(View arg0) {

        Toast.makeText(getApplicationContext(), "Loading Home Screen...", Toast.LENGTH_SHORT).show();
        homeIntent.putExtras(fromAll);
        startActivity(homeIntent);
    }
}

private class menuTabListener implements OnClickListener {

    @Override
    public void onClick(View arg0) {

        Toast.makeText(getApplicationContext(), "Loading Menu...", Toast.LENGTH_SHORT).show();
        menuIntent.putExtras(fromAll);
        startActivity(menuIntent);
    }
}
```

```
    private class customerOrderTabListener implements OnClickListener {

        @Override
        public void onClick(View arg0) {

            Toast.makeText(getApplicationContext(), "Loading your current Order...", Toast.LENGTH_SHORT).show();
            orderIntent.putExtras(fromAll);
            startActivity(orderIntent);
        }
    }

    private class queueTabListener implements OnClickListener {

        @Override
        public void onClick(View arg0) {

            Toast.makeText(getApplicationContext(), "Loading Queue progress display...", Toast.LENGTH_SHORT).show();
            queueIntent.putExtras(fromAll);
            startActivity(queueIntent);
        }
    }

    private class logoutTabListener implements OnClickListener {

        @Override
        public void onClick(View arg0) {

            Toast.makeText(getApplicationContext(), "Logging out...", Toast.LENGTH_SHORT).show();
            startActivity(logoutIntent);
        }
    }
}
```

A.Co_for_J.Co\src\com\specialproblem\acoforjco\menulist\ConfirmedOrder.java

```
package com.specialproblem.acoforjco.menulist;

import java.util.ArrayList;

import com.specialproblem.acoforjco.DashboardActivity;
import com.specialproblem.acoforjco.R;
import com.specialproblem.acoforjco.URLStorage;
import com.specialproblem.acoforjco.library.JSONParser3;

import android.app.Activity;
import android.app.ProgressDialog;
import android.content.Context;
import android.content.Intent;
import android.os.AsyncTask;
import android.os.Bundle;
import android.view.View;
import android.view.View.OnClickListener;
import android.widget.Button;
import android.widget.ListView;
import android.widget.TextView;
import android.widget.Toast;

public class ConfirmedOrder extends Activity {

    private JSONParser3 jsonParser3;
    private Context oContext;

    int category, quantity;
    double total = 0, taxrate = 0, discount = 0;;
    String name, urlTemp;

    Intent logoutIntent;
    Intent endIntent;

    Bundle fromAll;
    ArrayList<OrderList> items;

    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_confirmed_order);
```

```
        fromAll= getIntent().getExtras();

        Button homeTabButton = (Button)findViewById(R.id.button01);
        homeTabButton.setOnClickListener((OnClickListener) new homeTabListener());

            Button menuTabButton = (Button)findViewById(R.id.button02);
            menuTabButton.setOnClickListener((OnClickListener) new menuTabListener());

            Button customerOrderTabButton = (Button)findViewById(R.id.button03);
            customerOrderTabButton.setOnClickListener((OnClickListener) new customerOrderTabListener());

                Button queueTabButton = (Button)findViewById(R.id.button04);
            queueTabButton.setOnClickListener((OnClickListener) new queueTabListener());

            Button logoutTabButton = (Button)findViewById(R.id.button05);
            logoutIntent = new Intent(getBaseContext(),DashboardActivity.class);
            logoutTabButton.setOnClickListener((OnClickListener) new logoutTabListener());

        Button endButton = (Button)findViewById(R.id.Button06);
        endIntent = new Intent(getBaseContext(), DashboardActivity.class);
        endButton.setOnClickListener((OnClickListener) new endListener());

        int i, j;

        for (i = 0; i < 100; i++) {
            if(fromAll.getIntArray("tag")[i] == 0)
                break;
        }

        for (j = 0; j < i; j++) {
            category = fromAll.getIntArray("tag")[j];
            quantity = fromAll.getIntArray("quantity")[j];
            name = fromAll.getStringArray("name")[j];

            if(category == 1) {
                urlTemp = URLStorage.DonutsURL;
            } else if(category == 2) {
                urlTemp = URLStorage.CoffeeURL;
            } else if(category == 3) {
                urlTemp = URLStorage.ClubURL;
            } else if(category == 4) {
                urlTemp = URLStorage.PopsURL;
            } else if(category == 5) {
                urlTemp = URLStorage.CoolURL;
            }

            oContext = getApplicationContext();
            new ConfirmMenu().execute();
        }
    }

    private class ConfirmMenu extends AsyncTask<String, String, ArrayList<OrderList>>{
        private ProgressDialog pDialog;

        @Override
        protected void onPreExecute() {
            super.onPreExecute();

            pDialog = new ProgressDialog(ConfirmedOrder.this);
            pDialog.setTitle("Contacting Servers");
            pDialog.setMessage("Confirming orders ...");
            pDialog.setIndeterminate(false);
            pDialog.setCancelable(true);
            pDialog.show();
        }

        @Override
        protected ArrayList<OrderList> doInBackground(String... arg0) {
            jsonParser3 = new JSONParser3();
            items = jsonParser3.getJSONFromUrl3(name, quantity, total, urlTemp);

            pDialog.dismiss();
            return items;
        }

        @Override
        protected void onPostExecute (ArrayList<OrderList> items){
            discount = (10/100) * total;
```

```
                total = total - discount;
                taxrate = (12/100) * total;
                total = total + taxrate;

                TextView discountText = (TextView)findViewById(R.id.discount_text);
                discountText.setText("Discount (@10%) = " + Double.toString(discount) + " PHP");
                TextView taxrateText = (TextView)findViewById(R.id.taxrate_text);
                taxrateText.setText("Tax (@12%) = " + Double.toString(taxrate) + " PHP");
                TextView totalText = (TextView)findViewById(R.id.total_text);
                totalText.setText("Total = " + Double.toString(total) + " PHP");

                ListView listOrder = (ListView)findViewById(R.id.list_order);
                listOrder.setAdapter(new OrderAdapter(oContext, items));
            }
        }

        @Override
        public void onBackPressed() {
            return;
        }

        private class homeTabListener implements OnClickListener {

            @Override
            public void onClick(View arg0) {
                Toast.makeText(getApplicationContext(), "You already confirmed earlier. You can't go anywhere back now!",
Toast.LENGTH_LONG).show();
            }
        }

        private class menuTabListener implements OnClickListener {

            @Override
            public void onClick(View arg0) {
                Toast.makeText(getApplicationContext(), "You already confirmed earlier. You can't go anywhere back now!",
Toast.LENGTH_LONG).show();
            }
        }

        private class customerOrderTabListener implements OnClickListener {

            @Override
            public void onClick(View arg0) {
                Toast.makeText(getApplicationContext(), "You already confirmed earlier. You can't go anywhere back now!",
Toast.LENGTH_LONG).show();
            }
        }

        private class queueTabListener implements OnClickListener {

            @Override
            public void onClick(View arg0) {
                Toast.makeText(getApplicationContext(), "You already confirmed earlier. You can't go anywhere back now!",
Toast.LENGTH_LONG).show();
            }
        }

        private class logoutTabListener implements OnClickListener {

            @Override
            public void onClick(View arg0) {

                Toast.makeText(getApplicationContext(), "Logging out...", Toast.LENGTH_SHORT).show();
                startActivity(logoutIntent);
            }
        }

        private class endListener implements OnClickListener {

            @Override
            public void onClick(View arg0) {

                Toast.makeText(getApplicationContext(), "Ending Session...", Toast.LENGTH_SHORT).show();
                startActivity(endIntent);
            }
        }
}
```

A.Co_for_J.Co\src\com\specialproblem\acoforjco\menulist\Cool.java

```
package com.specialproblem.acoforjco.menulist;

import java.util.ArrayList;

import com.specialproblem.acoforjco.DashboardActivity;
import com.specialproblem.acoforjco.R;
import com.specialproblem.acoforjco.URLStorage;
import com.specialproblem.acoforjco.library.JSONParser2;

import android.app.Activity;
import android.app.ProgressDialog;
import android.content.Context;
import android.content.Intent;
import android.os.AsyncTask;
import android.os.Bundle;
import android.view.View;
import android.view.View.OnClickListener;
import android.widget.AdapterView;
import android.widget.Button;
import android.widget.ListView;
import android.widget.Toast;
import android.widget.AdapterView.OnItemClickListener;

public class Cool extends Activity {

     private JSONParser2 jsonParser2;
     private Context mContext;

     Intent homeIntent;
     Intent menuIntent;
     Intent orderIntent;
     Intent queueIntent;
     Intent logoutIntent;
     Intent detailsIntent;

     Bundle fromAll;
     ListView listOrder;
     ArrayList<MenuList> items;

     @Override
     public void onCreate(Bundle savedInstanceState) {
          super.onCreate(savedInstanceState);
          setContentView(R.layout.activity_food_items);

          fromAll = getIntent().getExtras();

               Button homeTabButton = (Button)findViewById(R.id.button01);
               homeIntent = new Intent(getBaseContext(), HomeActivity.class);
                    homeTabButton.setOnClickListener((OnClickListener) new homeTabListener());

               Button menuTabButton = (Button)findViewById(R.id.button02);
               menuIntent = new Intent(getBaseContext(), MenuActivity.class);
               menuTabButton.setOnClickListener((OnClickListener) new menuTabListener());

               Button customerOrderTabButton = (Button)findViewById(R.id.button03);
               orderIntent = new Intent(getBaseContext(), CustomerOrderActivity.class);
               customerOrderTabButton.setOnClickListener((OnClickListener) new customerOrderTabListener());

               Button queueTabButton = (Button)findViewById(R.id.button04);
               queueIntent = new Intent(getBaseContext(), QueueProgressActivity.class);
               queueTabButton.setOnClickListener((OnClickListener) new queueTabListener());

               Button logoutTabButton = (Button)findViewById(R.id.button05);
               logoutIntent = new Intent(getBaseContext(),DashboardActivity.class);
               logoutTabButton.setOnClickListener((OnClickListener) new logoutTabListener());

          detailsIntent  = new Intent(getBaseContext(), FoodDetails.class);

          listOrder = (ListView) findViewById (R.id.list_order);

          mContext = getApplicationContext();
          new CoolMenu().execute();
     }

     private class CoolMenu extends AsyncTask<String, String, ArrayList<MenuList>>{
```

```
    private ProgressDialog pDialog;

    @Override
    protected void onPreExecute() {
        super.onPreExecute();

        pDialog = new ProgressDialog(Cool.this);
        pDialog.setTitle("Contacting Servers");
        pDialog.setMessage("J.Cool ...");
        pDialog.setIndeterminate(false);
        pDialog.setCancelable(true);
        pDialog.show();
    }

    @Override
    protected ArrayList<MenuList> doInBackground(String... arg0) {
        jsonParser2 = new JSONParser2();
        items = jsonParser2.getJSONFromUrl2(URLStorage.CoolURL);

        pDialog.dismiss();
        return items;
    }

    @Override
    protected void onPostExecute (ArrayList<MenuList> items){
        listOrder.setAdapter(new MenuAdapter(mContext, items));
        listOrder.setOnItemClickListener((OnItemClickListener) new itemListener());
    }
}

@Override
public void onBackPressed() {
    menuIntent.putExtras(fromAll);
    startActivity(menuIntent);
    return;
}

public class itemListener implements OnItemClickListener {

    public void onItemLongClick1(AdapterView<?> parent, View view, int position, long id) {
    }

    @Override
    public void onItemClick(AdapterView<?> arg0, View arg1, int arg2, long arg3) {
        MenuList mL = items.get(arg2);
        String nn=mL.getName();

        int category = 5;

        detailsIntent.putExtras(fromAll);
        detailsIntent.putExtra("category", category);
        detailsIntent.putExtra("jObj2", nn);

        startActivity(detailsIntent);
    }
}

private class homeTabListener implements OnClickListener {

    @Override
    public void onClick(View arg0) {

        Toast.makeText(getApplicationContext(), "Loading Home Screen...", Toast.LENGTH_SHORT).show();
        homeIntent.putExtras(fromAll);
        startActivity(homeIntent);
    }
}

private class menuTabListener implements OnClickListener {

    @Override
    public void onClick(View arg0) {

        Toast.makeText(getApplicationContext(), "Loading Menu...", Toast.LENGTH_SHORT).show();
        menuIntent.putExtras(fromAll);
        startActivity(menuIntent);
    }
}
```

```
    private class customerOrderTabListener implements OnClickListener {

        @Override
        public void onClick(View arg0) {

            Toast.makeText(getApplicationContext(), "Loading your current Order...", Toast.LENGTH_SHORT).show();
            orderIntent.putExtras(fromAll);
            startActivity(orderIntent);
        }
    }

    private class queueTabListener implements OnClickListener {

        @Override
        public void onClick(View arg0) {

            Toast.makeText(getApplicationContext(), "Loading Queue progress display...", Toast.LENGTH_SHORT).show();
            queueIntent.putExtras(fromAll);
            startActivity(queueIntent);
        }
    }

    private class logoutTabListener implements OnClickListener {

        @Override
        public void onClick(View arg0) {

            Toast.makeText(getApplicationContext(), "Logging out...", Toast.LENGTH_SHORT).show();
            startActivity(logoutIntent);
        }
    }
}
```

A.Co_for_J.Co\src\com\specialproblem\acoforjco\menulist\CustomerOrderActivity.java

```
package com.specialproblem.acoforjco.menulist;

import java.io.InputStream;
import java.util.ArrayList;
import java.util.UUID;

import org.apache.http.HttpEntity;
import org.apache.http.HttpResponse;
import org.apache.http.NameValuePair;
import org.apache.http.client.HttpClient;
import org.apache.http.client.entity.UrlEncodedFormEntity;
import org.apache.http.client.methods.HttpPost;
import org.apache.http.impl.client.DefaultHttpClient;
import org.apache.http.message.BasicNameValuePair;

import com.specialproblem.acoforjco.DashboardActivity;
import com.specialproblem.acoforjco.R;
import com.specialproblem.acoforjco.URLStorage;
import com.specialproblem.acoforjco.library.JSONParser3;

import android.app.Activity;
import android.app.AlertDialog;
import android.app.ProgressDialog;
import android.content.Context;
import android.content.DialogInterface;
import android.content.Intent;
import android.os.AsyncTask;
import android.os.Bundle;
import android.telephony.TelephonyManager;
import android.text.InputType;
import android.view.View;
import android.view.View.OnClickListener;
import android.widget.AdapterView;
import android.widget.Button;
import android.widget.EditText;
import android.widget.ListView;
import android.widget.TextView;
import android.widget.Toast;
```

```
import android.widget.AdapterView.OnItemClickListener;

public class CustomerOrderActivity extends Activity {

     private JSONParser3 jsonParser3;
     private Context oContext;

     int category, quantity, price;
     double total = 0;

     Intent homeIntent;
     Intent menuIntent;
     Intent orderIntent;
     Intent queueIntent;
     Intent logoutIntent;
     Intent confirmIntent;

     Bundle fromAll;
     TextView totalText;
     ListView listOrder;
     ArrayList<OrderList> items = new ArrayList<OrderList>();

     @Override
     public void onCreate(Bundle savedInstanceState) {
          super.onCreate(savedInstanceState);
          setContentView(R.layout.activity_customer_order);

          fromAll = getIntent().getExtras();

          Button homeTabButton = (Button)findViewById(R.id.button01);
               homeIntent = new Intent(getBaseContext(), HomeActivity.class);
               homeTabButton.setOnClickListener((OnClickListener) new homeTabListener());

               Button menuTabButton = (Button)findViewById(R.id.button02);
               menuIntent = new Intent(getBaseContext(), MenuActivity.class);
               menuTabButton.setOnClickListener((OnClickListener) new menuTabListener());

               Button customerOrderTabButton = (Button)findViewById(R.id.button03);
               orderIntent = new Intent(getBaseContext(), CustomerOrderActivity.class);
               customerOrderTabButton.setOnClickListener((OnClickListener) new customerOrderTabListener());

                    Button queueTabButton = (Button)findViewById(R.id.button04);
               queueIntent = new Intent(getBaseContext(), QueueProgressActivity.class);
               queueTabButton.setOnClickListener((OnClickListener) new queueTabListener());

               Button logoutTabButton = (Button)findViewById(R.id.button05);
               logoutIntent = new Intent(getBaseContext(),DashboardActivity.class);
               logoutTabButton.setOnClickListener((OnClickListener) new logoutTabListener());

               Button orderConfirmButton = (Button)findViewById(R.id.Button06);
          confirmIntent = new Intent(getBaseContext(), ConfirmedOrder.class);
          orderConfirmButton.setOnClickListener((OnClickListener) new confirmListener());

          totalText = (TextView)findViewById(R.id.total_text);
          listOrder = (ListView)findViewById(R.id.list_order);

          oContext = getApplicationContext();
          new CustomerMenu().execute();

     }

     private class CustomerMenu extends AsyncTask<String, String, ArrayList<OrderList>>{
          private ProgressDialog pDialog;

          @Override
          protected void onPreExecute() {
               super.onPreExecute();

               pDialog = new ProgressDialog(CustomerOrderActivity.this);
               pDialog.setTitle("Contacting Servers");
               pDialog.setMessage("Gathering orders ...");
               pDialog.setIndeterminate(false);
               pDialog.setCancelable(true);
               pDialog.show();
          }

          @Override
          protected ArrayList<OrderList> doInBackground(String... arg0) {
               int i, j;
```

```
            String name, urlTemp = null;

            for(i = 0; i < 100; i++) {
                if(fromAll.getIntArray("tag")[i]==0)
                    break;
            }

            for(j = 0; j < i; j++) {
                category = fromAll.getIntArray("tag")[j];
                quantity = fromAll.getIntArray("quantity")[j];
                name = fromAll.getStringArray("name")[j];

                if(category == 1) {
                    urlTemp = URLStorage.DonutsURL;
                } else if(category == 2) {
                    urlTemp = URLStorage.CoffeeURL;
                } else if(category == 3) {
                    urlTemp = URLStorage.ClubURL;
                } else if(category == 4) {
                    urlTemp = URLStorage.PopsURL;
                } else if(category == 5) {
                    urlTemp = URLStorage.CoolURL;
                }

                jsonParser3 = new JSONParser3();
                items = jsonParser3.getJSONFromUrl3(name, quantity, total, urlTemp);

            }

            pDialog.dismiss();
            return items;
        }

        @Override
        protected void onPostExecute (ArrayList<OrderList> items){
            System.out.println("JSONParser3.total" +JSONParser3.total);
            totalText.setText("Total : " + Double.toString(JSONParser3.total) + " PHP");

            listOrder.setAdapter(new OrderAdapter(oContext, items));
            listOrder.setOnItemClickListener((OnItemClickListener) new orderListener());
        }
    }

    @Override
    public void onBackPressed() {
        menuIntent.putExtras(fromAll);
        startActivity(menuIntent);
        return;
    }

    private class homeTabListener implements OnClickListener {

        @Override
        public void onClick(View arg0) {

            Toast.makeText(getApplicationContext(), "Loading Home Screen...", Toast.LENGTH_SHORT).show();
            homeIntent.putExtras(fromAll);
            startActivity(homeIntent);
        }
    }

    private class menuTabListener implements OnClickListener {

        @Override
        public void onClick(View arg0) {

            Toast.makeText(getApplicationContext(), "Loading Menu...", Toast.LENGTH_SHORT).show();
            menuIntent.putExtras(fromAll);
            startActivity(menuIntent);
        }
    }

    private class customerOrderTabListener implements OnClickListener {

        @Override
        public void onClick(View arg0) {

            Toast.makeText(getApplicationContext(), "Loading your current Order...", Toast.LENGTH_SHORT).show();
            orderIntent.putExtras(fromAll);
```

```
            startActivity(orderIntent);
        }
    }

    private class queueTabListener implements OnClickListener {

        @Override
        public void onClick(View arg0) {

            Toast.makeText(getApplicationContext(), "Loading Queue progress display...", Toast.LENGTH_SHORT).show();
            queueIntent.putExtras(fromAll);
            startActivity(queueIntent);
        }
    }

    private class logoutTabListener implements OnClickListener {

        @Override
        public void onClick(View arg0) {

            Toast.makeText(getApplicationContext(), "Logging out...", Toast.LENGTH_SHORT).show();
            startActivity(logoutIntent);
        }
    }

    public class orderListener implements OnItemClickListener {
        AlertDialog.Builder alert;
        int z, k;

        public void onItemLongClick1(AdapterView<?> parent, View view, int position, long id) {

        }

        @Override
        public void onItemClick(AdapterView<?> arg0, View arg1, int arg2, long arg3) {
            OrderList oL = items.get(arg2);

            String nn = oL.getName();

            for (z = 0; z < 100; z++) {
                if(fromAll.getIntArray("tag")[z]==0)
                    break;
            }

            for (k = 0; k < z; k++) {
                if(nn.equals(fromAll.getStringArray("name")[k]))
                    break;
            }

            int q = fromAll.getIntArray("quantity")[k];
            alert = new AlertDialog.Builder(CustomerOrderActivity.this);
            alert.setTitle("Edit Order - "+ nn);
            alert.setMessage("Enter new quantity, or press remove to remove " + nn + " from your order\nCurrent
quantity is : " + q);

            //Set an EditText view to get user input
            final EditText input = new EditText(getBaseContext());
            input.setInputType(InputType.TYPE_CLASS_NUMBER);
            //   DialogInterface.OnClickListener
            alert.setView(input);

            alert.setPositiveButton("Change", new DialogInterface.OnClickListener() {
                public void onClick(DialogInterface dialog, int whichButton) {
                    String value = input.getText().toString();

                    if((value.matches("[0]*[0-9]{3}") || value.matches("[0]*[0-9]{2}") || value.matches("[0]*[0-
9]{1}")) && !(value.matches("[0]*"))) {
                        int abc = Integer.parseInt(value);

                        if(abc < 100) {
                            fromAll.getIntArray("quantity")[k] = Integer.parseInt(value);
                            fromAll.getIntArray("amount")[k] = Integer.parseInt(value) * price;
                            orderIntent.putExtras(fromAll);
                            startActivity(orderIntent);
                        }
                        else {
                            Toast.makeText(getApplicationContext(), "Enter a quantity less than 100",
Toast.LENGTH_SHORT).show();
                        }
```

```
                    }
                    else {
                        Toast.makeText(getApplicationContext(), "Enter a valid quantity!",
Toast.LENGTH_SHORT).show();
                    }
                }
            });

                alert.setNeutralButton("Remove", new DialogInterface.OnClickListener() {
                public void onClick(DialogInterface dialog, int whichButton) {
                    for(int m = k; k < z-1; k++) {
                        fromAll.getIntArray("tag")[m] = fromAll.getIntArray("tag")[m+1];
                        fromAll.getIntArray("quantity")[m] = fromAll.getIntArray("quantity")[m+1];
                        fromAll.getStringArray("name")[m] = fromAll.getStringArray("name")[m+1];
                        fromAll.getIntArray("amount")[m] = fromAll.getIntArray("amount")[m+1];
                    }

                        fromAll.getIntArray("tag")[z-1] = 0;
                    orderIntent.putExtras(fromAll);
                    startActivity(orderIntent);
                }
            });

            alert.setNegativeButton("Cancel", new DialogInterface.OnClickListener() {
                public void onClick(DialogInterface dialog, int whichButton) {
                    // Canceled.
                }
            });

            alert.show();
        }
    }

    private class confirmListener  implements OnClickListener {

        @Override
        public void onClick(View arg0) {

            int p;
            for (p = 0; p < 100; p++) {
                if (fromAll.getIntArray("tag")[p] == 0)
                    break;
            }

            if (p >= 1) {
                new CustomerOrderAsync().execute();

            } else
                Toast.makeText(getApplicationContext(), "Please add something to your order to confirm it!",
Toast.LENGTH_LONG).show();
        }
    }

    private class CustomerOrderAsync extends AsyncTask<String, String, ArrayList<OrderList>>{
        private ProgressDialog pDialog;

        @Override
        protected void onPreExecute() {
            super.onPreExecute();

            pDialog = new ProgressDialog(CustomerOrderActivity.this);
            pDialog.setTitle("Contacting Servers");
            pDialog.setMessage("Confirming orders ...");
            pDialog.setIndeterminate(false);
            pDialog.setCancelable(true);
            pDialog.show();
        }

        @Override
        protected ArrayList<OrderList> doInBackground(String... arg0) {
            ArrayList<NameValuePair> nameValuePairs = new ArrayList<NameValuePair>();

            int i , k = 0;
            for (i = 0; i < 100; i++) {
                if(fromAll.getIntArray("tag")[i] == 0)
                    break;
            }
```

```
            final TelephonyManager tm = (TelephonyManager)
getBaseContext().getSystemService(Context.TELEPHONY_SERVICE);

            final String tmDevice, tmSerial, androidId;
            tmDevice = "" + tm.getDeviceId();
            tmSerial = "" + tm.getSimSerialNumber();
            androidId = "" + android.provider.Settings.Secure.getString(getContentResolver(),
android.provider.Settings.Secure.ANDROID_ID);

            UUID deviceUuid = new UUID(androidId.hashCode(), ((long)tmDevice.hashCode() << 32) | tmSerial.hashCode());
            String deviceId = deviceUuid.toString();

            String name = "";
            String amount = "";
            String quantity = "";

            while (k < i) {
                 name = fromAll.getStringArray("name")[k];
                 amount = Integer.toString(fromAll.getIntArray("amount")[k]);
                 quantity = Integer.toString(fromAll.getIntArray("quantity")[k]);
                 ++k;

                 nameValuePairs.add(new BasicNameValuePair("name", name));
                 nameValuePairs.add(new BasicNameValuePair("price", amount));
                 nameValuePairs.add(new BasicNameValuePair("quantity", quantity));
                 nameValuePairs.add(new BasicNameValuePair("deviceId", deviceId));
                 System.out.println(deviceId);

                 try {
                      HttpClient httpclient = new DefaultHttpClient();
                      HttpPost httppost = new HttpPost(URLStorage.testURL);
                      httppost.setEntity(new UrlEncodedFormEntity(nameValuePairs));
                      HttpResponse response = httpclient.execute(httppost);
                      HttpEntity entity = response.getEntity();



                 } catch(Exception e) {
                      Toast.makeText(getBaseContext(),"Error in http connection "+e.toString(),
Toast.LENGTH_LONG).show();
                 } finally {
                 }
            }
            pDialog.dismiss();
            return items;
        }

        @Override
        protected void onPostExecute (ArrayList<OrderList> items){

            Toast.makeText(getApplicationContext(), "Confirming your order...", Toast.LENGTH_SHORT).show();
            confirmIntent.putExtras(fromAll);
            startActivity(confirmIntent);
        }
    }
}
```

A.Co_for_J.Co\src\com\specialproblem\acoforjco\menulist\Donuts.java

```
package com.specialproblem.acoforjco.menulist;

import java.util.ArrayList;

import com.specialproblem.acoforjco.DashboardActivity;
import com.specialproblem.acoforjco.R;
import com.specialproblem.acoforjco.URLStorage;
import com.specialproblem.acoforjco.library.JSONParser2;

import android.app.Activity;
import android.app.ProgressDialog;
import android.content.Intent;
import android.content.Context;
import android.os.AsyncTask;
import android.os.Bundle;
import android.view.View;
import android.view.View.OnClickListener;
```

```
import android.widget.AdapterView.OnItemClickListener;
import android.widget.AdapterView;
import android.widget.Button;
import android.widget.ListView;
import android.widget.Toast;

public class Donuts extends Activity {

     private JSONParser2 jsonParser2;
     private Context mContext;

     Intent homeIntent;
     Intent menuIntent;
     Intent orderIntent;
     Intent queueIntent;
     Intent logoutIntent;
     Intent detailsIntent;

     Bundle fromAll;
     ListView listOrder;
     ArrayList<MenuList> items;

     @Override
     public void onCreate(Bundle savedInstanceState) {
          super.onCreate(savedInstanceState);
          setContentView(R.layout.activity_food_items);

          fromAll = getIntent().getExtras();

          Button homeTabButton = (Button)findViewById(R.id.button01);
          homeIntent = new Intent(getBaseContext(), HomeActivity.class);
          homeTabButton.setOnClickListener((OnClickListener) new homeTabListener());

          Button menuTabButton = (Button)findViewById(R.id.button02);
          menuIntent = new Intent(getBaseContext(), MenuActivity.class);
          menuTabButton.setOnClickListener((OnClickListener) new menuTabListener());

          Button customerOrderTabButton = (Button)findViewById(R.id.button03);
          orderIntent = new Intent(getBaseContext(), CustomerOrderActivity.class);
          customerOrderTabButton.setOnClickListener((OnClickListener) new customerOrderTabListener());

          Button queueTabButton = (Button)findViewById(R.id.button04);
          queueIntent = new Intent(getBaseContext(), QueueProgressActivity.class);
          queueTabButton.setOnClickListener((OnClickListener) new queueTabListener());

          Button logoutTabButton = (Button)findViewById(R.id.button05);
          logoutIntent = new Intent(getBaseContext(),DashboardActivity.class);
          logoutTabButton.setOnClickListener((OnClickListener) new logoutTabListener());

          detailsIntent  = new Intent(getBaseContext(), FoodDetails.class);

          listOrder = (ListView) findViewById (R.id.list_order);

          mContext = getApplicationContext();
          new DonutMenu().execute();
     }

     private class DonutMenu extends AsyncTask<String, String, ArrayList<MenuList>>{
          private ProgressDialog pDialog;

          @Override
          protected void onPreExecute() {
               super.onPreExecute();

               pDialog = new ProgressDialog(Donuts.this);
               pDialog.setTitle("Contacting Servers");
               pDialog.setMessage("J.Co Donuts ...");
               pDialog.setIndeterminate(false);
               pDialog.setCancelable(true);
               pDialog.show();
          }

          @Override
          protected ArrayList<MenuList> doInBackground(String... arg0) {
               jsonParser2 = new JSONParser2();
               items = jsonParser2.getJSONFromUrl2(URLStorage.DonutsURL);

               System.out.println(items + "2");
               pDialog.dismiss();
```

```
            return items;
        }

        @Override
        protected void onPostExecute (ArrayList<MenuList> items){
            System.out.println(items + "3");
            listOrder.setAdapter(new MenuAdapter(mContext, items));
            listOrder.setOnItemClickListener((OnItemClickListener) new itemListener());
        }
    }

    @Override
    public void onBackPressed() {
        menuIntent.putExtras(fromAll);
        startActivity(menuIntent);
        return;
    }

    public class itemListener implements OnItemClickListener {

        public void onItemLongClick1(AdapterView<?> parent, View view, int position, long id) {

        }

        @Override
        public void onItemClick(AdapterView<?> arg0, View arg1, int arg2, long arg3) {
            MenuList mL = items.get(arg2);
            String nn = mL.getName();

            int category = 1;

            detailsIntent.putExtras(fromAll);
            detailsIntent.putExtra("category", category);
            detailsIntent.putExtra("jObj2", nn);

            startActivity(detailsIntent);
        }
    }

    private class homeTabListener implements OnClickListener {

        @Override
        public void onClick(View arg0) {

            Toast.makeText(getApplicationContext(), "Loading Home Screen...", Toast.LENGTH_SHORT).show();
            homeIntent.putExtras(fromAll);
            startActivity(homeIntent);
        }
    }

    private class menuTabListener implements OnClickListener {

        @Override
        public void onClick(View arg0) {

            Toast.makeText(getApplicationContext(), "Loading Menu...", Toast.LENGTH_SHORT).show();
            menuIntent.putExtras(fromAll);
            startActivity(menuIntent);
        }
    }

    private class customerOrderTabListener implements OnClickListener {

        @Override
        public void onClick(View arg0) {

            Toast.makeText(getApplicationContext(), "Loading your current Order...", Toast.LENGTH_SHORT).show();
            orderIntent.putExtras(fromAll);
            startActivity(orderIntent);
        }
    }

   private class queueTabListener implements OnClickListener {

        @Override
        public void onClick(View arg0) {

            Toast.makeText(getApplicationContext(), "Loading Queue progress display...", Toast.LENGTH_SHORT).show();
            queueIntent.putExtras(fromAll);
```

```
            startActivity(queueIntent);
        }
    }

    private class logoutTabListener implements OnClickListener {

        @Override
        public void onClick(View arg0) {

            Toast.makeText(getApplicationContext(), "Logging out...", Toast.LENGTH_SHORT).show();
            startActivity(logoutIntent);
        }
    }
}
```

A.Co_for_J.Co\src\com\specialproblem\acoforjco\menulist\FoodDetails.java

```
package com.specialproblem.acoforjco.menulist;

import java.io.BufferedInputStream;
import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStream;
import java.io.InputStreamReader;
import java.io.UnsupportedEncodingException;
import java.net.URL;
import java.net.URLConnection;

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 com.specialproblem.acoforjco.DashboardActivity;
import com.specialproblem.acoforjco.R;
import com.specialproblem.acoforjco.URLStorage;

import android.app.Activity;
import android.app.ProgressDialog;
import android.content.Intent;
import android.graphics.Bitmap;
import android.graphics.BitmapFactory;
import android.os.AsyncTask;
import android.os.Bundle;
import android.util.Log;
import android.view.View;
import android.view.View.OnClickListener;
import android.widget.Button;
import android.widget.EditText;
import android.widget.ImageView;
import android.widget.TextView;
import android.widget.Toast;

public class FoodDetails extends Activity {

    int counter, category, price1;
    String urlTemp = null, urlImage = null;
    String name, description, price, preptime;
    TextView nametext, detailstext, preptimetext, pricetext, quantiti;
    ImageView image;
    Bitmap bitMap = null;

    Intent homeIntent;
    Intent menuIntent;
    Intent orderIntent;
    Intent queueIntent;
    Intent logoutIntent;

    Bundle fromAll;

    @Override
```

```
public void onCreate(Bundle savedInstanceState) {
     super.onCreate(savedInstanceState);
     setContentView(R.layout.activity_food_details);

     fromAll = getIntent().getExtras();

     Button homeTabButton = (Button)findViewById(R.id.button01);
    homeIntent = new Intent(getBaseContext(), HomeActivity.class);
    homeTabButton.setOnClickListener((OnClickListener) new homeTabListener());

    Button menuTabButton = (Button)findViewById(R.id.button02);
    menuIntent = new Intent(getBaseContext(), MenuActivity.class);
    menuTabButton.setOnClickListener((OnClickListener) new menuTabListener());

    Button customerOrderTabButton = (Button)findViewById(R.id.button03);
    orderIntent = new Intent(getBaseContext(), CustomerOrderActivity.class);
    customerOrderTabButton.setOnClickListener((OnClickListener) new customerOrderTabListener());

    Button queueTabButton = (Button)findViewById(R.id.button04);
    queueIntent = new Intent(getBaseContext(), QueueProgressActivity.class);
    queueTabButton.setOnClickListener((OnClickListener) new queueTabListener());

    Button logoutTabButton = (Button)findViewById(R.id.button05);
    logoutIntent = new Intent(getBaseContext(),DashboardActivity.class);
    logoutTabButton.setOnClickListener((OnClickListener) new logoutTabListener());

    Button addButton = (Button)findViewById(R.id.Button06);
     addButton.setOnClickListener((OnClickListener) new addListener());

     nametext = (TextView)findViewById(R.id.name_text);
    detailstext = (TextView)findViewById(R.id.details_text);
    preptimetext = (TextView)findViewById(R.id.preptime_text);
    pricetext = (TextView)findViewById(R.id.price_text);

    counter = 0;
    Button add = (Button) findViewById(R.id.bAdd);
    Button sub = (Button) findViewById(R.id.bSub);
    quantiti = (TextView) findViewById(R.id.tvDisplay);

    add.setOnClickListener(new View.OnClickListener() {
        public void onClick(View v) {
        if(counter < 99 && counter >= 0)
          counter++;
        quantiti.setText( "" + counter);
        }
    });

    sub.setOnClickListener(new View.OnClickListener() {
        public void onClick(View v) {
     if(counter < 100 && counter >= 1)
          counter--;
        quantiti.setText( "" + counter);
        }
    });

     image = (ImageView)findViewById(R.id.home_image);

    category = getIntent().getIntExtra("category", 0);
     name = getIntent().getStringExtra("jObj2");

     if(category == 1) {
          urlTemp = URLStorage.DonutsURL;
          urlImage = URLStorage.DonutsFolder;
     } else if(category == 2) {
          urlTemp = URLStorage.CoffeeURL;
          urlImage = URLStorage.CoffeeFolder;
     } else if(category == 3) {
          urlTemp = URLStorage.ClubURL;
          urlImage = URLStorage.ClubFolder;
     } else if(category == 4) {
          urlTemp = URLStorage.PopsURL;
          urlImage = URLStorage.PopsFolder;
     } else if(category == 5) {
          urlTemp = URLStorage.CoolURL;
          urlImage = URLStorage.CoolFolder;
     }

     new DetailsMenu().execute();
}
```

```
private class DetailsMenu extends AsyncTask <String, String, String>{
     private ProgressDialog pDialog;

     @Override
     protected void onPreExecute() {
          super.onPreExecute();

          pDialog = new ProgressDialog(FoodDetails.this);
          pDialog.setTitle("Contacting Servers");
          pDialog.setMessage("Some information ...");
          pDialog.setIndeterminate(false);
          pDialog.setCancelable(true);
          pDialog.show();
     }

     @Override
     protected String doInBackground(String... arg0) {
          InputStream is2 = null;
          String json2 = "";

          try {
               HttpClient httpClient = new DefaultHttpClient();
               HttpGet httpGet = new HttpGet(urlTemp);
               HttpResponse httpResponse = httpClient.execute(httpGet);
               HttpEntity httpEntity = httpResponse.getEntity();

               if (httpEntity != null)
                    is2 = httpEntity.getContent();
          } catch (UnsupportedEncodingException e) {
               e.printStackTrace();
          } catch (ClientProtocolException e) {
               e.printStackTrace();
          } catch (IOException e) {
               e.printStackTrace();
          }

          try {
               BufferedReader reader = new BufferedReader(new InputStreamReader(is2));
               StringBuilder sb = new StringBuilder();
               String line = null;

               while ((line = reader.readLine()) != null) {
                    sb.append(line);
               }
               is2.close();
               json2 = sb.toString();
               Log.e("JSON", json2);
          } catch (Exception e) {
               Log.e("Buffer Error", "Error converting result " + e.toString());
          }

          return json2;
     }

     @Override
     protected void onPostExecute(String json2) {
          JSONObject jObj2 = null;

          try {
               JSONArray jArr = new JSONArray(json2);

               for(int i = 0; i < jArr.length(); i++) {
                    jObj2 = jArr.getJSONObject(i);
                    String name1 = jObj2.getString("name");

                  if(name.equals(name1)) {
                        urlImage = urlImage + (i+1);
                        nametext.setText("Name : " + name);

                        description = jObj2.getString("description");
                        detailstext.setText("Description : " + description);

                        price = jObj2.getString("price");
                        price1 = Integer.parseInt(price);
                        pricetext.setText("Price : " + price + " PHP");

                        preptime = jObj2.getString("preptime");
```

```
                        preptimetext.setText("Preptime : " + preptime + " minutes");
                        break;
                    }
                }
            } catch (JSONException e1) {
                Log.e("JSON Parser2", "Error parsing data " + e1.toString());
                e1.printStackTrace();
            }

            pDialog.dismiss();
            new imageShit().execute();
        }
    }

    private class imageShit extends AsyncTask <String, String, String>{

        @Override
        protected String doInBackground(String... arg0) {
            InputStream is3 = null;

            try {
                URL aURL = new URL(urlImage + ".png");
                URLConnection conn = aURL.openConnection();
                conn.connect();
                is3 = conn.getInputStream();
                BufferedInputStream bis = new BufferedInputStream(is3);
                /* Decode url-data to a bitmap. */
                bitMap = BitmapFactory.decodeStream(bis);
                bis.close();
                is3.close();
            } catch (IOException e) {

            }
            return null;
        }

        @Override
    protected void onPostExecute(String arg0){
            image.setImageBitmap(bitMap);
        }
    }

    @Override
    public void onBackPressed() {

        if(category == 1) {
            Intent donutsIntent = new Intent(getBaseContext(), Donuts.class);
           donutsIntent.putExtras(fromAll);
            startActivity(donutsIntent);
        } else if(category == 2) {
            Intent coffeeIntent = new Intent(getBaseContext(), Coffee.class);
           coffeeIntent.putExtras(fromAll);
            startActivity(coffeeIntent);
        } else if(category == 3) {
            Intent clubIntent = new Intent(getBaseContext(), Club.class);
            clubIntent.putExtras(fromAll);
            startActivity(clubIntent);
        } else if(category == 4) {
            Intent popsIntent = new Intent(getBaseContext(), Pops.class);
            popsIntent.putExtras(fromAll);
            startActivity(popsIntent);
        } else if(category == 5) {
            Intent coolIntent = new Intent(getBaseContext(), Cool.class);
            coolIntent.putExtras(fromAll);
            startActivity(coolIntent);
        }
        return;
    }

    private class addListener  implements OnClickListener {

        @Override
        public void onClick(View arg0) {

            String quantityString = quantiti.getText().toString();

            if ((quantityString.matches("[0]*[0-9]{3}") || quantityString.matches("[0]*[0-9]{2}") ||
quantityString.matches("[0]*[0-9]{1}")) && !(quantityString.matches("[0]*"))) {
                int quantityInt = Integer.parseInt(quantityString);
```

```
                    int flag2 = 0;

                    if (quantityInt < 100) {
                         int z, flag1 = 0;

                         for (z = 0; z < 100; z++) {
                              if(fromAll.getIntArray("tag")[z] == 0)
                                   break;
                         }

                         String quantityString2 = null;

                         for (int k = 0; k < z; k++) {
                              if(name.equals(fromAll.getStringArray("name")[k])) {
                                   quantityString2 = quantiti.getText().toString();
                                   int quantityInt2 = Integer.parseInt(quantityString2);
                                   int qu = fromAll.getIntArray("quantity")[k] + quantityInt2;

                                   if (qu < 100) {
                                        fromAll.getIntArray("tag")[k] = category;
                                        fromAll.getStringArray("name")[k] = name;
                                        fromAll.getIntArray("quantity")[k] = qu;
                                        fromAll.getIntArray("amount")[k] = qu * price1;
                                        flag1 = 1;
                                   } else {
                                        Toast.makeText(getApplicationContext(),"Maximum quatity allowed is
99!",Toast.LENGTH_SHORT).show();
                                        flag1 = 1;
                                        flag2 = 1;
                                   }
                              }
                         }

                         if (flag1 == 0) {
                              quantityString2 = quantiti.getText().toString();

                              int quantityInt2 = Integer.parseInt(quantityString2);
                              fromAll.getIntArray("tag")[z] = category;
                              fromAll.getStringArray("name")[z] = name;
                              fromAll.getIntArray("quantity")[z] = quantityInt2;
                              fromAll.getIntArray("amount")[z] = quantityInt2 * price1;

                         }

                         if (flag2 == 0) {
                              if(Integer.parseInt(quantityString2) == 1)
                                   Toast.makeText(getApplicationContext(),"Added " + quantityString2 + " " + name + " to
your order!",Toast.LENGTH_SHORT).show();
                              else
                                   Toast.makeText(getApplicationContext(),"Added " + quantityString2 + " " + name + "s to
your order!",Toast.LENGTH_SHORT).show();
                         }

                         menuIntent.putExtras(fromAll);
                         startActivity(menuIntent);
                    }
                    else {
                         Toast.makeText(getApplicationContext(),"Please enter quantity less than
100",Toast.LENGTH_SHORT).show();
                    }
               }
               else {
                    Toast.makeText(getApplicationContext(),"Please enter a valid quantity",Toast.LENGTH_SHORT).show();
               }
          }
     }

     private class homeTabListener implements OnClickListener {

     @Override
     public void onClick(View arg0) {

          Toast.makeText(getApplicationContext(), "Loading Home Screen...", Toast.LENGTH_SHORT).show();
          homeIntent.putExtras(fromAll);
          startActivity(homeIntent);
     }
    }

    private class menuTabListener implements OnClickListener {
```

```
        @Override
        public void onClick(View arg0) {

            Toast.makeText(getApplicationContext(), "Loading Menu...", Toast.LENGTH_SHORT).show();
            menuIntent.putExtras(fromAll);
            startActivity(menuIntent);
        }
    }

    private class customerOrderTabListener implements OnClickListener {

        @Override
        public void onClick(View arg0) {

            Toast.makeText(getApplicationContext(), "Loading your current Order...", Toast.LENGTH_SHORT).show();
            orderIntent.putExtras(fromAll);
            startActivity(orderIntent);
        }
    }

    private class queueTabListener implements OnClickListener {

        @Override
        public void onClick(View arg0) {

            Toast.makeText(getApplicationContext(), "Loading Queue progress display...", Toast.LENGTH_SHORT).show();
            queueIntent.putExtras(fromAll);
            startActivity(queueIntent);
        }
    }

    private class logoutTabListener implements OnClickListener {

        @Override
        public void onClick(View arg0) {

            Toast.makeText(getApplicationContext(), "Logging out...", Toast.LENGTH_SHORT).show();
            startActivity(logoutIntent);
        }
    }
}
```

A.Co_for_J.Co\src\com\specialproblem\acoforjco\menulist\HomeActivity.java

```
package com.specialproblem.acoforjco.menulist;

import com.specialproblem.acoforjco.DashboardActivity;
import com.specialproblem.acoforjco.R;

import android.app.Activity;
import android.content.Intent;
import android.os.Bundle;
import android.view.View;
import android.view.View.OnClickListener;
import android.widget.Button;
import android.widget.Toast;

public class HomeActivity extends Activity {

     Button homeTabButton;
     Button menuTabButton;
     Button customerOrderTabButton;
     Button queueTabButton;
     Button logoutTabButton;
     Intent homeIntent;
     Intent menuIntent;
     Intent orderIntent;
     Intent queueIntent;
     Intent logoutIntent;
     Bundle fromAll;

     @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_home);
```

```
    fromAll = getIntent().getExtras();

    homeTabButton = (Button)findViewById(R.id.button01);
    homeIntent = new Intent(getBaseContext(), HomeActivity.class);
    homeTabButton.setOnClickListener((OnClickListener) new homeTabListener());

    menuTabButton = (Button)findViewById(R.id.button02);
    menuIntent = new Intent(getBaseContext(), MenuActivity.class);
    menuTabButton.setOnClickListener((OnClickListener) new menuTabListener());

    customerOrderTabButton = (Button)findViewById(R.id.button03);
     orderIntent = new Intent(getBaseContext(), CustomerOrderActivity.class);
    customerOrderTabButton.setOnClickListener((OnClickListener) new customerOrderTabListener());

    queueTabButton = (Button)findViewById(R.id.button04);
    queueIntent = new Intent(getBaseContext(), QueueProgressActivity.class);
    queueTabButton.setOnClickListener((OnClickListener) new queueTabListener());

    logoutTabButton = (Button)findViewById(R.id.button05);
    logoutIntent = new Intent(getBaseContext(),DashboardActivity.class);
    logoutTabButton.setOnClickListener((OnClickListener) new logoutTabListener());
}

 @Override
 public void onBackPressed() {
      return;
 }

private class homeTabListener implements OnClickListener {

 @Override
 public void onClick(View arg0) {

      Toast.makeText(getApplicationContext(), "Loading Home Screen...", Toast.LENGTH_SHORT).show();
      homeIntent.putExtras(fromAll);
      startActivity(homeIntent);
 }
}

private class menuTabListener implements OnClickListener {

      @Override
      public void onClick(View arg0) {

           Toast.makeText(getApplicationContext(), "Loading Menu...", Toast.LENGTH_SHORT).show();
           menuIntent.putExtras(fromAll);
           startActivity(menuIntent);
      }
}

private class customerOrderTabListener implements OnClickListener {

      @Override
      public void onClick(View arg0) {

           Toast.makeText(getApplicationContext(), "Loading your current Order...", Toast.LENGTH_SHORT).show();
           orderIntent.putExtras(fromAll);
           startActivity(orderIntent);
      }
}

private class queueTabListener implements OnClickListener {

      @Override
      public void onClick(View arg0) {

           Toast.makeText(getApplicationContext(), "Loading Queue progress display...", Toast.LENGTH_SHORT).show();
           queueIntent.putExtras(fromAll);
           startActivity(queueIntent);
      }
}

private class logoutTabListener implements OnClickListener {

      @Override
      public void onClick(View arg0) {

           Toast.makeText(getApplicationContext(), "Logging out...", Toast.LENGTH_SHORT).show();
```

```
                startActivity(logoutIntent);
            }
        }
    }
```

A.Co_for_J.Co\src\com\specialproblem\acoforjco\menulist\MenuActivity.java

```
package com.specialproblem.acoforjco.menulist;

import com.specialproblem.acoforjco.DashboardActivity;
import com.specialproblem.acoforjco.R;

import android.app.Activity;
import android.content.Intent;
import android.os.Bundle;
import android.view.View;
import android.view.View.OnClickListener;
import android.widget.Button;
import android.widget.Toast;

public class MenuActivity extends Activity {

     Button homeTabButton;
     Button menuTabButton;
     Button customerOrderTabButton;
     Button queueTabButton;
     Button logoutTabButton;
     Button donutsButton;
     Button coffeeButton;
     Button clubButton;
     Button popsButton;
     Button coolButton;
     Intent homeIntent;
     Intent menuIntent;
     Intent orderIntent;
     Intent queueIntent;
     Intent logoutIntent;
     Intent donutsIntent;
     Intent coffeeIntent;
     Intent clubIntent;
     Intent popsIntent;
     Intent coolIntent;
     Bundle fromAll;

     @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_menu);

        fromAll = getIntent().getExtras();

        homeTabButton = (Button)findViewById(R.id.button01);
        homeIntent = new Intent(getBaseContext(), HomeActivity.class);
        homeTabButton.setOnClickListener((OnClickListener) new homeTabListener());

        menuTabButton = (Button)findViewById(R.id.button02);
        menuIntent = new Intent(getBaseContext(), MenuActivity.class);
        menuTabButton.setOnClickListener((OnClickListener) new menuTabListener());

        customerOrderTabButton = (Button)findViewById(R.id.button03);
        orderIntent = new Intent(getBaseContext(), CustomerOrderActivity.class);
        customerOrderTabButton.setOnClickListener((OnClickListener) new customerOrderTabListener());

        queueTabButton = (Button)findViewById(R.id.button04);
        queueIntent = new Intent(getBaseContext(), QueueProgressActivity.class);
        queueTabButton.setOnClickListener((OnClickListener) new queueTabListener());

        logoutTabButton = (Button)findViewById(R.id.button05);
        logoutIntent = new Intent(getBaseContext(),DashboardActivity.class);
        logoutTabButton.setOnClickListener((OnClickListener) new logoutTabListener());

        donutsButton = (Button)findViewById(R.id.button06);
        donutsIntent = new Intent(getBaseContext(), Donuts.class);
        donutsButton.setOnClickListener((OnClickListener) new donutsListener());

        coffeeButton = (Button)findViewById(R.id.button07);
```

```
    coffeeIntent = new Intent(getBaseContext(), Coffee.class);
    coffeeButton.setOnClickListener((OnClickListener) new coffeeListener());

    clubButton = (Button)findViewById(R.id.button08);
    clubIntent = new Intent(getBaseContext(), Club.class);
    clubButton.setOnClickListener((OnClickListener) new clubListener());

    popsButton = (Button)findViewById(R.id.button09);
    popsIntent = new Intent(getBaseContext(), Pops.class);
    popsButton.setOnClickListener((OnClickListener) new popsListener());

    coolButton = (Button)findViewById(R.id.button10);
    coolIntent = new Intent(getBaseContext(), Cool.class);
    coolButton.setOnClickListener((OnClickListener) new coolListener());

}

@Override
public void onBackPressed() {
     homeIntent.putExtras(fromAll);
     startActivity(homeIntent);
}

private class homeTabListener implements OnClickListener {

@Override
public void onClick(View arg0) {

     Toast.makeText(getApplicationContext(), "Loading Home Screen...", Toast.LENGTH_SHORT).show();
     homeIntent.putExtras(fromAll);
     startActivity(homeIntent);
}
}

private class menuTabListener implements OnClickListener {

     @Override
     public void onClick(View arg0) {

          Toast.makeText(getApplicationContext(), "Loading Menu...", Toast.LENGTH_SHORT).show();
          menuIntent.putExtras(fromAll);
          startActivity(menuIntent);
     }
}

private class customerOrderTabListener implements OnClickListener {

     @Override
     public void onClick(View arg0) {

          Toast.makeText(getApplicationContext(), "Loading your current Order...", Toast.LENGTH_SHORT).show();
          orderIntent.putExtras(fromAll);
          startActivity(orderIntent);
     }
}

private class queueTabListener implements OnClickListener {

     @Override
     public void onClick(View arg0) {

          Toast.makeText(getApplicationContext(), "Loading Queue progress display...", Toast.LENGTH_SHORT).show();
          queueIntent.putExtras(fromAll);
          startActivity(queueIntent);
     }
}

private class logoutTabListener implements OnClickListener {

     @Override
     public void onClick(View arg0) {

          Toast.makeText(getApplicationContext(), "Logging out...", Toast.LENGTH_SHORT).show();
          startActivity(logoutIntent);
     }
}

private class donutsListener implements OnClickListener {
```

```
        @Override
        public void onClick(View arg0) {

            Toast.makeText(getApplicationContext(), "Loading J.Co Donuts...", Toast.LENGTH_SHORT).show();
            donutsIntent.putExtras(fromAll);
            startActivity(donutsIntent);
        }
    }

    private class coffeeListener implements OnClickListener {

        @Override
        public void onClick(View arg0) {

            Toast.makeText(getApplicationContext(), "Loading J.Coffee...", Toast.LENGTH_SHORT).show();
            coffeeIntent.putExtras(fromAll);
            startActivity(coffeeIntent);
        }
    }

    private class clubListener implements OnClickListener {

        @Override
        public void onClick(View arg0) {

            Toast.makeText(getApplicationContext(), "Loading J.Club...", Toast.LENGTH_SHORT).show();
            clubIntent.putExtras(fromAll);
            startActivity(clubIntent);
        }
    }

    private class popsListener implements OnClickListener {

        @Override
        public void onClick(View arg0) {

            Toast.makeText(getApplicationContext(), "Loading J.Pops...", Toast.LENGTH_SHORT).show();
            popsIntent.putExtras(fromAll);
            startActivity(popsIntent);
        }
    }

    private class coolListener implements OnClickListener {

        @Override
        public void onClick(View arg0) {

            Toast.makeText(getApplicationContext(), "Loading J.Cool...", Toast.LENGTH_SHORT).show();
            coolIntent.putExtras(fromAll);
            startActivity(coolIntent);
        }
    }
}
```

A.Co_for_J.Co\src\com\specialproblem\acoforjco\menulist\MenuAdapter.java

```
package com.specialproblem.acoforjco.menulist;

import java.util.ArrayList;

import com.specialproblem.acoforjco.R;

import android.content.Context;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.BaseAdapter;
import android.widget.TextView;

public class MenuAdapter extends BaseAdapter {

    private static ArrayList<MenuList> menuItems;
    private LayoutInflater mInflater;

    public MenuAdapter(Context context, ArrayList<MenuList> results) {
        menuItems = results;
```

```
        mInflater = LayoutInflater.from(context);
    }

    @Override
    public int getCount() {
        return menuItems.size();
    }

    @Override
    public Object getItem(int position) {
        return menuItems.get(position);
    }

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

    @Override
    public View getView(int position, View convertView, ViewGroup parent) {
        ViewHolder holder;

        if (convertView == null) {
            convertView = mInflater.inflate(R.layout.activity_menulist, null);
            holder = new ViewHolder();

            holder.textName = (TextView) convertView.findViewById(R.id.menu_item_name);
            holder.textDescription = (TextView) convertView.findViewById(R.id.menu_item_description);
            holder.textPrice = (TextView) convertView.findViewById(R.id.menu_item_price);

            convertView.setTag(holder);
        }
        else {
            holder = (ViewHolder) convertView.getTag();
        }

        holder.textName.setText(menuItems.get(position).getName());
        holder.textDescription.setText(menuItems.get(position).getDescription());
        holder.textPrice.setText(menuItems.get(position).getPrice());

        return convertView;
    }

    static class ViewHolder
    {
        TextView textName;
        TextView textDescription;
        TextView textPrice;
    }
}
```

A.Co_for_J.Co\src\com\specialproblem\acoforjco\menulist\MenuList.java

```
package com.specialproblem.acoforjco.menulist;

public class MenuList {

    private String name = "";
    private String description = "";
    private String price = "";

    public void setName(String name) {
        this.name = name;
    }

    public String getName() {
        return name;
    }

    public void setDescription(String description) {
        this.description = description;
    }

    public String getDescription() {
        return description;
    }
```

```
    public void setPrice(String price) {
        this.price = price;
    }

    public String getPrice() {
        return price;
    }
}
```

A.Co_for_J.Co\src\com\specialproblem\acoforjco\menulist\OrderAdapter.java

```
package com.specialproblem.acoforjco.menulist;

import java.util.ArrayList;

import com.specialproblem.acoforjco.R;

import android.content.Context;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.BaseAdapter;
import android.widget.TextView;

public class OrderAdapter extends BaseAdapter {

    private static ArrayList<OrderList> menuItems;
    private LayoutInflater mInflater;

    public OrderAdapter(Context context, ArrayList<OrderList> results) {
        menuItems = results;
        mInflater = LayoutInflater.from(context);
    }

    @Override
    public int getCount() {
        return menuItems.size();
    }

    @Override
    public Object getItem(int position) {
        return menuItems.get(position);
    }

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

    @Override
    public View getView(int position, View convertView, ViewGroup parent) {
        ViewHolder holder;

        if (convertView == null) {
            convertView = mInflater.inflate(R.layout.activity_orderlist, null);
            holder = new ViewHolder();

            holder.textQuantity = (TextView) convertView.findViewById(R.id.text_quantity);
            holder.textName = (TextView) convertView.findViewById(R.id.text_name);
            holder.textAmount = (TextView) convertView.findViewById(R.id.text_amount);

            convertView.setTag(holder);
        }
        else {
            holder = (ViewHolder) convertView.getTag();
        }

        holder.textQuantity.setText(menuItems.get(position).getQuantity());
        holder.textName.setText(menuItems.get(position).getName());
        holder.textAmount.setText(menuItems.get(position).getAmount());

        return convertView;
    }

    static class ViewHolder {
```

```
        TextView textQuantity;
        TextView textName;
        TextView textAmount;
    }

}
```

A.Co_for_J.Co\src\com\specialproblem\acoforjco\menulist\OrderList.java

```
package com.specialproblem.acoforjco.menulist;

public class OrderList {

    private String quantity = "";
    private String name = "";
    private String amount = "";

    public void setQuantity(String quantity) {
        this.quantity = quantity;
    }

    public String getQuantity() {
        return quantity;
    }

    public void setName(String name) {
        this.name = name;
    }

    public String getName() {
        return name;
    }

    public void setAmount(String amount) {
        this.amount = amount;
    }

    public String getAmount() {
        return amount;
    }
}
```

A.Co_for_J.Co\src\com\specialproblem\acoforjco\menulist\Pops.java

```
package com.specialproblem.acoforjco.menulist;

import java.util.ArrayList;

import com.specialproblem.acoforjco.DashboardActivity;
import com.specialproblem.acoforjco.R;
import com.specialproblem.acoforjco.URLStorage;
import com.specialproblem.acoforjco.library.JSONParser2;

import android.app.Activity;
import android.app.ProgressDialog;
import android.content.Context;
import android.content.Intent;
import android.os.AsyncTask;
import android.os.Bundle;
import android.view.View;
import android.view.View.OnClickListener;
import android.widget.AdapterView;
import android.widget.Button;
import android.widget.ListView;
import android.widget.Toast;
import android.widget.AdapterView.OnItemClickListener;

public class Pops extends Activity {

    private JSONParser2 jsonParser2;
    private Context mContext;

    Intent homeIntent;
    Intent menuIntent;
```

```
Intent orderIntent;
Intent queueIntent;
Intent logoutIntent;
Intent detailsIntent;

Bundle fromAll;
ListView listOrder;
ArrayList<MenuList> items;

@Override
public void onCreate(Bundle savedInstanceState) {
     super.onCreate(savedInstanceState);
     setContentView(R.layout.activity_food_items);

     fromAll = getIntent().getExtras();

    Button homeTabButton = (Button)findViewById(R.id.button01);
    homeIntent = new Intent(getBaseContext(), HomeActivity.class);
    homeTabButton.setOnClickListener((OnClickListener) new homeTabListener());

    Button menuTabButton = (Button)findViewById(R.id.button02);
    menuIntent = new Intent(getBaseContext(), MenuActivity.class);
    menuTabButton.setOnClickListener((OnClickListener) new menuTabListener());

    Button customerOrderTabButton = (Button)findViewById(R.id.button03);
    orderIntent = new Intent(getBaseContext(), CustomerOrderActivity.class);
    customerOrderTabButton.setOnClickListener((OnClickListener) new customerOrderTabListener());

    Button queueTabButton = (Button)findViewById(R.id.button04);
    queueIntent = new Intent(getBaseContext(), QueueProgressActivity.class);
    queueTabButton.setOnClickListener((OnClickListener) new queueTabListener());

    Button logoutTabButton = (Button)findViewById(R.id.button05);
    logoutIntent = new Intent(getBaseContext(),DashboardActivity.class);
    logoutTabButton.setOnClickListener((OnClickListener) new logoutTabListener());

     detailsIntent  = new Intent(getBaseContext(), FoodDetails.class);

     listOrder = (ListView) findViewById (R.id.list_order);

     mContext = getApplicationContext();
     new PopsMenu().execute();
}

private class PopsMenu extends AsyncTask <String, String, ArrayList<MenuList>>{
     private ProgressDialog pDialog;

     @Override
     protected void onPreExecute(){
          super.onPreExecute();

          pDialog = new ProgressDialog(Pops.this);
          pDialog.setTitle("Contacting Servers");
          pDialog.setMessage("J.Pops...");
          pDialog.setIndeterminate(false);
          pDialog.setCancelable(true);
          pDialog.show();
     }

     @Override
     protected ArrayList<MenuList> doInBackground(String... arg0) {
          jsonParser2 = new JSONParser2();
          items = jsonParser2.getJSONFromUrl2(URLStorage.PopsURL);

          pDialog.dismiss();
          return items;
     }

     @Override
     protected void onPostExecute (ArrayList<MenuList> items){
          listOrder.setAdapter(new MenuAdapter(mContext, items));
          listOrder.setOnItemClickListener((OnItemClickListener) new itemListener());
     }
}

@Override
public void onBackPressed() {
     menuIntent.putExtras(fromAll);
     startActivity(menuIntent);
```

```
            return;
        }

        public class itemListener implements OnItemClickListener {

            public void onItemLongClick1(AdapterView<?> parent, View view, int position, long id) {
            }

            @Override
            public void onItemClick(AdapterView<?> arg0, View arg1, int arg2, long arg3) {
                MenuList mL = items.get(arg2);
                String nn = mL.getName();

                int category = 4;

                detailsIntent.putExtras(fromAll);
                detailsIntent.putExtra("category", category);
                detailsIntent.putExtra("jObj2", nn);

                startActivity(detailsIntent);
            }
        }

        private class homeTabListener implements OnClickListener {

        @Override
        public void onClick(View arg0) {

            Toast.makeText(getApplicationContext(), "Loading Home Screen...", Toast.LENGTH_SHORT).show();
            homeIntent.putExtras(fromAll);
            startActivity(homeIntent);
        }
       }

       private class menuTabListener implements OnClickListener {

            @Override
            public void onClick(View arg0) {

                Toast.makeText(getApplicationContext(), "Loading Menu...", Toast.LENGTH_SHORT).show();
                menuIntent.putExtras(fromAll);
                startActivity(menuIntent);
            }
       }

       private class customerOrderTabListener implements OnClickListener {

            @Override
            public void onClick(View arg0) {

                Toast.makeText(getApplicationContext(), "Loading your current Order...", Toast.LENGTH_SHORT).show();
                orderIntent.putExtras(fromAll);
                startActivity(orderIntent);
            }
       }

       private class queueTabListener implements OnClickListener {

            @Override
            public void onClick(View arg0) {

                Toast.makeText(getApplicationContext(), "Loading Queue progress display...", Toast.LENGTH_SHORT).show();
                queueIntent.putExtras(fromAll);
                startActivity(queueIntent);
            }
       }

       private class logoutTabListener implements OnClickListener {

            @Override
            public void onClick(View arg0) {

                Toast.makeText(getApplicationContext(), "Logging out...", Toast.LENGTH_SHORT).show();
                startActivity(logoutIntent);
            }
       }
}
```

A.Co_for_J.Co\src\com\specialproblem\acoforjco\menulist\QueueProgressActivity.java

```
package com.specialproblem.acoforjco.menulist;

import android.app.Activity;
import android.content.Intent;
import android.os.Bundle;
import android.view.View;
import android.view.View.OnClickListener;
import android.widget.Button;
import android.widget.Toast;

import com.specialproblem.acoforjco.DashboardActivity;
import com.specialproblem.acoforjco.R;

public class QueueProgressActivity extends Activity {

     Intent homeIntent;
     Intent menuIntent;
     Intent orderIntent;
     Intent queueIntent;
     Intent logoutIntent;
     Bundle fromAll;

     @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_queue_progress);

        fromAll = getIntent().getExtras();

        Button homeTabButton = (Button)findViewById(R.id.button01);
        homeIntent = new Intent(getBaseContext(), HomeActivity.class);
        homeTabButton.setOnClickListener((OnClickListener) new homeTabListener());

        Button menuTabButton = (Button)findViewById(R.id.button02);
        menuIntent = new Intent(getBaseContext(), MenuActivity.class);
        menuTabButton.setOnClickListener((OnClickListener) new menuTabListener());

        Button customerOrderTabButton = (Button)findViewById(R.id.button03);
        orderIntent = new Intent(getBaseContext(), CustomerOrderActivity.class);
        customerOrderTabButton.setOnClickListener((OnClickListener) new customerOrderTabListener());

        Button queueTabButton = (Button)findViewById(R.id.button04);
        queueIntent = new Intent(getBaseContext(), QueueProgressActivity.class);
        queueTabButton.setOnClickListener((OnClickListener) new queueTabListener());

        Button logoutTabButton = (Button)findViewById(R.id.button05);
        logoutIntent = new Intent(getBaseContext(),DashboardActivity.class);
        logoutTabButton.setOnClickListener((OnClickListener) new logoutTabListener());
    }

     @Override
     public void onBackPressed() {
          return;
     }

    private class homeTabListener implements OnClickListener {

     @Override
     public void onClick(View arg0) {

          Toast.makeText(getApplicationContext(), "Loading Home Screen...", Toast.LENGTH_SHORT).show();
          homeIntent.putExtras(fromAll);
          startActivity(homeIntent);
     }
    }

    private class menuTabListener implements OnClickListener {

          @Override
          public void onClick(View arg0) {

               Toast.makeText(getApplicationContext(), "Loading Menu...", Toast.LENGTH_SHORT).show();
```

```
            menuIntent.putExtras(fromAll);
            startActivity(menuIntent);
        }
    }

    private class customerOrderTabListener implements OnClickListener {

        @Override
        public void onClick(View arg0) {

            Toast.makeText(getApplicationContext(), "Loading your current Order...", Toast.LENGTH_SHORT).show();
            orderIntent.putExtras(fromAll);
            startActivity(orderIntent);
        }
    }

    private class queueTabListener implements OnClickListener {

        @Override
        public void onClick(View arg0) {

            Toast.makeText(getApplicationContext(), "Loading Queue progress display...", Toast.LENGTH_SHORT).show();
            queueIntent.putExtras(fromAll);
            startActivity(queueIntent);
        }
    }

    private class logoutTabListener implements OnClickListener {

        @Override
        public void onClick(View arg0) {

            Toast.makeText(getApplicationContext(), "Logging out...", Toast.LENGTH_SHORT).show();
            startActivity(logoutIntent);
        }
    }
}
```

A.Co_for_J.Co\src\com\specialproblem\acoforjco\orderlist\MainActivity.java

A.Co_for_J.Co\src\com\specialproblem\acoforjco\orderlist\MainClub.java

A.Co_for_J.Co\src\com\specialproblem\acoforjco\orderlist\MainCoffee.java

A.Co_for_J.Co\src\com\specialproblem\acoforjco\orderlist\MainCool.java

A.Co_for_J.Co\src\com\specialproblem\acoforjco\orderlist\MainDonuts.java

A.Co_for_J.Co\src\com\specialproblem\acoforjco\orderlist\MainMenuActivity.java

A.Co_for_J.Co\src\com\specialproblem\acoforjco\orderlist\MainOrderActivity.java

A.Co_for_J.Co\src\com\specialproblem\acoforjco\orderlist\MainPops.java

A.Co_for_J.Co\src\com\specialproblem\acoforjco\orderlist\MainQueueActivity.java

A.Co_for_J.Co\src\com\specialproblem\acoforjco\qrcode\camera\AutoFocusCallback.java

```
package com.specialproblem.acoforjco.qrcode.camera;

import android.hardware.Camera;
import android.os.Handler;
import android.os.Message;
import android.util.Log;

final class AutoFocusCallback implements Camera.AutoFocusCallback {

     private static final String TAG = AutoFocusCallback.class.getSimpleName();

    private static final long AUTOFOCUS_INTERVAL_MS = 1500L;

    private Handler autoFocusHandler;
    private int autoFocusMessage;

    void setHandler(Handler autoFocusHandler, int autoFocusMessage) {
        this.autoFocusHandler = autoFocusHandler;
        this.autoFocusMessage = autoFocusMessage;
    }

     @Override
    public void onAutoFocus(boolean success, Camera camera) {
        if (autoFocusHandler != null) {
            Message message = autoFocusHandler.obtainMessage(autoFocusMessage, success);
            // Simulate continuous autofocus by sending a focus request every
            // AUTOFOCUS_INTERVAL_MS milliseconds.
            // Log.d(TAG, "Got auto-focus callback; requesting another");
            autoFocusHandler.sendMessageDelayed(message, AUTOFOCUS_INTERVAL_MS);
            autoFocusHandler = null;
        } else {
            Log.d(TAG, "Got auto-focus callback, but no handler for it");
        }
    }

}
```

A.Co_for_J.Co\src\com\specialproblem\acoforjco\qrcode\camera\CameraConfigurationManager.java

```
package com.specialproblem.acoforjco.qrcode.camera;

import java.util.Collection;

import com.specialproblem.acoforjco.qrcode.data.Preferences;

import android.content.Context;
import android.hardware.Camera;
import android.graphics.Point;
import android.util.Log;
import android.view.Display;
import android.view.WindowManager;

/**
 * A class which deals with reading, parsing, and setting the camera parameters
 * which are used to configure the camera hardware.
 */
public final class CameraConfigurationManager {
```

```
    private static final String TAG = "CameraConfiguration";
    private static final int MIN_PREVIEW_PIXELS = 320 * 240; // small screen
    private static final int MAX_PREVIEW_PIXELS = 800 * 480; // large/HD screen

    private final Context context;
    private Point screenResolution;
    private Point cameraResolution;

     public CameraConfigurationManager(Context context) {
        this.context = context;
    }

    /**
     * Reads, one time, values from the camera that are needed by the app.
     */
    void initFromCameraParameters(Camera camera) {
        Camera.Parameters parameters = camera.getParameters();
        WindowManager manager = (WindowManager) context.getSystemService(Context.WINDOW_SERVICE);
        Display display = manager.getDefaultDisplay();
        int width = display.getWidth();
        int height = display.getHeight();
        // Landscape-only, may have seen issues with display thinking it's portrait when waking from sleep.
        // If it's not landscape, assume it's mistaken and reverse them:
        if (width < height) {
            Log.i(TAG, "Display reports portrait orientation; assuming this is incorrect");
            int temp = width;
            width = height;
            height = temp;
        }
        screenResolution = new Point(width, height);
        Log.i(TAG, "Screen resolution: " + screenResolution);
        cameraResolution = findBestPreviewSizeValue(parameters, screenResolution, false);
        Log.i(TAG, "Camera resolution: " + cameraResolution);
    }

    void setDesiredCameraParameters(Camera camera) {
        Camera.Parameters parameters = camera.getParameters();

        if (parameters == null) {
            Log.w(TAG, "Device error: no camera parameters are available. Proceeding without configuration.");
            return;
        }

        initializeTorch(parameters);
        String focusMode = findSettableValue(parameters.getSupportedFocusModes(), Camera.Parameters.FOCUS_MODE_AUTO,
Camera.Parameters.FOCUS_MODE_MACRO);
        if (focusMode != null) {
            parameters.setFocusMode(focusMode);
        }

        parameters.setPreviewSize(cameraResolution.x, cameraResolution.y);
        camera.setParameters(parameters);
    }

    public Point getCameraResolution() {
        return cameraResolution;
    }

    public Point getScreenResolution() {
        return screenResolution;
    }

    void setTorch(Camera camera, boolean newSetting) {
        Camera.Parameters parameters = camera.getParameters();
        doSetTorch(parameters, newSetting);
        camera.setParameters(parameters);
    }

    private static void initializeTorch(Camera.Parameters parameters) {
        doSetTorch(parameters, Preferences.KEY_FRONT_LIGHT);
    }

    private static void doSetTorch(Camera.Parameters parameters, boolean newSetting) {
        String flashMode;
        if (newSetting) {
            flashMode = findSettableValue(parameters.getSupportedFlashModes(), Camera.Parameters.FLASH_MODE_TORCH,
Camera.Parameters.FLASH_MODE_ON);
        } else {
            flashMode = findSettableValue(parameters.getSupportedFlashModes(), Camera.Parameters.FLASH_MODE_OFF);
```

```
        }
        if (flashMode != null) {
            parameters.setFlashMode(flashMode);
        }
    }

    private static Point findBestPreviewSizeValue(Camera.Parameters parameters, Point screenResolution, boolean portrait)
{
        Point bestSize = null;
        int diff = Integer.MAX_VALUE;
        for (Camera.Size supportedPreviewSize : parameters.getSupportedPreviewSizes()) {
            int pixels = supportedPreviewSize.height * supportedPreviewSize.width;
            if (pixels < MIN_PREVIEW_PIXELS || pixels > MAX_PREVIEW_PIXELS) {
                continue;
            }
            int supportedWidth = portrait ? supportedPreviewSize.height : supportedPreviewSize.width;
            int supportedHeight = portrait ? supportedPreviewSize.width : supportedPreviewSize.height;
            int newDiff = Math.abs(screenResolution.x * supportedHeight - supportedWidth * screenResolution.y);
            if (newDiff == 0) {
                bestSize = new Point(supportedWidth, supportedHeight);
                break;
            }
            if (newDiff < diff) {
                bestSize = new Point(supportedWidth, supportedHeight);
                diff = newDiff;
            }
        }
        if (bestSize == null) {
            Camera.Size defaultSize = parameters.getPreviewSize();
            bestSize = new Point(defaultSize.width, defaultSize.height);
        }
        return bestSize;
    }

    private static String findSettableValue(Collection<String> supportedValues, String... desiredValues) {
        Log.i(TAG, "Supported values: " + supportedValues);
        String result = null;
        if (supportedValues != null) {
            for (String desiredValue : desiredValues) {
                if (supportedValues.contains(desiredValue)) {
                    result = desiredValue;
                    break;
                }
            }
        }
        Log.i(TAG, "Settable value: " + result);
        return result;
    }
}
```

A.Co_for_J.Co\src\com\specialproblem\acoforjco\qrcode\camera\CameraManager.java

```
package com.specialproblem.acoforjco.qrcode.camera;

import java.io.IOException;

import com.specialproblem.acoforjco.qrcode.data.Preferences;
import com.specialproblem.acoforjco.qrcode.PlanarYUVLuminanceSource;

import android.content.Context;
import android.hardware.Camera;
import android.graphics.Point;
import android.graphics.Rect;
import android.os.Handler;
import android.util.Log;
import android.view.SurfaceHolder;

/**
 * This object wraps the Camera service object and expects to be the only one
 * talking to it. The implementation encapsulates the steps needed to take
 * preview-sized images, which are used for both preview and decoding.
 */
public final class CameraManager {
```

```
 private static final String TAG = CameraManager.class.getSimpleName();

private static final int MIN_FRAME_WIDTH = 240;
private static final int MIN_FRAME_HEIGHT = 240;
private static final int MAX_FRAME_WIDTH = 600;
private static final int MAX_FRAME_HEIGHT = 400;

private final CameraConfigurationManager configManager;
private Camera camera;
private Rect framingRect;
private Rect framingRectInPreview;
private boolean initialized;
private boolean previewing;
private int requestedFramingRectWidth;
private int requestedFramingRectHeight;

/**
 * Preview frames are delivered here, which we pass on to the registered
 * handler. Make sure to clear the handler so it will only receive one
 * message.
 */
private final PreviewCallback previewCallback;
/**
 * Autofocus callbacks arrive here, and are dispatched to the Handler which
 * requested them.
 */
private final AutoFocusCallback autoFocusCallback;

public CameraManager(Context context) {
    this.configManager = new CameraConfigurationManager(context);
    previewCallback = new PreviewCallback(configManager);
    autoFocusCallback = new AutoFocusCallback();
}

public CameraConfigurationManager getConfigurationManager() {
    return configManager;
}

public Camera getCamera() {
    return camera;
}

/**
 * Opens the camera driver and initializes the hardware parameters.
 *
 * @param holder
 *            The surface object which the camera will draw preview frames
 *            into.
 * @throws IOException
 *             Indicates the camera driver failed to open.
 */
public void openDriver(SurfaceHolder holder) throws IOException {
    Camera theCamera = camera;
    if (theCamera == null) {
        theCamera = Camera.open();
        if (theCamera == null) {
            throw new IOException();
        }
        camera = theCamera;
    }
    theCamera.setPreviewDisplay(holder);

    if (!initialized) {
        initialized = true;
        configManager.initFromCameraParameters(theCamera);
        if (requestedFramingRectWidth > 0 && requestedFramingRectHeight > 0) {
            setManualFramingRect(requestedFramingRectWidth, requestedFramingRectHeight);
            requestedFramingRectWidth = 0;
            requestedFramingRectHeight = 0;
        }
    }
    configManager.setDesiredCameraParameters(theCamera);
}

/**
 * Closes the camera driver if still in use.
 */
public void closeDriver() {
```

```
    if (camera != null) {
        camera.release();
        camera = null;
        // Make sure to clear these each time we close the camera, so that
        // any scanning rect requested by intent is forgotten.
        framingRect = null;
        framingRectInPreview = null;
    }
}

/**
 * Asks the camera hardware to begin drawing preview frames to the screen.
 */
public void startPreview() {
    Camera theCamera = camera;
    if (theCamera != null && !previewing) {
        theCamera.startPreview();
        previewing = true;
    }
}

/**
 * Tells the camera to stop drawing preview frames.
 */
public void stopPreview() {
    if (camera != null && previewing) {
        camera.stopPreview();
        previewCallback.setHandler(null, 0);
        autoFocusCallback.setHandler(null, 0);
        previewing = false;
    }
}

/**
 * A single preview frame will be returned to the handler supplied. The data
 * will arrive as byte[] in the message.obj field, with width and height
 * encoded as message.arg1 and message.arg2, respectively.
 *
 * @param handler
 *            The handler to send the message to.
 * @param message
 *            The what field of the message to be sent.
 */
public void requestPreviewFrame(Handler handler, int message) {
    Camera theCamera = camera;
    if (theCamera != null && previewing) {
        previewCallback.setHandler(handler, message);
        theCamera.setOneShotPreviewCallback(previewCallback);
    }
}

/**
 * Asks the camera hardware to perform an autofocus.
 *
 * @param handler
 *            The Handler to notify when the autofocus completes.
 * @param message
 *            The message to deliver.
 */
public void requestAutoFocus(Handler handler, int message) {
    if (camera != null && previewing) {
        autoFocusCallback.setHandler(handler, message);
        camera.autoFocus(autoFocusCallback);
    }
}

/**
 * Calculates the framing rect which the UI should draw to show the user
 * where to place the barcode. This target helps with alignment as well as
 * forces the user to hold the device far enough away to ensure the image
 * will be in focus.
 *
 * @return The rectangle to draw on screen in window coordinates.
 */
public Rect getFramingRect() {
    if (framingRect == null) {
        if (camera == null) {
            return null;
        }
```

```
        Point screenResolution = configManager.getScreenResolution();
        int width = screenResolution.x * 3 / 4;
        if (width < MIN_FRAME_WIDTH) {
            width = MIN_FRAME_WIDTH;
        } else if (width > MAX_FRAME_WIDTH) {
            width = MAX_FRAME_WIDTH;
        }
        int height = screenResolution.y * 3 / 4;
        if (height < MIN_FRAME_HEIGHT) {
            height = MIN_FRAME_HEIGHT;
        } else if (height > MAX_FRAME_HEIGHT) {
            height = MAX_FRAME_HEIGHT;
        }
        int leftOffset = (screenResolution.x - width) / 2;
        int topOffset = (screenResolution.y - height) / 2;
        framingRect = new Rect(leftOffset, topOffset, leftOffset + width, topOffset + height);
        Log.d(TAG, "Calculated framing rect: " + framingRect);
    }
    return framingRect;
}

/**
 * Like {@link #getFramingRect} but coordinates are in terms of the preview
 * frame, not UI / screen.
 */
public Rect getFramingRectInPreview() {
    if (framingRectInPreview == null) {
        Rect framingRect = getFramingRect();
        if (framingRect == null) {
            return null;
        }
        Rect rect = new Rect(framingRect);
        Point cameraResolution = configManager.getCameraResolution();
        Point screenResolution = configManager.getScreenResolution();
        rect.left = rect.left * cameraResolution.x / screenResolution.x;
        rect.right = rect.right * cameraResolution.x / screenResolution.x;
        rect.top = rect.top * cameraResolution.y / screenResolution.y;
        rect.bottom = rect.bottom * cameraResolution.y / screenResolution.y;
        framingRectInPreview = rect;
    }
    return framingRectInPreview;
}

/**
 * Allows third party apps to specify the scanning rectangle dimensions,
 * rather than determine them automatically based on screen resolution.
 *
 * @param width
 *            The width in pixels to scan.
 * @param height
 *            The height in pixels to scan.
 */
public void setManualFramingRect(int width, int height) {
    if (initialized) {
        Point screenResolution = configManager.getScreenResolution();
        if (width > screenResolution.x) {
            width = screenResolution.x;
        }
        if (height > screenResolution.y) {
            height = screenResolution.y;
        }
        int leftOffset = (screenResolution.x - width) / 2;
        int topOffset = (screenResolution.y - height) / 2;
        framingRect = new Rect(leftOffset, topOffset, leftOffset + width, topOffset + height);
        Log.d(TAG, "Calculated manual framing rect: " + framingRect);
        framingRectInPreview = null;
    } else {
        requestedFramingRectWidth = width;
        requestedFramingRectHeight = height;
    }
}

/**
 * A factory method to build the appropriate LuminanceSource object based on
 * the format of the preview buffers, as described by Camera.Parameters.
 *
 * @param data
 *            A preview frame.
 * @param width
```

```
     *            The width of the image.
     * @param height
     *            The height of the image.
     * @return A PlanarYUVLuminanceSource instance.
     */
    public PlanarYUVLuminanceSource buildLuminanceSource(byte[] data, int width, int height) {
        Rect rect = getFramingRectInPreview();
        if (rect == null) {
            return null;
        }
        // Assume it's YUV rather than die.
        return new PlanarYUVLuminanceSource(data, width, height, rect.left, rect.top, rect.width(), rect.height(),
                Preferences.KEY_REVERSE_IMAGE);
    }
}
```

A.Co_for_J.Co\src\com\specialproblem\acoforjco\qrcode\camera\PreviewCallback.java

```
package com.specialproblem.acoforjco.qrcode.camera;

import android.graphics.Point;
import android.hardware.Camera;
import android.os.Handler;
import android.os.Message;
import android.util.Log;

final class PreviewCallback implements Camera.PreviewCallback {

    private static final String TAG = PreviewCallback.class.getSimpleName();

    private final CameraConfigurationManager configManager;
    private Handler previewHandler;
    private int previewMessage;

    PreviewCallback(CameraConfigurationManager configManager) {
        this.configManager = configManager;
    }

    void setHandler(Handler previewHandler, int previewMessage) {
        this.previewHandler = previewHandler;
        this.previewMessage = previewMessage;
    }

    @Override
    public void onPreviewFrame(byte[] data, Camera camera) {
        Point cameraResolution = configManager.getCameraResolution();
        Handler thePreviewHandler = previewHandler;
        if (thePreviewHandler != null) {
            Message message = thePreviewHandler.obtainMessage(previewMessage, cameraResolution.x, cameraResolution.y,
data);
            message.sendToTarget();
            previewHandler = null;
        } else {
            Log.d(TAG, "Got preview callback, but no handler for it");
        }
    }

}
```

A.Co_for_J.Co\src\com\specialproblem\acoforjco\qrcode\data\Contents.java

```
package com.specialproblem.acoforjco.qrcode.data;

/**
 * The set of constants to use when sending the QR code scanner an Intent which
 * requests a barcode to be encoded.
 */
public final class Contents {
```

```
    private Contents() {
    }

    public static final class Type {

        /**
         * A plain text.
         *
         * Use Intent.putExtra(DATA, string).
         */
        public static final String SOMETEXT = "SOMETEXT_TYPE";

        /**
         * A customer information. Send a request to encode it as follows:
         *
         * Intent intent = new Intent(this, QRCodeEncoder.class);
         * Bundle customerBundle = new Bundle();
         * customerBundle.putString(Contents.NAMES, "Beverly Festejo");
         * customerBundle.putString(Contents.PHONES, "(02) 123-4568");
         * customerBundle.putString(Contents.NOTES, "Di ako maputi pero maganda ako hahaha");
         * intent.putExtras(customerBundle);
          */
        public static final String CUSTOMER = "CUSTOMER_TYPE";

        /**
         * A client information. Send a request to encode it as follows:
         *
         * Intent intent = new Intent(this, QRCodeEncoder.class);
         * Bundle customerBundle = new Bundle();
         * customerBundle.putString(Contents.QUEUES, "Customer No. 001");
         * customerBundle.putString(Contents.TIMES, "2014-03-22 09:50:58.007");
         * customerBundle.putString(Contents.URLS, "https://www.youtube.com/watch?v=oPQ3o14ksaM");
         * intent.putExtras(customerBundle);
         */
         public static final String CLIENT = "CLIENT_TYPE";

        private Type() {
        }
    }

    public static final String NAMES = "NAMES";
     public static final String PHONES = "PHONES";
    public static final String NOTES = "NOTES";
    public static final String QUEUES = "QUEUES";
    public static final String TIMES = "TIMES";
    public static final String URLS = "URLS";

}
```

A.Co_for_J.Co\src\com\specialproblem\acoforjco\qrcode\data\Preferences.java

```
package com.specialproblem.acoforjco.qrcode.data;

public class Preferences {

     private Preferences() {
    };

    public static boolean KEY_DECODE_QR = true;

    public static boolean KEY_REVERSE_IMAGE = false;
    public static boolean KEY_FRONT_LIGHT = false;
}
```

A.Co_for_J.Co\src\com\specialproblem\acoforjco\qrcode\encode\QRCodeEncoder.java

```
package com.specialproblem.acoforjco.qrcode.encode;

import java.util.EnumMap;
import java.util.Map;
```

```
import android.graphics.Bitmap;
import android.os.Bundle;
import android.telephony.PhoneNumberUtils;

import com.specialproblem.acoforjco.qrcode.data.Contents;
import com.google.zxing.BarcodeFormat;
import com.google.zxing.EncodeHintType;
import com.google.zxing.MultiFormatWriter;
import com.google.zxing.WriterException;
import com.google.zxing.common.BitMatrix;

/**
 * This class does the work of decoding the user's request and extracting all
 * the data to be encoded in a barcode.
 */
public final class QRCodeEncoder {

    private static final int WHITE = 0xFFFFFFFF;
    private static final int BLACK = 0xFF000000;

    private int dimension = Integer.MIN_VALUE;
    private String contents = null;
    private String displayContents = null;
    private String title = null;
    private BarcodeFormat format = null;
    private boolean encoded = false;


    public QRCodeEncoder(String data, Bundle bundle, String type, String format, int dimension) {
        this.dimension = dimension;
        this.format = BarcodeFormat.QR_CODE;
        encoded = encodeContents(data, bundle, type);
    }

    public String getContents() {
        return contents;
    }

    public String getDisplayContents() {
        return displayContents;
    }

    public String getTitle() {
        return title;
    }

    private boolean encodeContents(String data, Bundle bundle, String type) {
        // Default to QR_CODE.
        encodeQRCodeContents(data, bundle, type);
        return contents != null && contents.length() > 0;
    }

    private void encodeQRCodeContents(String data, Bundle bundle, String type) {
        if (type.equals(Contents.Type.SOMETEXT)) {
            if (data != null && data.length() > 0) {
                contents = data;
                displayContents = data;
                title = "Text";
            }
        } else if (type.equals(Contents.Type.CUSTOMER)) {
            if (bundle != null) {
                StringBuilder newContents = new StringBuilder(500);
                StringBuilder newDisplayContents = new StringBuilder(500);

                newContents.append("MECARD:");

                String name = trim(bundle.getString(Contents.NAMES));
                if (name != null) {
                    newContents.append("N:").append(escapeMECARD(name)).append(';');
                    newDisplayContents.append(name);
                }

                String phone = trim(bundle.getString(Contents.PHONES));
                if (phone != null) {
                    newContents.append("TEL:").append(escapeMECARD(phone)).append(';');
                    newDisplayContents.append('\n').append(PhoneNumberUtils.formatNumber(phone));
                }
```

```
            String note = trim(bundle.getString(Contents.NOTES));
            if (note != null) {
                newContents.append("NOTE:").append(escapeMECARD(note)).append(';');
                newDisplayContents.append('\n').append(note);
            }

            // Make sure we've encoded at least one field.
            if (newDisplayContents.length() > 0) {
                newContents.append(';');
                contents = newContents.toString();
                displayContents = newDisplayContents.toString();
                title = "Customer";
            } else {
                contents = null;
                displayContents = null;
            }
        }
    }else if (type.equals(Contents.Type.CLIENT)) {
        if (bundle != null) {
            StringBuilder newContents = new StringBuilder(500);
            StringBuilder newDisplayContents = new StringBuilder(500);

            newContents.append("MECARD:");

            String queueNumber = trim(bundle.getString(Contents.QUEUES));
            if (queueNumber != null) {
                newContents.append("N:").append(escapeMECARD(queueNumber)).append(';');
                newDisplayContents.append(queueNumber);
            }

            String timeStamp = trim(bundle.getString(Contents.TIMES));
            if (timeStamp != null) {
                newContents.append("NOTE:").append(escapeMECARD(timeStamp)).append(';');
                newDisplayContents.append('\n').append(timeStamp);
            }

            String redirection = trim(bundle.getString(Contents.URLS));
            if (redirection != null) {
                // escapeMECARD(url) -> wrong escape e.g. http\://zxing.google.com
                newContents.append("URL:").append(redirection).append(';');
                newDisplayContents.append('\n').append(redirection);
            }

            // Make sure we've encoded at least one field.
            if (newDisplayContents.length() > 0) {
                newContents.append(';');
                contents = newContents.toString();
                displayContents = newDisplayContents.toString();
                title = "Customer";
            } else {
                contents = null;
                displayContents = null;
            }
        }
    }
}

public Bitmap encodeAsBitmap() throws WriterException {
    if (!encoded)
        return null;

    Map<EncodeHintType, Object> hints = null;
    String encoding = guessAppropriateEncoding(contents);
    if (encoding != null) {
        hints = new EnumMap<EncodeHintType, Object>(EncodeHintType.class);
        hints.put(EncodeHintType.CHARACTER_SET, encoding);
    }

    MultiFormatWriter writer = new MultiFormatWriter();
    BitMatrix result = writer.encode(contents, format, dimension, dimension, hints);
    int width = result.getWidth();
    int height = result.getHeight();
    int[] pixels = new int[width * height];
    // All are 0, or black, by default
    for (int y = 0; y < height; y++) {
        int offset = y * width;
        for (int x = 0; x < width; x++) {
            pixels[offset + x] = result.get(x, y) ? BLACK : WHITE;
        }
```

```
        }

        Bitmap bitmap = Bitmap.createBitmap(width, height, Bitmap.Config.ARGB_8888);
        bitmap.setPixels(pixels, 0, width, 0, 0, width, height);
        return bitmap;
    }

    private static String guessAppropriateEncoding(CharSequence contents) {
        // Very crude at the moment
        for (int i = 0; i < contents.length(); i++) {
            if (contents.charAt(i) > 0xFF) {
                return "UTF-8";
            }
        }
        return null;
    }

    private static String trim(String s) {
        if (s == null) {
            return null;
        }
        String result = s.trim();
        return result.length() == 0 ? null : result;
    }

    private static String escapeMECARD(String input) {
        if (input == null || (input.indexOf(':') < 0 && input.indexOf(';') < 0)) {
            return input;
        }
        int length = input.length();
        StringBuilder result = new StringBuilder(length);
        for (int i = 0; i < length; i++) {
            char c = input.charAt(i);
            if (c == ':' || c == ';') {
                result.append('\\');
            }
            result.append(c);
        }
        return result.toString();
    }
}
```

A.Co_for_J.Co\src\com\specialproblem\acoforjco\qrcode\history\DBHelper.java

```
package com.specialproblem.acoforjco.qrcode.history;

import android.database.sqlite.SQLiteOpenHelper;
import android.database.sqlite.SQLiteDatabase;
import android.content.Context;

final class DBHelper extends SQLiteOpenHelper {

     private static final int DB_VERSION = 5;
     private static final String DB_NAME = "qr_code_scanner_history.db";
     static final String TABLE_NAME = "history";
     static final String ID_COL = "id";
     static final String TEXT_COL = "text";
     static final String FORMAT_COL = "format";
     static final String DISPLAY_COL = "display";
     static final String TIMESTAMP_COL = "timestamp";
     static final String DETAILS_COL = "details";

     DBHelper(Context context) {
          super(context, DB_NAME, null, DB_VERSION);
     }

     @Override
     public void onCreate(SQLiteDatabase sqLiteDatabase) {
          sqLiteDatabase.execSQL(
                    "CREATE TABLE " + TABLE_NAME + " (" +
                    ID_COL + " INTEGER PRIMARY KEY, " +
                    TEXT_COL + " TEXT, " +
                    FORMAT_COL + " TEXT, " +
                    DISPLAY_COL + " TEXT, " +
                    TIMESTAMP_COL + " INTEGER, " +
```

```
                DETAILS_COL + " TEXT);");
    }

    @Override
    public void onUpgrade(SQLiteDatabase sqLiteDatabase, int oldVersion, int newVersion) {
        sqLiteDatabase.execSQL("DROP TABLE IF EXISTS " + TABLE_NAME);
        onCreate(sqLiteDatabase);
    }

}
```

A.Co_for_J.Co\src\com\specialproblem\acoforjco\qrcode\history\HistoryActivity.java

```
package com.specialproblem.acoforjco.qrcode.history;

import android.app.Activity;
import android.app.AlertDialog;
import android.app.ListActivity;
import android.content.DialogInterface;
import android.content.Intent;
import android.os.Bundle;
import android.view.ContextMenu;
import android.view.Menu;
import android.view.MenuInflater;
import android.view.MenuItem;
import android.view.View;
import android.widget.AdapterView;
import android.widget.ArrayAdapter;
import android.widget.ListView;
import com.specialproblem.acoforjco.qrcode.CaptureActivity;
import com.specialproblem.acoforjco.qrcode.Intents;
import com.specialproblem.acoforjco.R;

public final class HistoryActivity extends ListActivity {

    private HistoryManager historyManager;
    private ArrayAdapter<HistoryItem> adapter;
    private CharSequence originalTitle;

    @Override
    protected void onCreate(Bundle icicle) {
        super.onCreate(icicle);
        this.historyManager = new HistoryManager(this);
        adapter = new HistoryItemAdapter(this);
        setListAdapter(adapter);
        View listview = getListView();
        registerForContextMenu(listview);
        originalTitle = getTitle();
    }

    @Override
    protected void onResume() {
        super.onResume();
        reloadHistoryItems();
    }

    private void reloadHistoryItems() {
        Iterable<HistoryItem> items = historyManager.buildHistoryItems();
        adapter.clear();
        for (HistoryItem item : items) {
            adapter.add(item);
        }
        setTitle(originalTitle + " [" + adapter.getCount() + ']');
        if (adapter.isEmpty()) {
            adapter.add(new HistoryItem(null, null, null));
        }
    }

    @Override
    protected void onListItemClick(ListView l, View v, int position, long id) {
        if (adapter.getItem(position).getResult() != null) {
            Intent intent = new Intent(this, CaptureActivity.class);
            intent.putExtra(Intents.History.ITEM_NUMBER, position);
```

```
                setResult(Activity.RESULT_OK, intent);
                finish();
            }
        }

        @Override
        public void onCreateContextMenu(ContextMenu menu,
                View v,
                ContextMenu.ContextMenuInfo menuInfo) {
            int position = ((AdapterView.AdapterContextMenuInfo) menuInfo).position;
            if (position >= adapter.getCount() || adapter.getItem(position).getResult() != null) {
                menu.add(Menu.NONE, position, position, R.string.history_clear_one_history);
            } // else it's just that dummy "Empty" message
        }

        @Override
        public boolean onContextItemSelected(MenuItem item) {
            int position = item.getItemId();
            historyManager.deleteHistoryItem(position);
            reloadHistoryItems();
            return true;
        }

        @Override
        public boolean onCreateOptionsMenu(Menu menu) {
            if (historyManager.hasHistoryItems()) {
                MenuInflater menuInflater = getMenuInflater();
                menuInflater.inflate(R.menu.history, menu);
            }
            return super.onCreateOptionsMenu(menu);
        }

        @Override
        public boolean onOptionsItemSelected(MenuItem item) {
            switch (item.getItemId()) {
            case R.id.menu_history_clear:
                AlertDialog.Builder builder = new AlertDialog.Builder(this);
                builder.setMessage(R.string.msg_sure);
                builder.setCancelable(true);
                builder.setPositiveButton(R.string.button_ok, new DialogInterface.OnClickListener() {
                    @Override
                    public void onClick(DialogInterface dialog, int i2) {
                        historyManager.clearHistory();
                        dialog.dismiss();
                        finish();
                    }
                });
                builder.setNegativeButton(R.string.button_cancel, null);
                builder.show();
                break;
            default:
                return super.onOptionsItemSelected(item);
            }
            return true;
        }

}
```

A.Co_for_J.Co\src\com\specialproblem\acoforjco\qrcode\history\HistoryItem.java

```
package com.specialproblem.acoforjco.qrcode.history;

import android.annotation.SuppressLint;
import com.google.zxing.Result;

public final class HistoryItem {

    private final Result result;
    private final String display;
    private final String details;

    HistoryItem(Result result, String display, String details) {
        this.result = result;
        this.display = display;
        this.details = details;
    }
```

```
    public Result getResult() {
        return result;
    }

    @SuppressLint("NewApi")
    public String getDisplayAndDetails() {
        StringBuilder displayResult = new StringBuilder();
        if (display == null || display.isEmpty()) {
            displayResult.append(result.getText());
        } else {
            displayResult.append(display);
        }
        if (details != null && !details.isEmpty()) {
            displayResult.append(" : ").append(details);
        }
        return displayResult.toString();
    }

}
```

A.Co_for_J.Co\src\com\specialproblem\acoforjco\qrcode\history\HistoryItemAdapter.java

```
package com.specialproblem.acoforjco.qrcode.history;

import android.content.Context;
import android.content.res.Resources;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.ArrayAdapter;
import android.widget.LinearLayout;
import android.widget.TextView;
import com.google.zxing.Result;
import com.specialproblem.acoforjco.R;

import java.text.DateFormat;
import java.util.ArrayList;
import java.util.Date;

final class HistoryItemAdapter extends ArrayAdapter<HistoryItem> {

    private final Context activity;

    HistoryItemAdapter(Context activity) {
        super(activity, R.layout.activity_history, new ArrayList<HistoryItem>());
        this.activity = activity;
    }

    @Override
    public View getView(int position, View view, ViewGroup viewGroup) {
        View layout;
        if (view instanceof LinearLayout) {
            layout = view;
        } else {
            LayoutInflater factory = LayoutInflater.from(activity);
            layout = factory.inflate(R.layout.activity_history, viewGroup, false);
        }

        HistoryItem item = getItem(position);
        Result result = item.getResult();
        DateFormat formatter = DateFormat.getDateTimeInstance(DateFormat.SHORT, DateFormat.SHORT);

        CharSequence title;
        CharSequence detail;
        if (result != null) {
            title = "A.CO QR Code produced on " + formatter.format(new Date(result.getTimestamp()));
            detail = item.getDisplayAndDetails();
        } else {
            Resources resources = getContext().getResources();
            title = resources.getString(R.string.history_empty);
            detail = resources.getString(R.string.history_empty_detail);
        }
```

```
        ((TextView) layout.findViewById(R.id.history_title)).setText(title);
        ((TextView) layout.findViewById(R.id.history_detail)).setText(detail);

        return layout;
    }

}
```

A.Co_for_J.Co\src\com\specialproblem\acoforjco\qrcode\history\HistoryManager.java

```
package com.specialproblem.acoforjco.qrcode.history;

import android.database.sqlite.SQLiteException;
import com.google.zxing.BarcodeFormat;
import com.google.zxing.Result;
import com.specialproblem.acoforjco.qrcode.result.ResultHandler;

import android.app.Activity;
import android.content.ContentValues;
import android.database.Cursor;
import android.database.sqlite.SQLiteDatabase;
import android.database.sqlite.SQLiteOpenHelper;
import android.util.Log;

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

/**
 * Manages functionality related to scan history.
 */
public final class HistoryManager {

    private static final String TAG = HistoryManager.class.getSimpleName();

    private static final int MAX_ITEMS = 2000;

    private static final String[] COLUMNS = {
        DBHelper.TEXT_COL,
        DBHelper.DISPLAY_COL,
        DBHelper.FORMAT_COL,
        DBHelper.TIMESTAMP_COL,
        DBHelper.DETAILS_COL,
    };

    private static final String[] COUNT_COLUMN = { "COUNT(1)" };

    private static final String[] ID_COL_PROJECTION = { DBHelper.ID_COL };

    private final Activity activity;

    public HistoryManager(Activity activity) {
        this.activity = activity;
    }

    public boolean hasHistoryItems() {
        SQLiteOpenHelper helper = new DBHelper(activity);
        SQLiteDatabase db = null;
        Cursor cursor = null;
        try {
            db = helper.getReadableDatabase();
            cursor = db.query(DBHelper.TABLE_NAME, COUNT_COLUMN, null, null, null, null, null);
            cursor.moveToFirst();
            return cursor.getInt(0) > 0;
        } finally {
            close(cursor, db);
        }
    }

    public List<HistoryItem> buildHistoryItems() {
        SQLiteOpenHelper helper = new DBHelper(activity);
        List<HistoryItem> items = new ArrayList<HistoryItem>();
        SQLiteDatabase db = null;
```

```
        Cursor cursor = null;
        try {
            db = helper.getReadableDatabase();
            cursor = db.query(DBHelper.TABLE_NAME, COLUMNS, null, null, null, null, DBHelper.TIMESTAMP_COL + " DESC");
            while (cursor.moveToNext()) {
                String text = cursor.getString(0);
                String display = cursor.getString(1);
                String format = cursor.getString(2);
                long timestamp = cursor.getLong(3);
                String details = cursor.getString(4);
                Result result = new Result(text, null, null, BarcodeFormat.valueOf(format), timestamp);
                items.add(new HistoryItem(result, display, details));
            }
        } finally {
            close(cursor, db);
        }
        return items;
    }

    public HistoryItem buildHistoryItem(int number) {
        SQLiteOpenHelper helper = new DBHelper(activity);
        SQLiteDatabase db = null;
        Cursor cursor = null;
        try {
            db = helper.getReadableDatabase();
            cursor = db.query(DBHelper.TABLE_NAME, COLUMNS, null, null, null, null, DBHelper.TIMESTAMP_COL + " DESC");
            cursor.move(number + 1);
            String text = cursor.getString(0);
            String display = cursor.getString(1);
            String format = cursor.getString(2);
            long timestamp = cursor.getLong(3);
            String details = cursor.getString(4);
            Result result = new Result(text, null, null, BarcodeFormat.valueOf(format), timestamp);
            return new HistoryItem(result, display, details);
        } finally {
            close(cursor, db);
        }
    }

    public void deleteHistoryItem(int number) {
        SQLiteOpenHelper helper = new DBHelper(activity);
        SQLiteDatabase db = null;
        Cursor cursor = null;
        try {
            db = helper.getWritableDatabase();
            cursor = db.query(DBHelper.TABLE_NAME,
                    ID_COL_PROJECTION,
                    null, null, null, null,
                    DBHelper.TIMESTAMP_COL + " DESC");
            cursor.move(number + 1);
            db.delete(DBHelper.TABLE_NAME, DBHelper.ID_COL + '=' + cursor.getString(0), null);
        } finally {
            close(cursor, db);
        }
    }

    public void addHistoryItem(Result result, ResultHandler handler) {
        ContentValues values = new ContentValues();
        values.put(DBHelper.TEXT_COL, result.getText());
        values.put(DBHelper.FORMAT_COL, result.getBarcodeFormat().toString());
        values.put(DBHelper.DISPLAY_COL, handler.getDisplayContents().toString());
        values.put(DBHelper.TIMESTAMP_COL, System.currentTimeMillis());

        SQLiteOpenHelper helper = new DBHelper(activity);
        SQLiteDatabase db = null;
        try {
            db = helper.getWritableDatabase();
            // Insert the new entry into the DB.
            db.insert(DBHelper.TABLE_NAME, DBHelper.TIMESTAMP_COL, values);
        } finally {
            close(null, db);
        }
    }

    public void trimHistory() {
        SQLiteOpenHelper helper = new DBHelper(activity);
        SQLiteDatabase db = null;
        Cursor cursor = null;
        try {
```

```
            db = helper.getWritableDatabase();
            cursor = db.query(DBHelper.TABLE_NAME,
                     ID_COL_PROJECTION,
                     null, null, null, null,
                     DBHelper.TIMESTAMP_COL + " DESC");
            cursor.move(MAX_ITEMS);
            while (cursor.moveToNext()) {
                 String id = cursor.getString(0);
                 Log.i(TAG, "Deleting scan history ID " + id);
                 db.delete(DBHelper.TABLE_NAME, DBHelper.ID_COL + '=' + id, null);
            }
        } catch (SQLiteException sqle) {
            // We're seeing an error here when called in CaptureActivity.onCreate() in rare cases
            // and don't understand it. First theory is that it's transient so can be safely ignored.
            Log.w(TAG, sqle);
            // continue
        } finally {
            close(cursor, db);
        }
    }

    void clearHistory() {
        SQLiteOpenHelper helper = new DBHelper(activity);
        SQLiteDatabase db = null;
        try {
            db = helper.getWritableDatabase();
            db.delete(DBHelper.TABLE_NAME, null, null);
        } finally {
            close(null, db);
        }
    }

    private static void close(Cursor cursor, SQLiteDatabase database) {
        if (cursor != null) {
            cursor.close();
        }
        if (database != null) {
            database.close();
        }
    }

}
```

A.Co_for_J.Co\src\com\specialproblem\acoforjco\qrcode\result\AddressBookResultHandler.java

```
package com.specialproblem.acoforjco.qrcode.result;

import java.text.DateFormat;
import java.text.ParsePosition;
import java.text.SimpleDateFormat;
import java.util.Date;
import java.util.Locale;

import android.app.Activity;
import android.telephony.PhoneNumberUtils;
import android.text.Spannable;
import android.text.SpannableString;
import android.text.style.StyleSpan;

import com.specialproblem.acoforjco.R;
import com.google.zxing.client.result.AddressBookParsedResult;
import com.google.zxing.client.result.ParsedResult;

/**
 * Handles address book entries.
 */
public final class AddressBookResultHandler extends ResultHandler {

    private static final DateFormat[] DATE_FORMATS = {
     new SimpleDateFormat("yyyyMMdd", Locale.ENGLISH),
        new SimpleDateFormat("yyyyMMdd HHmmss", Locale.ENGLISH),
        new SimpleDateFormat("yyyy-MM-dd", Locale.ENGLISH),
        new SimpleDateFormat("yyyy-MM-dd HH:mm:ss.SSS", Locale.ENGLISH), };
```

```
    private final boolean[] fields;

    public AddressBookResultHandler(Activity activity, ParsedResult result) {
        super(activity, result);
        AddressBookParsedResult addressResult = (AddressBookParsedResult) result;
        String[] addresses = addressResult.getAddresses();
        boolean hasAddress = addresses != null && addresses.length > 0 && addresses[0].length() > 0;
        String[] phoneNumbers = addressResult.getPhoneNumbers();
        boolean hasPhoneNumber = phoneNumbers != null && phoneNumbers.length > 0;
        String[] emails = addressResult.getEmails();
        boolean hasEmailAddress = emails != null && emails.length > 0;

        fields = new boolean[4];
        fields[0] = true; // Add contact is always available
        fields[1] = hasAddress;
        fields[2] = hasPhoneNumber;
        fields[3] = hasEmailAddress;
    }

    // Overriden so we can hyphenate phone numbers, format time ins, and bold the name.
    @Override
    public CharSequence getDisplayContents() {
        AddressBookParsedResult result = (AddressBookParsedResult) getResult();
        StringBuilder contents = new StringBuilder(500);
        ParsedResult.maybeAppend(result.getNames(), contents);
        int namesLength = contents.length();

        String[] numbers = result.getPhoneNumbers();
        if (numbers != null) {
            for (String number : numbers) {
                ParsedResult.maybeAppend(PhoneNumberUtils.formatNumber(number), contents);
            }
        }

        ParsedResult.maybeAppend(result.getURL(), contents);

        String dayIn = result.getBirthday();
        if (dayIn != null && dayIn.length() > 0) {
            Date date = parseDate(dayIn);
            if (date != null) {
                ParsedResult.maybeAppend(DateFormat.getDateInstance().format(date.getTime()), contents);
            }
        }

        ParsedResult.maybeAppend(result.getNote(), contents);

        if (namesLength > 0) {
            // Bold the full name to make it stand out a bit.
            Spannable styled = new SpannableString(contents.toString());
            styled.setSpan(new StyleSpan(android.graphics.Typeface.BOLD), 0, namesLength, 0);
            return styled;
        }

        return contents.toString();
    }

    private static Date parseDate(String s) {
        for (DateFormat currentFormat : DATE_FORMATS) {
            synchronized (currentFormat) {
                currentFormat.setLenient(false);
                Date result = currentFormat.parse(s, new ParsePosition(0));
                if (result != null) {
                    return result;
                }
            }
        }
        return null;
    }

    @Override
    public int getDisplayTitle() {
        return R.string.result_address_book;
    }
}
```

A.Co_for_J.Co\src\com\specialproblem\acoforjco\qrcode\result\ResultHandler.java

```java
package com.specialproblem.acoforjco.qrcode.result;

import android.app.Activity;

import com.google.zxing.Result;
import com.google.zxing.client.result.ParsedResult;
import com.google.zxing.client.result.ParsedResultType;

/**
 * A base class for the Android-specific barcode handlers. These allow the app
 * to polymorphically suggest the appropriate actions for each data type.
 *
 * This class also contains a bunch of utility methods to take common actions
 * like opening a URL. They could easily be moved into a helper object, but it
 * can't be static because the Activity instance is needed to launch an intent.
 */

public abstract class ResultHandler {

    private final ParsedResult result;
    private final Activity activity;
    private final Result rawResult;

    ResultHandler(Activity activity, ParsedResult result) {
        this(activity, result, null);
    }

    ResultHandler(Activity activity, ParsedResult result, Result rawResult) {
        this.result = result;
        this.activity = activity;
        this.rawResult = rawResult;
    }

    public ParsedResult getResult() {
        return result;
    }

    Activity getActivity() {
        return activity;
    }

    public Result getRawResult() {
        return rawResult;
    }

    /**
     * Some barcode contents are considered secure, and should not be saved to
     * history, copied to the clipboard, or otherwise persisted.
     *
     * @return If true, do not create any permanent record of these contents.
     */
    public boolean areContentsSecure() {
        return false;
    }

    /**
     * Create a possibly styled string for the contents of the current barcode.
     *
     * @return The text to be displayed.
     */
    public CharSequence getDisplayContents() {
        String contents = result.getDisplayResult();
        return contents.replace("\r", "");
    }

    /**
     * A string describing the kind of barcode that was found, e.g.
     * "Found contact info".
     *
     * @return The resource ID of the string.
     */
    public abstract int getDisplayTitle();

    /**
     * A convenient method to get the parsed type. Should not be overridden.
     *
     * @return The parsed type, e.g. URI or ISBN
     */
    public final ParsedResultType getType() {
```

```
        return result.getType();
    }

}
```

A.Co_for_J.Co\src\com\specialproblem\acoforjco\qrcode\result\ResultHandlerFactory.java

```
package com.specialproblem.acoforjco.qrcode.result;

import android.app.Activity;

import com.google.zxing.Result;
import com.google.zxing.client.result.ParsedResult;
import com.google.zxing.client.result.ResultParser;

/**
 * Manufactures Android-specific handlers based on the barcode content's type.
 */

public final class ResultHandlerFactory {

    private ResultHandlerFactory() {
    }

    public static ResultHandler makeResultHandler(Activity activity, Result rawResult) {
        ParsedResult result = parseResult(rawResult);
        switch (result.getType()) {
        case ADDRESSBOOK:
            return new AddressBookResultHandler(activity, result);
        default:
          return new TextResultHandler(activity, result, rawResult);
        }
    }

    private static ParsedResult parseResult(Result rawResult) {
        return ResultParser.parseResult(rawResult);
    }
}
```

A.Co_for_J.Co\src\com\specialproblem\acoforjco\qrcode\result\TextResultHandler.java

```
package com.specialproblem.acoforjco.qrcode.result;

import android.app.Activity;

import com.google.zxing.Result;
import com.google.zxing.client.result.ParsedResult;
import com.google.zxing.client.result.TextParsedResult;
import com.specialproblem.acoforjco.R;

/**
 * This class handles TextParsedResult.
 */
public final class TextResultHandler extends ResultHandler {

     public TextResultHandler(Activity activity, ParsedResult result, Result rawResult) {
        super(activity, result, rawResult);
    }

    @Override
    public CharSequence getDisplayContents() {
        TextParsedResult result = (TextParsedResult) getResult();
        StringBuilder contents = new StringBuilder(500);
        ParsedResult.maybeAppend(result.getText(), contents);
        contents.trimToSize();
        return contents.toString();
```

```
    }

    @Override
    public int getDisplayTitle() {
        return R.string.result_text;
    }

}
```

A.Co_for_J.Co\src\com\specialproblem\acoforjco\qrcode\CaptureActivity.java

```
package com.specialproblem.acoforjco.qrcode;

import java.io.IOException;
import java.net.HttpURLConnection;
import java.net.MalformedURLException;
import java.net.URL;
import java.text.DateFormat;
import java.text.SimpleDateFormat;
import java.util.Collection;
import java.util.Date;
import java.util.EnumSet;
import java.util.Map;
import java.util.Set;

import org.json.JSONException;
import org.json.JSONObject;

import com.google.zxing.BarcodeFormat;
import com.google.zxing.Result;
import com.google.zxing.ResultMetadataType;
import com.google.zxing.ResultPoint;
import com.specialproblem.acoforjco.library.DatabaseHandler;
import com.specialproblem.acoforjco.library.UserFunctions;
import com.specialproblem.acoforjco.qrcode.history.HistoryActivity;
import com.specialproblem.acoforjco.qrcode.history.HistoryItem;
import com.specialproblem.acoforjco.qrcode.history.HistoryManager;
import com.specialproblem.acoforjco.qrcode.camera.CameraManager;
import com.specialproblem.acoforjco.qrcode.result.ResultHandler;
import com.specialproblem.acoforjco.qrcode.result.ResultHandlerFactory;
import com.specialproblem.acoforjco.DashboardActivity;
import com.specialproblem.acoforjco.R;

import android.app.Activity;
import android.app.ProgressDialog;
import android.content.Context;
import android.content.Intent;
import android.graphics.Bitmap;
import android.graphics.BitmapFactory;
import android.graphics.Canvas;
import android.graphics.Paint;
import android.graphics.Rect;
import android.net.ConnectivityManager;
import android.net.NetworkInfo;
import android.os.AsyncTask;
import android.os.Bundle;
import android.os.Handler;
import android.os.Message;
import android.util.Log;
import android.util.TypedValue;
import android.view.KeyEvent;
import android.view.Menu;
import android.view.MenuInflater;
import android.view.MenuItem;
import android.view.SurfaceHolder;
import android.view.SurfaceView;
import android.view.View;
import android.view.Window;
import android.view.WindowManager;
import android.widget.Button;
import android.widget.ImageView;
import android.widget.TextView;

/**
 * This activity opens the camera and does the actual scanning on a background thread. It draws a
 * viewfinder to help the user place the barcode correctly, shows feedback as the image processing
```

```
 * is happening, and then overlays the results when a scan is successful.
 */
public class CaptureActivity extends Activity implements IDecoderActivity, SurfaceHolder.Callback {

     private static final String TAG = CaptureActivity.class.getSimpleName();
     public static final int HISTORY_REQUEST_CODE = 0x0000bacc;

     private static final Set<ResultMetadataType> DISPLAYABLE_METADATA_TYPES =
EnumSet.of(ResultMetadataType.ISSUE_NUMBER, ResultMetadataType.SUGGESTED_PRICE,
          ResultMetadataType.ERROR_CORRECTION_LEVEL, ResultMetadataType.POSSIBLE_COUNTRY);

     private CaptureActivityHandler handler = null;
    private Result savedResultToShow;
    private ViewfinderView viewfinderView = null;
    private CameraManager cameraManager = null;
    private boolean hasSurface = false;
    private Collection<BarcodeFormat> decodeFormats = null;
    private String characterSet = null;
    private HistoryManager historyManager;

    private TextView statusView = null;
    private TextView captureErrorMsg;
    private View resultView = null;
    private boolean inScanMode = false;

    private static String KEY_SUCCESS = "success";
     private static String KEY_UID = "uid";
     private static String KEY_FIRSTNAME = "fname";
     private static String KEY_LASTNAME = "lname";
     private static String KEY_IDENTIFIER = "identifier";
     private static String KEY_DESCRIPTION = "description";
     private static String KEY_CREATED_AT = "created_at";

     private String identifier, password;


     @Override
     public void onCreate(Bundle savedInstanceState) {
          super.onCreate(savedInstanceState);
          setContentView(R.layout.activity_capture);
        Log.v(TAG, "onCreate()");

        Window window = getWindow();
        window.addFlags(WindowManager.LayoutParams.FLAG_KEEP_SCREEN_ON);

        handler = null;
        hasSurface = false;

        resultView = findViewById(R.id.result_view);
        statusView = (TextView) findViewById(R.id.status_view);
        inScanMode = false;

        historyManager = new HistoryManager(this);
        historyManager.trimHistory();

        captureErrorMsg = (TextView) findViewById(R.id.capture_error);
    }

     @Override
    protected void onDestroy() {
        super.onDestroy();
        Log.v(TAG, "onDestroy()");
    }

     @Override
    protected void onResume() {
        super.onResume();
        Log.v(TAG, "onResume()");

        // CameraManager must be initialized here.
        if (cameraManager == null) cameraManager = new CameraManager(getApplication());

        if (viewfinderView == null) {
            viewfinderView = (ViewfinderView) findViewById(R.id.viewfinder_view);
            viewfinderView.setCameraManager(cameraManager);
        }

        showScanner();
```

```
    SurfaceView surfaceView = (SurfaceView) findViewById(R.id.preview_view);
    SurfaceHolder surfaceHolder = surfaceView.getHolder();
    if (hasSurface) {
        // The activity was paused but not stopped, so the surface still exists.
     // Therefore surfaceCreated() won't be called, so init the camera here.
        initCamera(surfaceHolder);
    } else {
        // Install the callback and wait for surfaceCreated() to init the
        // camera.
        surfaceHolder.addCallback(this);
        surfaceHolder.setType(SurfaceHolder.SURFACE_TYPE_PUSH_BUFFERS);
    }
}

@Override
protected void onPause() {
    super.onPause();
    Log.v(TAG, "onPause()");

    if (handler != null) {
        handler.quitSynchronously();
        handler = null;
    }

    cameraManager.closeDriver();

    if (!hasSurface) {
        SurfaceView surfaceView = (SurfaceView) findViewById(R.id.preview_view);
        SurfaceHolder surfaceHolder = surfaceView.getHolder();
        surfaceHolder.removeCallback(this);
    }
}

@Override
public boolean onKeyDown(int keyCode, KeyEvent event) {
    if (keyCode == KeyEvent.KEYCODE_BACK) {
        if (inScanMode)
            finish();
        else
            onResume();
        return true;
    }
    return super.onKeyDown(keyCode, event);
}

@Override
public boolean onCreateOptionsMenu(Menu menu){
     MenuInflater inflater = getMenuInflater();
     inflater.inflate(R.menu.capture, menu);
     return super.onCreateOptionsMenu(menu);
}

@Override
public boolean onOptionsItemSelected(MenuItem item){
     Intent intent = new Intent(Intent.ACTION_VIEW);
     intent.addFlags(Intent.FLAG_ACTIVITY_CLEAR_WHEN_TASK_RESET);

     // Handles presses on the action bar item
     switch (item.getItemId()) {
     case R.id.menu_history:
          intent.setClassName(this, HistoryActivity.class.getName());
          startActivityForResult(intent, HISTORY_REQUEST_CODE);
          break;
     default:
          return super.onOptionsItemSelected(item);
     }
     return true;
}

@Override
public void onActivityResult(int requestCode, int resultCode, Intent intent) {
     if (resultCode == RESULT_OK) {
          if (requestCode == HISTORY_REQUEST_CODE) {
               int itemNumber = intent.getIntExtra(Intents.History.ITEM_NUMBER, -1);
               if (itemNumber >= 0) {
                    HistoryItem historyItem = historyManager.buildHistoryItem(itemNumber);
                    decodeOrStoreSavedBitmap(null, historyItem.getResult());
               }
          }
```

```
        }
    }

    private void decodeOrStoreSavedBitmap(Bitmap bitmap, Result result) {
        // Bitmap isn't used yet -- will be used soon
        if (handler == null) {
            savedResultToShow = result;
        } else {
            if (result != null) {
                savedResultToShow = result;
            }
            if (savedResultToShow != null) {
                Message message = Message.obtain(handler, R.id.decode_succeeded, savedResultToShow);
                handler.sendMessage(message);
            }
            savedResultToShow = null;
        }
    }

    @Override
    public void surfaceCreated(SurfaceHolder holder) {
        if (holder == null)
            Log.e(TAG, "*** WARNING *** surfaceCreated() gave us a null surface!");
        if (!hasSurface) {
            hasSurface = true;
            initCamera(holder);
        }
    }

    @Override
    public void surfaceDestroyed(SurfaceHolder holder) {
        hasSurface = false;
    }

    @Override
    public void surfaceChanged(SurfaceHolder holder, int format, int width, int height) {
        // Ignore
    }

    @Override
    public ViewfinderView getViewfinder() {
        return viewfinderView;
    }

    @Override
    public Handler getHandler() {
        return handler;
    }

    @Override
    public CameraManager getCameraManager() {
        return cameraManager;
    }

    /**
     * A valid barcode has been found, so give an indication of success and show the results.
     *
     * @param rawResult The contents of the barcode.
     * @param barcode   A greyscale bitmap of the camera data which was decoded.
     */
    @Override
    public void handleDecode(Result rawResult, Bitmap barcode) {
        ResultHandler resultHandler = ResultHandlerFactory.makeResultHandler(this, rawResult);

        boolean fromLiveScan = barcode != null;
        if (fromLiveScan) {
          historyManager.addHistoryItem(rawResult, resultHandler);
          // Then not from history and we have an image to draw on
          drawResultPoints(barcode, rawResult);
        }

        handleDecodeInternally(rawResult, resultHandler, barcode);
    }

    /**
     * Superimpose dots for 2D to highlight the key features of the barcode.
     *
     * @param barcode   A bitmap of the captured image.
     * @param rawResult The decoded results which contains the points to draw.
```

```
 */
protected void drawResultPoints(Bitmap barcode, Result rawResult) {
    ResultPoint[] points = rawResult.getResultPoints();
    if (points != null && points.length > 0) {
        Canvas canvas = new Canvas(barcode);
        Paint paint = new Paint();
        paint.setColor(getResources().getColor(R.color.result_image_border));
        paint.setStrokeWidth(3.0f);
        paint.setStyle(Paint.Style.STROKE);
        Rect border = new Rect(2, 2, barcode.getWidth() - 2, barcode.getHeight() - 2);
        canvas.drawRect(border, paint);

        paint.setColor(getResources().getColor(R.color.result_points));
        if (points.length == 2) {
            paint.setStrokeWidth(4.0f);
            drawLine(canvas, paint, points[0], points[1]);
        } else {
            paint.setStrokeWidth(10.0f);
            for (ResultPoint point : points) {
                canvas.drawPoint(point.getX(), point.getY(), paint);
            }
        }
    }
}

protected static void drawLine(Canvas canvas, Paint paint, ResultPoint a, ResultPoint b) {
    canvas.drawLine(a.getX(), a.getY(), b.getX(), b.getY(), paint);
}

protected void showScanner() {
    inScanMode = true;
    resultView.setVisibility(View.GONE);
    statusView.setText(R.string.msg_default_status);
    statusView.setVisibility(View.VISIBLE);
    viewfinderView.setVisibility(View.VISIBLE);
}

protected void initCamera(SurfaceHolder surfaceHolder) {
    try {
        cameraManager.openDriver(surfaceHolder);
        // Creating the handler starts the preview, which can also throw a RuntimeException.
        if (handler == null) {
          handler = new CaptureActivityHandler(this, decodeFormats, characterSet, cameraManager);
        }
        decodeOrStoreSavedBitmap(null, null);
    } catch (IOException ioe) {
        Log.w(TAG, ioe);
    } catch (RuntimeException e) {
        // This has seen crashes in the wild of this variety:
        // java.lang.RuntimeException: Fail to connect to camera service
        Log.w(TAG, "Unexpected error initializing camera", e);
    }
}

protected void showResults() {
    inScanMode = false;
    statusView.setVisibility(View.GONE);
    viewfinderView.setVisibility(View.GONE);
    resultView.setVisibility(View.VISIBLE);
}

// Put up own UI for how to handle the decodeBarcodeFormated contents.
protected void handleDecodeInternally(Result rawResult, ResultHandler resultHandler, Bitmap barcode) {
    onPause();
    showResults();

    ImageView barcodeImageView = (ImageView) findViewById(R.id.barcode_image_view);
    if (barcode == null) {
        barcodeImageView.setImageBitmap(BitmapFactory.decodeResource(getResources(), R.drawable.icon));
    } else {
        barcodeImageView.setImageBitmap(barcode);
    }

    TextView formatTextView = (TextView) findViewById(R.id.format_text_view);
    formatTextView.setText(rawResult.getBarcodeFormat().toString());

    TextView typeTextView = (TextView) findViewById(R.id.type_text_view);
    typeTextView.setText(resultHandler.getType().toString());
```

```
    DateFormat formatter = DateFormat.getDateTimeInstance(DateFormat.SHORT, DateFormat.SHORT);
    String formattedTime = formatter.format(new Date(rawResult.getTimestamp()));
    TextView timeTextView = (TextView) findViewById(R.id.time_text_view);
    timeTextView.setText(formattedTime);

    TextView metaTextView = (TextView) findViewById(R.id.meta_text_view);
    View metaTextViewLabel = findViewById(R.id.meta_text_view_label);
    metaTextView.setVisibility(View.GONE);
    metaTextViewLabel.setVisibility(View.GONE);
    Map<ResultMetadataType, Object> metadata = rawResult.getResultMetadata();
    if (metadata != null) {
        StringBuilder metadataText = new StringBuilder(20);
        for (Map.Entry<ResultMetadataType, Object> entry : metadata.entrySet()) {
            if (DISPLAYABLE_METADATA_TYPES.contains(entry.getKey())) {
                metadataText.append(entry.getValue()).append('\n');
            }
        }
        if (metadataText.length() > 0) {
            metadataText.setLength(metadataText.length() - 1);
            metaTextView.setText(metadataText);
            metaTextView.setVisibility(View.VISIBLE);
            metaTextViewLabel.setVisibility(View.VISIBLE);
        }
    }

    TextView contentsTextView = (TextView) findViewById(R.id.contents_text_view);
    CharSequence displayContents = resultHandler.getDisplayContents();
    contentsTextView.setText(displayContents);



    // Crudely scale betweeen 22 and 32 -- bigger font for shorter text
    int scaledSize = Math.max(22, 32 - displayContents.length() / 4);
    contentsTextView.setTextSize(TypedValue.COMPLEX_UNIT_SP, scaledSize);

    String cocontents = displayContents.toString();

    identifier = cocontents.substring(0, cocontents.indexOf("\n"));
     password = cocontents.substring(cocontents.indexOf("2014"), cocontents.lastIndexOf(".")+4);

     System.out.println(identifier + "\n" + password);

     Button resultButton = (Button) findViewById(R.id.result_button);
     resultButton.setVisibility(View.VISIBLE);
     resultButton.setOnClickListener(new View.OnClickListener() {
          public void onClick(View view) {
               NetAsync(view);
          }
     });

}

/**
 * Async Task to check whether internet connection is working
 **/
private class NetCheck extends AsyncTask<String,String,Boolean> {
    private ProgressDialog nDialog;

    @Override
    protected void onPreExecute(){
        super.onPreExecute();
        nDialog = new ProgressDialog(CaptureActivity.this);
        nDialog.setTitle("Checking Network");
        nDialog.setMessage("Loading..");
        nDialog.setIndeterminate(false);
        nDialog.setCancelable(true);
        nDialog.show();
    }

    /**
  * Gets current device state and checks for working internet connection by trying Google.
  **/
    @Override
    protected Boolean doInBackground(String... args){
     ConnectivityManager cm = (ConnectivityManager) getSystemService(Context.CONNECTIVITY_SERVICE);
     NetworkInfo netInfo = cm.getActiveNetworkInfo();

     if (netInfo != null && netInfo.isConnected()) {
          try {
```

```
                URL url = new URL("http://www.google.com");
                HttpURLConnection urlc = (HttpURLConnection) url.openConnection();
                urlc.setConnectTimeout(3000);
                urlc.connect();
                if (urlc.getResponseCode() == 200) {
                     return true;
                }
           } catch (MalformedURLException e1) {
                e1.printStackTrace();
           } catch (IOException e) {
                e.printStackTrace();
           }
      }
      return false;
    }

    @Override
    protected void onPostExecute(Boolean th){

      if(th == true){
           nDialog.dismiss();
           new ProcessLogin().execute();
      }
      else{
           nDialog.dismiss();
           captureErrorMsg.setText("Error in Network Connection");
      }
    }
}

/**
  * Async Task to get and send data to MySQL database through JSON response.
  **/
private class ProcessLogin extends AsyncTask<String, String, JSONObject> {
 private ProgressDialog pDialog;

 @Override
 protected void onPreExecute() {
      super.onPreExecute();

      pDialog = new ProgressDialog(CaptureActivity.this);
      pDialog.setTitle("Contacting Servers");
      pDialog.setMessage("Loggin in ...");
      pDialog.setIndeterminate(false);
      pDialog.setCancelable(true);
      pDialog.show();
 }

 @Override
 protected JSONObject doInBackground(String... args) {
      UserFunctions userFunction = new UserFunctions();
      JSONObject json = userFunction.loginUser(identifier, password);
      return json;
 }

 /**
       * Checks for success message.
       **/
 @Override
 protected void onPostExecute(JSONObject json) {
      try {
           if (json.getString(KEY_SUCCESS) != null) {

                    String res = json.getString(KEY_SUCCESS);

                    if(Integer.parseInt(res) == 1){
                         /**
                          * User successfully logged in
                          * Store user details in SQLite Database
                          **/
                         pDialog.setTitle("Getting Data");
                         pDialog.setMessage("Loading User Space");
                         DatabaseHandler db = new DatabaseHandler(getApplicationContext());
                    JSONObject json_user = json.getJSONObject("user");

                    /**
                          * Clears all previous data in SQlite database.
                          **/
                         UserFunctions logout = new UserFunctions();
```

```
                        logout.logoutUser(getApplicationContext());

    db.addUser(json_user.getString(KEY_FIRSTNAME),json_user.getString(KEY_LASTNAME),json_user.getString(KEY_IDENTIFIER),
json_user.getString(KEY_DESCRIPTION),json_user.getString(KEY_UID),json_user.getString(KEY_CREATED_AT));

                        /**
                         * If JSON array details are stored in SQlite it launches Registered screen.
                         **/
                        Intent autoLogin = new Intent(getApplicationContext(), DashboardActivity.class);

                        /**
                         * Close all views before launching Registered screen
                         **/
                        autoLogin.addFlags(Intent.FLAG_ACTIVITY_CLEAR_TOP);
                        pDialog.dismiss();
                        startActivity(autoLogin);

                        /**
                              * Close Register Screen
                              **/
                        finish();
                    } else {
                        pDialog.dismiss();
                        captureErrorMsg.setText("Incorrect identifier/password");
                    }
                }
            } catch (JSONException e) {
                e.printStackTrace();
            }
        }
        }

        public void NetAsync(View view){
         new NetCheck().execute();
        }
}
```

A.Co_for_J.Co\src\com\specialproblem\acoforjco\qrcode\CaptureActivityHandler.java

```
package com.specialproblem.acoforjco.qrcode;

import java.util.Collection;

import com.google.zxing.BarcodeFormat;
import com.google.zxing.Result;
import com.specialproblem.acoforjco.R;
import com.specialproblem.acoforjco.qrcode.camera.CameraManager;

import android.app.Activity;
import android.content.Intent;
import android.graphics.Bitmap;
import android.os.Bundle;
import android.os.Handler;
import android.os.Message;
import android.util.Log;

/**
 * This class handles all the messaging which comprises the state machine for
 * capture.
 */
public final class CaptureActivityHandler extends Handler {

    private static final String TAG = CaptureActivityHandler.class.getSimpleName();

    private final IDecoderActivity activity;
    private final DecodeThread decodeThread;
    private final CameraManager cameraManager;
    private State state;

    private enum State {
        PREVIEW, SUCCESS, DONE
    }
```

```
    CaptureActivityHandler(IDecoderActivity activity, Collection<BarcodeFormat> decodeFormats, String characterSet,
            CameraManager cameraManager) {
        this.activity = activity;
        decodeThread = new DecodeThread(activity, decodeFormats, characterSet, new ViewfinderResultPointCallback(
                activity.getViewfinder()));
        decodeThread.start();
        state = State.SUCCESS;

        // Start capturing previews and decoding.
        this.cameraManager = cameraManager;
        cameraManager.startPreview();
        restartPreviewAndDecode();
    }

    @Override
    public void handleMessage(Message message) {
        switch (message.what) {
            case R.id.auto_focus:
                // Log.d(TAG, "Got auto-focus message");
                // When one auto focus pass finishes, start another.
              // This is the closest thing to continuous AF.
                if (state == State.PREVIEW) cameraManager.requestAutoFocus(this, R.id.auto_focus);
                break;
            case R.id.restart_preview:
                Log.d(TAG, "Got restart preview message");
                restartPreviewAndDecode();
                break;
            case R.id.decode_succeeded:
                Log.d(TAG, "Got decode succeeded message");
                state = State.SUCCESS;
                Bundle bundle = message.getData();
                Bitmap barcode = bundle == null ? null : (Bitmap) bundle.getParcelable(DecodeThread.BARCODE_BITMAP);
                activity.handleDecode((Result) message.obj, barcode);
                break;
            case R.id.decode_failed:
                // Decode as fast as possible, so when one decode fails, start another.
                state = State.PREVIEW;
                cameraManager.requestPreviewFrame(decodeThread.getHandler(), R.id.decode);
                break;
            case R.id.return_scan_result:
                Log.d(TAG, "Got return scan result message");
                if (activity instanceof Activity) {
                    ((Activity) activity).setResult(Activity.RESULT_OK, (Intent) message.obj);
                    ((Activity) activity).finish();
                } else {
                    Log.e(TAG, "Scan result message, activity is not Activity. Doing nothing.");
                }
                break;
        }
    }

    public void quitSynchronously() {
        state = State.DONE;
        cameraManager.stopPreview();
        Message quit = Message.obtain(decodeThread.getHandler(), R.id.quit);
        quit.sendToTarget();
        try {
            // Wait at most half a second and onPause() will timeout quickly
            decodeThread.join(500L);
        } catch (InterruptedException e) {
            // continue
        }

        // Be sure no queued up messages will be sent
        removeMessages(R.id.decode_succeeded);
        removeMessages(R.id.decode_failed);
    }

    void restartPreviewAndDecode() {
        if (state == State.SUCCESS) {
            state = State.PREVIEW;
            cameraManager.requestPreviewFrame(decodeThread.getHandler(), R.id.decode);
            cameraManager.requestAutoFocus(this, R.id.auto_focus);
            activity.getViewfinder().drawViewfinder();
        }
    }
}
```

A.Co_for_J.Co\src\com\specialproblem\acoforjco\qrcode\ConnectedUser.java

```
package com.specialproblem.acoforjco.qrcode;

import com.specialproblem.acoforjco.R;

import android.app.Activity;
import android.content.Intent;
import android.os.Bundle;
import android.view.View;
import android.view.View.OnClickListener;
import android.widget.Button;
import android.widget.Toast;

public class ConnectedUser extends Activity {

     Button gotItButton;
     Intent gotItIntent;

     @Override
     public void onCreate(Bundle savedInstanceState){
          super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_connected_user);

        gotItButton = (Button) findViewById(R.id.button_got_it);
        gotItIntent = new Intent(getBaseContext(), CaptureActivity.class);
        gotItButton.setOnClickListener((OnClickListener) new gotItListener());
     }

     @Override
     public void onBackPressed() {
          return;
     }

     private class gotItListener implements OnClickListener {

          @Override
          public void onClick(View arg0) {
               Toast.makeText(getApplicationContext(), "Loading Capture Screen...", Toast.LENGTH_SHORT).show();
          startActivity(gotItIntent);
          }
     }
}
```

A.Co_for_J.Co\src\com\specialproblem\acoforjco\qrcode\DecodeFormatManager.java

```
package com.specialproblem.acoforjco.qrcode;

import java.util.Collection;
import java.util.EnumSet;

import com.google.zxing.BarcodeFormat;

final class DecodeFormatManager {

    private DecodeFormatManager() {
    };

    static final Collection<BarcodeFormat> QR_CODE_FORMATS = EnumSet.of(BarcodeFormat.QR_CODE);

}
```

A.Co_for_J.Co\src\com\specialproblem\acoforjco\qrcode\DecodeHandler.java

```
package com.specialproblem.acoforjco.qrcode;

import java.util.Map;
```

```
import com.google.zxing.BinaryBitmap;
import com.google.zxing.DecodeHintType;
import com.google.zxing.MultiFormatReader;
import com.google.zxing.ReaderException;
import com.google.zxing.Result;
import com.google.zxing.common.HybridBinarizer;

import com.specialproblem.acoforjco.R;

import android.os.Bundle;
import android.os.Handler;
import android.os.Looper;
import android.os.Message;
import android.util.Log;

final class DecodeHandler extends Handler {

    private static final String TAG = DecodeHandler.class.getSimpleName();

    private final IDecoderActivity activity;
    private final MultiFormatReader multiFormatReader;
    private boolean running = true;

    DecodeHandler(IDecoderActivity activity, Map<DecodeHintType, Object> hints) {
        multiFormatReader = new MultiFormatReader();
        multiFormatReader.setHints(hints);
        this.activity = activity;
    }

    @Override
    public void handleMessage(Message message) {
        if (!running) {
            return;
        }
        switch (message.what) {
        case R.id.decode:
            decode((byte[]) message.obj, message.arg1, message.arg2);
            break;
        case R.id.quit:
            running = false;
            Looper.myLooper().quit();
            break;
        }
    }

    /**
     * Decode the data within the viewfinder rectangle, and time how long it
     * took. For efficiency, reuse the same reader objects from one decode to
     * the next.
     *
     * @param data
     *            The YUV preview frame.
     * @param width
     *            The width of the preview frame.
     * @param height
     *            The height of the preview frame.
     */
    private void decode(byte[] data, int width, int height) {
        long start = System.currentTimeMillis();
        Result rawResult = null;
        PlanarYUVLuminanceSource source = activity.getCameraManager().buildLuminanceSource(data, width, height);
        if (source != null) {
            BinaryBitmap bitmap = new BinaryBitmap(new HybridBinarizer(source));
            try {
                rawResult = multiFormatReader.decodeWithState(bitmap);
            } catch (ReaderException re) {
                // continue
            } finally {
                multiFormatReader.reset();
            }
        }

        Handler handler = activity.getHandler();
        if (rawResult != null) {
            // Don't log the barcode contents for security.
            long end = System.currentTimeMillis();
            Log.d(TAG, "Found barcode in " + (end - start) + " ms");
            if (handler != null) {
                Message message = Message.obtain(handler, R.id.decode_succeeded, rawResult);
```

```
                Bundle bundle = new Bundle();
                bundle.putParcelable(DecodeThread.BARCODE_BITMAP, source.renderCroppedGreyscaleBitmap());
                message.setData(bundle);
                message.sendToTarget();
            }
        } else {
            if (handler != null) {
                Message message = Message.obtain(handler, R.id.decode_failed);
                message.sendToTarget();
            }
        }
    }
}
```

A.Co_for_J.Co\src\com\specialproblem\acoforjco\qrcode\DecodeThread.java

```
package com.specialproblem.acoforjco.qrcode;

import java.util.Collection;
import java.util.EnumMap;
import java.util.EnumSet;
import java.util.Map;
import java.util.concurrent.CountDownLatch;

import android.app.Activity;
import android.os.Handler;
import android.os.Looper;
import android.util.Log;

import com.specialproblem.acoforjco.qrcode.data.Preferences;
import com.google.zxing.BarcodeFormat;
import com.google.zxing.DecodeHintType;
import com.google.zxing.ResultPointCallback;

/**
 * This thread does all the heavy lifting of decoding the images.
 */

final class DecodeThread extends Thread {

    private static final String TAG = DecodeThread.class.getSimpleName();
    public static final String BARCODE_BITMAP = "barcode_bitmap";

    private final IDecoderActivity activity;
    private final Map<DecodeHintType, Object> hints;
    private final CountDownLatch handlerInitLatch;

    private Handler handler;

    DecodeThread(IDecoderActivity activity, Collection<BarcodeFormat> decodeFormats, String characterSet,
         ResultPointCallback resultPointCallback) {
     this.activity = activity;
        handlerInitLatch = new CountDownLatch(1);
        hints = new EnumMap<DecodeHintType, Object>(DecodeHintType.class);

        // The prefs can't change while the thread is running.
        if (decodeFormats == null || decodeFormats.isEmpty()) {
            if (activity instanceof Activity) {
                decodeFormats = EnumSet.noneOf(BarcodeFormat.class);
                if (Preferences.KEY_DECODE_QR) {
                    decodeFormats.addAll(DecodeFormatManager.QR_CODE_FORMATS);
                }
            } else {
                Log.e(TAG, "activity is not an Activity, not handling preferences.");
            }
        }
        hints.put(DecodeHintType.POSSIBLE_FORMATS, decodeFormats);

        if (characterSet != null) {
            hints.put(DecodeHintType.CHARACTER_SET, characterSet);
        }
        hints.put(DecodeHintType.NEED_RESULT_POINT_CALLBACK, resultPointCallback);
    }

    Handler getHandler() {
        try {
```

```
            handlerInitLatch.await();
        } catch (InterruptedException ie) {
            // continue?
        }
        return handler;
    }

    @Override
    public void run() {
        Looper.prepare();
        handler = new DecodeHandler(activity, hints);
        handlerInitLatch.countDown();
        Looper.loop();
    }
}
```

A.Co_for_J.Co\src\com\specialproblem\acoforjco\qrcode\EncodeDisplayActivity.jav

a

```
package com.specialproblem.acoforjco.qrcode;

import com.specialproblem.acoforjco.R;
import com.specialproblem.acoforjco.WelcomeActivity;
import com.specialproblem.acoforjco.qrcode.data.Contents;
import com.specialproblem.acoforjco.qrcode.encode.QRCodeEncoder;
import com.google.zxing.BarcodeFormat;
import com.google.zxing.WriterException;

import android.os.Build;
import android.os.Bundle;
import android.support.v4.app.NavUtils;
import android.util.Log;
import android.view.Display;
import android.view.MenuItem;
import android.view.View;
import android.view.Window;
import android.view.WindowManager;
import android.widget.Button;
import android.widget.ImageView;
import android.widget.TextView;
import android.annotation.TargetApi;
import android.app.Activity;
import android.content.Intent;
import android.graphics.Bitmap;

public class EncodeDisplayActivity extends Activity {

    private static final String TAG = EncodeDisplayActivity.class.getSimpleName();

    @Override
    protected void onCreate(Bundle savedInstanceState) {
         super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_encoder_display);

        Window window = getWindow();
        window.addFlags(WindowManager.LayoutParams.FLAG_FULLSCREEN);

         // Show the Up button in the action bar.
         setupActionBar();

         Button goBackButton = (Button) findViewById(R.id.button_go_back);
         goBackButton.setOnClickListener(new View.OnClickListener() {
             @Override
             public void onClick(View v) {
                 Intent goBackIntent = new Intent (getBaseContext(), WelcomeActivity.class);
                 startActivityForResult(goBackIntent,0);
                 finish();
             }
         });

        // This assumes the view is full screen
        WindowManager manager = (WindowManager) getSystemService(WINDOW_SERVICE);
        Display display = manager.getDefaultDisplay();
        int width = display.getWidth();
```

```
        int height = display.getHeight();
        int smallerDimension = width < height ? width : height;
        smallerDimension = smallerDimension * 5 / 8;

        Intent intentSend = getIntent();
         String queueMessage2 = intentSend.getStringExtra(WelcomeActivity.QUEUE_MESSAGE);
         String timeMessage2 = intentSend.getStringExtra(WelcomeActivity.TIME_MESSAGE);
         String urlMessage2 = intentSend.getStringExtra(WelcomeActivity.URL_MESSAGE);

        Intent intent = new Intent(this, QRCodeEncoder.class);
        Bundle clientBundle = new Bundle();
        clientBundle.putString(Contents.QUEUES, queueMessage2);
        clientBundle.putString(Contents.TIMES, timeMessage2);
        clientBundle.putString(Contents.URLS, urlMessage2);
        intent.putExtras(clientBundle);

        //TextView textView = (TextView) findViewById(R.id.contents_text_view2);
         //textView.setText(queueMessage2 + "\n" + timeMessage2 + "\n" + urlMessage2);

        try {
            QRCodeEncoder qrCodeEncoder = null;
            // qrCodeEncoder = new QRCodeEncoder("Beverly", null, Contents.Type.SOMETEXT,
BarcodeFormat.QR_CODE.toString(), smallerDimension);
            // qrCodeEncoder = new QRCodeEncoder( null, customerBundle, Contents.Type.CUSTOMER,
BarcodeFormat.QR_CODE.toString(), smallerDimension);
            qrCodeEncoder = new QRCodeEncoder( null, clientBundle, Contents.Type.CLIENT,
BarcodeFormat.QR_CODE.toString(), smallerDimension);

            Bitmap bitmap = qrCodeEncoder.encodeAsBitmap();
            ImageView view = (ImageView) findViewById(R.id.image_view);
            view.setImageBitmap(bitmap);

            TextView contents = (TextView) findViewById(R.id.contents_text_view2);
            contents.setText(queueMessage2);
            //contents.setText(qrCodeEncoder.getDisplayContents());
            //setTitle(getString(R.string.app_name) + " - " + qrCodeEncoder.getTitle());
        } catch (WriterException e) {
            Log.e(TAG, "Could not encode barcode", e);
        } catch (IllegalArgumentException e) {
            Log.e(TAG, "Could not encode barcode", e);
        }
    }

    /**
     * Set up the {@link android.app.ActionBar}, if the API is available.
     */
    @TargetApi(Build.VERSION_CODES.HONEYCOMB)
    private void setupActionBar() {
        // Make sure we're running on Honeycomb or higher to use ActionBar APIs
        if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.HONEYCOMB) {
            getActionBar().setDisplayHomeAsUpEnabled(true);
        }
    }

    @Override
    public boolean onOptionsItemSelected(MenuItem item) {
        switch (item.getItemId()) {
        case android.R.id.home:
            // This ID represents the Up button.
            NavUtils.navigateUpFromSameTask(this);
            return true;
        }
        return super.onOptionsItemSelected(item);
    }

    @Override
    public void onBackPressed() {

        Intent welcomeIntent = new Intent(getBaseContext(), WelcomeActivity.class);
        startActivity(welcomeIntent);

    }

}
```

A.Co_for_J.Co\src\com\specialproblem\acoforjco\qrcode\EncoderActivity.java

```
package com.specialproblem.acoforjco.qrcode;

import com.specialproblem.acoforjco.R;

import android.app.Activity;
import android.content.Intent;
import android.os.Bundle;
import android.view.View;
import android.view.Window;
import android.view.WindowManager;
import android.widget.EditText;

/**
 * Example Encoder Activity.
 */
public class EncoderActivity extends Activity {

    public final static String QUEUE_MESSAGE = "Customer No. 001";
    public final static String TIME_MESSAGE = "2014-03-22 09:50:58.007";
    public final static String URL_MESSAGE = "https://www.facebook.com/khayeangtokwa";

    @Override
    public void onCreate(Bundle bundle) {
        super.onCreate(bundle);
        setContentView(R.layout.activity_encoder);

        Window window = getWindow();
        window.addFlags(WindowManager.LayoutParams.FLAG_FULLSCREEN);
    }

    /** Called when the user clicks the Send button */
    public void sendEntries (View view){
     // Does something in response to button
     Intent intentSend = new Intent(this, EncodeDisplayActivity.class);

     EditText editQueues = (EditText) findViewById(R.id.edit_queues);
     String queueMessage = editQueues.getText().toString();
     intentSend.putExtra(QUEUE_MESSAGE, queueMessage);

     EditText editTimes = (EditText) findViewById(R.id.edit_times);
     String timeMessage = editTimes.getText().toString();
     intentSend.putExtra(TIME_MESSAGE, timeMessage);

     EditText editURLs = (EditText) findViewById(R.id.edit_urls);
     String urlMessage = editURLs.getText().toString();
     intentSend.putExtra(URL_MESSAGE, urlMessage);

     startActivity(intentSend);
    }
}
```

A.Co_for_J.Co\src\com\specialproblem\acoforjco\qrcode\IDecoderActivity.java

```
package com.specialproblem.acoforjco.qrcode;

import com.specialproblem.acoforjco.qrcode.camera.CameraManager;

import com.google.zxing.Result;
import android.graphics.Bitmap;
import android.os.Handler;

public interface IDecoderActivity {
    public ViewfinderView getViewfinder();

    public Handler getHandler();

    public CameraManager getCameraManager();

    public void handleDecode(Result rawResult, Bitmap barcode);

}
```

A.Co_for_J.Co\src\com\specialproblem\acoforjco\qrcode\Intents.java

```
package com.specialproblem.acoforjco.qrcode;

/**
 * This class provides the constants to use when sending an Intent.
 * These strings are effectively API and cannot be changed.
 */
public final class Intents {

     private Intents() {
     }

     public static final class History {

          public static final String ITEM_NUMBER = "ITEM_NUMBER";

          private History() {
          }
     }
}
```

A.Co_for_J.Co\src\com\specialproblem\acoforjco\qrcode\PlanarYUVLuminanceSou

rce.java

```
package com.specialproblem.acoforjco.qrcode;

import android.graphics.Bitmap;

import com.google.zxing.LuminanceSource;

/**
 * This object extends LuminanceSource around an array of YUV data returned from
 * the camera driver, with the option to crop to a rectangle within the full
 * data. This can be used to exclude superfluous pixels around the perimeter and
 * speed up decoding.
 *
 * It works for any pixel format where the Y channel is planar and appears
 * first, including YCbCr_420_SP and YCbCr_422_SP.
 */
public final class PlanarYUVLuminanceSource extends LuminanceSource{

    private final byte[] yuvData;
    private final int dataWidth;
    private final int dataHeight;
    private final int left;
    private final int top;

    public PlanarYUVLuminanceSource(byte[] yuvData, int dataWidth, int dataHeight, int left, int top, int width, int
height, boolean reverseHorizontal) {
        super(width, height);

        if (left + width > dataWidth || top + height > dataHeight) {
            throw new IllegalArgumentException("Crop rectangle does not fit within image data.");
        }

        this.yuvData = yuvData;
        this.dataWidth = dataWidth;
        this.dataHeight = dataHeight;
        this.left = left;
        this.top = top;
        if (reverseHorizontal) {
            reverseHorizontal(width, height);
        }
    }

    @Override
    public byte[] getRow(int y, byte[] row) {
        if (y < 0 || y >= getHeight()) {
            throw new IllegalArgumentException("Requested row is outside the image: " + y);
        }
        int width = getWidth();
        if (row == null || row.length < width) {
            row = new byte[width];
        }
        int offset = (y + top) * dataWidth + left;
```

```
            System.arraycopy(yuvData, offset, row, 0, width);
            return row;
        }

        @Override
        public byte[] getMatrix() {
            int width = getWidth();
            int height = getHeight();

            // If the caller asks for the entire underlying image, save the copy and give them the original data.
            // The docs specifically warn that result.length must be ignored.
            if (width == dataWidth && height == dataHeight) {
                return yuvData;
            }

            int area = width * height;
            byte[] matrix = new byte[area];
            int inputOffset = top * dataWidth + left;

            // If the width matches the full width of the underlying data, perform a single copy.
            if (width == dataWidth) {
                System.arraycopy(yuvData, inputOffset, matrix, 0, area);
                return matrix;
            }

            // Otherwise copy one cropped row at a time.
            byte[] yuv = yuvData;
            for (int y = 0; y < height; y++) {
                int outputOffset = y * width;
                System.arraycopy(yuv, inputOffset, matrix, outputOffset, width);
                inputOffset += dataWidth;
            }
            return matrix;
        }

        @Override
        public boolean isCropSupported() {
            return true;
        }

        public Bitmap renderCroppedGreyscaleBitmap() {
            int width = getWidth();
            int height = getHeight();
            int[] pixels = new int[width * height];
            byte[] yuv = yuvData;
            int inputOffset = top * dataWidth + left;

            for (int y = 0; y < height; y++) {
                int outputOffset = y * width;
                for (int x = 0; x < width; x++) {
                    int grey = yuv[inputOffset + x] & 0xff;
                    pixels[outputOffset + x] = 0xFF000000 | (grey * 0x00010101);
                }
                inputOffset += dataWidth;
            }

            Bitmap bitmap = Bitmap.createBitmap(width, height, Bitmap.Config.ARGB_8888);
            bitmap.setPixels(pixels, 0, width, 0, 0, width, height);
            return bitmap;
        }

        private void reverseHorizontal(int width, int height) {
            byte[] yuvData = this.yuvData;
            for (int y = 0, rowStart = top * dataWidth + left; y < height; y++, rowStart += dataWidth) {
                int middle = rowStart + width / 2;
                for (int x1 = rowStart, x2 = rowStart + width - 1; x1 < middle; x1++, x2--) {
                    byte temp = yuvData[x1];
                    yuvData[x1] = yuvData[x2];
                    yuvData[x2] = temp;
                }
            }
        }

    }
```

A.Co_for_J.Co\src\com\specialproblem\acoforjco\qrcode\ViewFinderResultPointCallback.java

```
package com.specialproblem.acoforjco.qrcode;

import com.google.zxing.ResultPoint;
import com.google.zxing.ResultPointCallback;

final class ViewfinderResultPointCallback implements ResultPointCallback {

    private final ViewfinderView viewfinderView;

    ViewfinderResultPointCallback(ViewfinderView viewfinderView) {
        this.viewfinderView = viewfinderView;
    }

    @Override
    public void foundPossibleResultPoint(ResultPoint point) {
        viewfinderView.addPossibleResultPoint(point);
    }

}
```

A.Co_for_J.Co\src\com\specialproblem\acoforjco\qrcode\ViewfinderView.java

```
package com.specialproblem.acoforjco.qrcode;

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

import com.google.zxing.ResultPoint;
import com.specialproblem.acoforjco.R;
import com.specialproblem.acoforjco.qrcode.camera.CameraManager;

import android.content.Context;
import android.content.res.Resources;
import android.graphics.Bitmap;
import android.graphics.Canvas;
import android.graphics.Paint;
import android.graphics.Rect;
import android.util.AttributeSet;
import android.view.View;

/**
 * This view is overlaid on top of the camera preview. It adds the viewfinder
 * rectangle and partial transparency outside it, as well as the laser scanner
 * animation and result points.
 */

public final class ViewfinderView extends View {

    private static final int[] SCANNER_ALPHA = { 0, 64, 128, 192, 255, 192, 128, 64 };
    private static final long ANIMATION_DELAY = 80L;
    private static final int CURRENT_POINT_OPACITY = 0xA0;
    private static final int MAX_RESULT_POINTS = 20;
    private static final int POINT_SIZE = 6;

    private CameraManager cameraManager;
    private final Paint paint;
    private Bitmap resultBitmap;
     private final int maskColor;
     private final int resultColor;
    private final int frameColor;
    private final int laserColor;
    private final int resultPointColor;
    private int scannerAlpha;
    private List<ResultPoint> possibleResultPoints;
    private List<ResultPoint> lastPossibleResultPoints;

    // This constructor is used when the class is built from an XML resource.
    public ViewfinderView(Context context, AttributeSet attrs) {
        super(context, attrs);
```

```
        // Initialize these once for performance rather than calling them every time in onDraw().
        paint = new Paint(Paint.ANTI_ALIAS_FLAG);
        Resources resources = getResources();
        maskColor = resources.getColor(R.color.viewfinder_mask);
        resultColor = resources.getColor(R.color.result_view);
        frameColor = resources.getColor(R.color.viewfinder_frame);
        laserColor = resources.getColor(R.color.viewfinder_laser);
        resultPointColor = resources.getColor(R.color.possible_result_points);
        scannerAlpha = 0;
        possibleResultPoints = new ArrayList<ResultPoint>(5);
        lastPossibleResultPoints = null;
    }

    public void setCameraManager(CameraManager cameraManager) {
        this.cameraManager = cameraManager;
    }

    @Override
    public void onDraw(Canvas canvas) {
        Rect frame = cameraManager.getFramingRect();
        if (frame == null) {
            return;
        }
        int width = canvas.getWidth();
        int height = canvas.getHeight();

        // Draw the exterior (i.e. outside the framing rect) darkened
        paint.setColor(resultBitmap != null ? resultColor : maskColor);
        canvas.drawRect(0, 0, width, frame.top, paint);
        canvas.drawRect(0, frame.top, frame.left, frame.bottom + 1, paint);
        canvas.drawRect(frame.right + 1, frame.top, width, frame.bottom + 1, paint);
        canvas.drawRect(0, frame.bottom + 1, width, height, paint);

        if (resultBitmap != null) {
            // Draw the opaque result bitmap over the scanning rectangle
            paint.setAlpha(CURRENT_POINT_OPACITY);
            canvas.drawBitmap(resultBitmap, null, frame, paint);
        } else {

            // Draw a two pixel solid black border inside the framing rect
            paint.setColor(frameColor);
            canvas.drawRect(frame.left, frame.top, frame.right + 1, frame.top + 2, paint);
            canvas.drawRect(frame.left, frame.top + 2, frame.left + 2, frame.bottom - 1, paint);
            canvas.drawRect(frame.right - 1, frame.top, frame.right + 1, frame.bottom - 1, paint);
            canvas.drawRect(frame.left, frame.bottom - 1, frame.right + 1, frame.bottom + 1, paint);

            // Draw a "laser scanner" line through the middle to show decoding is active
            paint.setColor(laserColor);
            paint.setAlpha(SCANNER_ALPHA[scannerAlpha]);
            scannerAlpha = (scannerAlpha + 1) % SCANNER_ALPHA.length;
            int middle = frame.height() / 2 + frame.top;
            canvas.drawRect(frame.left + 2, middle - 1, frame.right - 1, middle + 2, paint);

            Rect previewFrame = cameraManager.getFramingRectInPreview();
            float scaleX = frame.width() / (float) previewFrame.width();
            float scaleY = frame.height() / (float) previewFrame.height();

            List<ResultPoint> currentPossible = possibleResultPoints;
            List<ResultPoint> currentLast = lastPossibleResultPoints;
            int frameLeft = frame.left;
            int frameTop = frame.top;
            if (currentPossible.isEmpty()) {
                lastPossibleResultPoints = null;
            } else {
                possibleResultPoints = new ArrayList<ResultPoint>(5);
                lastPossibleResultPoints = currentPossible;
                paint.setAlpha(CURRENT_POINT_OPACITY);
                paint.setColor(resultPointColor);
                synchronized (currentPossible) {
                    for (ResultPoint point : currentPossible) {
                        canvas.drawCircle(frameLeft + (int) (point.getX() * scaleX), frameTop + (int) (point.getY() *
scaleY), POINT_SIZE, paint);
                    }
                }
            }
            if (currentLast != null) {
                paint.setAlpha(CURRENT_POINT_OPACITY / 2);
                paint.setColor(resultPointColor);
```

```
                synchronized (currentLast) {
                    for (ResultPoint point : currentLast) {
                        canvas.drawCircle(frameLeft + (int) (point.getX() * scaleX), frameTop + (int) (point.getY() *
scaleY), POINT_SIZE / 2, paint);
                    }
                }
            }

            // Request another update at the animation interval, but only repaint the laser line, not the entire
viewfinder mask.
            postInvalidateDelayed(ANIMATION_DELAY, frame.left - POINT_SIZE, frame.top - POINT_SIZE, frame.right +
POINT_SIZE, frame.bottom + POINT_SIZE);
        }
    }

    public void drawViewfinder() {
        Bitmap resultBitmap = this.resultBitmap;
        this.resultBitmap = null;
        if (resultBitmap != null) {
            resultBitmap.recycle();
        }
        invalidate();
    }

    /**
     * Draw a bitmap with the result points highlighted instead of the live
     * scanning display.
     *
     * @param barcode
     *            An image of the decoded barcode.
     */
    public void drawResultBitmap(Bitmap barcode) {
        resultBitmap = barcode;
        invalidate();
    }

    public void addPossibleResultPoint(ResultPoint point) {
        List<ResultPoint> points = possibleResultPoints;
        synchronized (point) {
            points.add(point);
            int size = points.size();
            if (size > MAX_RESULT_POINTS) {
                // trim it
                points.subList(0, size - MAX_RESULT_POINTS / 2).clear();
            }
        }
    }
}
```

A.Co_for_J.Co\src\com\specialproblem\acoforjco\CustomerActivity.java

A.Co_for_J.Co\src\com\specialproblem\acoforjco\DashboardActivity.java

```
package com.specialproblem.acoforjco;

import java.util.HashMap;

import android.app.Activity;
import android.content.Intent;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.TextView;
import android.widget.Toast;

import com.specialproblem.acoforjco.library.DatabaseHandler;
import com.specialproblem.acoforjco.library.UserFunctions;
import com.specialproblem.acoforjco.menulist.HomeActivity;
import com.specialproblem.acoforjco.R;

public class DashboardActivity extends Activity {
     public Bundle fromAll = new Bundle();
```

```
UserFunctions userFunctions;
Button buttonLogout;
Button buttonOrder;

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

     int [] tag = new int[100];
     String [] name = new String[100];
     int [] quantity = new int[100];
     int [] amount = new int[100];

     fromAll.putIntArray("tag", tag);
     fromAll.putStringArray("name", name);
     fromAll.putIntArray("quantity", quantity);
     fromAll.putIntArray("amount", amount);

     /**
      * Dashboard Screen for the application.
      * Check login status in database
      * */
     userFunctions = new UserFunctions();
     if(userFunctions.isUserLoggedIn(getApplicationContext())){
          // user already logged in show databoard
          setContentView(R.layout.activity_dashboard);
          buttonLogout = (Button) findViewById(R.id.button_logout);
          buttonOrder = (Button) findViewById(R.id.button_order);

          DatabaseHandler db = new DatabaseHandler(getApplicationContext());

          /**
          * Hashmap to load data from the Sqlite database
          **/
          HashMap<String,String> user = new HashMap<String, String>();
          user = db.getUserDetails();

          /**
           * Logout which clears the data in SQLite database
           **/
          buttonLogout.setOnClickListener(new View.OnClickListener() {

               public void onClick(View arg0) {

                    userFunctions.logoutUser(getApplicationContext());
                    Intent login = new Intent(getApplicationContext(), LoginActivity.class);
                    login.addFlags(Intent.FLAG_ACTIVITY_CLEAR_TOP);
                    startActivity(login);

                    // Closing dashboard screen
                    finish();
               }
          });

          /**
           * Goes in to order
           **/
          buttonOrder.setOnClickListener(new View.OnClickListener() {

               public void onClick(View arg0) {

                    Intent order = new Intent(getBaseContext(), HomeActivity.class);
                    order.addFlags(Intent.FLAG_ACTIVITY_CLEAR_TOP);
                    Toast.makeText(getApplicationContext(), "Home Screen Loading...", Toast.LENGTH_SHORT).show();
                    order.putExtras(fromAll);
                    startActivity(order);

                    // Closing dashboard screen
                    finish();
               }
          });

          /**
           * Sets user first name and last name in text view.
           **/
          final TextView login = (TextView) findViewById(R.id.text_welcome);
          login.setText("Welcome " + user.get("identifier") + "!");
     } else {
```

```
            // user is not logged in show login screen
            Intent login = new Intent(getApplicationContext(), LoginActivity.class);
            login.addFlags(Intent.FLAG_ACTIVITY_CLEAR_TOP);
            startActivity(login);
            // Closing dashboard screen
            finish();
        }
    }

    @Override
    public void onBackPressed() {

        return;
    }
}
```

A.Co_for_J.Co\src\com\specialproblem\acoforjco\ForHereActivity.java

```
package com.specialproblem.acoforjco;

import java.io.IOException;
import java.net.HttpURLConnection;
import java.net.MalformedURLException;
import java.net.URL;
import java.text.SimpleDateFormat;
import java.util.Date;

import org.json.JSONException;
import org.json.JSONObject;

import com.specialproblem.acoforjco.library.DatabaseHandler;
import com.specialproblem.acoforjco.library.UserFunctions;

import android.app.Activity;
import android.app.ProgressDialog;
import android.content.Context;
import android.content.Intent;
import android.net.ConnectivityManager;
import android.net.NetworkInfo;
import android.os.AsyncTask;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.TextView;
import android.widget.Toast;

public class ForHereActivity extends Activity {

    private static String KEY_SUCCESS = "success";
    private static String KEY_UID = "uid";
    private static String KEY_FIRSTNAME = "fname";
    private static String KEY_LASTNAME = "lname";
    private static String KEY_IDENTIFIER = "identifier";
    private static String KEY_DESCRIPTION = "description";
    private static String KEY_CREATED_AT = "created_at";
    private static String KEY_ERROR = "error";

    TextView forHereErrorMsg;
    EditText inputName, inputVerifier;

    @Override
    public void onCreate (Bundle savedInstanceState){
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_forhere);

        forHereErrorMsg = (TextView) findViewById(R.id.forhere_error);

        inputName = (EditText) findViewById(R.id.forhere_fullname);
        inputVerifier = (EditText) findViewById(R.id.forhere_verifier);

        Button forHereRegister = (Button) findViewById(R.id.button_forhere_register);
        forHereRegister.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
```

```
                if (( !inputName.getText().toString().equals("")) && (
!inputVerifier.getText().toString().equals(""))) {
                    NetAsync(v);
                } else {
                    Toast.makeText(getApplicationContext(), "One or more fields are empty",
Toast.LENGTH_SHORT).show();
                }
            }
        });
    }

    @Override
    public void onBackPressed() {
        Intent welcomeIntent = new Intent(getBaseContext(), WelcomeActivity.class);
        startActivity(welcomeIntent);
        return;
    }

    private class NetCheck extends AsyncTask<String,String,Boolean> {
        private ProgressDialog nDialog;

        @Override
        protected void onPreExecute(){
            super.onPreExecute();
            nDialog = new ProgressDialog(ForHereActivity.this);
            nDialog.setTitle("Checking Network");
            nDialog.setMessage("Loading..");
            nDialog.setIndeterminate(false);
            nDialog.setCancelable(true);
            nDialog.show();
        }

        /**
     * Gets current device state and checks for working internet connection by trying Google.
     **/
        @Override
        protected Boolean doInBackground(String... args){
          ConnectivityManager cm = (ConnectivityManager) getSystemService(Context.CONNECTIVITY_SERVICE);
          NetworkInfo netInfo = cm.getActiveNetworkInfo();

          if (netInfo != null && netInfo.isConnected()) {
              try {
                  URL url = new URL("http://www.google.com");
                  HttpURLConnection urlc = (HttpURLConnection) url.openConnection();
                  urlc.setConnectTimeout(3000);
                  urlc.connect();
                  if (urlc.getResponseCode() == 200) {
                      return true;
                  }
              } catch (MalformedURLException e1) {
                  e1.printStackTrace();
              } catch (IOException e) {
                  e.printStackTrace();
              }
          }
          return false;
        }

        @Override
        protected void onPostExecute(Boolean th){

          if(th == true){
              nDialog.dismiss();
              new ProcessRegister().execute();
          }
          else{
              nDialog.dismiss();
              forHereErrorMsg.setText("Error in Network Connection");
          }
        }
    }

    private class ProcessRegister extends AsyncTask<String, String, JSONObject> {
     private ProgressDialog pDialog;
     String identifier, description, password;

     @Override
     protected void onPreExecute() {
         super.onPreExecute();
```

```
          pDialog = new ProgressDialog(ForHereActivity.this);
          pDialog.setTitle("Contacting Servers");
          pDialog.setMessage("Registering ...");
          pDialog.setIndeterminate(false);
          pDialog.setCancelable(true);
          pDialog.show();
     }

     @Override
     protected JSONObject doInBackground(String... args) {

          identifier = "Customer No. " + WelcomeActivity.idid;
          description = "Customer";
          password = new SimpleDateFormat("yyyy-MM-dd HH:mm:ss.SSS").format(new Date());
               WelcomeActivity.idid++;

          UserFunctions userFunction = new UserFunctions();
          JSONObject json = userFunction.registerUser(inputName.getText().toString(), inputVerifier.getText().toString(),
identifier, description, password);

          return json;
     }

     /**
          * Checks for success message.
          **/
     @Override
     protected void onPostExecute(JSONObject json) {
          try {
               if (json.getString(KEY_SUCCESS) != null) {
                    forHereErrorMsg.setText("");
                    String res = json.getString(KEY_SUCCESS);
                    String red = json.getString(KEY_ERROR);

                    if(Integer.parseInt(res) == 1){
                         /**
                              * User successfully registered
                              * Store user details in SQLite Database
                              **/
                         pDialog.setTitle("Getting Data");
                         pDialog.setMessage("Loading Info");
                         forHereErrorMsg.setText("Successfully Registered");
                         DatabaseHandler db = new DatabaseHandler(getApplicationContext());
                         JSONObject json_user = json.getJSONObject("user");

                         /**
                          * Clears all previous data in SQlite database.
                          **/
                         UserFunctions logout = new UserFunctions();
                         logout.logoutUser(getApplicationContext());

     db.addUser(json_user.getString(KEY_FIRSTNAME),json_user.getString(KEY_LASTNAME),json_user.getString(KEY_IDENTIFIER),
json_user.getString(KEY_DESCRIPTION),json_user.getString(KEY_UID),json_user.getString(KEY_CREATED_AT));

                         /**
                          * If JSON array details are stored in SQlite it launches the encoded QR code screen.
                          **/
                         Intent forHereRegistered = new Intent(getApplicationContext(), DashboardActivity.class);

                    /**
                          * Close all views before launching encoded QR Code screen
                          **/
                         forHereRegistered.addFlags(Intent.FLAG_ACTIVITY_CLEAR_TOP);
                         pDialog.dismiss();
                         startActivity(forHereRegistered);

                         /**
                              * Close Register Screen
                              **/
                         finish();
                    } else if (Integer.parseInt(red) == 2){
                         pDialog.dismiss();
                         new ProcessRegister().execute();
                    } else {
                         pDialog.dismiss();
                         forHereErrorMsg.setText("Error occured in registrationhaha");
                    }
               } else {
```

```
                pDialog.dismiss();
                forHereErrorMsg.setText("Error occured in registration");
            }
        } catch (JSONException e) {
            e.printStackTrace();
        }
    }
    }

    public void NetAsync(View view){
        new NetCheck().execute();
        }
}
```

A.Co_for_J.Co\src\com\specialproblem\acoforjco\LoginActivity.java

```
package com.specialproblem.acoforjco;

import java.io.IOException;
import java.net.HttpURLConnection;
import java.net.MalformedURLException;
import java.net.URL;

import org.json.JSONException;
import org.json.JSONObject;

import android.app.Activity;
import android.app.ProgressDialog;
import android.content.Context;
import android.content.Intent;
import android.net.ConnectivityManager;
import android.net.NetworkInfo;
import android.os.AsyncTask;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.TextView;
import android.widget.Toast;

import com.specialproblem.acoforjco.library.DatabaseHandler;
import com.specialproblem.acoforjco.library.UserFunctions;
import com.specialproblem.acoforjco.R;

public class LoginActivity extends Activity {
    Button buttonLogin;
    Button buttonLinkRegister;
    EditText inputIdentifier;
    EditText inputPassword;
    TextView loginErrorMsg;

    // JSON Response node names
    private static String KEY_SUCCESS = "success";
    private static String KEY_UID = "uid";
    private static String KEY_FIRSTNAME = "fname";
    private static String KEY_LASTNAME = "lname";
    private static String KEY_IDENTIFIER = "identifier";
    private static String KEY_DESCRIPTION = "description";
    private static String KEY_CREATED_AT = "created_at";

    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_login);

        // Importing all assets like buttons, text fields
        inputIdentifier = (EditText) findViewById(R.id.login_identifier);
        inputPassword = (EditText) findViewById(R.id.login_password);
        buttonLogin = (Button) findViewById(R.id.button_login);
        buttonLinkRegister = (Button) findViewById(R.id.button_link_register);
        loginErrorMsg = (TextView) findViewById(R.id.login_error);

        // Link to Register Screen
        buttonLinkRegister.setOnClickListener(new View.OnClickListener() {
```

```
            public void onClick(View view) {
                Intent i = new Intent(view.getContext(), RegisterActivity.class);
                startActivityForResult(i, 0);
                finish();
            }
        });

        /**
         * Login button click event
         * A Toast is set to alert when the Identifier or Password field is empty
         **/
        buttonLogin.setOnClickListener(new View.OnClickListener() {

            public void onClick(View view) {
                if ((!inputIdentifier.getText().toString().equals("")) &&
(!inputPassword.getText().toString().equals(""))) {
                    NetAsync(view);
                }
                else if ((!inputIdentifier.getText().toString().equals(""))) {
                    Toast.makeText(getApplicationContext(), "Password field empty", Toast.LENGTH_SHORT).show();
                }
                else if ((!inputPassword.getText().toString().equals(""))) {
                    Toast.makeText(getApplicationContext(), "Identifier field empty", Toast.LENGTH_SHORT).show();
                }
                else {
                    Toast.makeText(getApplicationContext(), "Identifier and Password fields are empty",
Toast.LENGTH_SHORT).show();
                }
            }
        });
    }

    /**
     * Async Task to check whether internet connection is working.
     **/
    private class NetCheck extends AsyncTask<String,String,Boolean> {
        private ProgressDialog nDialog;

        @Override
        protected void onPreExecute(){
            super.onPreExecute();
            nDialog = new ProgressDialog(LoginActivity.this);
            nDialog.setTitle("Checking Network");
            nDialog.setMessage("Loading..");
            nDialog.setIndeterminate(false);
            nDialog.setCancelable(true);
            nDialog.show();
        }

        /**
         * Gets current device state and checks for working Internet connection by trying Google.
         **/
        @Override
        protected Boolean doInBackground(String... args){
            ConnectivityManager cm = (ConnectivityManager) getSystemService(Context.CONNECTIVITY_SERVICE);
            NetworkInfo netInfo = cm.getActiveNetworkInfo();

            if (netInfo != null && netInfo.isConnected()) {
                try {
                    URL url = new URL("http://www.google.com");
                    HttpURLConnection urlc = (HttpURLConnection) url.openConnection();
                    urlc.setConnectTimeout(3000);
                    urlc.connect();
                    if (urlc.getResponseCode() == 200) {
                        return true;
                    }
                } catch (MalformedURLException e1) {
                    e1.printStackTrace();
                } catch (IOException e) {
                    e.printStackTrace();
                }
            }
            return false;
        }

        @Override
        protected void onPostExecute(Boolean th){

            if(th == true){
```

```
                    nDialog.dismiss();
                    new ProcessLogin().execute();
               }
               else{
                    nDialog.dismiss();
                    loginErrorMsg.setText("Error in Network Connection");
               }
          }
     }

     /**
      * Async Task to get and send data to MySQL database through JSON response.
      **/
     private class ProcessLogin extends AsyncTask<String, String, JSONObject> {
          private ProgressDialog pDialog;
          String identifier, password;

          @Override
          protected void onPreExecute() {
               super.onPreExecute();

               inputIdentifier = (EditText) findViewById(R.id.login_identifier);
               inputPassword = (EditText) findViewById(R.id.login_password);
               identifier = inputIdentifier.getText().toString();
               password = inputPassword.getText().toString();
               pDialog = new ProgressDialog(LoginActivity.this);
               pDialog.setTitle("Contacting Servers");
               pDialog.setMessage("Logging in ...");
               pDialog.setIndeterminate(false);
               pDialog.setCancelable(true);
               pDialog.show();
          }

          @Override
          protected JSONObject doInBackground(String... args) {

               UserFunctions userFunction = new UserFunctions();
               JSONObject json = userFunction.loginUser(identifier, password);
               return json;
          }

          @Override
          protected void onPostExecute(JSONObject json) {
               try {
                    if (json.getString(KEY_SUCCESS) != null) {

                         String res = json.getString(KEY_SUCCESS);

                         if(Integer.parseInt(res) == 1){
                              /**
                               * User successfully logged in
                               * Store user details in SQLite Database
                               **/
                              pDialog.setTitle("Getting Data");
                              pDialog.setMessage("Loading User Space");
                              DatabaseHandler db = new DatabaseHandler(getApplicationContext());
                              JSONObject json_user = json.getJSONObject("user");

                              /**
                               * Clears all previous data in SQlite database.
                               **/
                              UserFunctions logout = new UserFunctions();
                              logout.logoutUser(getApplicationContext());

     db.addUser(json_user.getString(KEY_FIRSTNAME),json_user.getString(KEY_LASTNAME),json_user.getString(KEY_IDENTIFIER),
json_user.getString(KEY_DESCRIPTION),json_user.getString(KEY_UID),json_user.getString(KEY_CREATED_AT));

                              /**
                               * If JSON array details are stored in SQlite it launches the Dashboard screen.
                               **/
                              Intent dashboard = new Intent(getApplicationContext(), DashboardActivity.class);
                              dashboard.addFlags(Intent.FLAG_ACTIVITY_CLEAR_TOP);
                              pDialog.dismiss();
                              startActivity(dashboard);

                              /**
                               * Close Login Screen
                               **/
                              finish();
```

```
                    } else {
                        pDialog.dismiss();
                        loginErrorMsg.setText("Incorrect identifier/password");
                    }
                }
            } catch (JSONException e) {
                e.printStackTrace();
            }
        }
    }

    public void NetAsync(View view){
       new NetCheck().execute();
    }
}
```

A.Co_for_J.Co\src\com\specialproblem\acoforjco\RegisterActivity.java

```
package com.specialproblem.acoforjco;

import java.io.IOException;
import java.net.HttpURLConnection;
import java.net.MalformedURLException;
import java.net.URL;

import org.json.JSONException;
import org.json.JSONObject;

import com.specialproblem.acoforjco.library.DatabaseHandler;
import com.specialproblem.acoforjco.library.UserFunctions;
import com.specialproblem.acoforjco.R;

import android.app.Activity;
import android.app.ProgressDialog;
import android.content.Context;
import android.content.Intent;
import android.net.ConnectivityManager;
import android.net.NetworkInfo;
import android.os.AsyncTask;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.TextView;
import android.widget.Toast;

public class RegisterActivity extends Activity {
    Button buttonRegister;
    Button buttonLinkLogin;
    EditText inputFirstName;
    EditText inputLastName;
    EditText inputIdentifier;
    EditText inputDescription;
    EditText inputPassword;
    TextView registerErrorMsg;

    // JSON Response node names
    private static String KEY_SUCCESS = "success";
    private static String KEY_UID = "uid";
    private static String KEY_FIRSTNAME = "fname";
    private static String KEY_LASTNAME = "lname";
    private static String KEY_IDENTIFIER = "identifier";
    private static String KEY_DESCRIPTION = "description";
    private static String KEY_CREATED_AT = "created_at";
    private static String KEY_ERROR = "error";

    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_register);

        // Importing all assets like buttons, text fields
        inputFirstName = (EditText) findViewById(R.id.register_fname);
        inputLastName = (EditText) findViewById(R.id.register_lname);
        inputIdentifier = (EditText) findViewById(R.id.register_identifier);
```

```
        inputDescription = (EditText) findViewById(R.id.register_description);
        inputPassword = (EditText) findViewById(R.id.register_password);
        buttonRegister = (Button) findViewById(R.id.button_register);
        buttonLinkLogin = (Button) findViewById(R.id.button_link_login);
        registerErrorMsg = (TextView) findViewById(R.id.register_error);

        // Link to Login Screen
        buttonLinkLogin.setOnClickListener(new View.OnClickListener() {

            public void onClick(View view) {
                Intent i = new Intent(view.getContext(), LoginActivity.class);
                startActivityForResult(i,0);
                // Close Registration View
                finish();
            }
        });

        /**
         * Register Button click event.
         * A Toast is set to alert when the fields are empty.
         * Another toast is set to alert Description must be a valid user role.
         **/
        buttonRegister.setOnClickListener(new View.OnClickListener() {
            public void onClick(View view) {
                if ((!inputDescription.getText().toString().equals("")) && (
!inputPassword.getText().toString().equals("")) && ( !inputFirstName.getText().toString().equals("")) && (
!inputLastName.getText().toString().equals("")) && ( !inputIdentifier.getText().toString().equals(""))) {
                    if ((inputDescription.getText().toString().equals("Customer")) ||
(inputDescription.getText().toString().equals("Cashier")) || (inputDescription.getText().toString().equals("Counter"))){
                        NetAsync(view);
                    } else {
                        Toast.makeText(getApplicationContext(), "Position should be your role!",
Toast.LENGTH_SHORT).show();
                    }
                } else {
                    Toast.makeText(getApplicationContext(), "One or more fields are empty",
Toast.LENGTH_SHORT).show();
                }
            }
        });
    }

    /**
     * Async Task to check whether internet connection is working
     **/
    private class NetCheck extends AsyncTask<String,String,Boolean> {
        private ProgressDialog nDialog;

        @Override
        protected void onPreExecute(){
            super.onPreExecute();
            nDialog = new ProgressDialog(RegisterActivity.this);
            nDialog.setTitle("Checking Network");
            nDialog.setMessage("Loading..");
            nDialog.setIndeterminate(false);
            nDialog.setCancelable(true);
            nDialog.show();
        }

        /**
      * Gets current device state and checks for working internet connection by trying Google.
      **/
        @Override
        protected Boolean doInBackground(String... args){
          ConnectivityManager cm = (ConnectivityManager) getSystemService(Context.CONNECTIVITY_SERVICE);
          NetworkInfo netInfo = cm.getActiveNetworkInfo();

          if (netInfo != null && netInfo.isConnected()) {
            try {
                URL url = new URL("http://www.google.com");
                HttpURLConnection urlc = (HttpURLConnection) url.openConnection();
                urlc.setConnectTimeout(3000);
                urlc.connect();
                if (urlc.getResponseCode() == 200) {
                    return true;
                }
            } catch (MalformedURLException e1) {
                e1.printStackTrace();
            } catch (IOException e) {
```

```
                e.printStackTrace();
            }
        }
        return false;
    }

    @Override
    protected void onPostExecute(Boolean th){

        if(th == true){
            nDialog.dismiss();
            new ProcessRegister().execute();
        }
        else{
            nDialog.dismiss();
            registerErrorMsg.setText("Error in Network Connection");
        }
    }
}

/**
 * Async Task to get and send data to MySQL database through JSON response.
 **/
private class ProcessRegister extends AsyncTask<String, String, JSONObject> {
 private ProgressDialog pDialog;
 String identifier, password, fname, lname, description;

 @Override
 protected void onPreExecute() {
     super.onPreExecute();

     inputIdentifier = (EditText) findViewById(R.id.register_identifier);
     inputPassword = (EditText) findViewById(R.id.register_password);
     fname = inputFirstName.getText().toString();
     lname = inputLastName.getText().toString();
     identifier = inputIdentifier.getText().toString();
     description= inputDescription.getText().toString();
     password = inputPassword.getText().toString();
     pDialog = new ProgressDialog(RegisterActivity.this);
     pDialog.setTitle("Contacting Servers");
     pDialog.setMessage("Registering ...");
     pDialog.setIndeterminate(false);
     pDialog.setCancelable(true);
     pDialog.show();
 }

 @Override
 protected JSONObject doInBackground(String... args) {
     UserFunctions userFunction = new UserFunctions();
     JSONObject json = userFunction.registerUser(fname, lname, identifier, description, password);

     return json;
 }

 /**
      * Checks for success message.
      **/
 @Override
 protected void onPostExecute(JSONObject json) {
     try {
         if (json.getString(KEY_SUCCESS) != null) {
             registerErrorMsg.setText("");
             String res = json.getString(KEY_SUCCESS);
             String red = json.getString(KEY_ERROR);

             if(Integer.parseInt(res) == 1){
                 /**
                       * User successfully registered
                       * Store user details in SQLite Database
                       **/
                 pDialog.setTitle("Getting Data");
                 pDialog.setMessage("Loading Info");
                 registerErrorMsg.setText("Successfully Registered");
                 DatabaseHandler db = new DatabaseHandler(getApplicationContext());
                 JSONObject json_user = json.getJSONObject("user");

                 /**
                  * Clears all previous data in SQlite database.
                  **/
```

```
                        UserFunctions logout = new UserFunctions();
                        logout.logoutUser(getApplicationContext());

    db.addUser(json_user.getString(KEY_FIRSTNAME),json_user.getString(KEY_LASTNAME),json_user.getString(KEY_IDENTIFIER),
json_user.getString(KEY_DESCRIPTION),json_user.getString(KEY_UID),json_user.getString(KEY_CREATED_AT));

                        /**
                         * If JSON array details are stored in SQlite it launches Registered screen.
                         **/
                        Intent registered = new Intent(getApplicationContext(), DashboardActivity.class);

                        /**
                         * Close all views before launching Registered screen
                         **/
                        registered.addFlags(Intent.FLAG_ACTIVITY_CLEAR_TOP);
                        pDialog.dismiss();
                        startActivity(registered);

                        /**
                              * Close Register Screen
                              **/
                        finish();
                    } else if (Integer.parseInt(red) == 2){
                        pDialog.dismiss();
                        registerErrorMsg.setText("User already exists");
                    } else {
                        pDialog.dismiss();
                        registerErrorMsg.setText("Error occured in registrationhaha");
                    }
                } else {
                    pDialog.dismiss();
                    registerErrorMsg.setText("Error occured in registration");
                }
            } catch (JSONException e) {
                e.printStackTrace();
            }
        }
        }

        public void NetAsync(View view){
         new NetCheck().execute();
        }
}
```

A.Co_for_J.Co\src\com\specialproblem\acoforjco\URLStorage.java

```
package com.specialproblem.acoforjco;

public final class URLStorage {

     private URLStorage() {
    }

     public static final String APIFolder = "http://agila.upm.edu.ph/~bcfestejo/acoforjco_api/";
     public static final String NewAPIFolder = "http://agila.upm.edu.ph/~bcfestejo/acoforjco_api/";

     // public static final String APIFolder = "http://172.16.121.178/acoforjco_api/";
     // public static final String NewAPIFolder = "http://172.16.121.178/new_acoforjco_api/";

     // public static final String APIFolder = "http://192.168.0.10/acoforjco_api/";
     // public static final String NewAPIFolder = "http://192.168.0.10/new_acoforjco_api/";

     public static final String testURL = NewAPIFolder + "test.php";

     public static final String DonutsURL = NewAPIFolder + "donuts.json";
     public static final String CoffeeURL = NewAPIFolder + "coffee.json";
     public static final String ClubURL = NewAPIFolder + "club.json";
     public static final String PopsURL = NewAPIFolder + "pops.json";
     public static final String CoolURL = NewAPIFolder + "cool.json";

     public static final String DonutsFolder = NewAPIFolder + "donuts/";
     public static final String CoffeeFolder = NewAPIFolder + "coffee/";
     public static final String ClubFolder = NewAPIFolder + "club/";
     public static final String PopsFolder = NewAPIFolder + "pops/";
     public static final String CoolFolder = NewAPIFolder + "cool/";
```

```
}
```

A.Co_for_J.Co\src\com\specialproblem\acoforjco\WelcomeActivity.java

```
package com.specialproblem.acoforjco;

import java.io.IOException;
import java.net.HttpURLConnection;
import java.net.MalformedURLException;
import java.net.URL;
import java.text.SimpleDateFormat;
import java.util.Date;

import org.json.JSONException;
import org.json.JSONObject;

import com.specialproblem.acoforjco.library.DatabaseHandler;
import com.specialproblem.acoforjco.library.UserFunctions;
import com.specialproblem.acoforjco.qrcode.EncodeDisplayActivity;

import android.app.Activity;
import android.app.ProgressDialog;
import android.content.Context;
import android.content.Intent;
import android.net.ConnectivityManager;
import android.net.NetworkInfo;
import android.os.AsyncTask;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.TextView;

public class WelcomeActivity extends Activity {

     public final static String QUEUE_MESSAGE = "Customer No. 012";
    public final static String TIME_MESSAGE = "2014-03-22 09:50:58.007";
    public final static String URL_MESSAGE = "http://agila.upm.edu.ph/~bcfestejo/acoforjco_api/";

    private static String KEY_SUCCESS = "success";
     private static String KEY_UID = "uid";
     private static String KEY_FIRSTNAME = "fname";
     private static String KEY_LASTNAME = "lname";
     private static String KEY_IDENTIFIER = "identifier";
     private static String KEY_DESCRIPTION = "description";
     private static String KEY_CREATED_AT = "created_at";
     private static String KEY_ERROR = "error";
     public static int idid = 12;

     TextView welcomeErrorMsg;

     @Override
     public void onCreate(Bundle savedInstanceState){
          super.onCreate(savedInstanceState);
          setContentView(R.layout.activity_welcome);

          welcomeErrorMsg = (TextView) findViewById(R.id.welcome_error);

          Button forHereButton = (Button) findViewById(R.id.button_for_here);
          forHereButton.setOnClickListener(new View.OnClickListener() {
               @Override
               public void onClick(View v) {
                    Intent forHereIntent = new Intent (getBaseContext(), ForHereActivity.class);
                    startActivityForResult(forHereIntent,0);
                    finish();
               }
          });

          Button toGoButton = (Button) findViewById(R.id.button_to_go);
          toGoButton.setOnClickListener(new View.OnClickListener() {
               @Override
               public void onClick(View v) {
                    NetAsync(v);
               }
          });
     }
```

```
@Override
public void onBackPressed() {
     return;
}

private class NetCheck extends AsyncTask<String,String,Boolean> {
    private ProgressDialog nDialog;

    @Override
    protected void onPreExecute(){
        super.onPreExecute();
        nDialog = new ProgressDialog(WelcomeActivity.this);
        nDialog.setTitle("Checking Network");
        nDialog.setMessage("Loading..");
        nDialog.setIndeterminate(false);
        nDialog.setCancelable(true);
        nDialog.show();
    }

    /**
 * Gets current device state and checks for working internet connection by trying Google.
 **/
    @Override
    protected Boolean doInBackground(String... args){
      ConnectivityManager cm = (ConnectivityManager) getSystemService(Context.CONNECTIVITY_SERVICE);
      NetworkInfo netInfo = cm.getActiveNetworkInfo();

      if (netInfo != null && netInfo.isConnected()) {
           try {
                URL url = new URL("http://www.google.com");
                HttpURLConnection urlc = (HttpURLConnection) url.openConnection();
                urlc.setConnectTimeout(3000);
                urlc.connect();
                if (urlc.getResponseCode() == 200) {
                     return true;
                }
           } catch (MalformedURLException e1) {
                e1.printStackTrace();
           } catch (IOException e) {
                e.printStackTrace();
           }
      }
      return false;
    }

    @Override
    protected void onPostExecute(Boolean th){

      if(th == true){
           nDialog.dismiss();
           new ProcessRegister().execute();
      }
      else{
           nDialog.dismiss();
           welcomeErrorMsg.setText("Error in Network Connection");
      }
    }
}

private class ProcessRegister extends AsyncTask<String, String, JSONObject> {
 private ProgressDialog pDialog;
 String fname, lname, identifier, description, password;

 @Override
 protected void onPreExecute() {
      super.onPreExecute();

      pDialog = new ProgressDialog(WelcomeActivity.this);
      pDialog.setTitle("Contacting Servers");
      pDialog.setMessage("Registering ...");
      pDialog.setIndeterminate(false);
      pDialog.setCancelable(true);
      pDialog.show();
 }

 @Override
 protected JSONObject doInBackground(String... args) {

      fname = "Customer No. " + idid;
```

```
            lname = "Customer No. " + idid;
            identifier = "Customer No. " + idid;
            description = "Customer";
            password = new SimpleDateFormat("yyyy-MM-dd HH:mm:ss.SSS").format(new Date());
                idid++;


                System.out.println("welcome" + identifier + "\n" + password);

            UserFunctions userFunction = new UserFunctions();
            JSONObject json = userFunction.registerUser(fname, lname, identifier, description, password);

            return json;
        }

        /**
              * Checks for success message.
              **/
        @Override
        protected void onPostExecute(JSONObject json) {
            try {
                if (json.getString(KEY_SUCCESS) != null) {
                    welcomeErrorMsg.setText("");
                    String res = json.getString(KEY_SUCCESS);
                    String red = json.getString(KEY_ERROR);

                    if(Integer.parseInt(res) == 1){
                        /**
                              * User successfully registered
                              * Store user details in SQLite Database
                              **/
                        pDialog.setTitle("Getting Data");
                        pDialog.setMessage("Loading Info");
                        welcomeErrorMsg.setText("Successfully Registered");
                        DatabaseHandler db = new DatabaseHandler(getApplicationContext());
                        JSONObject json_user = json.getJSONObject("user");

                        /**
                         * Clears all previous data in SQlite database.
                         **/
                        UserFunctions logout = new UserFunctions();
                        logout.logoutUser(getApplicationContext());

    db.addUser(json_user.getString(KEY_FIRSTNAME),json_user.getString(KEY_LASTNAME),json_user.getString(KEY_IDENTIFIER),
json_user.getString(KEY_DESCRIPTION),json_user.getString(KEY_UID),json_user.getString(KEY_CREATED_AT));

                        /**
                         * If JSON array details are stored in SQlite it launches the encoded QR code screen.
                         **/
                        Intent displayQR = new Intent(getApplicationContext(), EncodeDisplayActivity.class);
                        displayQR.putExtra(QUEUE_MESSAGE, identifier);
                    displayQR.putExtra(TIME_MESSAGE, password);
                    displayQR.putExtra(URL_MESSAGE, URL_MESSAGE);

                    /**
                          * Close all views before launching encoded QR Code screen
                          **/
                        displayQR.addFlags(Intent.FLAG_ACTIVITY_CLEAR_TOP);
                        pDialog.dismiss();
                        startActivity(displayQR);

                        /**
                              * Close Register Screen
                              **/
                        finish();
                    } else if (Integer.parseInt(red) == 2){
                        pDialog.dismiss();
                        new ProcessRegister().execute();
                    } else {
                        pDialog.dismiss();
                        welcomeErrorMsg.setText("Error occured in registrationhaha");
                    }
                } else {
                    pDialog.dismiss();
                    welcomeErrorMsg.setText("Error occured in registration");
                }
            } catch (JSONException e) {
                e.printStackTrace();
            }
```

```
     }
    }

     public void NetAsync(View view){
          new NetCheck().execute();
         }
}
```

A.Co_for_J.Co\res\layout\ activity_capture.xml

```
<?xml version="1.0" encoding="UTF-8"?>

<FrameLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent">

    <SurfaceView
        android:id="@+id/preview_view"
        android:layout_width="fill_parent"
        android:layout_height="fill_parent">
    </SurfaceView>

    <com.specialproblem.acoforjco.qrcode.ViewfinderView
        android:id="@+id/viewfinder_view"
         android:layout_width="fill_parent"
         android:layout_height="fill_parent"
         android:background="@color/transparent">
    </com.specialproblem.acoforjco.qrcode.ViewfinderView>

     <LinearLayout
          android:id="@+id/result_view"
         android:orientation="horizontal"
         android:layout_width="fill_parent"
         android:layout_height="fill_parent"
         android:gravity="center"
         android:padding="@dimen/half_padding"
         android:background="@color/result_view"
        android:visibility="gone"
        android:baselineAligned="false">

     <LinearLayout
          android:orientation="vertical"
               android:layout_width="wrap_content"
               android:layout_height="fill_parent"
               android:gravity="right|center_vertical">

            <ImageView
            android:id="@+id/barcode_image_view"
             android:layout_width="160dip"
                android:layout_height="wrap_content"
                android:maxWidth="160dip"
                android:maxHeight="160dip"
                android:layout_marginBottom="@dimen/half_padding"
                android:adjustViewBounds="true"
                android:scaleType="centerInside"
                android:contentDescription="@string/barcode_image">
            </ImageView>

            <LinearLayout
                android:orientation="horizontal"
              android:layout_width="wrap_content"
              android:layout_height="wrap_content">

                <TextView
                    android:id="@+id/format_text_view_label"
                    android:layout_width="wrap_content"
                    android:layout_height="wrap_content"
                    android:text="@string/msg_default_format"
                    android:textColor="@color/result_minor_text"
                    android:textStyle="bold"
                    android:textSize="14sp"
                    android:paddingRight="@dimen/half_padding">
                </TextView>

                <TextView
```

```
        android:id="@+id/format_text_view"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textColor="@color/result_minor_text"
        android:textSize="14sp">
    </TextView>
</LinearLayout>

<LinearLayout
    android:orientation="horizontal"
  android:layout_width="wrap_content"
  android:layout_height="wrap_content">

    <TextView
        android:id="@+id/type_text_view_label"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="@string/msg_default_type"
        android:textColor="@color/result_minor_text"
        android:textStyle="bold"
        android:textSize="14sp"
        android:paddingRight="@dimen/half_padding">
    </TextView>

    <TextView
        android:id="@+id/type_text_view"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textColor="@color/result_minor_text"
        android:textSize="14sp">
    </TextView>
</LinearLayout>

<LinearLayout
    android:orientation="horizontal"
  android:layout_width="wrap_content"
  android:layout_height="wrap_content">

    <TextView
        android:id="@+id/time_text_view_label"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="@string/msg_default_time"
        android:textColor="@color/result_minor_text"
        android:textStyle="bold"
        android:textSize="14sp"
        android:paddingRight="@dimen/half_padding">
    </TextView>

    <TextView
        android:id="@+id/time_text_view"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textColor="@color/result_minor_text"
        android:textSize="14sp">
    </TextView>
</LinearLayout>

<LinearLayout
    android:orientation="horizontal"
    android:layout_width="wrap_content"
  android:layout_height="wrap_content">

    <TextView
        android:id="@+id/meta_text_view_label"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="@string/msg_default_meta"
        android:textColor="@color/result_minor_text"
        android:textStyle="bold"
        android:textSize="14sp"
        android:paddingRight="@dimen/half_padding">
    </TextView>

    <TextView
        android:id="@+id/meta_text_view"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textColor="@color/result_minor_text"
```

```
                android:textSize="14sp">
            </TextView>
        </LinearLayout>
        </LinearLayout>

        <ScrollView
            android:layout_width="wrap_content"
            android:layout_height="wrap_content">

            <LinearLayout
                android:layout_width="wrap_content"
              android:layout_height="wrap_content"
              android:orientation="vertical">

                <TextView
                    android:id="@+id/contents_text_view"
                    android:layout_width="wrap_content"
                    android:layout_height="wrap_content"
                    android:textColor="@color/result_text"
                    android:textColorLink="@color/result_text"
                    android:textSize="40sp"
                    android:paddingLeft="12dip"
                    android:autoLink="web">
                </TextView>

            </LinearLayout>
        </ScrollView>
    </LinearLayout>


        <Button
            android:id="@+id/result_button"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_gravity="bottom|end"
            android:paddingLeft="@dimen/double_padding"
            android:paddingRight="@dimen/double_padding"
            android:text="@string/button_result"
            android:textSize="14sp"
            android:textStyle="bold"
            android:visibility="gone" >
        </Button>

    <!-- Error message -->
    <TextView
        android:id="@+id/capture_error"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
         android:layout_gravity="bottom|center_horizontal"
        android:padding="@dimen/standard_padding"
        android:textColor="@color/login_error_message"
        android:textStyle="bold" >
    </TextView>

     <TextView
         android:id="@+id/status_view"
         android:layout_width="wrap_content"
         android:layout_height="wrap_content"
         android:layout_gravity="bottom|center_horizontal"
         android:background="@color/transparent"
         android:text="@string/msg_default_status"
         android:textColor="@color/status_text">
     </TextView>
</FrameLayout>
```

A.Co_for_J.Co\res\layout\ activity_confirmed_order.xml

```
<?xml version="1.0" encoding="utf-8"?>

<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:background="@color/encode_view"
    android:orientation="horizontal" >
```

```
<LinearLayout
    android:id="@+id/tab_buttons"
    android:layout_width="wrap_content"
    android:layout_height="fill_parent"
     android:layout_gravity="center_vertical"
    android:background="@color/login_background"
    android:gravity="center"
    android:orientation="vertical" >

    <Button
        android:id="@+id/button01"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:layout_marginTop="@dimen/standard_padding"
        android:layout_marginBottom="@dimen/standard_padding"
        android:padding="@dimen/standard_padding"
        android:background="@color/login_background"
        android:text="@string/button_home"
        android:textColor="@color/encode_view" >
    </Button>

    <Button
        android:id="@+id/button02"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:layout_marginTop="@dimen/standard_padding"
        android:layout_marginBottom="@dimen/standard_padding"
        android:padding="@dimen/standard_padding"
        android:background="@color/login_background"
        android:text="@string/button_menu"
        android:textColor="@color/encode_view" >
    </Button>

    <Button
        android:id="@+id/button03"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:layout_marginTop="@dimen/standard_padding"
        android:layout_marginBottom="@dimen/standard_padding"
        android:padding="@dimen/standard_padding"
        android:background="@color/result_points"
        android:text="@string/button_order"
        android:textColor="@color/encode_view" >
    </Button>

    <Button
        android:id="@+id/button04"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:layout_marginTop="@dimen/standard_padding"
        android:layout_marginBottom="@dimen/standard_padding"
        android:padding="@dimen/standard_padding"
        android:background="@color/login_background"
        android:text="@string/button_queue"
        android:textColor="@color/encode_view" >
    </Button>

    <Button
        android:id="@+id/button05"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:layout_marginTop="@dimen/standard_padding"
        android:layout_marginBottom="@dimen/standard_padding"
        android:padding="@dimen/standard_padding"
        android:background="@color/login_background"
        android:text="@string/button_logout"
        android:textColor="@color/encode_view" >
    </Button>
</LinearLayout>

<LinearLayout
    android:id="@+id/body_customer_order"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:orientation="vertical" >

     <TextView
        android:id="@+id/customer_order_title"
        android:layout_width="fill_parent"
```

```
    android:layout_height="wrap_content"
    android:gravity="center"
   android:padding="@dimen/double_padding"
    android:text="@string/customer_order_title"
    android:textColor="@color/font_color"
    android:textSize="30sp"
    android:textStyle="bold" >
</TextView>

 <LinearLayout
    android:id="@+id/customer_order_tabs"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:orientation="horizontal" >

    <TextView
        android:id="@+id/order_quantity"
       android:layout_height="fill_parent"
       android:layout_width="0dip"
      android:layout_weight="1"
       android:gravity="center"
        android:text="@string/order_quantity"
        android:textColor="@color/font_color"
        android:textSize="40sp" >
    </TextView>

    <TextView
        android:id="@+id/order_name"
       android:layout_height="fill_parent"
       android:layout_width="0dip"
      android:layout_weight="1"
       android:gravity="center"
        android:text="@string/order_name"
        android:textColor="@color/font_color"
        android:textSize="40sp" >
    </TextView>

    <TextView
        android:id="@+id/order_price"
       android:layout_height="fill_parent"
       android:layout_width="0dip"
      android:layout_weight="1"
       android:gravity="center"
        android:text="@string/order_price"
        android:textColor="@color/font_color"
        android:textSize="40sp" >
    </TextView>
</LinearLayout>

 <LinearLayout
    android:id="@+id/customer_orders"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:orientation="vertical">

    <ListView
        android:id="@+id/list_order"
        android:layout_width="fill_parent"
        android:layout_height="fill_parent"
        android:textSize="30sp" >
    </ListView>
</LinearLayout>

 <TextView
    android:id="@+id/discount_text"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_gravity="right"
    android:textColor="@color/font_color"
    android:text="@string/label_discount">
</TextView>

 <TextView
    android:id="@+id/taxrate_text"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_gravity="right"
    android:textColor="@color/font_color"
    android:text="@string/label_taxrate">
```

```
        </TextView>

         <TextView
            android:id="@+id/total_text"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_gravity="right"
            android:textColor="@color/font_color"
            android:text="@string/label_total">
        </TextView>

         <Button
            android:id="@+id/Button06"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_gravity="bottom|center_horizontal"
             android:paddingLeft="@dimen/double_padding"
             android:paddingRight="@dimen/double_padding"
         android:text="@string/button_endsession" >
        </Button>

    </LinearLayout>
</LinearLayout>
```

A.Co_for_J.Co\res\layout\ activity_connected_user.xml

```
<?xml version="1.0" encoding="utf-8"?>

<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:background="@color/encode_view"
    android:orientation="vertical" >

     <Button
        android:id="@+id/button_got_it"
        android:layout_width="wrap_content"
         android:layout_height="wrap_content"
        android:layout_marginTop="@dimen/standard_padding"
        android:layout_marginBottom="@dimen/standard_padding"
        android:textColor="@color/encode_view"
        android:background="@drawable/connected_user">
    </Button>
</LinearLayout>
```

A.Co_for_J.Co\res\layout\ activity_customer_dashboard.xml

A.Co_for_J.Co\res\layout\ activity_customer_order.xml

```
<?xml version="1.0" encoding="utf-8"?>

<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:background="@color/encode_view"
    android:orientation="horizontal" >

    <LinearLayout
        android:id="@+id/tab_buttons"
        android:layout_width="wrap_content"
        android:layout_height="fill_parent"
         android:layout_gravity="center_vertical"
        android:background="@color/login_background"
        android:gravity="center"
        android:orientation="vertical" >

        <Button
            android:id="@+id/button01"
```

```
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:layout_marginTop="@dimen/standard_padding"
        android:layout_marginBottom="@dimen/standard_padding"
        android:padding="@dimen/standard_padding"
        android:background="@color/login_background"
        android:text="@string/button_home"
        android:textColor="@color/encode_view" >
    </Button>

    <Button
        android:id="@+id/button02"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:layout_marginTop="@dimen/standard_padding"
        android:layout_marginBottom="@dimen/standard_padding"
        android:padding="@dimen/standard_padding"
        android:background="@color/login_background"
        android:text="@string/button_menu"
        android:textColor="@color/encode_view" >
    </Button>

    <Button
        android:id="@+id/button03"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:layout_marginTop="@dimen/standard_padding"
        android:layout_marginBottom="@dimen/standard_padding"
        android:padding="@dimen/standard_padding"
        android:background="@color/result_points"
        android:text="@string/button_order"
        android:textColor="@color/encode_view" >
    </Button>

    <Button
        android:id="@+id/button04"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:layout_marginTop="@dimen/standard_padding"
        android:layout_marginBottom="@dimen/standard_padding"
        android:padding="@dimen/standard_padding"
        android:background="@color/login_background"
        android:text="@string/button_queue"
        android:textColor="@color/encode_view" >
    </Button>

    <Button
        android:id="@+id/button05"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:layout_marginTop="@dimen/standard_padding"
        android:layout_marginBottom="@dimen/standard_padding"
        android:padding="@dimen/standard_padding"
        android:background="@color/login_background"
        android:text="@string/button_logout"
        android:textColor="@color/encode_view" >
    </Button>
</LinearLayout>

<LinearLayout
    android:id="@+id/body_customer_order"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:orientation="vertical" >

     <TextView
        android:id="@+id/customer_order_title"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:gravity="center"
       android:padding="@dimen/double_padding"
        android:text="@string/customer_order_title"
        android:textColor="@color/font_color"
        android:textSize="30sp"
        android:textStyle="bold" >
    </TextView>

    <LinearLayout
        android:id="@+id/customer_order_tabs"
```

```
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:orientation="horizontal" >

            <TextView
                android:id="@+id/order_quantity"
               android:layout_height="fill_parent"
               android:layout_width="0dip"
              android:layout_weight="1"
                android:gravity="center"
                android:text="@string/order_quantity"
                android:textColor="@color/font_color"
                android:textSize="22sp" >
            </TextView>

            <TextView
                android:id="@+id/order_name"
               android:layout_height="fill_parent"
               android:layout_width="0dip"
              android:layout_weight="1"
                android:gravity="center"
                android:text="@string/order_name"
                android:textColor="@color/font_color"
                android:textSize="22sp" >
            </TextView>

            <TextView
                android:id="@+id/order_price"
               android:layout_height="fill_parent"
               android:layout_width="0dip"
              android:layout_weight="1"
                android:gravity="center"
                android:text="@string/order_price"
                android:textColor="@color/font_color"
                android:textSize="22sp" >
            </TextView>
        </LinearLayout>

        <LinearLayout
            android:id="@+id/customer_orders"
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:orientation="vertical">

            <ListView
                android:id="@+id/list_order"
                android:layout_width="fill_parent"
                android:layout_height="fill_parent" >
            </ListView>
        </LinearLayout>

        <TextView
            android:id="@+id/total_text"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_gravity="right"
            android:paddingRight="@dimen/double_padding"
            android:textColor="@color/font_color" >
        </TextView>

        <Button
            android:id="@+id/Button06"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_gravity="bottom|center_horizontal"
             android:paddingLeft="@dimen/double_padding"
             android:paddingRight="@dimen/double_padding"
         android:text="@string/button_confirm" >
        </Button>

    </LinearLayout>
</LinearLayout>
```

A.Co_for_J.Co\res\layout\ activity_dashboard.xml

```
<?xml version="1.0" encoding="utf-8"?>
```

```
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:paddingBottom="@dimen/standard_padding"
    android:paddingLeft="@dimen/standard_padding"
    android:paddingRight="@dimen/standard_padding"
    android:paddingTop="@dimen/standard_padding"
    android:background="@color/login_background"
    android:orientation="vertical">

    <TextView
        android:id="@+id/text_welcome"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_gravity="center_horizontal"
        android:layout_marginTop="@dimen/standard_padding"
        android:layout_marginBottom="@dimen/standard_padding"
        android:textColor="@color/encode_view"
        android:textSize="40sp" >
    </TextView>

    <LinearLayout
         android:layout_width="fill_parent"
         android:layout_height="fill_parent"
        android:gravity="bottom"
         android:orientation="horizontal">

         <Button
             android:id="@+id/button_logout"
             android:layout_width="fill_parent"
             android:layout_height="wrap_content"
             android:layout_weight="1"
             android:layout_marginLeft="@dimen/standard_padding"
             android:layout_marginRight="@dimen/standard_padding"
             android:layout_marginBottom="@dimen/double_padding"
          android:gravity="center_horizontal"
             android:padding="@dimen/double_padding"
             android:background="@color/result_points"
            android:text="@string/button_link_logout"
             android:textColor="@color/encode_view"
             android:textStyle="bold">
          </Button>

         <Button
             android:id="@+id/button_order"
             android:layout_width="fill_parent"
             android:layout_height="wrap_content"
             android:layout_weight="1"
             android:layout_marginLeft="@dimen/standard_padding"
             android:layout_marginRight="@dimen/standard_padding"
             android:layout_marginBottom="@dimen/double_padding"
          android:gravity="center_horizontal"
             android:padding="@dimen/double_padding"
             android:background="@color/result_points"
             android:text="@string/button_order_now"
             android:textColor="@color/encode_view"
             android:textStyle="bold">
          </Button>
     </LinearLayout>
</LinearLayout>
```

A.Co_for_J.Co\res\layout\ activity_encoder.xml

```
<?xml version="1.0" encoding="UTF-8"?>

<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingBottom="@dimen/standard_padding"
    android:paddingLeft="@dimen/standard_padding"
    android:paddingRight="@dimen/standard_padding"
    android:paddingTop="@dimen/standard_padding"
```

```
    android:orientation="vertical">

    <TextView
         android:id="@+id/text_label_queues"
         android:layout_width="wrap_content"
         android:layout_height="wrap_content"
         android:text="@string/label_queues">
     </TextView>

    <EditText
        android:id="@+id/edit_queues"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:singleLine="true"
        android:hint="@string/edit_queues">
    </EditText>

    <TextView
         android:id="@+id/text_label_times"
         android:layout_width="wrap_content"
         android:layout_height="wrap_content"
         android:text="@string/label_times">
     </TextView>

    <EditText
        android:id="@+id/edit_times"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:singleLine="true"
        android:hint="@string/edit_times">
    </EditText>

    <TextView
         android:id="@+id/text_label_urls"
         android:layout_width="wrap_content"
         android:layout_height="wrap_content"
         android:text="@string/label_urls">
     </TextView>

    <EditText
        android:id="@+id/edit_urls"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:singleLine="true"
        android:hint="@string/edit_urls">
    </EditText>

     <Button
         android:layout_width="wrap_content"
         android:layout_height="wrap_content"
         android:paddingLeft="@dimen/double_padding"
         android:paddingRight="@dimen/double_padding"
         android:layout_gravity="bottom|center_horizontal"
         android:onClick="sendEntries"
         android:text="@string/button_done" >
     </Button>

</LinearLayout>
```

A.Co_for_J.Co\res\layout\ activity_encoder_display.xml

```
<?xml version="1.0" encoding="UTF-8"?>

 <LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:id="@+id/encode_view"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:fillViewport="true"
    android:background="@color/encode_view"
    android:orientation="vertical"
    android:gravity="center">

    <ImageView
        android:id="@+id/image_view"
```

```
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_gravity="center_horizontal"
        android:scaleType="center"
        android:contentDescription="@string/barcode_image">
    </ImageView>

    <ScrollView
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:layout_gravity="center_horizontal"
        android:gravity="center">

        <TextView
             android:id="@+id/contents_text_view2"
             android:layout_width="fill_parent"
             android:layout_height="wrap_content"
             android:layout_gravity="center_horizontal"
             android:gravity="center"
             android:textColor="@color/contents_text"
             android:textSize="18sp"
             android:paddingBottom="@dimen/standard_padding"
             android:paddingLeft="@dimen/standard_padding"
             android:paddingRight="@dimen/standard_padding"/>
     </ScrollView>

    <Button
        android:id="@+id/button_go_back"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginLeft="@dimen/double_padding"
        android:layout_marginRight="@dimen/double_padding"
        android:background="@color/result_points"
        android:padding="@dimen/double_padding"
        android:text="@string/button_got_it"
        android:textColor="@color/login_background"
        android:textStyle="bold" >
    </Button>
</LinearLayout>
```

A.Co_for_J.Co\res\layout\ activity_food_details.xml

```
<?xml version="1.0" encoding="utf-8"?>

<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:background="@color/encode_view"
    android:orientation="horizontal" >

     <LinearLayout
        android:id="@+id/tab_buttons"
        android:layout_width="wrap_content"
        android:layout_height="fill_parent"
         android:layout_gravity="center_vertical"
        android:background="@color/login_background"
        android:gravity="center"
        android:orientation="vertical" >

        <Button
            android:id="@+id/button01"
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:layout_marginTop="@dimen/standard_padding"
            android:layout_marginBottom="@dimen/standard_padding"
            android:padding="@dimen/standard_padding"
            android:background="@color/login_background"
            android:text="@string/button_home"
            android:textColor="@color/encode_view" >
        </Button>

        <Button
            android:id="@+id/button02"
            android:layout_width="fill_parent"
```

```
        android:layout_height="wrap_content"
        android:layout_marginTop="@dimen/standard_padding"
        android:layout_marginBottom="@dimen/standard_padding"
        android:padding="@dimen/standard_padding"
        android:background="@color/result_points"
        android:text="@string/button_menu"
        android:textColor="@color/encode_view" >
    </Button>

    <Button
        android:id="@+id/button03"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:layout_marginTop="@dimen/standard_padding"
        android:layout_marginBottom="@dimen/standard_padding"
        android:padding="@dimen/standard_padding"
        android:background="@color/login_background"
        android:text="@string/button_order"
        android:textColor="@color/encode_view" >
    </Button>

    <Button
        android:id="@+id/button04"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:layout_marginTop="@dimen/standard_padding"
        android:layout_marginBottom="@dimen/standard_padding"
        android:padding="@dimen/standard_padding"
        android:background="@color/login_background"
        android:text="@string/button_queue"
        android:textColor="@color/encode_view" >
    </Button>

    <Button
        android:id="@+id/button05"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:layout_marginTop="@dimen/standard_padding"
        android:layout_marginBottom="@dimen/standard_padding"
        android:padding="@dimen/standard_padding"
        android:background="@color/login_background"
        android:text="@string/button_logout"
        android:textColor="@color/encode_view" >
    </Button>
</LinearLayout>

 <LinearLayout
    android:id="@+id/body_food_details"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:orientation="vertical" >

     <TextView
        android:id="@+id/food_title"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
       android:padding="@dimen/standard_padding"
        android:gravity="center"
        android:text="@string/food_title"
        android:textColor="@color/font_color"
        android:textSize="30sp"
        android:textStyle="bold" >
    </TextView>

     <LinearLayout
        android:id="@+id/body_food_image"
        android:layout_width="fill_parent"
        android:layout_height="fill_parent"
        android:orientation="horizontal" >

         <ImageView
            android:id="@+id/home_image"
            android:layout_width="0dip"
            android:layout_height="fill_parent"
            android:layout_weight="1"
            android:orientation="vertical"
            android:src="@drawable/icon" >
         </ImageView>
```

```
<LinearLayout
    android:id="@+id/body_food_information"
    android:layout_width="0dip"
    android:layout_height="fill_parent"
    android:layout_weight="1"
    android:layout_gravity="right"
    android:orientation="vertical" >

    <TextView
        android:id="@+id/name_text"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
     android:textColor="@color/font_color"
     android:textSize="20sp" >
    </TextView>

    <TextView
        android:id="@+id/details_text"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
     android:textColor="@color/font_color"
     android:textSize="15sp" >
    </TextView>

    <TextView
        android:id="@+id/preptime_text"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
     android:textColor="@color/font_color"
     android:textSize="15sp" >
    </TextView>

    <TextView
        android:id="@+id/price_text"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
     android:textColor="@color/font_color"
     android:textSize="15sp" >
    </TextView>

     <LinearLayout
         android:id="@+id/body_food_yes"
         android:layout_width="fill_parent"
         android:layout_height="0dip"
         android:layout_weight="1"
         android:layout_gravity="bottom|end"
         android:gravity="bottom|end"
          android:layout_marginBottom="@dimen/double_padding"
         android:orientation="horizontal" >

         <Button
             android:id="@+id/bSub"
             android:layout_width="wrap_content"
             android:layout_height="wrap_content"
               android:padding="@dimen/double_padding"
          android:background="@color/result_points"
          android:textColor="@color/login_background"
               android:textStyle="bold"
               android:text="@string/button_minus" >
         </Button>

         <TextView
             android:id="@+id/tvDisplay"
             android:layout_width="wrap_content"
             android:layout_height="wrap_content"
               android:paddingLeft="@dimen/double_padding"
               android:layout_marginRight="@dimen/double_padding"
               android:textStyle="bold"
               android:textColor="@color/login_background"
               android:text="@string/button_zero" >
         </TextView>

         <Button
             android:id="@+id/bAdd"
             android:layout_width="wrap_content"
             android:layout_height="wrap_content"
               android:padding="@dimen/double_padding"
          android:background="@color/result_points"
          android:textColor="@color/login_background"
```

```
                    android:textStyle="bold"
                    android:text="@string/button_plus" >
                </Button>

                <Button
                    android:id="@+id/Button06"
                    android:layout_width="wrap_content"
                    android:layout_height="wrap_content"
                     android:layout_marginRight="@dimen/double_padding"
                     android:layout_marginLeft="@dimen/double_padding"
                     android:padding="@dimen/double_padding"
                  android:background="@color/result_points"
                  android:textColor="@color/login_background"
                  android:text="@string/button_add" >
                </Button>
            </LinearLayout>
        </LinearLayout>
    </LinearLayout>
</LinearLayout>
```

A.Co_for_J.Co\res\layout\ activity_food_items.xml

```
<?xml version="1.0" encoding="utf-8"?>

<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:background="@color/encode_view"
    android:orientation="horizontal" >

    <LinearLayout
        android:id="@+id/tab_buttons"
        android:layout_width="wrap_content"
        android:layout_height="fill_parent"
         android:layout_gravity="center_vertical"
        android:background="@color/login_background"
        android:gravity="center"
        android:orientation="vertical" >

        <Button
            android:id="@+id/button01"
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:layout_marginTop="@dimen/standard_padding"
            android:layout_marginBottom="@dimen/standard_padding"
            android:padding="@dimen/standard_padding"
            android:background="@color/login_background"
            android:text="@string/button_home"
            android:textColor="@color/encode_view" >
        </Button>

        <Button
            android:id="@+id/button02"
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:layout_marginTop="@dimen/standard_padding"
            android:layout_marginBottom="@dimen/standard_padding"
            android:padding="@dimen/standard_padding"
            android:background="@color/result_points"
            android:text="@string/button_menu"
            android:textColor="@color/encode_view" >
        </Button>

        <Button
            android:id="@+id/button03"
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:layout_marginTop="@dimen/standard_padding"
            android:layout_marginBottom="@dimen/standard_padding"
            android:padding="@dimen/standard_padding"
            android:background="@color/login_background"
            android:text="@string/button_order"
            android:textColor="@color/encode_view" >
        </Button>
```

```
        <Button
            android:id="@+id/button04"
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:layout_marginTop="@dimen/standard_padding"
            android:layout_marginBottom="@dimen/standard_padding"
            android:padding="@dimen/standard_padding"
            android:background="@color/login_background"
            android:text="@string/button_queue"
            android:textColor="@color/encode_view" >
        </Button>

        <Button
            android:id="@+id/button05"
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:layout_marginTop="@dimen/standard_padding"
            android:layout_marginBottom="@dimen/standard_padding"
            android:padding="@dimen/standard_padding"
            android:background="@color/login_background"
            android:text="@string/button_logout"
            android:textColor="@color/encode_view" >
        </Button>
    </LinearLayout>

    <LinearLayout
        android:id="@+id/body_menu"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:orientation="vertical" >

         <TextView
            android:id="@+id/home_title"
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:gravity="center"
           android:padding="@dimen/double_padding"
            android:text="@string/menu_title"
            android:textColor="@color/font_color"
            android:textSize="30sp"
            android:textStyle="bold" >
        </TextView>

         <ListView
           android:id="@+id/list_order"
           android:layout_width="fill_parent"
           android:layout_height="fill_parent"
           android:padding="@dimen/double_padding" >
        </ListView>
    </LinearLayout>
</LinearLayout>
```

A.Co_for_J.Co\res\layout\ activity_forhere.xml

```
<?xml version="1.0" encoding="utf-8"?>

<ScrollView
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:paddingBottom="@dimen/standard_padding"
    android:paddingLeft="@dimen/standard_padding"
    android:paddingRight="@dimen/standard_padding"
    android:paddingTop="@dimen/standard_padding"
    android:background="@color/login_background"
    android:orientation="vertical" >

    <LinearLayout
        android:layout_width="fill_parent"
    android:layout_height="wrap_content"
        android:orientation="vertical"
        android:padding="@dimen/standard_padding" >

        <!--  Fullame Label -->
        <TextView
```

```
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:padding="@dimen/double_padding"
            android:text="@string/label_fullname" >
         </TextView>
        <!--  Fullname TextField -->
        <EditText
            android:id="@+id/forhere_fullname"
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:padding="@dimen/double_padding"
            android:singleLine="true"
              android:hint="@string/edit_fullname" >
    </EditText>

        <!--  Verifier Label -->
        <TextView
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:padding="@dimen/double_padding"
            android:text="@string/label_verifier" >
         </TextView>
        <!--  Verifier TextField -->
        <EditText
            android:id="@+id/forhere_verifier"
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:padding="@dimen/double_padding"
              android:hint="@string/edit_verifier" >
    </EditText>

        <!--  Error message -->
        <TextView
            android:id="@+id/forhere_error"
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:textColor="@color/login_error_message"
            android:padding="@dimen/standard_padding"
            android:textStyle="bold">
         </TextView>

        <!--  Login Button -->
        <Button
            android:id="@+id/button_forhere_register"
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:layout_marginTop="@dimen/double_padding"
            android:layout_marginBottom="@dimen/double_padding"
            android:padding="@dimen/double_padding"
         android:text="@string/button_register" >
         </Button>
    </LinearLayout>
</ScrollView>
```

A.Co_for_J.Co\res\layout\ activity_history.xml

```
<?xml version="1.0" encoding="UTF-8"?>

<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="vertical"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:padding="@dimen/standard_padding">

    <TextView
        android:id="@+id/history_title"
         android:layout_width="fill_parent"
         android:layout_height="wrap_content"
        android:textAppearance="?android:attr/textAppearanceLarge"
        android:singleLine="true">
    </TextView>

    <TextView
        android:id="@+id/history_detail"
         android:layout_width="fill_parent"
```

```
     android:layout_height="wrap_content"
        android:textAppearance="?android:attr/textAppearanceSmall"
        android:singleLine="false">
     </TextView>

</LinearLayout>
```

A.Co_for_J.Co\res\layout\ activity_home.xml

```
<?xml version="1.0" encoding="utf-8"?>

<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:background="@color/encode_view"
    android:orientation="horizontal" >

    <LinearLayout
        android:id="@+id/tab_buttons"
        android:layout_width="wrap_content"
        android:layout_height="fill_parent"
         android:layout_gravity="center_vertical"
        android:background="@color/login_background"
        android:gravity="center"
        android:orientation="vertical" >

        <Button
            android:id="@+id/button01"
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:layout_marginTop="@dimen/standard_padding"
            android:layout_marginBottom="@dimen/standard_padding"
            android:padding="@dimen/standard_padding"
            android:background="@color/result_points"
            android:text="@string/button_home"
            android:textColor="@color/encode_view" >
        </Button>

        <Button
            android:id="@+id/button02"
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:layout_marginTop="@dimen/standard_padding"
            android:layout_marginBottom="@dimen/standard_padding"
            android:padding="@dimen/standard_padding"
            android:background="@color/login_background"
            android:text="@string/button_menu"
            android:textColor="@color/encode_view" >
        </Button>

        <Button
            android:id="@+id/button03"
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:layout_marginTop="@dimen/standard_padding"
            android:layout_marginBottom="@dimen/standard_padding"
            android:padding="@dimen/standard_padding"
            android:background="@color/login_background"
            android:text="@string/button_order"
            android:textColor="@color/encode_view" >
        </Button>

        <Button
            android:id="@+id/button04"
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:layout_marginTop="@dimen/standard_padding"
            android:layout_marginBottom="@dimen/standard_padding"
            android:padding="@dimen/standard_padding"
            android:background="@color/login_background"
            android:text="@string/button_queue"
            android:textColor="@color/encode_view" >
        </Button>
```

```
        <Button
            android:id="@+id/button05"
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:layout_marginTop="@dimen/standard_padding"
            android:layout_marginBottom="@dimen/standard_padding"
            android:padding="@dimen/standard_padding"
            android:background="@color/login_background"
            android:text="@string/button_logout"
            android:textColor="@color/encode_view" >
        </Button>
    </LinearLayout>

    <LinearLayout
        android:id="@+id/body_home"
        android:layout_width="fill_parent"
        android:layout_height="fill_parent"
        android:orientation="vertical" >

         <TextView
            android:id="@+id/home_title"
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:gravity="center"
           android:padding="@dimen/double_padding"
            android:text="@string/home_title"
            android:textColor="@color/font_color"
            android:textSize="30sp"
            android:textStyle="bold" >
        </TextView>

        <ImageView
            android:id="@+id/home_image"
            android:layout_width="fill_parent"
            android:layout_height="fill_parent"
            android:layout_gravity="center_vertical"
            android:gravity="center"
            android:orientation="vertical"
            android:src="@drawable/homepage" >
         </ImageView>
    </LinearLayout>
</LinearLayout>
```

A.Co_for_J.Co\res\layout\ activity_login.xml

```
<?xml version="1.0" encoding="utf-8"?>

<ScrollView
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:paddingBottom="@dimen/standard_padding"
    android:paddingLeft="@dimen/standard_padding"
    android:paddingRight="@dimen/standard_padding"
    android:paddingTop="@dimen/standard_padding"
    android:background="@color/login_background"
    android:orientation="vertical" >

    <LinearLayout
         android:layout_width="fill_parent"
     android:layout_height="wrap_content"
        android:orientation="vertical"
        android:padding="@dimen/standard_padding" >

        <!--  Identifier Label -->
        <TextView
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:padding="@dimen/double_padding"
            android:text="@string/label_identifier" >
         </TextView>
        <!--  Identifier TextField -->
        <EditText
            android:id="@+id/login_identifier"
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
```

```
            android:padding="@dimen/double_padding"
            android:singleLine="true"
              android:hint="@string/edit_identifier" >
    </EditText>

        <!--  Password Label -->
        <TextView
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:padding="@dimen/double_padding"
            android:text="@string/label_password" >
         </TextView>
        <!--  Password TextField -->
        <EditText
            android:id="@+id/login_password"
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:padding="@dimen/double_padding"
            android:singleLine="true"
            android:inputType="textPassword"
              android:hint="@string/edit_password" >
    </EditText>

        <!--  Error message -->
        <TextView
            android:id="@+id/login_error"
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:textColor="@color/login_error_message"
            android:padding="@dimen/standard_padding"
            android:textStyle="bold">
         </TextView>

        <!--  Login Button -->
        <Button
            android:id="@+id/button_login"
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:layout_marginTop="@dimen/double_padding"
            android:layout_marginBottom="@dimen/double_padding"
         android:text="@string/button_login" >
         </Button>

        <!--  Link to Registration Screen -->
        <Button
            android:id="@+id/button_link_register"
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:layout_marginTop="@dimen/double_padding"
            android:layout_gravity="bottom|center_horizontal"
            android:background="@null"
            android:text="@string/button_link_register"
            android:textColor="@color/login_register_text"
            android:textStyle="bold" >
         </Button>
    </LinearLayout>
</ScrollView>
```

A.Co_for_J.Co\res\layout\ activity_main.xml

A.Co_for_J.Co\res\layout\ activity_main_menu.xml

A.Co_for_J.Co\res\layout\ activity_main_order.xml

A.Co_for_J.Co\res\layout\ activity_main_queue.xml

A.Co_for_J.Co\res\layout\ activity_menu.xml

```
<?xml version="1.0" encoding="utf-8"?>

<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:background="@color/encode_view"
    android:orientation="horizontal" >

     <LinearLayout
        android:id="@+id/tab_buttons"
        android:layout_width="wrap_content"
        android:layout_height="fill_parent"
         android:layout_gravity="center_vertical"
        android:background="@color/login_background"
        android:gravity="center"
        android:orientation="vertical" >

        <Button
            android:id="@+id/button01"
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:layout_marginTop="@dimen/standard_padding"
            android:layout_marginBottom="@dimen/standard_padding"
            android:padding="@dimen/standard_padding"
            android:background="@color/login_background"
            android:text="@string/button_home"
            android:textColor="@color/encode_view">
        </Button>

        <Button
            android:id="@+id/button02"
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:layout_marginTop="@dimen/standard_padding"
            android:layout_marginBottom="@dimen/standard_padding"
            android:padding="@dimen/standard_padding"
            android:background="@color/result_points"
            android:text="@string/button_menu"
            android:textColor="@color/encode_view" >
        </Button>

        <Button
            android:id="@+id/button03"
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:layout_marginTop="@dimen/standard_padding"
            android:layout_marginBottom="@dimen/standard_padding"
            android:padding="@dimen/standard_padding"
            android:background="@color/login_background"
            android:text="@string/button_order"
            android:textColor="@color/encode_view" >
        </Button>

        <Button
            android:id="@+id/button04"
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:layout_marginTop="@dimen/standard_padding"
            android:layout_marginBottom="@dimen/standard_padding"
            android:padding="@dimen/standard_padding"
            android:background="@color/login_background"
            android:text="@string/button_queue"
            android:textColor="@color/encode_view" >
        </Button>

        <Button
            android:id="@+id/button05"
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:layout_marginTop="@dimen/standard_padding"
            android:layout_marginBottom="@dimen/standard_padding"
            android:padding="@dimen/standard_padding"
```

```
        android:background="@color/login_background"
        android:text="@string/button_logout"
        android:textColor="@color/encode_view" >
    </Button>
</LinearLayout>

 <LinearLayout
    android:id="@+id/body_menu"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:orientation="vertical" >

     <TextView
        android:id="@+id/home_title"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:gravity="center"
       android:padding="@dimen/double_padding"
        android:text="@string/menu_title"
        android:textColor="@color/font_color"
        android:textSize="30sp"
        android:textStyle="bold" >
     </TextView>

     <LinearLayout
        android:id="@+id/menu_options"
        android:layout_width="fill_parent"
        android:layout_height="fill_parent"
        android:orientation="vertical" >

        <LinearLayout
            android:id="@+id/upper_menu"
            android:layout_width="fill_parent"
            android:layout_height="0dip"
            android:layout_weight="1"
            android:gravity="center_horizontal" >

            <Button
                android:id="@+id/button06"
              android:layout_weight="1"
                android:layout_width="0dip"
                android:layout_height="fill_parent"
                android:background="@drawable/jdonut" >
            </Button>

            <Button
                android:id="@+id/button07"
              android:layout_weight="1"
                android:layout_width="0dip"
                android:layout_height="fill_parent"
                android:background="@drawable/jcoffee" >
            </Button>

            <Button
                android:id="@+id/button08"
              android:layout_weight="1"
                android:layout_width="0dip"
                android:layout_height="fill_parent"
                android:background="@drawable/jclub" >
            </Button>
        </LinearLayout>

        <LinearLayout
            android:id="@+id/lower_menu"
            android:layout_width="fill_parent"
            android:layout_height="0dip"
            android:layout_weight="1"
            android:gravity="center_horizontal" >

            <Button
                android:id="@+id/button09"
              android:layout_weight="1"
                android:layout_width="0dip"
                android:layout_height="fill_parent"
                android:background="@drawable/jpops">
            </Button>

            <Button
                android:id="@+id/button10"
```

```
                  android:layout_weight="1"
                    android:layout_width="0dip"
                    android:layout_height="fill_parent"
                    android:background="@drawable/jcool" >
                </Button>

                <Button
                    android:id="@+id/button11"
                  android:layout_weight="1"
                    android:layout_width="0dip"
                    android:layout_height="fill_parent"
                    android:background="@color/encode_view" >
                </Button>
            </LinearLayout>
        </LinearLayout>
    </LinearLayout>
</LinearLayout>
```

A.Co_for_J.Co\res\layout\ activity_menulist.xml

```
<?xml version="1.0" encoding="utf-8"?>

<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:background="@color/encode_view"
    android:orientation="horizontal" >

    <LinearLayout
        android:id="@+id/menu_item"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:orientation="vertical" >

        <TextView
            android:id="@+id/menu_item_name"
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:text="@string/order_name"
             android:textSize="30sp"
             android:textStyle="bold"
            android:textColor="@color/font_color" >
        </TextView>

        <TextView
            android:id="@+id/menu_item_description"
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:text="@string/order_name"
            android:textSize="20sp"
             android:textStyle="bold"
            android:textColor="@color/font_color" >
        </TextView>
    </LinearLayout>

    <TextView
     android:id="@+id/menu_item_price"
        android:layout_width="0dip"
        android:layout_height="wrap_content"
        android:layout_weight="1"
        android:gravity="right"
          android:paddingLeft="@dimen/double_padding"
          android:paddingRight="@dimen/double_padding"
        android:text="@string/order_price"
        android:textStyle="bold"
        android:textSize="14sp"
        android:textColor="@color/font_color" >
     </TextView>

</LinearLayout>
```

A.Co_for_J.Co\res\layout\ activity_orderlist.xml

```
<?xml version="1.0" encoding="utf-8"?>

<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:background="@color/encode_view"
    android:orientation="horizontal" >

    <LinearLayout
        android:id="@+id/tab_button1"
        android:layout_width="0dip"
       android:layout_height="wrap_content"
       android:layout_weight="1" >

        <TextView
            android:id="@+id/text_quantity"
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:gravity="center_horizontal"
           android:padding="@dimen/standard_padding"
           android:background="@color/login_background"
           android:text="@string/order_quantity" >
        </TextView>
    </LinearLayout>

    <LinearLayout
        android:id="@+id/tab_button2"
        android:layout_width="0dip"
       android:layout_height="wrap_content"
       android:layout_weight="1" >

        <TextView
            android:id="@+id/text_name"
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:gravity="center_horizontal"
           android:padding="@dimen/standard_padding"
           android:background="@color/login_background"
           android:text="@string/order_name" >
        </TextView>
    </LinearLayout>

    <LinearLayout
        android:id="@+id/tab_button3"
        android:layout_width="0dip"
       android:layout_height="wrap_content"
       android:layout_weight="1" >

        <TextView
            android:id="@+id/text_amount"
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:gravity="center_horizontal"
           android:padding="@dimen/standard_padding"
           android:background="@color/login_background"
           android:text="@string/order_price" >
        </TextView>
    </LinearLayout>
</LinearLayout>
```

A.Co_for_J.Co\res\layout\ activity_queue_progress.xml

```
<?xml version="1.0" encoding="utf-8"?>

<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:background="@color/encode_view"
    android:orientation="horizontal" >

        <LinearLayout
        android:id="@+id/tab_buttons"
        android:layout_width="wrap_content"
        android:layout_height="fill_parent"
```

```
     android:layout_gravity="center_vertical"
    android:background="@color/login_background"
    android:gravity="center"
    android:orientation="vertical" >

    <Button
        android:id="@+id/button01"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:layout_marginTop="@dimen/standard_padding"
        android:layout_marginBottom="@dimen/standard_padding"
        android:padding="@dimen/standard_padding"
        android:background="@color/login_background"
        android:text="@string/button_home"
        android:textColor="@color/encode_view" >
    </Button>

    <Button
        android:id="@+id/button02"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:layout_marginTop="@dimen/standard_padding"
        android:layout_marginBottom="@dimen/standard_padding"
        android:padding="@dimen/standard_padding"
        android:background="@color/login_background"
        android:text="@string/button_menu"
        android:textColor="@color/encode_view" >
    </Button>

    <Button
        android:id="@+id/button03"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:layout_marginTop="@dimen/standard_padding"
        android:layout_marginBottom="@dimen/standard_padding"
        android:padding="@dimen/standard_padding"
        android:background="@color/result_points"
        android:text="@string/button_order"
        android:textColor="@color/encode_view" >
    </Button>

    <Button
        android:id="@+id/button04"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:layout_marginTop="@dimen/standard_padding"
        android:layout_marginBottom="@dimen/standard_padding"
        android:padding="@dimen/standard_padding"
        android:background="@color/login_background"
        android:text="@string/button_queue"
        android:textColor="@color/encode_view" >
    </Button>

    <Button
        android:id="@+id/button05"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:layout_marginTop="@dimen/standard_padding"
        android:layout_marginBottom="@dimen/standard_padding"
        android:padding="@dimen/standard_padding"
        android:background="@color/login_background"
        android:text="@string/button_logout"
        android:textColor="@color/encode_view" >
    </Button>
</LinearLayout>

<LinearLayout
    android:id="@+id/body_home"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:orientation="vertical" >

     <TextView
        android:id="@+id/queue_title"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:gravity="center"
       android:padding="@dimen/double_padding"
        android:text="@string/queue_title"
```

```
            android:textColor="@color/font_color"
            android:textSize="40sp"
            android:textStyle="bold" >
        </TextView>

        <TextView
            android:id="@+id/now_serving"
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:gravity="center"
           android:padding="@dimen/double_padding"
            android:text="Customer No. 12"
           android:background="@color/result_points"
            android:textColor="@color/font_color"
            android:textSize="30sp"
            android:textStyle="bold" >
        </TextView>

        <LinearLayout
            android:id="@+id/body_queue"
            android:layout_width="fill_parent"
            android:layout_height="fill_parent"
            android:orientation="horizontal" >


            <TextView
                android:id="@+id/waiting_04"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:gravity="center"
               android:padding="@dimen/double_padding"
                android:text="QUEUE: "
                android:textColor="@color/font_color"
                android:textSize="30sp"
                android:textStyle="bold" >
            </TextView>

            <TextView
                android:id="@+id/waiting_01"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:gravity="center"
               android:padding="@dimen/double_padding"
                android:text="No. 13"
                android:textColor="@color/font_color"
                android:textSize="30sp"
                android:textStyle="bold" >
            </TextView>

            <TextView
                android:id="@+id/waiting_02"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:gravity="center"
               android:padding="@dimen/double_padding"
                android:text="No. 14"
                android:textColor="@color/font_color"
                android:textSize="30sp"
                android:textStyle="bold" >
            </TextView>

            <TextView
                android:id="@+id/waiting_03"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:gravity="center"
               android:padding="@dimen/double_padding"
                android:text="No. 15"
                android:textColor="@color/font_color"
                android:textSize="30sp"
                android:textStyle="bold" >
            </TextView>
        </LinearLayout>
    </LinearLayout>
</LinearLayout>
```

A.Co_for_J.Co\res\layout\ activity_register.xml

```xml
<?xml version="1.0" encoding="utf-8"?>

<ScrollView
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:paddingBottom="@dimen/standard_padding"
    android:paddingLeft="@dimen/standard_padding"
    android:paddingRight="@dimen/standard_padding"
    android:paddingTop="@dimen/standard_padding"
    android:background="@color/login_background"
    android:orientation="vertical" >

    <LinearLayout
        android:layout_width="fill_parent"
     android:layout_height="wrap_content"
        android:orientation="vertical"
        android:padding="@dimen/standard_padding" >

        <!--  First name Label -->
        <TextView
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:text="@string/label_fname" >
         </TextView>
        <!--  First name TextField -->
        <EditText
            android:id="@+id/register_fname"
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:singleLine="true"
              android:hint="@string/edit_fname" >
    </EditText>

        <!--  Last name Label -->
        <TextView
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:text="@string/label_lname" >
         </TextView>
        <!--  Last name TextField -->
        <EditText
            android:id="@+id/register_lname"
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:singleLine="true"
              android:hint="@string/edit_lname" >
    </EditText>

        <!--  Identifier Label -->
        <TextView
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:text="@string/label_identifier" >
         </TextView>
        <!--  Identifier TextField -->
        <EditText
            android:id="@+id/register_identifier"
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:singleLine="true"
              android:hint="@string/edit_identifier" >
    </EditText>

        <!--  Description Label -->
        <TextView
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:text="@string/label_description" >
         </TextView>
        <!--  Description TextField -->
        <EditText
            android:id="@+id/register_description"
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:singleLine="true"
              android:hint="@string/edit_description" >
    </EditText>
```

```
        <!--  Password Label -->
        <TextView
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:text="@string/label_password" >
         </TextView>
        <!--  Password TextField -->
        <EditText
            android:id="@+id/register_password"
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:singleLine="true"
            android:inputType="textPassword"
              android:hint="@string/edit_password" >
    </EditText>

        <!--  Error message -->
        <TextView
            android:id="@+id/register_error"
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:textColor="@color/login_error_message"
            android:padding="@dimen/standard_padding"
            android:textStyle="bold">
         </TextView>

        <!--  Login Button -->
        <Button
            android:id="@+id/button_register"
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:layout_marginTop="@dimen/double_padding"
            android:layout_marginBottom="@dimen/double_padding"
            android:text="@string/button_register" >
         </Button>

        <!--  Link to Login Screen -->
        <Button
            android:id="@+id/button_link_login"
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:layout_marginTop="@dimen/double_padding"
            android:layout_gravity="bottom|center_horizontal"
            android:background="@null"
            android:text="@string/button_link_login"
            android:textColor="@color/login_register_text"
            android:textStyle="bold" >
         </Button>
    </LinearLayout>

</ScrollView>
```

A.Co_for_J.Co\res\layout\ activity_welcome.xml

```
<?xml version="1.0" encoding="utf-8"?>

<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:background="@color/encode_view"
    android:orientation="vertical" >

    <ImageView
        android:id="@+id/banner"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:src="@drawable/banner" >
    </ImageView>

    <LinearLayout
        android:layout_width="fill_parent"
        android:layout_height="fill_parent"
        android:layout_weight="1"
        android:background="@color/encode_view"
```

```
        android:orientation="horizontal" >

          <Button
               android:id="@+id/button_to_go"
               android:layout_weight="1"
               android:layout_width="wrap_content"
               android:layout_height="wrap_content"
               android:background="@drawable/togo_customer">
          </Button>

          <Button
               android:id="@+id/button_for_here"
               android:layout_weight="1"
               android:layout_width="wrap_content"
               android:layout_height="wrap_content"
               android:background="@drawable/forhere_customer">
          </Button>
     </LinearLayout>

     <!-- Error message -->
     <TextView
          android:id="@+id/welcome_error"
          android:layout_width="fill_parent"
          android:layout_height="wrap_content"
          android:layout_gravity="bottom|center_horizontal"
          android:padding="@dimen/standard_padding"
          android:textColor="@color/login_error_message"
          android:textStyle="bold" >
     </TextView>
</LinearLayout>
```

A.Co_for_J.Co\res\menu\capture.xml

```
<?xml version="1.0" encoding="UTF-8"?>

<menu xmlns:android="http://schemas.android.com/apk/res/android" >

     <item
        android:id="@+id/menu_history_clear"
        android:icon="@android:drawable/ic_menu_delete"
        android:orderInCategory="1"
        android:showAsAction="withText|ifRoom"
        android:title="@string/history_clear">
     </item>

</menu>
```

A.Co_for_J.Co\res\menu\history.xml

```
<?xml version="1.0" encoding="UTF-8"?>

<menu
     xmlns:android="http://schemas.android.com/apk/res/android">

     <item
          android:id="@+id/menu_history"
          android:title="@string/menu_history"
          android:icon="@android:drawable/ic_menu_recent_history"
          android:orderInCategory="1"
          android:showAsAction="withText|ifRoom"/>
</menu>
```

A.Co_for_J.Co\res\values\colors.xml

```
<?xml version="1.0" encoding="utf-8"?>

<resources>
     <color name="transparent">#00000000</color>
```

```
    <color name="font_color">#6A200D</color>

    <color name="contents_text">#ff000000</color>

    <color name="encode_view">#ffffffff</color>

    <color name="possible_result_points">#c0FCa200</color>

    <color name="login_background">#3b3b3b</color>
    <color name="login_error_message">#e30000</color>
    <color name="login_register_text">#21dbd4</color>

    <color name="result_image_border">#ff4c3100</color>
    <color name="result_minor_text">#ffc0c0c0</color>
    <color name="result_points">#c0FCa200</color>
    <color name="result_text">#ffffffff</color>
    <color name="result_view">#b0000000</color>

    <color name="status_text">#ffffffff</color>

    <color name="viewfinder_frame">#ff4c3100</color>
    <color name="viewfinder_laser">#ffFCa200</color>
    <color name="viewfinder_mask">#60FFa500</color>

</resources>
```

A.Co_for_J.Co\res\values\dimens.xml

```
<?xml version="1.0" encoding="UTF-8"?>

<resources>

    <dimen name="double_padding">16dip</dimen>
    <dimen name="standard_padding">8dip</dimen>
    <dimen name="half_padding">4dip</dimen>

</resources>
```

A.Co_for_J.Co\res\values\ids.xml

```
<?xml version="1.0" encoding="utf-8"?>

<resources>

    <!-- Messages IDs -->
    <item type="id" name="auto_focus"/>
    <item type="id" name="decode"/>
    <item type="id" name="decode_failed"/>
    <item type="id" name="decode_succeeded"/>
    <item type="id" name="quit"/>
    <item type="id" name="restart_preview"/>
    <item type="id" name="return_scan_result"/>

</resources>
```

A.Co_for_J.Co\res\values\strings.xml

```
<?xml version="1.0" encoding="utf-8"?>

<resources>

    <string name="app_name">A.Co for J.Co :D</string>
    <string name="for_customer">A.Co: Customer</string>
    <string name="for_entrance">A.Co: Entrance</string>
    <string name="for_counter">A.Co: Counter</string>
    <string name="capture_name">A.Co: Capture</string>
    <string name="login_name">A.Co for J.Co</string>

    <string name="history_title">History</string>
    <string name="login_title">Login User</string>
```

```
<string name="register_title">Register User</string>
<string name="home_title">A.CO for J.CO</string>
<string name="menu_title">MENU LIST</string>
<string name="queue_title">NOW SERVING...</string>
<string name="food_title">YES?</string>
<string name="customer_order_title">YOUR ORDER</string>

<string name="order_quantity">QUANTITY</string>
<string name="order_name">ITEM NAME</string>
<string name="order_price">PRICE</string>

<string name="button_cancel">Cancel</string>
<string name="button_done">Done</string>
<string name="button_login">Log in</string>
<string name="button_ok">OK</string>
<string name="button_register">Register</string>
<string name="button_result">Order Now</string>
<string name="button_link_register">I don\'t have an account. Register Me!</string>
<string name="button_link_login">I have already registered. Log in Me!</string>
<string name="button_link_logout">Log out Me!</string>
<string name="button_order_now">Let me in!</string>
<string name="button_home">HOME</string>
<string name="button_menu">MENU</string>
<string name="button_order">ORDER</string>
<string name="button_queue">QUEUE</string>
<string name="button_logout">LEAVE</string>
<string name="button_confirm">CONFIRM ORDER</string>
<string name="button_endsession">END SESSION</string>
<string name="button_add">ADD</string>
<string name="button_got_it">GOT IT</string>
<string name="button_plus"> + </string>
<string name="button_zero"> 0 </string>
<string name="button_minus"> - </string>


<string name="history_clear">Clear history</string>
<string name="history_clear_one_history">Clear</string>
<string name="history_empty">Empty</string>
<string name="history_empty_detail">No barcode scans have been recorded</string>

<string name="menu_history">History</string>

<string name="msg_default_format">Format</string>
<string name="msg_default_meta">Metadata</string>
<string name="msg_default_status">Place a QR code inside the viewfinder rectangle to scan it.</string>
<string name="msg_default_time">Time</string>
<string name="msg_default_type">Type</string>
<string name="msg_sure">Are you sure?</string>

<string name="result_address_book">Found contact info</string>
<string name="result_text">Found plain text</string>

<string name="barcode_image">QR code Image</string>

<string name="label_queues">Customer Number:</string>
<string name="label_times">Time of Arrival:</string>
<string name="label_urls">URL Redirection:</string>
<string name="label_identifier">Identifier:</string>
<string name="label_password">Password:</string>
<string name="label_description">Position:</string>
<string name="label_fname">First Name:</string>
<string name="label_lname">Last Name:</string>
<string name="label_discount">Discount:</string>
<string name="label_taxrate">Taxrate:</string>
<string name="label_total">Total:</string>
<string name="label_fullname">Fullname:</string>
<string name="label_verifier">Verifier:</string>

<string name="edit_queues">Customer No. XXX…</string>
<string name="edit_times">yyyy-MM-dd HH:mm:ss.SSS…</string>
<string name="edit_urls">http://agila.upm.edu.ph/~bcfestejo/acoforjco_api…</string>
<string name="edit_identifier">Identification Number…</string>
<string name="edit_password">Password…</string>
<string name="edit_description">Position…</string>
<string name="edit_fname">First name…</string>
<string name="edit_lname">Last name…</string>
<string name="edit_quantity">quantity…</string>
<string name="edit_fullname">Enter your fullname…</string>
```

```
    <string name="edit_verifier">Enter as much physical description as you can like what you are wearing, or what you are carrying for the staff to verify your identity upon your claiming your order…</string>

    <string name="connected_welcome">Welcome On-the-go Customer!</string>

</resources>
```

A.Co_for_J.Co\res\values\themes.xml

```
<?xml version="1.0" encoding="UTF-8"?>

<resources>
    <style name="CaptureTheme" parent="android:Theme">
        <item name="android:windowFullscreen">true</item>
        <item name="android:windowActionBar">true</item>
        <item name="android:actionBarStyle">@style/CaptureActionBar</item>
        <item name="android:windowActionBarOverlay">true</item>
        <item name="android:windowActionModeOverlay">true</item>
    </style>

    <style name="EncoderTheme" parent="android:Theme">
        <item name="android:windowActionBar">true</item>
        <item name="android:actionBarStyle">@style/MyActionBar</item>
    </style>

    <style name="HistoryTheme" parent="android:Theme">
        <item name="android:windowFullscreen">true</item>
        <item name="android:windowActionBar">true</item>
        <item name="android:actionBarStyle">@style/MyActionBar</item>
    </style>

    <!-- ActionBar styles -->
    <style name="CaptureActionBar" parent="android:Widget.ActionBar">
        <item name="android:background">#60FFa500</item>
    </style>

    <style name="MyActionBar" parent="android:Widget.ActionBar">
        <item name="android:background">#c0FCa200</item>
    </style>

</resources>
```

A.Co_for_J.Co\AndroidManifest.xml

```
<?xml version="1.0" encoding="utf-8"?>

<manifest
    xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.specialproblem.acoforjco"
    android:versionCode="1"
    android:versionName="1.0" >

    <uses-sdk
        android:minSdkVersion="8"
        android:targetSdkVersion="19" />

    <uses-feature
        android:name="android.hardware.camera" />
    <uses-feature
        android:name="android.hardware.camera.autofocus"
        android:required="false" />
    <uses-feature
        android:name="android.hardware.touchscreen"
        android:required="false" />
    <uses-feature
        android:name="android.hardware.wifi"
        android:required="false" />

    <uses-permission
        android:name="android.permission.CAMERA" />
    <uses-permission
        android:name="android.permission.INTERNET" />
    <uses-permission
```

```
    android:name="android.permission.ACCESS_NETWORK_STATE" />
<uses-permission
    android:name="android.permission.READ_PHONE_STATE" />

<application
    android:allowBackup="true"
    android:icon="@drawable/icon"
    android:label="@string/app_name" >

    <activity
        android:name="com.specialproblem.acoforjco.qrcode.ConnectedUser"
        android:clearTaskOnLaunch="true"
        android:configChanges="orientation|keyboardHidden"
        android:icon="@drawable/icon"
        android:label="@string/for_customer"
        android:screenOrientation="Landscape"
        android:stateNotNeeded="true"
        android:theme="@style/HistoryTheme"
        android:windowSoftInputMode="stateAlwaysHidden" >

        <intent-filter>
            <action
                android:name="android.intent.action.MAIN" />
            <category
                android:name="android.intent.category.LAUNCHER" />
        </intent-filter>
    </activity>

    <activity
        android:name="com.specialproblem.acoforjco.WelcomeActivity"
        android:clearTaskOnLaunch="true"
        android:configChanges="orientation|keyboardHidden"
        android:icon="@drawable/icon"
        android:label="@string/for_entrance"
        android:screenOrientation="Landscape"
        android:stateNotNeeded="true"
        android:theme="@style/HistoryTheme"
        android:windowSoftInputMode="stateAlwaysHidden" >

        <intent-filter>
            <action
                android:name="android.intent.action.MAIN" />
            <category
                android:name="android.intent.category.LAUNCHER" />
        </intent-filter>
    </activity>

    <activity
        android:name="com.specialproblem.acoforjco.ForHereActivity"
        android:label="@string/app_name"
        android:screenOrientation="Landscape"
        android:stateNotNeeded="true"
        android:theme="@style/HistoryTheme">
    </activity>

    <activity
        android:name="com.specialproblem.acoforjco.qrcode.CaptureActivity"
        android:configChanges="orientation|keyboardHidden"
        android:icon="@drawable/icon"
        android:label="@string/capture_name"
        android:screenOrientation="Landscape"
        android:stateNotNeeded="true"
        android:theme="@style/CaptureTheme"
        android:windowSoftInputMode="stateAlwaysHidden" >
    </activity>

    <activity
        android:name="com.specialproblem.acoforjco.qrcode.EncoderActivity"
        android:clearTaskOnLaunch="true"
        android:configChanges="orientation|keyboardHidden"
        android:icon="@drawable/icon"
        android:label="@string/app_name"
        android:screenOrientation="Landscape"
        android:stateNotNeeded="true"
        android:theme="@style/EncoderTheme"
        android:windowSoftInputMode="stateAlwaysHidden" >
    </activity>

    <activity
```

```
    android:name="com.specialproblem.acoforjco.qrcode.EncodeDisplayActivity"
    android:label="@string/app_name"
    android:theme="@style/HistoryTheme"
    android:parentActivityName="com.specialproblem.acoforjco.WelcomeActivity"
    android:screenOrientation="landscape" >

    <meta-data
        android:name="android.support.PARENT_ACTIVITY"
        android:value="com.specialproblem.acoforjco.WelcomeActivity" />
</activity>

<activity
    android:name="com.specialproblem.acoforjco.qrcode.history.HistoryActivity"
    android:label="@string/history_title"
    android:screenOrientation="landscape"
    android:stateNotNeeded="true"
    android:theme="@style/HistoryTheme" >
</activity>

<activity
    android:name="com.specialproblem.acoforjco.DashboardActivity"
    android:clearTaskOnLaunch="true"
    android:configChanges="orientation|keyboardHidden"
    android:icon="@drawable/icon"
    android:label="@string/login_name"
    android:screenOrientation="landscape"
    android:stateNotNeeded="true"
    android:theme="@style/HistoryTheme"
    android:windowSoftInputMode="stateAlwaysHidden" >
</activity>

<activity
    android:name="com.specialproblem.acoforjco.LoginActivity"
    android:clearTaskOnLaunch="true"
    android:configChanges="orientation|keyboardHidden"
    android:icon="@drawable/icon"
    android:label="@string/for_counter"
    android:screenOrientation="landscape"
    android:stateNotNeeded="true"
    android:theme="@style/HistoryTheme"
    android:windowSoftInputMode="stateAlwaysHidden">

    <intent-filter>
        <action
            android:name="android.intent.action.MAIN" />
        <category
            android:name="android.intent.category.LAUNCHER" />
     </intent-filter>
</activity>

<activity
    android:name="com.specialproblem.acoforjco.RegisterActivity"
    android:label="@string/register_title"
    android:screenOrientation="landscape"
    android:stateNotNeeded="true"
    android:theme="@style/HistoryTheme">
</activity>

<activity
    android:name="com.specialproblem.acoforjco.menulist.HomeActivity"
    android:clearTaskOnLaunch="true"
    android:configChanges="orientation|keyboardHidden"
    android:icon="@drawable/icon"
    android:label="@string/app_name"
    android:screenOrientation="landscape"
    android:stateNotNeeded="true"
    android:theme="@style/HistoryTheme"
    android:windowSoftInputMode="stateAlwaysHidden" >
</activity>

<activity
    android:name="com.specialproblem.acoforjco.menulist.MenuActivity"
    android:label="@string/login_name"
    android:screenOrientation="landscape"
    android:stateNotNeeded="true"
    android:theme="@style/HistoryTheme">
</activity>

<activity
```

```
            android:name="com.specialproblem.acoforjco.menulist.QueueProgressActivity"
            android:label="@string/login_name"
            android:screenOrientation="landscape"
            android:stateNotNeeded="true"
            android:theme="@style/HistoryTheme">
        </activity>

        <activity
            android:name="com.specialproblem.acoforjco.menulist.CustomerOrderActivity"
            android:label="@string/login_name"
            android:screenOrientation="landscape"
            android:stateNotNeeded="true"
            android:theme="@style/HistoryTheme">
        </activity>

        <activity
            android:name="com.specialproblem.acoforjco.menulist.ConfirmedOrder"
            android:label="@string/login_name"
            android:screenOrientation="landscape"
            android:stateNotNeeded="true"
            android:theme="@style/HistoryTheme">
        </activity>

        <activity
            android:name="com.specialproblem.acoforjco.menulist.FoodDetails"
            android:label="@string/login_name"
            android:screenOrientation="landscape"
            android:stateNotNeeded="true"
            android:theme="@style/HistoryTheme">
        </activity>

        <activity
            android:name="com.specialproblem.acoforjco.menulist.Donuts"
            android:label="@string/login_name"
            android:screenOrientation="landscape"
            android:stateNotNeeded="true"
            android:theme="@style/HistoryTheme">
        </activity>

        <activity
            android:name="com.specialproblem.acoforjco.menulist.Coffee"
            android:label="@string/login_name"
            android:screenOrientation="landscape"
            android:stateNotNeeded="true"
            android:theme="@style/HistoryTheme">
        </activity>

        <activity
            android:name="com.specialproblem.acoforjco.menulist.Club"
            android:label="@string/login_name"
            android:screenOrientation="landscape"
            android:stateNotNeeded="true"
            android:theme="@style/HistoryTheme">
        </activity>

        <activity
            android:name="com.specialproblem.acoforjco.menulist.Pops"
            android:label="@string/login_name"
            android:screenOrientation="landscape"
            android:stateNotNeeded="true"
            android:theme="@style/HistoryTheme">
        </activity>

        <activity
            android:name="com.specialproblem.acoforjco.menulist.Cool"
            android:label="@string/login_name"
            android:screenOrientation="landscape"
            android:stateNotNeeded="true"
            android:theme="@style/HistoryTheme">
        </activity>
    </application>
</manifest>
```

XI. Acknowledgement

I would like to take this opportunity to express my profound gratitude and deep regards to everyone who has been a part of my wonderful life as a college student.

Firstly, I would like to thank my family who has always been there for me and who has simply supported me on all of my endeavors. My Mama and Papa who had worked so hard just to be able to raise me as the person I am today.

Second, all my blockmates and close friends who have basically shared with me the most wonderful years of my life. My lovely Lalay and Renzy, who has always been there to simply give me that gentle push whenever I stumble; to my dear Chelsea and Lara, who has shared my wackiness even from afar; to my cool Gilbert, who has always shared a good laugh with me as we stay up late at nights that we both had to study; and to Ate Leonie, who has always been in her office to welcome me and all the goofiness, and all the stress, and all mood swings that I carry.

Third, to all my professors who were simply there to hone all of us to become not only a great computer scientist, but also a responsible and mature individual. Specifically my old adviser Sir Richard Bryann Chua, who has always been there to give me constructive criticisms. Thank you so much Sir. And also, my very patient adviser Sir Aldrich Colin Co, the one and only—the inspiration of this project's name—for his exemplary guidance, monitoring and constant encouragement throughout the course of this thesis. The blessing, help and guidance given by him time to time will indeed carry me a long way in the journey of life on which I am about to embark.

Fourth, to my most ridiculously insistent bugger of a lover, my lovely Michael, who has always been there to give me that gentle kick to make me move, that warm hug to offer me comfort, and that unconditional love to make the rest of my stay here in this world a more vibrant and colorful one.

Fifth, to Almighty Google, who has consistently stirred me to the right search result and helped me to understand all the complexities of this programming world.

And last, but definitely not the least, to the Higher Being that I believe in, Thank You.