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

Trends.Net Remote Laboratory Access 2.0

A special problem in partial fulfillment
of the requirements for the degree of
Bachelor of Science in Computer Science

Submitted by:

Angelo A. Rosal

June 2015

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 "Trends.Net Remote Laboratory Access 2.0" prepared and submitted by Angelo A. Rosal in partial fulfillment of the requirements for the degree of Bachelor of Science in Computer Science has been examined and is recommended for acceptance.

_____

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

EXAMINERS:

	Approved	**Disapproved**
1. Richard Bryann L. Chua, M.Sc.	_____	_____
2. Avegail D. Carpio, M.Sc.	_____	_____
3. Perlita E. Gasmen, M.Sc. (*candidate*)	_____	_____
4. Ma. Sheila A. Magboo, M.Sc.	_____	_____
5. Vincent Peter C. Magboo, M.D., M.Sc.	_____	_____
6. 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

Trends.Net, Inc. Education Center (TNET) is a training provider for Information Technology and Management Professionals. TNET offers networking certification trainings that can help associates and professionals prepare for a career with any company that requires each certification. Contrarily, there are cases where a company sends students to expand their area of expertise or aim for a higher level certification. Most of the trainings are composed of two parts: a lecture and a laboratory class which usually lasts for a week. The laboratory classes are subdivided into two types: local and remote. The local laboratory is accessed within the internal network of the company while the remote laboratory is accessed through an external network (i.e. the Internet). The remote laboratory is either a Virtual Private Network (VPN) or a public Internet Protocol (IP) address mapped to a private IP address using Network Address Translation (NAT) depending on whether the packets has to pass through an external unsecured network or not.

Keywords: Remote Access, Scalable Vector Graphics, Document Object Model, Distance Learning

Contents

List of Figures

List of Tables

I. Introduction

A. Background of the Study

Trends.Net, Inc. Education Center (TNET) is an independent affiliate of Trends & Technologies Holdings, Inc. (TTHI) [1]. TNET is a training provider for Information Technology and Management Professionals. They offer networking certification trainings that can help associates and professionals prepare for a career with any company that requires each certification [2]. Contrarily, there are cases where a client company enrols their employees to expand their area of expertise or to aim for a higher level certification (e.g. CCNA to CCNP). The curriculum for each training is determined by the company that creates the certification exam for that training (e.g. Cisco, Microsoft, etc.). TNET uses the provided curriculum to conduct trainings. TNET also serves as an exam center for the students. They conduct mock exams and deliver the actual online certification exam created by the Cisco and other companies to the students.

Most of the trainings are composed of two parts: a lecture and a laboratory class which usually lasts for a week. The laboratory classes are subdivided into two types: local and remote. The local laboratory is accessed within the internal network of the company while the remote laboratory is accessed through an external network (i.e. the Internet) [3]. Each training has a kit which includes a book module containing the theoretical concepts of the course and laboratory questions among others.

Each course uses a unique physical arrangement, which shall be henceforth referred to as the *main topology* or *physical connection*, of network nodes and media, referred to as *pods*, such as switches, routers, servers, and sometimes firewalls, within an enterprise network structure. Each main topology is represented virtually through a *logical connection* which will be generated by the system. The lecture is conducted by a professor hired by the company. After each lecture, the professor prepares a laboratory activity which is either taken from their respective

curriculum providers (e.g. Cisco, Microsoft, etc.) or, in some cases, made by the professor himself/herself as requested by the company the students represent [3].

Trends.Net Remote Laboratory Access 1.0 was an internship project of a Dela Salle University student. Given the limited timeframe, he was unable to write a documentation of the program so there is nothing to go by except for the source code itself.

The system only applies to two (2) Cisco courses: ICND1 and ICND2. In each laboratory activity, the pod in question is the only one displayed. So to speak, the students do not see the entire Physical Connection and is only limited to the devices within the pod s/he is assigned to.

The logical topology in this version is hardcoded. It follows the same implementation as discussed in this version except for the build logical topology feature. Everytime a professor account is created, it creates a directory and fills it up with its own copy of the activities making it a bit messy as it keeps multiple copies of the same activity files in each account. It uses command line arguments saved as a windows batch files and executed via php. It does not provide a student with an account as will be implemented with version 2.0.

B. Statement of the Problem

The existing system is working but is only limited to two Cisco courses. As previously stated, TNET offers trainings which are composed of a hierarchy of courses. Cisco, for example, has seven courses for the associate training level and eight courses for the professional training level [1] each of which has its own main topology while each device, depending on its type, has its own set of Internet Protocol (IP) address, username, password, and host name which are needed to access it.

The current system uses an HTML image map generated using Microsoft Office Visio to represent each course's logical connection thereby preventing any changes to it and the configuration of the pods within it without generating a new image

map which requires editing the system's code. Some of the exam providers have implemented a new curriculum changing the structure of the topologies hence, the current system is unable to cope up [3].

C. Objectives of the Study

The aim of the project is to recreate the system which should be flexible enough to enable the administrator(s) to edit the logical connections as they see fit and the professors to define the access level of their students.

The system will have three user types:

1. The administrator, who is an employee at TNET, should have the privilege to:
   (a) Add, edit, and delete user accounts with no active classes
   (b) Add, edit, and delete devices
   (c) Add, edit, and delete courses
   (d) Add, edit, and delete the logical connection per course
   (e) Set the maximum number of students per pod
2. The professor, who is hired by the company, should have the privilege to:
   (a) Add, edit, and delete classes of a course which includes:
      i. Setting the class password
      ii. Setting the IP address, username, and password of the class' terminal server
      iii. Specifying the number of students in his/her class
      iv. Set class' expiration date
   (b) Add, edit, and delete pods within their classes which includes:
      i. Specifying the pod number
      ii. Setting each pod's password

iii. Selecting the devices included per pod

3. The students should have the privilege to:

   (a) Access their class by specifying the name of the professor and the class/-course where they belong

   (b) Select the pod they will use

   (c) Perform the laboratory activities using the pod they selected

D. Significance of the Project

The logical and physical connection does not follow a one-to-one correspondence. So to speak, the logical connection may vary from one student to another during laboratory activities depending on the pod assigned to them while maintaining a single logical connection. The idea, then, is to make the logical connections visible through implementing a Graphical User Interface (GUI) based model.

The new system will no longer be based on a MS Visio template as compared to what is currently implemented. Instead, the logical connection will be built inside the system making it more flexible and enabling it to accommodate additional courses and curricular changes.

E. Scope and Limitations

1. A Logical Connection is consisted of at least one (1) Terminal Server.

2. Non-accessible devices in the Logical Connection will be rendered in grayscale.

3. The expiration date of a class detemines whether the user account of the professor holding that class may already be deleted. It also automatically deletes the class once the expiration date is reached.

4. A student will not have a *de facto* user account and his/her login credentials will be the name of the professor, the course and class s/he belongs to, and the class password.

5. The distribution of the pod and class passwords to the students will not be addressed by the system.

6. Should the student lose his/her password, the system will not provide a means to recover it, as the student will not have a user account, and shall be resolved by the student and professor themselves.

7. The activities' questions will no longer be included in the system as they are already written in the laboratory manual.

8. The domain of system shall not include how the students interact with the devices and shall be, as it has always been, through command line arguments.

9. The students must use Internet Explorer as their browser as it is the only browser to support ActiveXObject.

F. Assumptions

1. The devices are already connected physically and are already configured before starting a class.

2. The system shall not bear the security risks of giving access to the students and shall be addressed by the company itself.

3. Any new device that will be added may be accessed using either Microsoft Windows Remote Desktop Protocol or PuTTy Secure Shell Protocol, or not at all (e.g. Serial Cable).

4. A non-portable version of PuTTy is already installed in the client side under Program Files.

5. Internet Explorer is already configured to allow ActiveXObject scripting.

II. Review of Related Literature

A remote access lab can be defined as a laboratory that uses a communications network, where users and lab computers are geographically separated and telecommunications technologies are used for accessing to these devices. The Mitsubishi RV-2AJ Robot is an industrial manipulator with 5 degrees of freedom and with an anthropomorphic articulation that offers a load capacity of up to 2 kg. The robot has a reach of 410 mm and combines a maximum speed of 2.100 mm/s with a repeatability of $\pm$ 0.02 mm. This robot is equipped with Mitsubishi CR1-571 controller, which is the interface to the system server. The Mitsubishi MELFA CR1-571 controller has a 64-bit DSP/RISC microprocessor, which allows the execution of 32 programs simultaneously in multitask mode. This robot has been used in the University of Quindio at Armenia, along with a middleware that is based on standard Java Servlets, to build a tool that supports remote access for experimenting with the Mitsubishi RV-2AJ manipulator robot [4].

GigaBOT is a Web Lab for mobile robotics education. It operates two ActivMedia's Pioneer P3-DX robots with sixteen sonars and protection bumpers. One robot is fitted with on-board processor, wireless network interface (802.11g), and a Canon VCC4 on-board camera with PTZ (pan/tilt/zoom) and connected to frame grabber. The second robot does not have on-board processor, being controlled via serial port by an HP IPaq H5555 handheld with 802.11b wireless network. Six experiment classes were developed for the Web Lab: Basic Telemetry, Navigation on Structured Environments, Navigation on Non Structured Environments, Vision-based Navigation, Code Submission, and Cooperative Robotics. Since this experiment runs on the client side, the network bandwidth and delay strongly impacts on the performance of the experiment. In order to assess the bandwidth and delay demanded by the Web Lab, the vision-based navigation experiment was performed from four different access networks: local area, campus, residential ADSL (Asymmetric Digital Subscriber Line), and virtual private networks [5].

Virtual Engineering Laboratory (VE-LAB) is Web-based remote control to

the fuzzy efficiency optimization of the hybrid electric vehicle (HEV) starter/alternator. The Web enabled motor experimentation platform provides fully digital control of an 8-kW induction motor starter/alternator and a programmable load. An outer loop fuzzy logic controller is used to automatically tune the starter/alternator for optimum efficiency given a user specified operating point. Induction machines are normally operated at rated flux to achieve their best dynamic response. The fuzzy logic controller is used to automatically determine the proper flux level for optimum efficiency operation under varying load and temperature [6].

A platform for testing electric motors over the Internet, which enables users to test multiple motors remotely, has been developed by Tedesco and Emami. The system is divided into three principle subsystems namely Motor Test Platform for positioning and coupling the test motors to the Load Module, Target Software Application for executing and controlling the test operations, and Server Software Application for enabling remote access and maintaining user accounts. The user gains access to the experimental setup controls by logging onto the experiment via a website. S/he is presented with the Testing GUI. The Testing GUI accesses the motor testing Dynamic Linked Library (DLL), which contains the classes and methods for sending commands to the test platform via the PCI-2517 DAQ board from Measurement Computing Corp [7].

With the availability of high performance networks and the increase in the number of online scientific instruments, remote instrumentation is a topic with much popularity lately, promising better instrument utilization, easier collaboration between distant organizations and diminution of travel-related costs and overhead [8].

Tokamaks (TOroidal'naya KAMera s AKsial'nym magnitnym polem, *toroidal chamber with an axial magnetic field* [9]) are equipped with many diagnostics and control devices. For example, the DIII-D National Fusion Facility located at San Diego, California has more than 50 diagnostics devices. Tokamak diagnostic set-

tings are repeatedly modified to meet the changing needs of each experiment. Enabling the remote diagnostic control has significant challenges due to security and efficiency requirements. The Operation Request Gatekeeper (ORG) is a software system that addresses the challenges of remotely but securely submitting modification requests. A prototype ORG was developed for the ITER CODAC that satisfies their initial requirements for remote request submission and has been tested with remote control of the KSTAR Plasma Control System [10].

The National Ultrahigh-Field NMR Facility for Solids houses a state-of-the-art 900 MHz Nuclear Magnetic Resonance (NMR) instrument, among several other smaller NMR instruments. It is located in Ottawa, and funded by the Canada Foundation for Innovation (CFI), the Natural Sciences and Engineering Research Council of Canada (NSERC) and the National Research Council Canada (NRC), the facility provides access to leading-edge NMR instruments for the Canadian research community. As a sub-project of the NRC Grid Infrastructure Project, SpectroGrid (formerly named NMRConnect) was implemented and released for production use in 2004. SpectroGrid allows a researcher to remotely access NMR instruments located at NRC in Ottawa. The system provides a Remote Display System (RDS) by creating a Virtual Network Computing (VNC) sessions after a successful grid certificate authentication [8].

The European X-ray Free Electron Laser (European XFEL) is an international project with 12 participating countries that is located in Hamburg, Germany at the site of the DESY research center. In XFEL electrons will be accelerated to energies of up to 17.5 GeV by a 2.1 km long superconducting linear accelerator. At the end of the linear accelerator the electrons are forced into curved trajectories resulting in the emission of x-ray radiation. The control system for the XFEL is designed as a distributed system with network nodes. The embedded controllers are running Distributed Object Oriented Control System (DOOCS) servers and Java DOOCS Data Display (JDDD) terminals for control [11].

Distance learning is an emerging paradigm where students, teachers, and equip-

ment may be at different geographic locations. Practical experience is a very important part of control engineering education, but it is resource intensive. Designing and constructing state-of-the-art control experiments can take time, money, and energy. Sharing experiments locally and remotely allows unique laboratory equipment to be utilized more fully, brings down the experiment cost per student, and makes more experiments available to students [12].

Gurkan et. al. of the IEEE developed a remote laboratory implementation for optical circuit courses. The Introduction to Optical Communications of the University of Colorado and the Optical Circuits course of the University of Houston are subjected to the use of the system. The students are given the theoretical background on concepts in class and then are asked to perform pre-laboratory activities, such as orientation video and simulation, and laboratory procedures online. The simulations are prepared using the VPI software (manufactured by VPI systems, Holmdel, NJ) for optical systems. Orientation video has the recordings of the instructions for the simulation software and the remote access mechanism outlined by the instructor. Student connects to the experiment using the Web-based client that connects to the LabView Web server. The student will have access directly to the virtual experiment and will control the remote experiment. Students have multiple parameters to configure the experiment and collect data and record the results [13].

The delivery of laboratory exercises to students that are unable to attend in person due to physical disabilities is a significant issue. iNetSim is an accessible network simulator, created to allow both vision-impaired and sighted users to complete CCNA 2 laboratory sessions without access to the networking hardware. The system is capable of representing several generic network devices including routers, switches, hubs and PCs [14].

Traditionally, Remote Access Laboratory (RAL) have been defined as methods to provide off site access to remotely controlled hardware, such as robots or laboratories, thereby deeming RALs based learning activities to be commonplace

in engineering faculties. At the University of Southern Queensland, the RAL technology is used for learning activities in the Department of Nursing and Midwifery. In a typical clinical situation, nurses are given an intravenous (IV) fluid order form as part of a patient's medication chart. In the RAL environment, the nurse must configure the IV infusion pump with correct rates and volumes of fluids to be administered including standard patient and infusion site checks [15].

III. Theoretical Framework

As discussed in chapter I, TNET provides trainings which are composed of a laboratory class which is subdivided into two categories: local and remote. The remote laboratory is either a Virtual Private Network (VPN) or a public Internet Protocol (IP) address mapped to a private IP address using Network Address Translation (NAT) [3] depending on whether the packets have to pass through an external unsecured network or not [16]. This bases on the fact that TNET uses the concept of private and public networks to manage their laboratory classes.

A. Network Architecture

A *public network*, like the public telephone system and the Internet, is a large collection of unrelated peers that exchange information more or less freely with each other. The people with access to the public network may or may not have anything in common, and any given person on that network may only communicate with a small fraction of his potential users [17].

A *private network* is composed of computers owned by a single organization that share information specifically with each other. They're assured that they are going to be the only ones using the network, and that information sent between them will (at worst) only be seen by others in the group [17]. A private network may be classified into two types:

1. *Intranets* are essentially when the company treats its internal network infrastructure as though it was its very own World Wide Web [16]. Intranets serve the employees at the corporation site, but not employees on the road or telecommunicating from home. To accommodate the remote access needs of "road warriors" and telecommuters, companies have set up remote access servers to extend intranets over the field [18].

2. *Extranets* are simply an extension of the intranet concept and a subset of the Internet. So to speak, it involves the collaboration of the organizations

or companies that share a common interest and are willing to connect each of their intranets into a miniature isolated version of the global Internet in order to share their resources (within arbitrarily specified limits set by their respective systems administrators) [18].

With the increasing number of systems connecting to the Internet, it has become extremely difficult for organizations to get their own class C or B block of IP addresses. There are simply not enough IPv4 address ranges for every network-connected device on the planet to have its own unique IP address. This has created the need for companies and organizations to allow multiple computers on the Internet network to access the Internet with a single IP address [19].

VPN. Protection of private corporate information is of utmost importance when designing an information structure. However, the separate private networking solutions are expensive and cannot be updated to quickly adapt to changes in business requirements. Internet, on the other hand, is inexpensive but does not, by itself, ensure privacy [18].

A *Virtual Private Network* (VPN) is a "virtual" network that is kept private by "tunneling" private data through the underlying infrastructure of the public Internet [16]. A VPN allows the provisioning of private network services (intranet) for an organization or organizations over a public or shared infrastructure (extranet) such as the Internet or service provider backbone network [20]. VPNs use obfuscation through secure tunneling, rather than physical separation, to keep communications private [16].

Tunneling is defined as the encapsulation of a certain data packet (the original, of inner packet) into another data packet (the encapsulating, or outer packet) so that the inner packet is opaque to the network over which the outer packet is routed [18]. Before the encapsulation takes place, the packets are encrypted so that the data is unreadable to anyone monitoring the network [16].

NAT stands for *Network Address Translation*. NAT was originally designed to address problems being created by the exponential growth of the Internet and the

inevitable shortage of public IP address that would eventually follow. In March 1994, the organization responsible for the public access scheme of the Internet, the Internet Assigned Numbers Authority (IANA), had an idea about conserving the unique IP addressing space on the Internet (public space) by setting aside (reserving) a large set of addresses to be used to be used in corporate private networks (private space). These addresses can be used arbitrarily and at the complete discretion of those corporations. To make this solution viable, the IANA made sure that this pool of reserved addresses would not be routable on the public Internet [16]. NAT was created to allow many organizations to use so-called "non-routable" private IP addresses (the ranges 10.0.0.0/8, 172.[16-31].0.0/12, and 192.168.[1-254].0/24) in the organizations internal network [19].

NAT is a process whereby any internal private addresses (such as those defined in RFC 1597) could be mapped or "translated" by an intermediate device (usually a router) to an external assigned public address. The intention was that, regardless of the addressing scheme used internally by a given organization, it would not conflict with any of the publicly assigned IP addresses on the Internet [16].

B. System

The system, on the other hand, will provide a logical connection for each course which remains constant until such time that the curriculum providers deemed it necessary to change them. The logical connection which serves as a template for each course will be built and saved by the Administrator and will be loaded during a class. So to speak, two classes may have the same logical connection but will have a different physical connection. These two are necessary for a class to actually work. The logical connection serves as a middleware between the student and the physical connection. Without the physical connection, the logical connection will have nothing to connect to.

During a class, the students will be grouped into sets. The number of students per set varies from one course to another. Each set of students will have an explicit

pod to work on. A set of students will see the whole main topology but their access privileges will be determined by the professor. So to speak, they will see all the devices but can only access the devices included within the pod assigned to them. No two sets of students will work on the same pod unless the curriculum says so, during which the course will be labeled as *paired*. In a manner of speaking, it means that a set of students, with or without a partner set, will be entitled to two pods.

A device shown within the system during a class has a one-to-one correspondence with a real device (i.e. a router image will represent a real router, a switch image will represent a real switch, and so on). The lines will represent a physical wire that connects the devices (and the numbers on each end will define the port of the module they're connected to) but will not be categorized like the devices. So to speak, a line might represent a straight-through[1], crossover[2], or octal cable[3]. The only wire which will be explicitly identified is the serial cable[4]. Any other object which needs to be explicitly represented will have to be defined using the Add Device feature. So to speak, if there arises a need to represent the wireless connection as a separate entity within the system, the Administrator will have to add its Device Name and Image. All of these are under the assumption that the new device can be accessed via Secure Shell Protocol or Remote Desktop Protocol, otherwise not at all (e.g. Serial Cable).

C. Document Object Model

The system utilizes javascript's Document Object Model (DOM) when building Logical Connections. The Document Object Model is a platform- and language-

[1]A straight-through cable is used to connect dissimilar devices together (i.e. computer-to-switch and switch-to-router) [21].

[2]A crossover cable has wires 1 and 2 switch positions with wires 3 and 6 on one end and is used to connect similar devices together (i.e. computer-to-computer, switch-to-switch, and computer-to-router) [21].

[3]A rollover cable is used to connect a computer to the console port or auxiliary port of the router for administration purposes [21].

[4]The back-to-back serial cable is used to connect two routers directly together over a serial link. A back-to-back serial link will have one router act as the DCE device with the clock rate set and the other router act as the DTE device [21].

neutral interface that will allow programs and scripts to dynamically access and update the content, structure and style of documents [22].

DOM is used to load and manipulate the Device Images upon adding a device. It is also used to store the device's attributes and save it to the database should there arise the need to edit the Logical Connection in a later point in time. This only applies to devices which are connectable. (e.g. Router, Switch, PC, Server, erminal Server, etc.)

In cases where the device only exists in the Logical Connection as an image (i.e. under no circumstances will there be a need to connect to this device), the system renders this as a Scalable Vector Graphic (SVG) DOM and is manipulated using RaphäelJS.

It should be noted that aside from storing these devices' attributes on the database, it is also saved as php files which are loaded during a class. The DOM and SVG rendering are only used when the Administrator needs to add or edit the Logical Connection of a course.

D. Protocols

The behavior of a device depends upon the Device Type. For example, clicking a router or switch opens the device's terminal while clicking a PC or server establishes a remote desktop connection to that device. The system uses PuTTy's SSH protocol to access a device's terminal remotely and uses Microsoft Windows Remote Desktop Protocol (RDP) for Remote Desktop Access. SSH (which stands for 'secure shell') is a recently designed, high-security protocol. It uses strong cryptography to protect your connection against eavesdropping, hijacking and other attacks [23].

IV. Design and Implementation

A. Context Diagram

The entities that interact with the Remote Laboratory Access are the Administrator, Professor, and Student as shown in Figure 1.

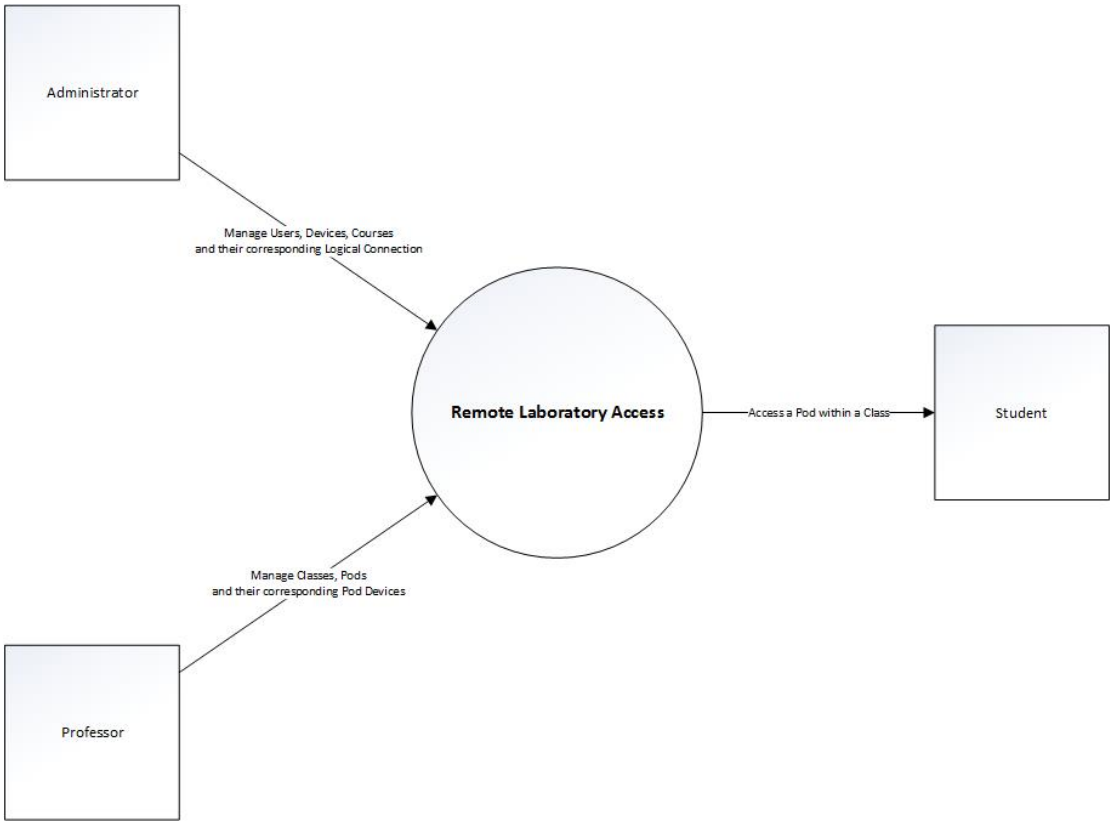


Figure 1: Context Diagram

B. Use Cases

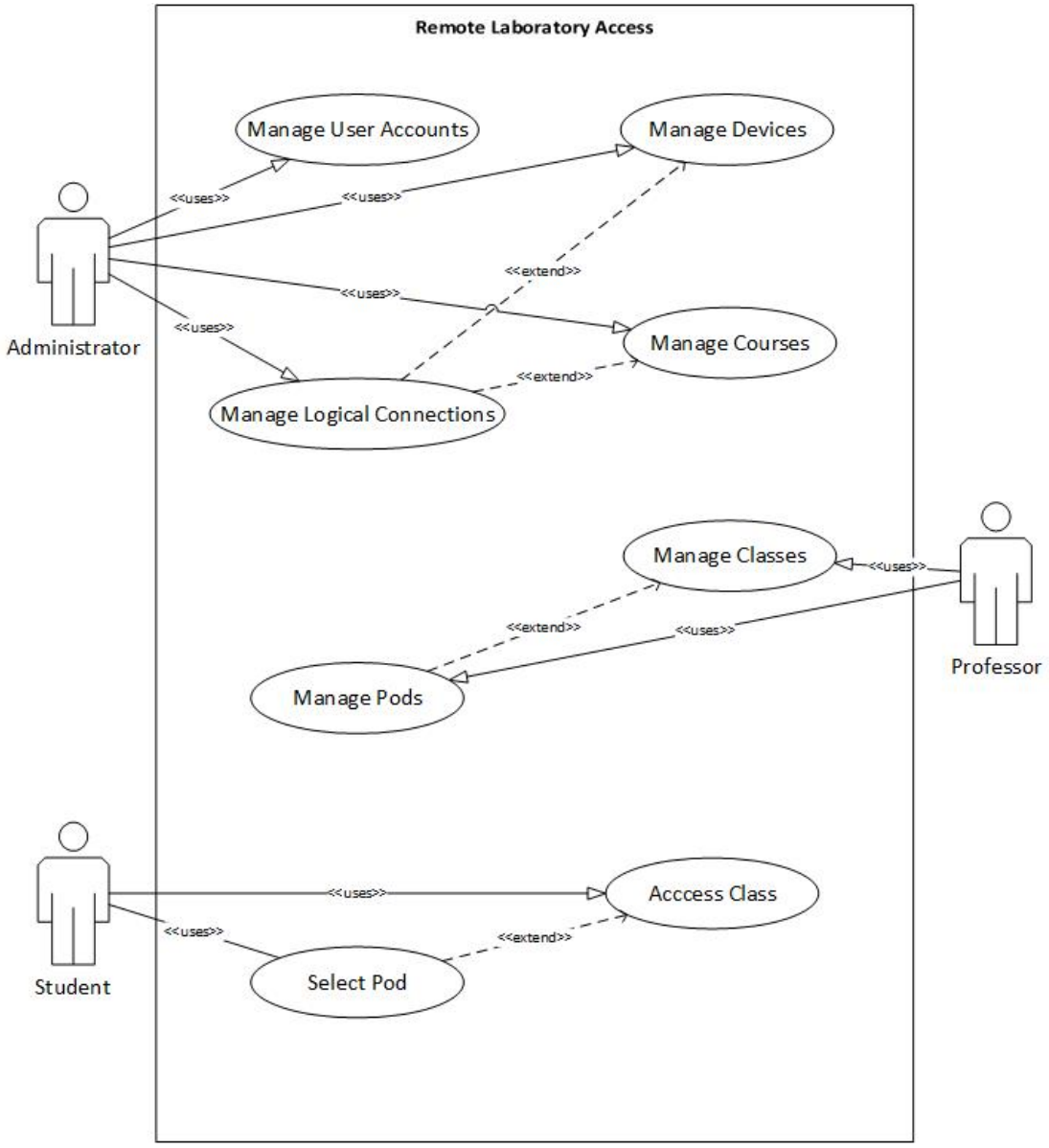


Figure 2: Use Case Diagram

The system includes three user types: the Administrator, the Professor, and the Student. The Student's login page will have a different Uniform Resource Locator (URL) from the Administrator and the Professor as requested by TNET.

The role of the Administrator is to manage the user accounts and logical connections. Managing the user accounts follows the usual add, edit, and delete database functions. Adding a user involves entering the user's Employee ID, Password, Name; and choosing the User Type. Editing a user involves selecting the

user to be edited and editing any of the user's information except the Employee ID. Deleting a user involves selecting the user to be deleted and confirming the delete operation. There are two (2) cases when the delete operation will not work: first, if the administrator is trying to delete his/her own account; second, if the administrator is trying to delete a professor account with active classes. Deleting a professor deletes all their classes and all the classes' respective pods.

Building the Logical Connection starts by defining the Course Name and Course Code it represents after which, the Professor will be asked for the maximum number of students per pod which will be used to compute the number of pods given by:

$$number\ of\ pods\ = \begin{cases} \text{paired;} & 2\lceil \frac{number\ of\ students\ enrolled}{maximum\ number\ of\ students\ per\ pod} \rceil \\ \text{not paired;} & \lceil \frac{number\ of\ students\ enrolled}{maximum\ number\ of\ students\ per\ pod} \rceil \end{cases} \quad (1)$$

The draggable Device Image (which may be a Firewall, Layer 2 Switch, Layer 3 Switch, PC, Router, Server, or Serial) will be loaded on the browser screen upon selecting it from the dropdown menu and clicking Add. A Device Image, except the Serial Cable, may assume a Device Type, which may be a Terminal Server, Router/Switch, or PC/Server. Depending on the Device Type, with the exception of the Terminal Server, a device requires a set of properties to be accessed. A Router/Switch needs a Host Name and a Display Name while a PC/Server needs an IP Address and a Display Name. Selecting the Device Image will display its corresponding Delete button and attributes. The built Logical Connection may be saved by clicking Save Topology.

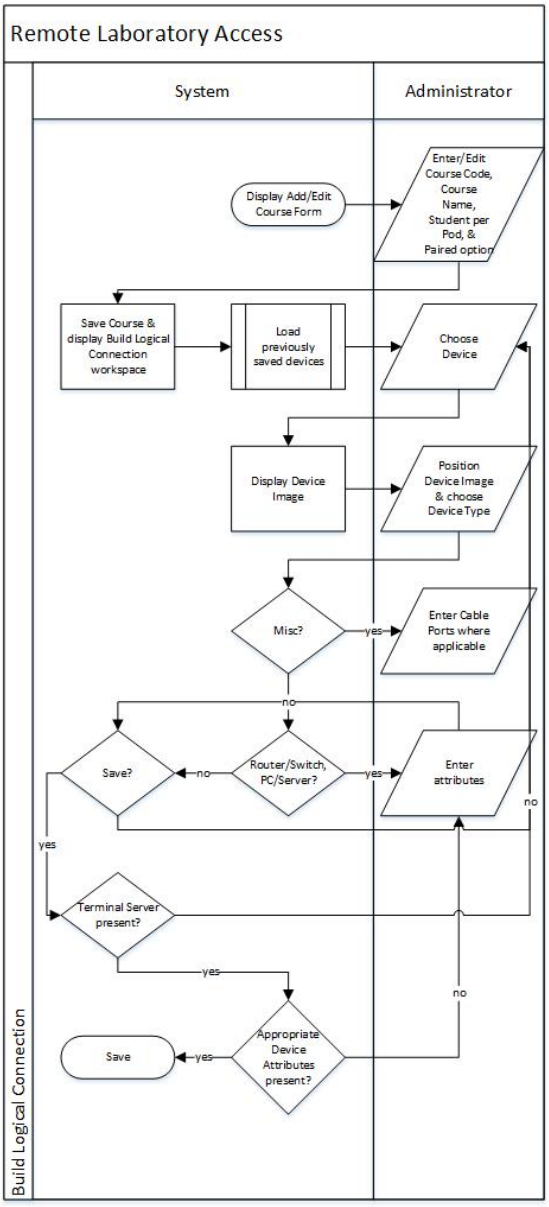


Figure 3: Build Logical Connection

In the event, however, where a device is not available within the system, the Administrator will be provided with a separate form which will enable him/her to add a new device. Adding a new device involves defining the Device Name, uploading the Device Image.

After the Logical Connection has been built, the Professor manages classes. Again, "manage" here refers to the usual database functions while "class" refers to an instance of a course. Adding a class involves choosing the class' Course and setting the class' Password and the terminal server's IP Address, Username,

Password, and Number of Students per Pod. Editing a class involves selecting the class to be edited and editing the class' information. Deleting a class involves selecting the class to be deleted and confirming the delete operation. An active class cannot be deleted. That is, if the expiration date of a class is yet to pass. Deleting a class deletes their corresponding pods.

A class can have many pods depending on the equation shown above. After a class has been created, the Professor will have to manage Pods within the class. Adding a pod involves entering the Pod Number and the pod's Password. The Logical Connection of the class, which is based on its course, will then be shown to the Professor. After which, the Professor will be asked to choose the devices included in the pod.

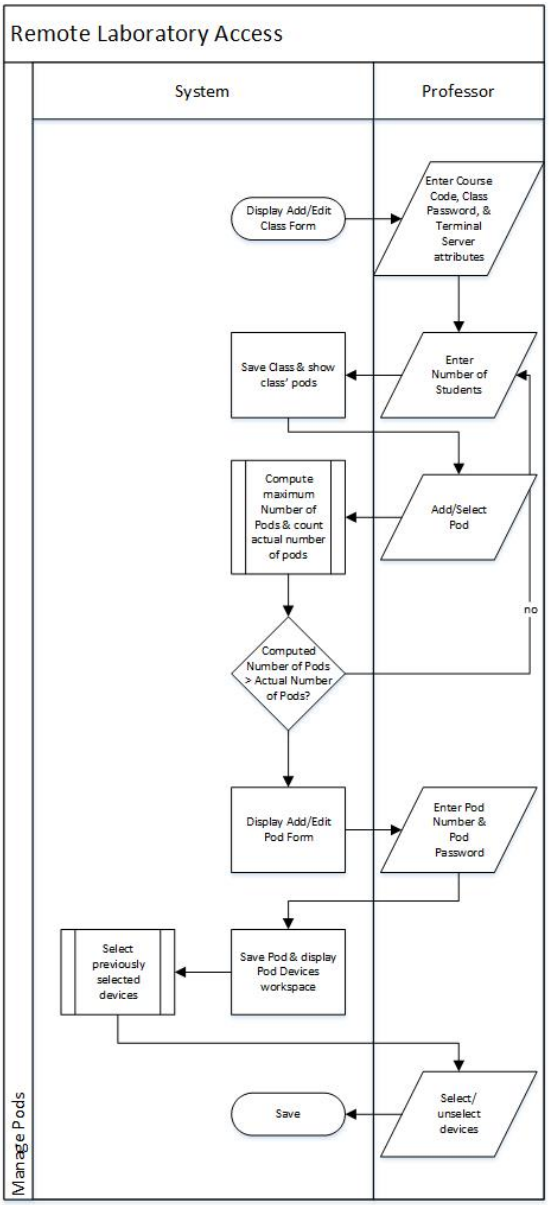


Figure 4: Manage Pods

After the devices included in each pod are defined and each pod has its own password, the Professor will have to distribute the class' password among the Students, assign them pods and distribute the pods' respective passwords.

The Student may access their class by selecting the professor and class, and providing the correct class password. On the other hand, the pods within each class may be accessed by clicking the pod and providing the correct pod password after which, they may perform the activities provided by the laboratory manual.

C. Database Design

Aside from storing the usual user and other common database tables, the system database will be used to store previously saved Logical Connections, so as it can be loaded via javascript's Document Object Model (DOM) and the Administrator can edit it accordingly. It will also store previously selected Pod Devices that it may also be loaded and edited by the Professor as needed.

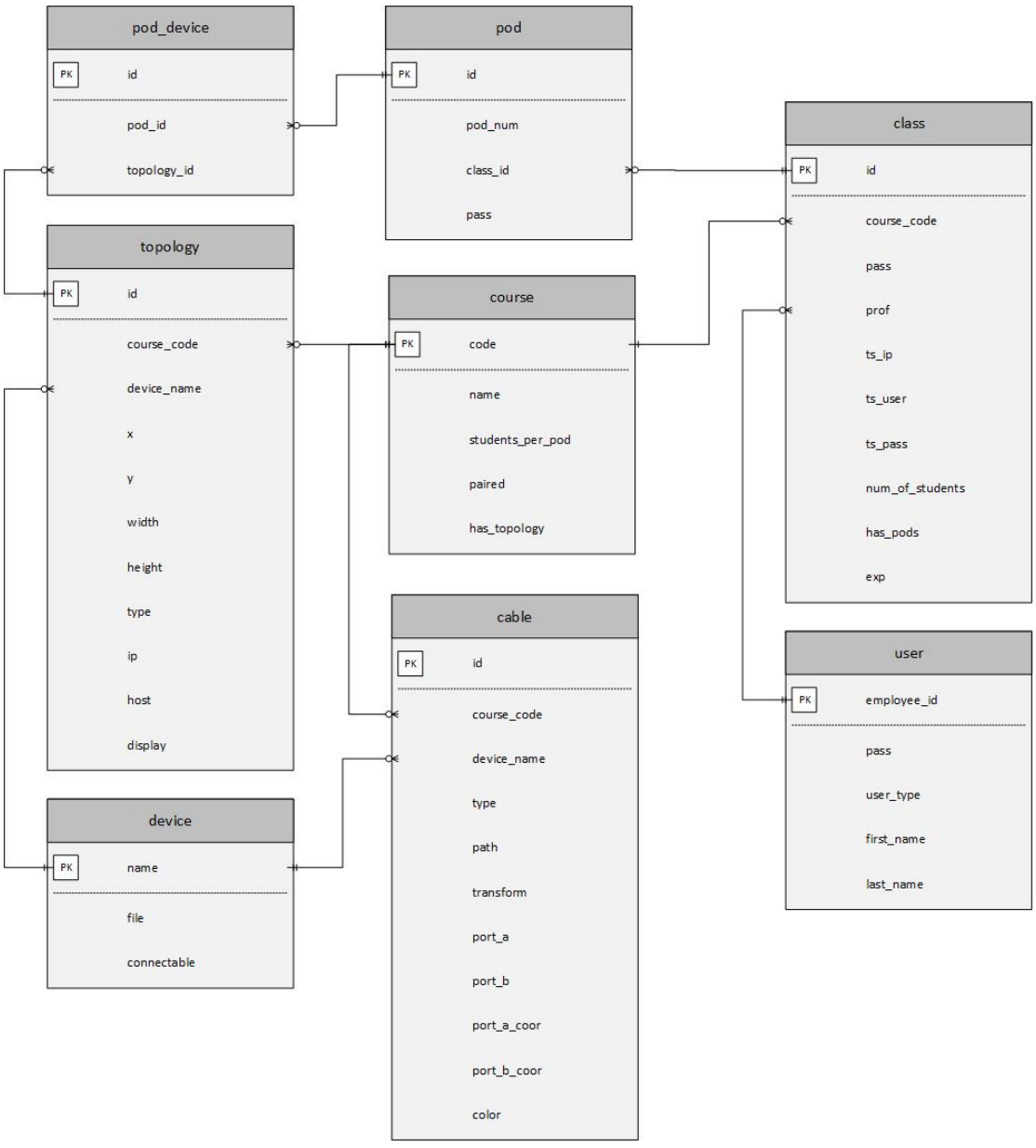


Figure 5: Entity Relationship Diagram

D. Data Dictionary

Field	Type	Description
id	int(11)	Primary Key; Unique identifier of the cable
course_code	varchar(64)	Foreign Key; The course where the cable belongs
device_name	varchar(64)	Foreign Key; The Device Image loaded by the SVG
type	int(11)	Cable type
path	varchar(128)	SVG line path/SVG image dimensions
transform	varchar(64)	Device Image rotation
port_a	varchar(16)	The port where the cable is connected
port_b	varchar(16)	The port where the cable is connected
port_a_coor	varchar(128)	Port A dimensions
port_b_coor	varchar(128)	Port B dimensions
color	varchar(32)	SVG line color attribute

Table 1: cable table

Field	Type	Description
id	int(11)	Primary Key; Unique identifier of the class
course_code	varchar(64)	Foreign Key; Class course
pass	varchar(32)	Class password
prof	varchar(32)	Foreign Key; Class professor
ts_ip	varchar(32)	Terminal server IP address
ts_user	varchar(32)	Terminal server username
ts_pass	varchar(32)	Terminal server password
num_of_students	int(11)	Number of students presently enrolled
has_pods	enum('Yes','No')	Determines whether the class has at least one pod
exp	date	Expiration date of the class

Table 2: class table

Field	Type	Description
id	int(11)	Primary key; Unique identifier of the logical connection
course_code	varchar(64)	Foreign Key; The course which owns this logical connection
device_name	varchar(64)	Foreign Key; The device to be loaded
x	int(11)	X position of the device
y	int(11)	Y position of the device
width	int(11)	Device width
height	int(11)	Device height
type	enum('1','2','3')	Device type; Detemines whether the device is a Terminal Server, Router/Switch, or PC/Server
ip	varchar(32)	Device IP address
host	varchar(16)	Device hostname
display	varchar(16)	Device display name

Table 3: topology table

Field	Type	Description
employee_id	varchar(32)	Primary Key; Unique identifier of the user
pass	varchar(128)	User password; Generated using php password_hash function
user_type	enum('Administrator','Professor')	User type
first_name	varchar(64)	First name of the user
last_name	varchar(64)	Last name of the user

Table 4: user table

Field	Type	Description
name	varchar(64)	Primary Key; Device name
file	varchar(128)	Device Image filename
connectable	enum('Yes','No')	Determines whether the device can be accessed via RDP or SSH

Table 5: device table

Field	Type	Description
code	varchar(64)	Primary Key; Course code
name	text	Course name
students_per_pod	int(11)	Maximum number of students per pod
paired	enum('Yes','No')	Used for computing the maximum number of pods
has_topology	enum('Yes','No')	Determines whether the course has a logical connection

Table 6: course table

Field	Type	Description
id	int(11)	Primary Key; Unique identifier of the pod
pod_num	int(11)	Pod number
class_id	int(11)	Foreign Key; The class where the pod belongs
pass	varchar(32)	Pod password

Table 7: pod table

Field	Type	Description
id	int(11)	Primary Key; Unique identifier of the pod device
pod_id	int(11)	Foreign Key; The pod where the pod device belongs
topology_id	int(11)	Foreign Key; The logical connection where the pod device belongs

Table 8: pod device table

V. Architecture

A. System Architecture

Trends.Net Remote Laboratory Access 2.0 was implemented using PHP's Yes It Is (Yii) Framework which uses the standard Model-View-Controller (MVC) architecture. The framework generates the basic functions for the MVC (i.e. it generates the Models from a MySQL database along with its rules and entity relations, the coresponding View forms, and the typical Add, Edit, and Delete Controller functions). Everything else; like uploading a file and saving it, deleting the uploaded file upon deletion of its corresponding database entry, and additional Model rules for user input error handling; are coded after the basic code is generated. The Logical Connection GUI and everything else that extends it are rendered using CSS, JavaScript DOM, jQuery, and RaphäelJS SVG manipulation. The system also provides an initial database synchronization and administrator account setup which was inspired by Zend Framework's initial setup, though Yii provides no such feature and was coded separately.

B. Technical Architecture

The minimum system requirements for the server are:

1. XAMPP with PHP 5.5
2. Microsoft Windows Server 2008
3. 1.0 GB free hard disk space
4. 1.0 GB RAM
5. 1.67GHz single core processor

The minimum system requirements for the client are:

1. Microsoft Windows XP Service Pack 1 or higher
2. 1.0 GB free hard disk space
3. 1.0 GB RAM
4. 1.67GHz single core processor
5. A browser that supports SVG for the Administrator and Professor
6. A browser that supports ActiveXObject for the student (i.e. Internet Explorer).

VI. Results

Trends.Net Remote Laboratory Access 2.0 is composed of two(2) major functionalities: the typical database part where the usual add, edit, and delete functions are; and the course's Logical Connection manager which makes use of the database to store and load dynamic DOM data and generates the Logical Connection PHP file which will be loaded for the Professor and Student upon logging in.

As mentioned before, the system provides an initial setup so there is no need for the Administrator to upload an sql file (for there is already an sql template to be loaded) unless the system is being migrated to another server. The first step of the said setup involves defining the database's Host Name, Username, and Password.

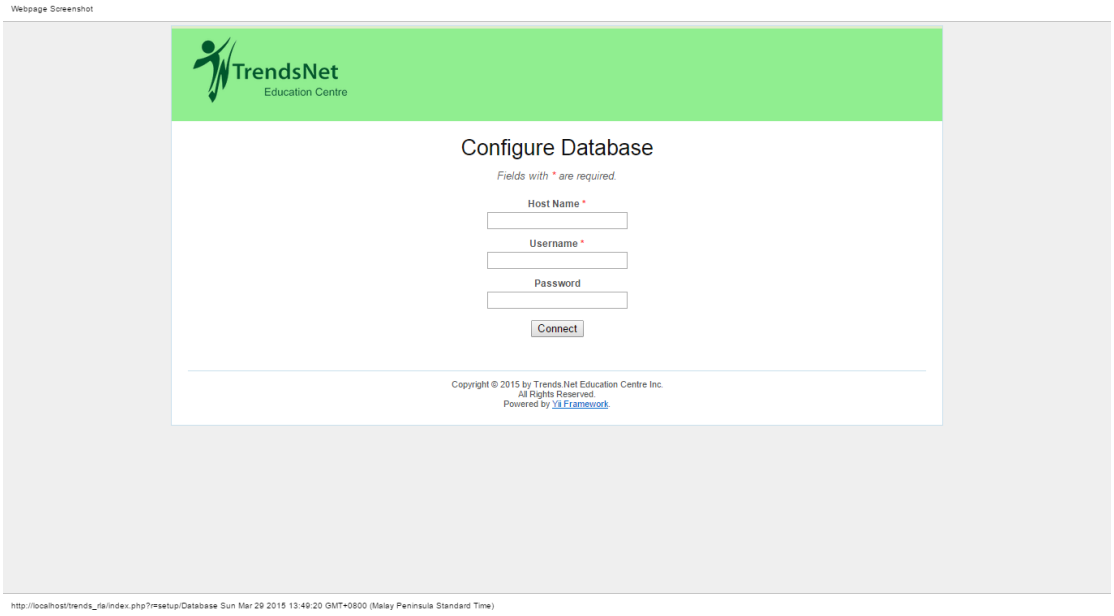


Figure 6: Database Setup

The system will then try to connect to the database server using the given credentials and create the database template located at /trends_rla/protected/data/trends_rla.sql. After the database has been successfully created, the system will ask for the first Administrator's credentials and save it in the database.

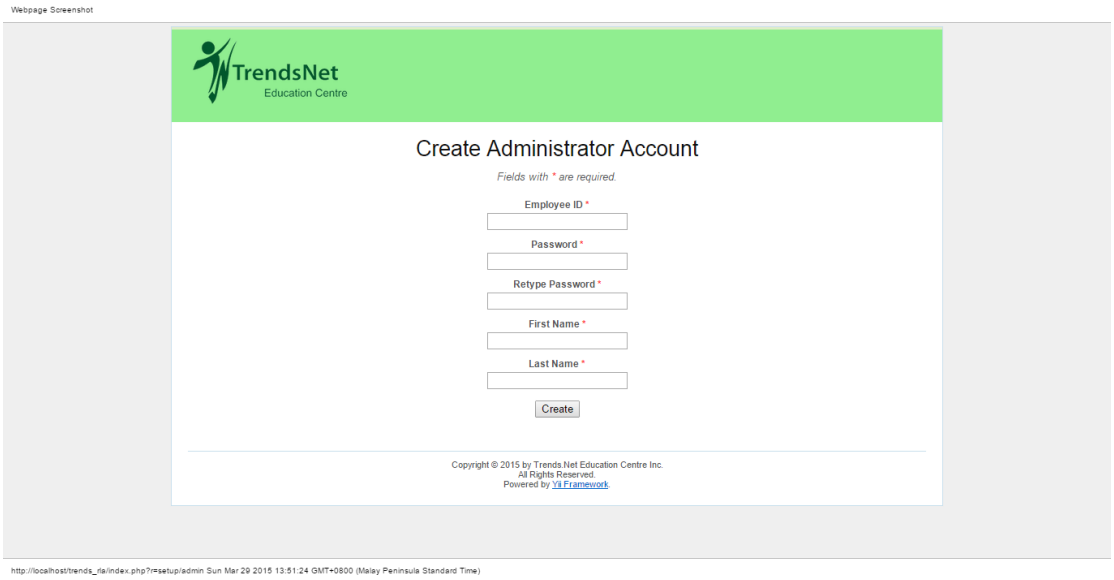


Figure 7: Administrator Setup

Now that the database and administrator account are ready, the system will redirect the browser to the Login Page.

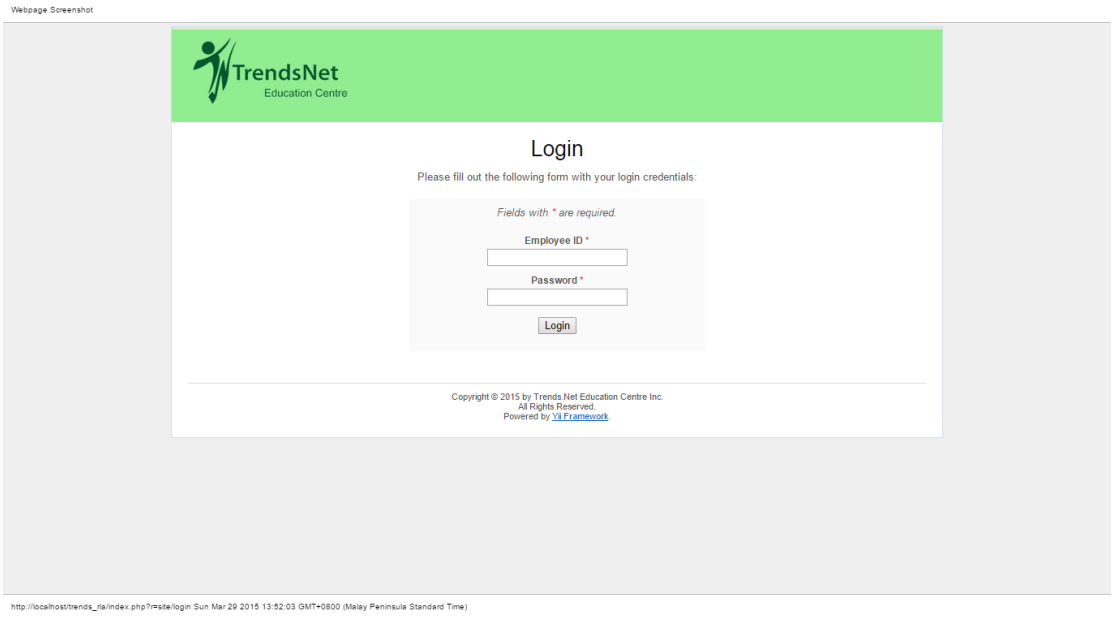


Figure 8: Login

Figure 9 shows the Administrator's Home Page which will be displayed after a sucessful Login attempt. The page has a menu bar which is composed of Home, which redirects the Administrator to the Home Page; Users; Courses, where the Logical Connection builder is; Devices, where the devices that will be used for building Logical Connections will be; and Change Password.

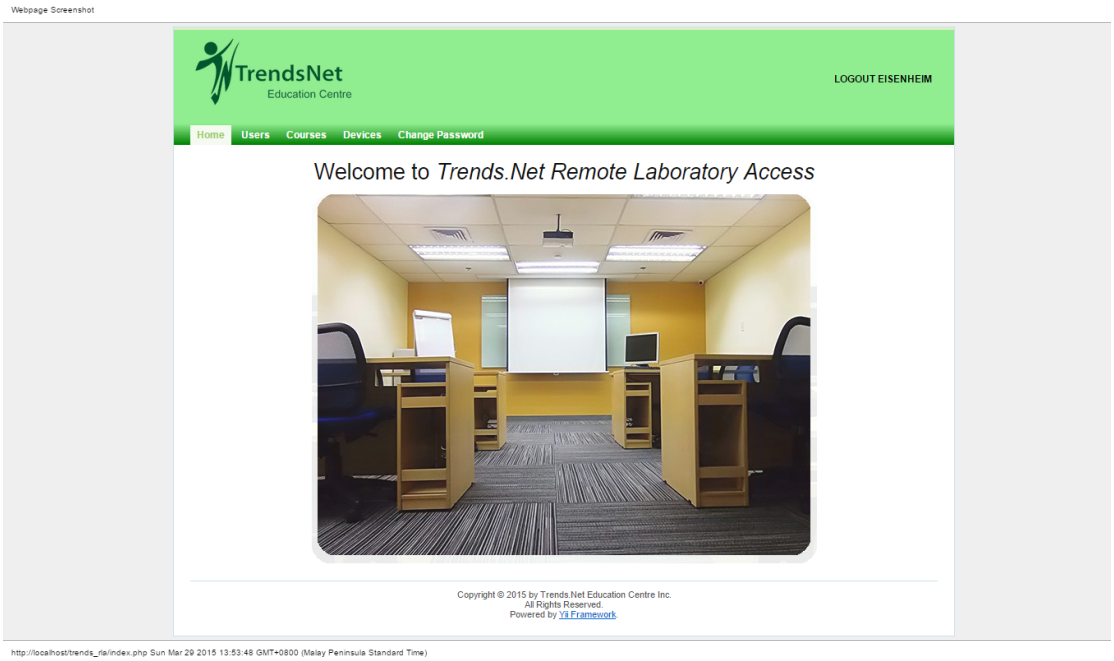


Figure 9: Administrator Home Page

Under Users, the common database functions are used to store the user's Employee ID, which will serve as the Primary key; Password; User Type, which can only be either "Administrator" or "Professor"; First Name; and Last Name. The user's password will be hashed using PHP's password_hash function before it is stored in the database.

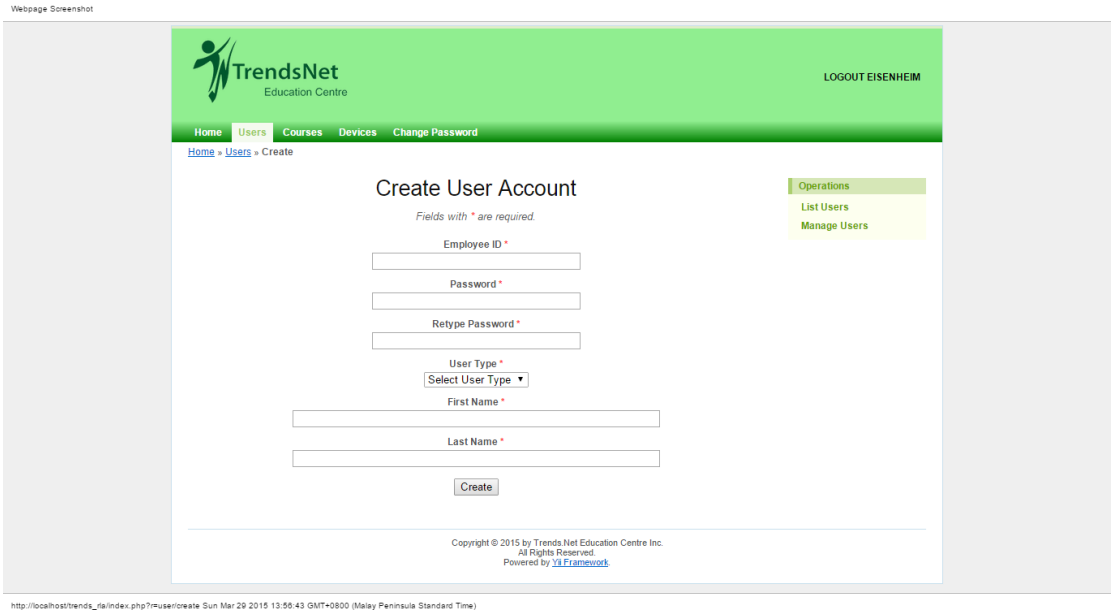


Figure 10: Create User

Should any of the Professors forget their password, there will be no means to

recover it. Instead, they can ask the Administrator(s) to change their password for them.

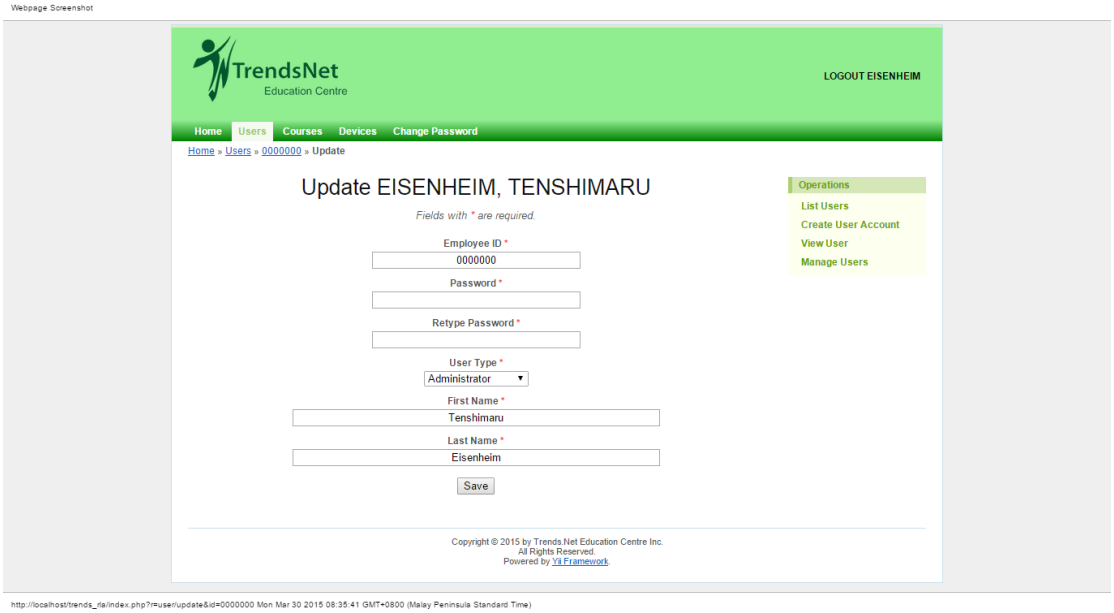


Figure 11: Update User Account

The Courses button provides the common database functions to store the Course Code, which will serve as the Primary key; the Course Name; the Students Per Pod and the Paired option, which will both be used to compute the maximum number of pods for the classes that fall under this course.

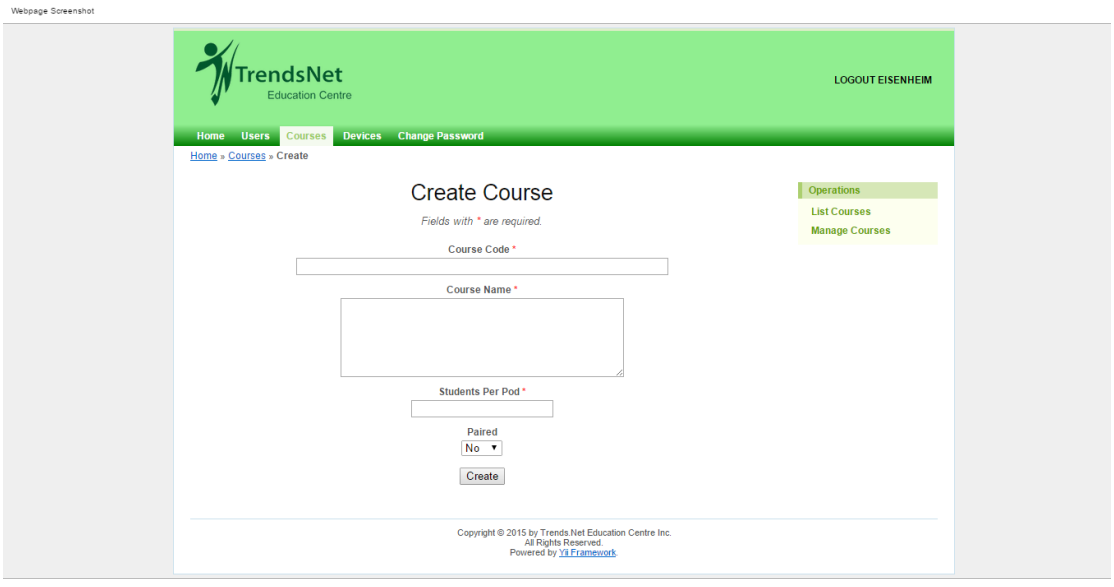


Figure 12: Create Course

Before the Logical Connection of the course can be built, the devices needed

to build it must first be uploaded. The Devices button uses the common database functions to store the Device Name, which will serve as the Primary Key; the Image File, which will be saved at /trends_rla/protected/devices and will be deleted upon the deletion of its correspoding database entry; and the Connectable option, which determines whether the device will be placed under the *Devices* tab in the Logical Connection builder where it will be required to have a Device Type or under the *Misc* tab where it will not have a Device Type but may have a maximum of two ports. Cables, and possibly the Device Image for Wireless Connection, will fall under the latter category.

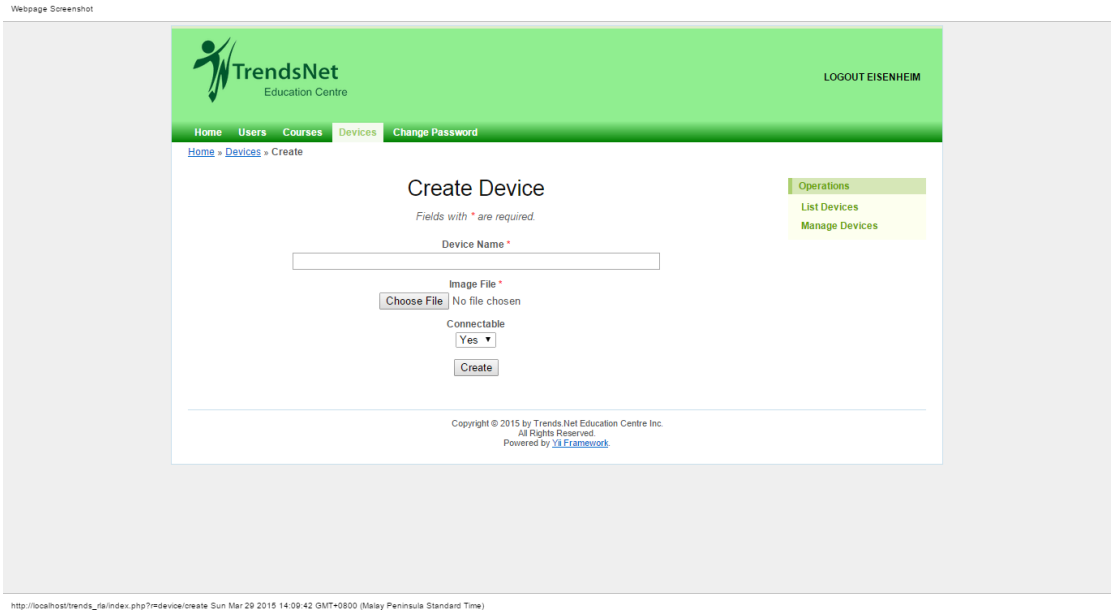


Figure 13: Create Device

Now that the Course and Devices are ready, the Logical Connection for the Course may be built. The *Topology* button on the right panel upon clicking *View Course* or *Update Course* loads this functionality. As previously stated, the entries under the Devices table are categorized into whether they are connectable or not. Connectable devices go under the Devices tab where they may be a Terminal Server; a Router/Switch, which requires a Host Name and Display Name (the former will be saved as a text file which goes to the "-m" argument of PuTTy's ssh command); or a PC/Server, which requires an IP Address and Display Name (the former will be used as an argument to Microsoft Window's RDP connection).

These devices uses jQuery's draggable widget and therefore can be dragged around the workspace.

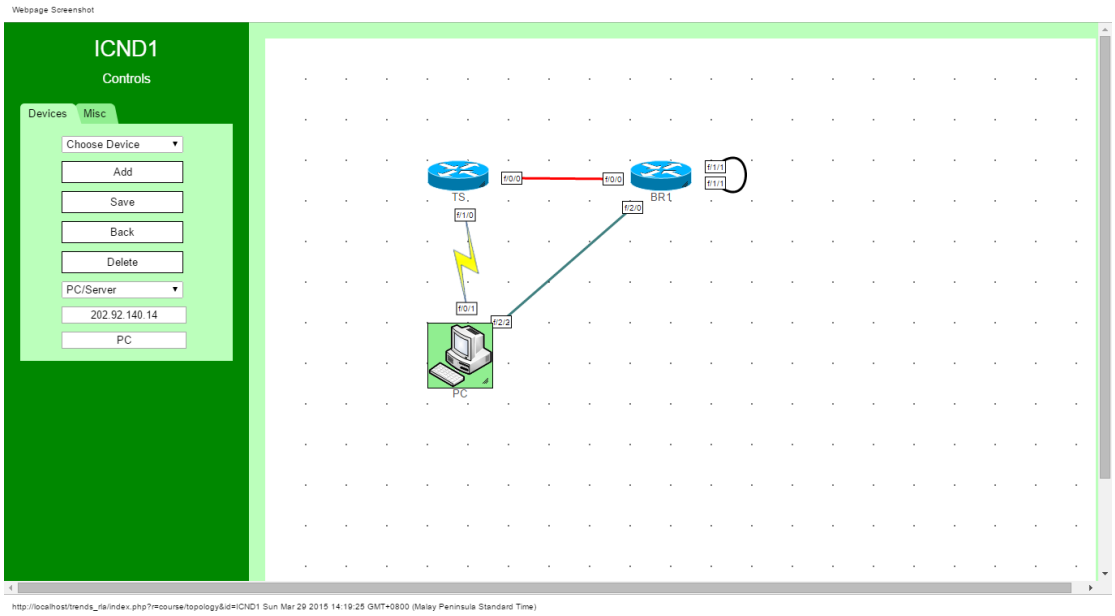


Figure 14: Connectable Device Example

The non-connectable devices under the Misc tab can be further classified into two(2): the hardcoded ones and the ones fetched from the database. The former are either the Default Cable, which is a straight line, or the Loopback Cable, which is a parabola that opens either to the left or to the right (Figure 14). The latter, on the other hand, are Devices with their *Connectable* attribute set to "No" (Figure 15). Both of these can have at least one port which will be placed on either ends for the hardcoded ones, and left and right for the database-fetched ones.

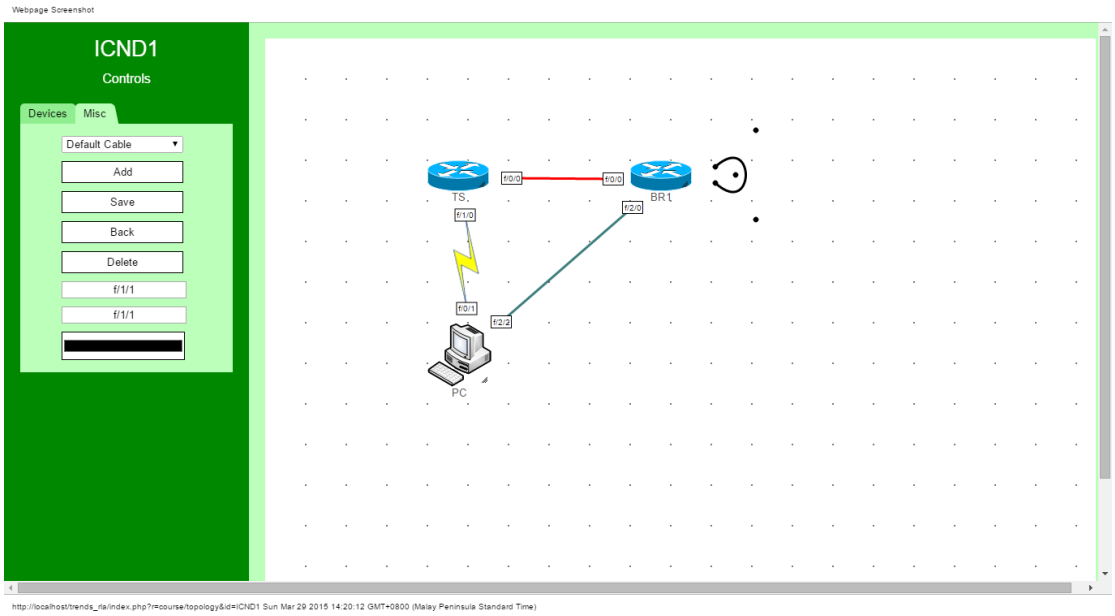


Figure 15: Hardcoded Cable Example

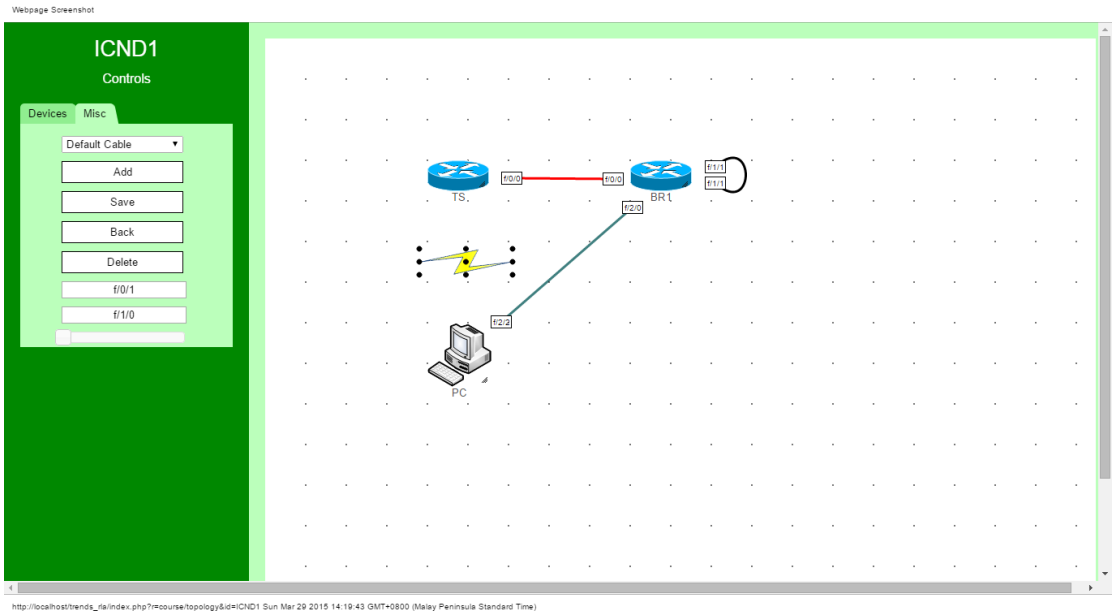


Figure 16: Non Connectable Device Example

Upon saving the Logical Connection, the system saves the coordinates, height, width, and other attributes of the devices in the database. Afterwards, it generates a PHP file under the class view, which contains the DOM elements converted into hardcode, and a folder under the same directory containing all the text files for the Terminal Server and all the Routers and Switches.

After creating the Professors' user accounts and setting up the Courses and

their corresponding Logical Connections, the Professor will have to create his Classes and their respective Pods. The Professor will be logging in at the same page as the Administrator (Figure 8). After which, the system will display the Professor's Home Page. The page's menu bar consists of the same Home and Change Password button, the only difference is the Classes button under which the Select Pod feature is contained.

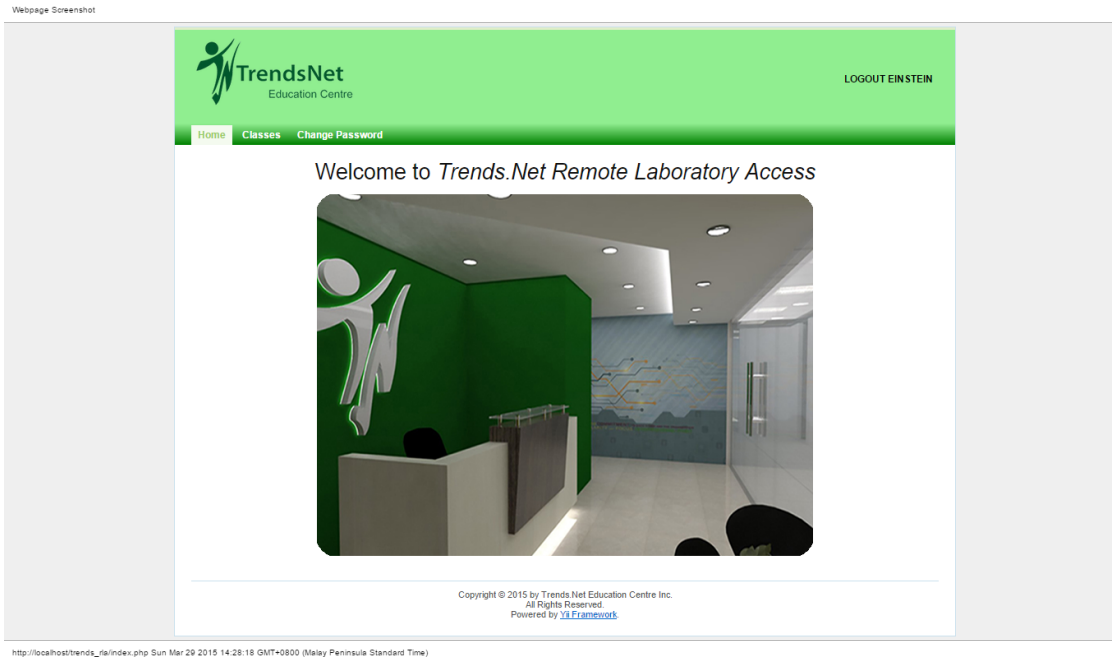


Figure 17: Professor Home Page

The Classes menu option consists of the same database functions which, in this case, stores the Course Code of the class, which is a dropdown menu containing the all the courses created by the Administrator; the Class Password; the Terminal Server credentials; and the Number of Students, where the maximum number of pods that can be created is based.

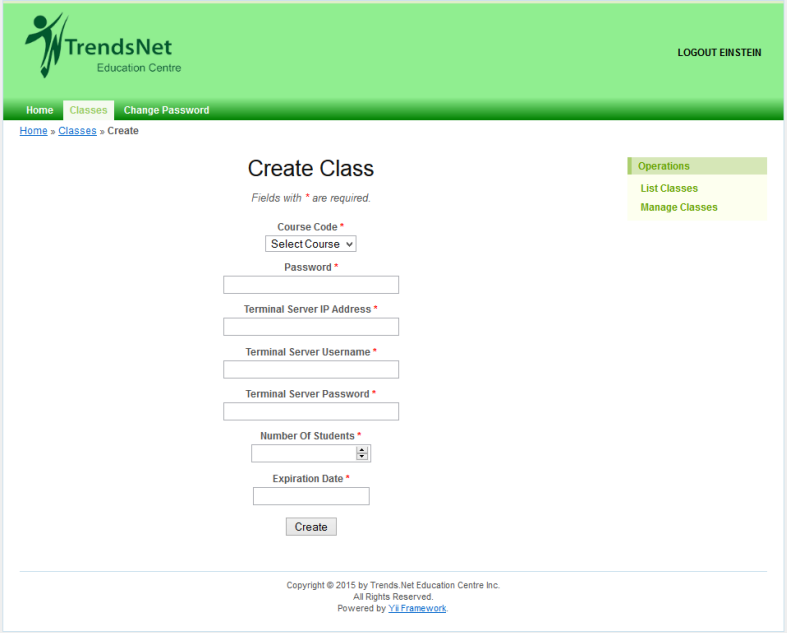


Figure 18: Create Class

Each Class contains at least one Pod and each Pod consists of a set of devices which is a subset of the Logical Connection. After the Class is created, the Professor will have to create Pods whose number ranges from one(1) up to the computed value given by the equation discussed in Chapter 1. If, in case, the maximum number is reached s/he can either create a new Class or add more students.

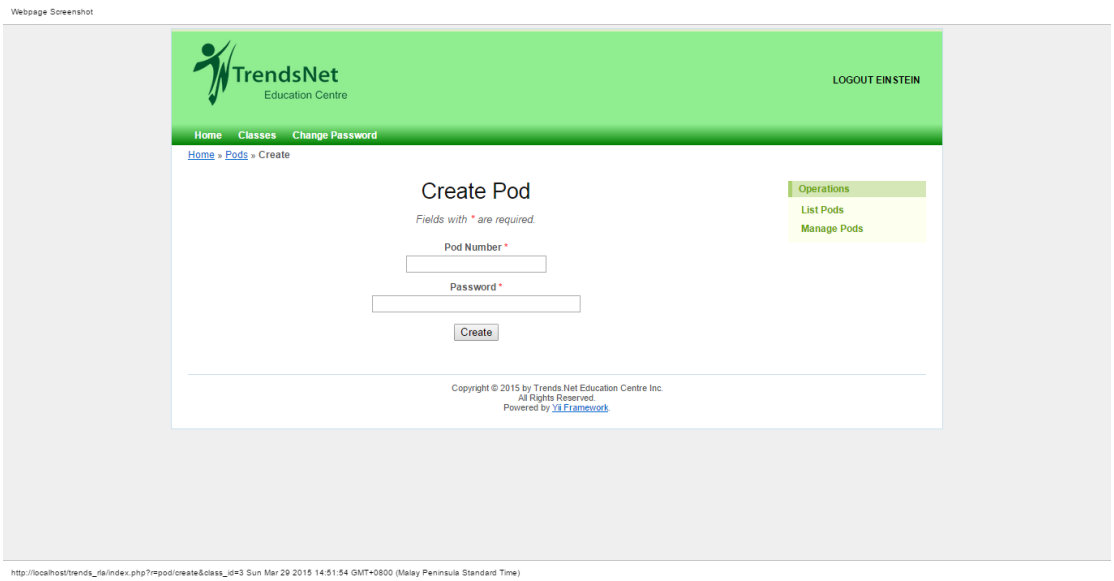


Figure 19: Create Pod

After creating the Pod, clicking the *Pod Devices* button on the right panel after clicking *View Pod* or *Update Pod* loads the Logical Connection of the Class' Course. The Professor may then click the Connectable Devices on the rendered Logical Connection. The selected devices will be saved in the database as part of the Pod in question and may be accessed by the Student assigned to that Pod.

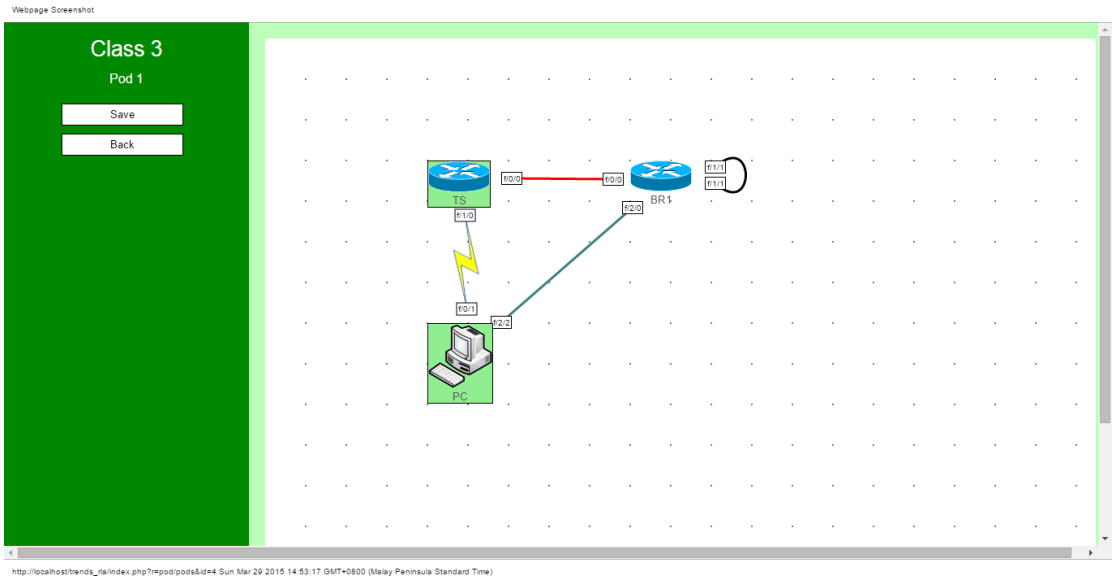


Figure 20: Select Pod Devices Example

All the User Types discussed so far have the option to change their password. It should be noted that every form within the system have their own set of rules. These rules catches any user input error and are implemented at the client side. Any invalid user input detected sets the password field in any form to null. In the change password form, the user is asked for the current password to verify that the one changing the password is the user itself and not someone impersonating him/her. Then, s/he is asked for the new password and a retype to cancel any typographical error. The form's fields are reset to null if any of the following errors are encountered by the system: the current password entered does not match the hashed password in the database; the new password has less than six(6) characters, which applies to any create password form within the system (e.g. Create Administrator Account); or the new password and the retype did not match.

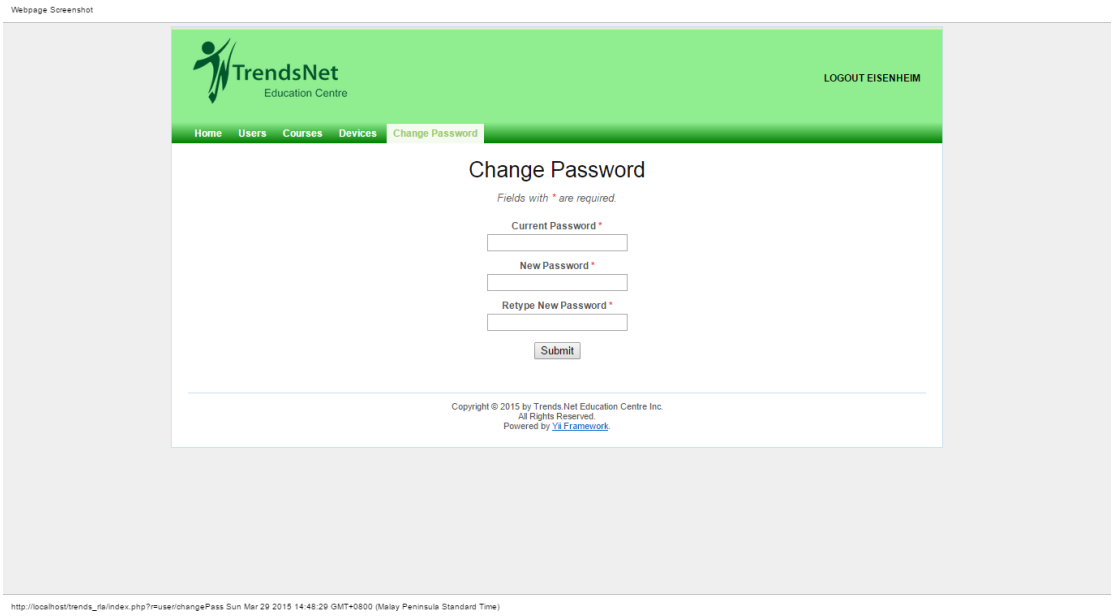


Figure 21: Change Password

Now that there are accessible Pod Devices in each Class, the Student may Login to his/her Class and select a Pod. The Student Login Page will be different, by layout and URL, from the Professor and Administrator as requested by Trends.Net Education Centre Inc. This is the very first page that will be loaded each time you go to the website. This page has three(3) fields: the Professor, which contains all the Professors in the database; the Subject, whose contents will be fetched from the database using XMLHttpRequest depending on the Professor field; and the Password, which accepts the Class Password.

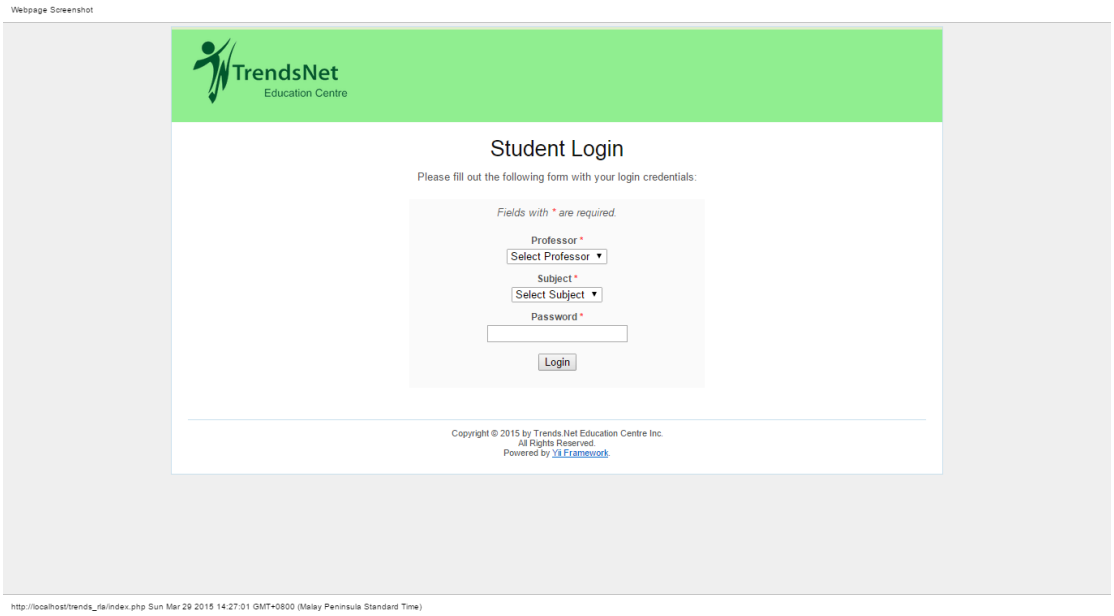


Figure 22: Student Login Page

After the Student successfully enters his/her Class credentials, he will be prompted for the Pod Number he will use and its corresponding password.

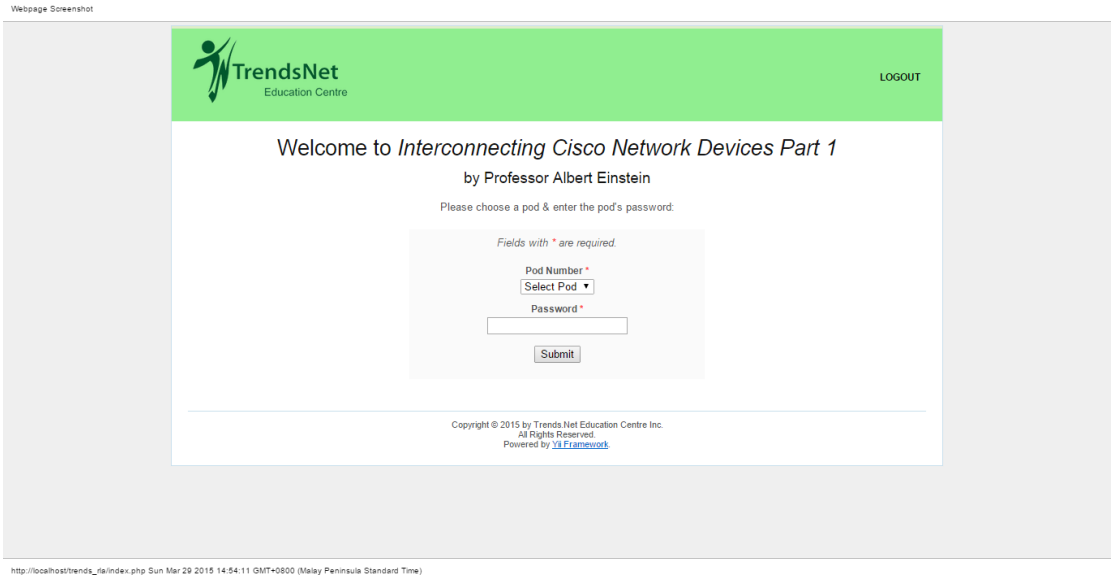


Figure 23: Pod Login

After a successful two-layer login, the Logical Connection for the Course will be rendered with the accessible connectable devices (previously selected by the Professor for that Pod) are colored while the inaccessible connectable devices are in grayscale. The actual connection may be initiated by double clicking any of the accessible connectable devices. Double clicking a Terminal Server, Router,

or Switch opens a PuTTy ssh connection to that device; while double clicking a PC or Server opens a Windows RDP connection to that device. The black close button on the upper-right corner returns the Student to the Pod Login page.

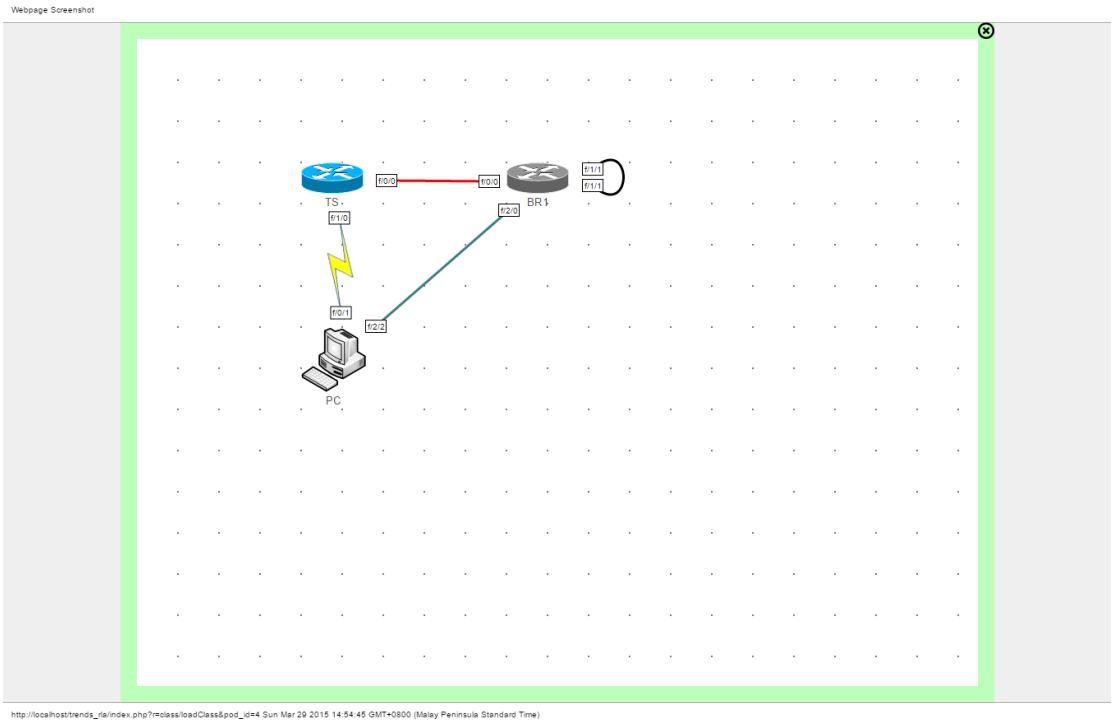


Figure 24: Access Pod Example

VII. Discussions

Trends.Net Remote Laboratory Access 2.0 aims to provide a more flexible framework expanding over multitudes of courses currently taught by Trends.Net Education Centre Inc. It intends to achieve a ductile environment capable of coping up with the curricular changes of each course. It was coded using Yii Framework that utilizes the MVC architecture to divert the difficulty of the problem from the tedious and repetitive database functions and user input error handling to the Logical Connection builder.

As previously mentioned, the Logical and Physical Connection does not follow a one-to-one correspondence. That is, while each Course maintains a single Logical Connection, each pod is only a subset of this Logical Connection. So to speak, given a Logical Connection with five(5) Routers, five(5) Switches, and a Terminal Server, a pod may contain only one(1) Router, one(1) Switch, and the Terminal Server. In the previous version, a separate image map is made for every activity while in 2.0, the same Logical Connection is loaded everytime. Essentially, everyone sees the same workspace but possess different access levels. In the example above, the Student will see all five(5) Routers and Switches including the Terminal Server but can only access the one(1) Router and Switch assigned to his/her pod as well as the Terminal Server.

Since there is a need for the transition from static html image map to image manipulation functions, the very first asset that was considered was JavaScript's DOM and was supplemented by jQuery's widgets. In particular, the draggable widget allows the image not only to be dragged but also to be contained in a particular container. The resizable widget, on the other hand, provides easy manipulation of the image's dimensions. DOM also implements a dynamic manipulation of html form fields which proved essential for saving the volatile DOM environment to the database where it will be retained and loaded during editing. It also served as a way of storing the devices' credentials (i.e. IP Address, Host Name, etc.).

The problem of the non connectable devices also arose. Since the administrator must be able to add devices that only serve as images and have no other immediate function (like accepting commands through the network) and may, or may not, be a cable during which it will need to have ports; the use of HTML5's canvas element was first tested. However, the elements within the canvas cannot be updated (i.e. drag, resize, etc.) without reloading the canvas element itself. Due to the demand of a more dynamic structure, the use of SVG was attempted. Even though it supported realtime image updating (the SVG container need not be refreshed to see the changes applied on the elements within it), the actual image managing proved to be difficult without using a predefined framework (like jQuery on JavaScript). So, with this in mind, it was found that RaphäelJS provides a set of functions that simplifies maniputaing SVG elements as DOM and, consequently, the interactions between DOM elements inside and out of the SVG environment.

There also arose a need to enable browsers to run command line arguments. This poses a major threat since you're giving the server power over your local drives. Any malicious code can and will ruin your local computer. Hence, Mozilla Firefox, Google Chrome, and all other browsers except Internet Explorer deliberately disabled this functionality. In this setting, however, there is an utmost need to perform RDP and ssh from the students' devices to the physical connection. Internet Explorer's ActiveXObject is the only remaining option.

VIII. Conclusions

Trends.Net Remote Laboratory Access 2.0 is no longer limited to just two(2) courses and may adjust accordingly with the only limitation of the protocols used to connect to the devices. It provides a better GUI for the users, especially the Administrator who oversees the workings of the courses and other users. The design is no longer a rigid hardcoded image map generated from an external application and may be updated on-the-fly. Furthermore, it provides a device manager that lets the administrator add new devices given that it is reachable through the aforementioned protocols.

The Professor, on the other hand, is given the power over the devices' accessibility within his/her class. S/he may now limit the Student's access to their pods. Before, s/he can only control the Students' interaction through the class password and pod password and cannot go beyond what is predetermined by the activity's image map. The class terminal server credentials' management is also passed to the Professor which, before, had to be prepared manually for the PuTTy to connect before, or while, the Logical Connection is built. It also eliminated the messy file repetition with each account by creating a common ground among the resources and loads it only when necessary.

The system, like its previous version, still provides device remote access across the public network so shipping the Physical Topology to the actual testing site is not necessary. Also, the use of MS Visio has been eliminated because the Logical Connection is generated automatically within the system.

One minor thing that may come into observation is that the non connectable image reverts to 0 degree transform when dragging and resizing. This is due to the fact that the orientation of the x and y-axis of the rotated Device Image and its handles become different. Meaning, if it's not reverted to transform 0 degree, dragging the handles along the x-axis will cause the Device Image to translate along whatever angle it is rotated.

IX. Recommendations

One possible improvement might be suggested by the fact that with the current design, the image quality is inversely proportional to the image dimensions, so to speak, as the image gets bigger, the more pixelated it becomes. One solution is to intentionally upload a very large Device Image and set the initial values for its height and width so that the image quality will not degrade with the increase in size.

Another possible improvement lies on the system's protocol and operating system limitation. Should there be a new way to connect two(2) devices within a network enterprise or if there arise a need for two(2) devices to connect via RDP with different operating systems (e.g. Linux to Windows and vice versa), the system has no means to adapt. So it would be very useful if there could be a more generic way of connecting devices without the use of Windows RDP and PuTTy.

X. Bibliography

[1] "Trends and technologies holdings, inc.." http://www.tthi.com.ph.

[2] "Trends.net education center." https://www.trendsnet.com.ph.

[3] J. P. Oira, 2013. Interview.

[4] J. A. Buitrago, F. D. Giraldo, and J. A. Lamprea, "Remote access lab for mitsubishi rv-2aj robot," *IEEE IX Latin American and IEEE Colombian Conference on Automatic Control and Industry Applications*, 2011.

[5] P. R. S. L. Coelho *et al.*, "A web lab for mobile robotics education," *2007 IEEE International Conference on Robotics and Automation*, 2007.

[6] R. B. Sepe, M. Chamberland, and N. Short, "Web-based virtual engineering laboratory (ve-lab) for collaborative experimentation on a hybrid electric vehicle starter/alternator," *IEEE Transactions on Industry Applications*, vol. 36, 2000.

[7] M. A. Tedesco and M. R. Emami, "A remote access platform for evaluating electric motors," *2010 XIX International Conference on Electrical Machines*, vol. 36, 2010.

[8] A. Charbonneau and V. Terskikh, "Spectrogrid: Providing simple secure remote access to scientific instruments," *2010 XIX International Conference on Electrical Machines*, 2008.

[9] "Dictionary and thesaurus - merriam-webster online." http://www.merriam-webster.com.

[10] G. Abla *et al.*, "Operation request gatekeeper: A software system for remote access control of diagnostic instruments in fusion experiments," *Review of Scientific Instruments*, vol. 81, 2010.

[11] C. C. W. Robson *et al.*, "Secure and reliable remote access for the european xfel control system," *2012 18th IEEE-NPSS Real Time Conference*, 2012.

[12] B. Aktan *et al.*, "Distance learning applied to control engineering laboratories," *IEEE Transactions on Education*, vol. 39, 1996.

[13] D. Gurkan *et al.*, "Remote laboratories for optical circuits," *IEEE Transactions on Education*, vol. 51, 2008.

[14] I. Murray and H. Armstrong, "Remote laboratory access for students with vision impairment," *5th International Conference on Networking and Services*, 2009.

[15] L. Bowtell *et al.*, "Using remote access laboratories in nursing education," *9th International Conference on Remote Engineering and Virtual Instrumentation*, 2012.

[16] J. L. Mairs, *VPNs: A Beginners Guide*. New York, USA: McGraw-Hill/OsborneMedia, 2001.

[17] M. Erwin *et al.*, *Virtual Private Networks*. OReilly Media, 2 ed., 1998.

[18] R. Yuan and W. T. Strayer, *Virtual Private Networks: Technologies and Solutions*. Addison-Wesley Professional, 2001.

[19] R. Tibbs and E. Oakes, *Firewalls and VPNs: Principles and Practices*. Prentice Hall, 2005.

[20] M. Lewis, *Comparing, Designing, and Deploying VPNs*. Cisco Press, 2006.

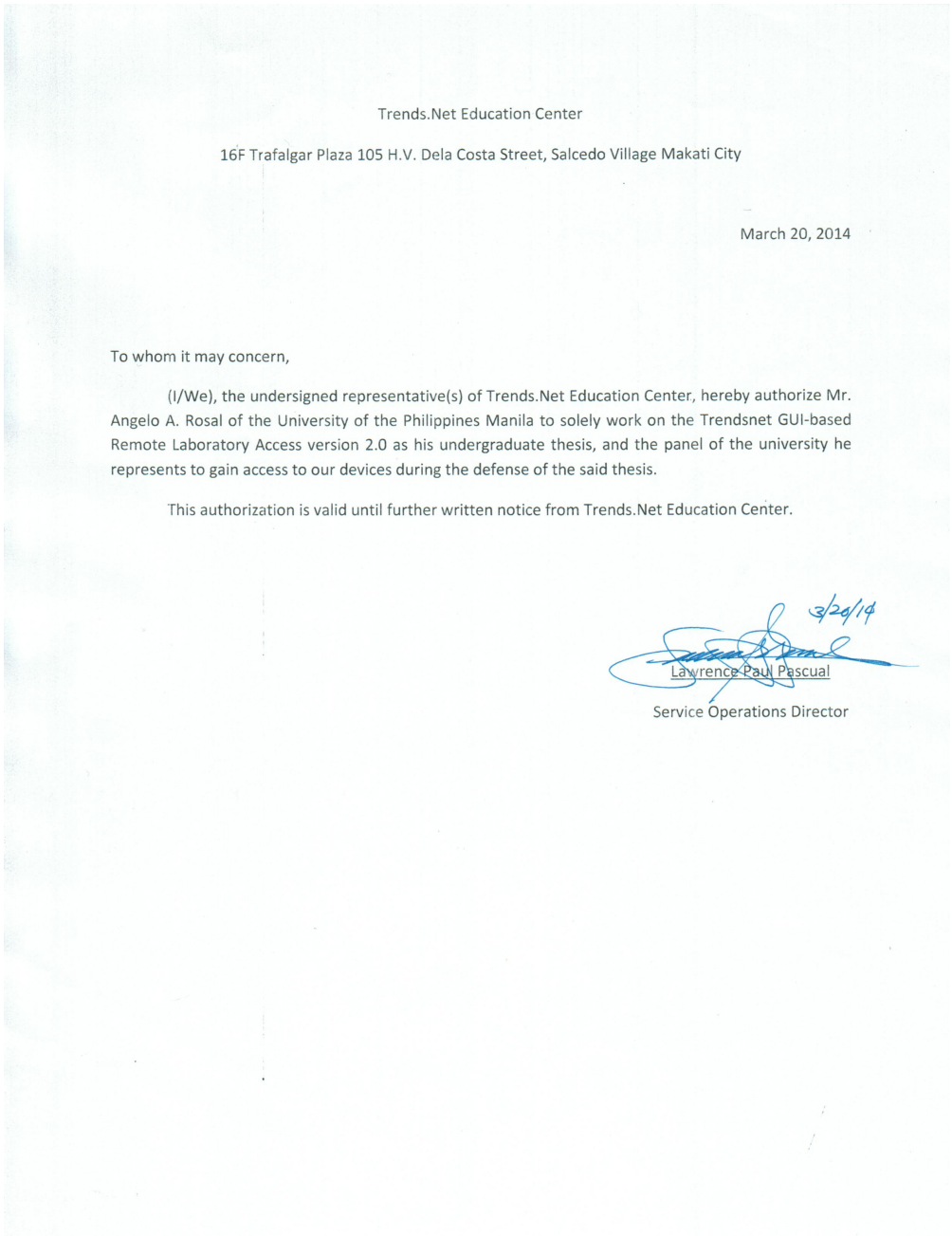
[21] "Network cabling." `http://www.dummies.com`.

[22] "Document object model (dom)." `http://www.w3.org/DOM`.

[23] "Putty user manual." `http://tartarus.org`.

XI. Appendix

A. Consent Letter



Trends.Net Education Center

16F Trafalgar Plaza 105 H.V. Dela Costa Street, Salcedo Village Makati City

March 20, 2014

To whom it may concern,

(I/We), the undersigned representative(s) of Trends.Net Education Center, hereby authorize Mr. Angelo A. Rosal of the University of the Philippines Manila to solely work on the Trendsnet GUI-based Remote Laboratory Access version 2.0 as his undergraduate thesis, and the panel of the university he represents to gain access to our devices during the defense of the said thesis.

This authorization is valid until further written notice from Trends.Net Education Center.

3/24/14

Lawrence Paul Pascual

Service Operations Director

Figure 25: Consent Letter

B. Cascading Style Sheets

Listing 1: Topology CSS

```
#topology-canvas {
        background-color: #FFFFFF;
}

#topology-div
{
        position: absolute;
        top: 0px;
        left: 0px;
        background-color: #BBFFBB;
}

#control-div
{
        position: fixed;
        top: 0px;
        left: 0px;
        height: 100%;
        padding: 20px;
        background-color: #008800;
        z-index: 5;
        overflow: auto;
        text-align: center;
}

#control-div h1,
#control-div h4
{
        color: #FFFFFF
}

#tab-controls
{
        padding: 0px;
        text-align: left;
}

#tab-controls a
{
        color: #000000;
        position: relative;
        display: inline-block;
        padding: 5px 10px;
        cursor: pointer;
        text-decoration: none;
}

#tab-controls a::before
{
        content: ''; /* To generate the box */
        position: absolute;
        top: 0;
        right: 0;
        bottom: 0;
        left: 0;
        z-index: -1;
        background-color: #90EE90;
        transform: perspective(20px) rotateX(5deg);
        transform-origin: top left;
        border-radius: 5px 5px 0px 0px;
}

#tab-controls a.active
{
        z-index: 2;
}

#tab-controls a.active::before
{
        background-color: #BBFFBB;
        border-bottom: none;
}

#tab-contents
{
        background-color: #BBFFBB;
        padding: 5px;
}

#tab-contents div.tab-content
{
        display: none;
}

#tab-contents div.tab-content.active
{
        display: block;
}

#tab-contents div select,
#tab-contents div input[type="button"],
```

```
#tab-contents div input[type="submit"],
#tab-contents div input[type="text"],
#tab-contents div input[type="color"],
#control-div input[type="button"],
#control-div input[type="submit"]
{
        width: 150px;
        display: block;
        margin: 10px;
}

#tab-contents div input[type="button"],
#tab-contents div input[type="submit"],
#tab-contents div input[type="color"],
#control-div input[type="button"],
#control-div input[type="submit"]
{
        padding: 5px 0px;
        background-color: #FFFFFF;
        border: 1px solid #000000;
        cursor: pointer;
}

#tab-contents div input[type="button"]:hover,
#tab-contents div input[type="submit"]:hover,
#tab-contents div input[type="color"]:hover,
#control-div input[type="button"]:hover,
#control-div input[type="submit"]:hover
{
        color: #FFFFFF;
        background-color: #000000;
}

#tab-contents div input[type="button"]:active,
#tab-contents div input[type="submit"]:active,
#tab-contents div input[type="color"]:active,
#control-div input[type="button"]:active,
#control-div input[type="submit"]:active
{
        color: #000000;
        background-color: #BBBBBB;
}

#canvas-div
{
        position: absolute;
        padding: 20px;
        top: 0px;
        left: 0px;
}

input[type="text"] {
        text-align: center
}

div.draggable {
        cursor: move;
}

div.draggable input[type="button"] {
        display: block;
}

div.selected {
        z-index: 2;
        border: 1px solid black;
        background-color: lightgreen;
}

img.grayscale {
        filter: url("data:image/svg+xml;utf8,<svg xmlns=\'http://www.w3.org/2000/svg\'><filter
             id=\'grayscale\'><feColorMatrix type=\'matrix\' values=\'0.3333 0.3333 0.3333 0 0
             0.3333 0.3333 0.3333 0 0 0.3333 0.3333 0.3333 0 0 0 0 0 1 0\'/></filter></svg>#
            grayscale");
        -webkit-filter: grayscale(100%);
}
```

C. JavaScripts

Listing 2: Cable JS

```
var paper;

function cableInit() {
        paper = Raphael(document.getElementById("topology-canvas"), $("#topology-canvas").
            width(), $("#topology-canvas").height());
}

function drawCable(cable_id, type, path, port_a, port_b, color, cableImg, cable_name,
    transform, port_a_coor, port_b_coor) {
        var line, colorForm, transForm, cableForm, a, b, rotator, coorA, coorB;

        if(type < 2) {
                line = paper.path(path);
                line.attr({
```

```
                stroke: color,
                "stroke-width": 3
        })
        .data("type", type)
        .mouseover(function(event) {
                this[0].style.cursor = "pointer";
        })
        .mouseout(function(event) {
                this[0].style.cursor = "";
        })
        .click(function() {
                toggleCableSelect(line);
        });

        colorForm = document.createElement("input");
        $(colorForm).attr({
                type: "color",
                id: "line_"+line.id+"_color",
                name: "line["+line.id+"][color]",
                style: "margin-left: "+((10 + excess) / 2)+"px"
        })
        .val(color)
        .change(function() {
                line.attr("stroke", $(this).val());
        });
} else {
        var dim = path.split(","), i;

        for(i = 0; i < dim.length; i++)
                dim[i] = parseFloat(dim[i]);

        line = paper.image(cableImg, dim[0], dim[1], dim[2], dim[3]);
        line.data("type", type)
        .mouseover(function(event) {
                this[0].style.cursor = "pointer";
        })
        .mouseout(function(event) {
                this[0].style.cursor = "";
        })
        .click(function() {
                toggleCableSelect(line);
        });

        transForm = document.createElement("input");
        $(transForm).attr({
                type: "hidden",
                id: "line_"+line.id+"_transform",
                name: "line["+line.id+"][transform]",
                style: "margin-left: "+((10 + excess) / 2)+"px"
        }).val(transform);

        rotator = document.createElement("div");
        $(rotator).attr({
                id: "rotator_"+line.id,
                style: "margin-left: "+((10 + excess) / 2)+"px; width: 150px"
        })
        .slider({
                orientation: "horizontal",
                min: -91,
                max: 91,
                value: transform,
                slide: function(event, ui) {
                        var pa = paper.getById(line.data("portA")),
                        ra = paper.getById(line.data("rectA")),
                        pb = paper.getById(line.data("portB")),
                        rb = paper.getById(line.data("rectB")),
                        circle = paper.path("M "+(line.attr("x") + line.attr("width")
                             / 2)+" "+(line.attr("y") + line.attr("height") / 2)+
                                " m "+(-line.attr("width") / 2)+", 0"+
                                " a "+(line.attr("width") / 2)+", "+(line.attr("width
                                     ") / 2)+" 0 1, 0 "+line.attr("width")+", 0"+
                                " a "+(line.attr("width") / 2)+", "+(line.attr("width
                                     ") / 2)+" 0 1, 0 "+(-line.attr("width"))+", 0")
                        .attr("opacity", 0);

                        line.attr("transform", "r"+$(this).slider("value"));

                        if($(this).slider("value") > 0)
                                pa.attr({
                                        x: circle.getPointAtLength(circle.
                                             getTotalLength() * ((360 - $(this).slider
                                             ("value")) / 360)).x,
                                        y: circle.getPointAtLength(circle.
                                             getTotalLength() * ((360 - $(this).slider
                                             ("value")) / 360)).y
                                });
                        else
                                pa.attr({
                                        x: circle.getPointAtLength(circle.
                                             getTotalLength() * ((0 - $(this).slider("
                                             value")) / 360)).x,
                                        y: circle.getPointAtLength(circle.
                                             getTotalLength() * ((0 - $(this).slider("
                                             value")) / 360)).y
                                });

                        var ba = pa.getBBox();

                        ra.attr({
```

```
                                        x: ba.x - 2,
                                        y: ba.y - 2,
                                        width: ba.width + 4,
                                        height: ba.height + 4
                                });

                                pb.attr({
                                        x: circle.getPointAtLength(circle.getTotalLength() *
                                             ((180 - $(this).slider("value")) / 360)).x,
                                        y: circle.getPointAtLength(circle.getTotalLength() *
                                             ((180 - $(this).slider("value")) / 360)).y
                                });

                                var bb = pb.getBBox();

                                rb.attr({
                                        x: bb.x - 2,
                                        y: bb.y - 2,
                                        width: bb.width + 4,
                                        height: bb.height + 4
                                });

                                $("#line_"+line.id+"_rect_a").val(ra.attr("x")+","+ra.attr("y
                                     ")+","+ra.attr("width")+","+ra.attr("height"));
                                $("#line_"+line.id+"_rect_b").val(rb.attr("x")+","+rb.attr("y
                                     ")+","+rb.attr("width")+","+rb.attr("height"));
                                $("#line_"+line.id+"_port_a_coor").val(pa.attr("x")+","+pa.
                                     attr("y"));
                                $("#line_"+line.id+"_port_b_coor").val(pb.attr("x")+","+pb.
                                     attr("y"));

                                circle.remove();
                        },
                        change: function(event, ui) {
                                $("#line_"+line.id+"_transform").val($(this).slider("value"));
                        }
                })
                .mouseover(function() {
                        paper.forEach(function(el) {
                                if(el.type == "circle")
                                        el.hide();
                        });
                        line.animate({transform: "r"+$(this).slider("value")}, 100);
                        paper.getById(line.data("portA")).show();
                        paper.getById(line.data("rectA")).show();
                        paper.getById(line.data("portB")).show();
                        paper.getById(line.data("rectB")).show();
                })
                .mouseout(function() {
                        paper.forEach(function(el) {
                                if(el.type == "circle")
                                        el.show();
                        });
                        line.animate({transform: "r0"}, 100);
                        paper.getById(line.data("portA")).hide();
                        paper.getById(line.data("rectA")).hide();
                        paper.getById(line.data("portB")).hide();
                        paper.getById(line.data("rectB")).hide();
                });

                cableForm = document.createElement("input");
                $(cableForm).attr({
                        type: "hidden",
                        id: "line_"+line.id+"_device_name",
                        name: "line["+line.id+"][device_name]",
                        style: "margin-left: "+((10 + excess) / 2)+"px"
                }).val(cable_name);

                coorA = document.createElement("input");
                $(coorA).attr({
                        type: "hidden",
                        id: "line_"+line.id+"_port_a_coor",
                        name: "line["+line.id+"][port_a_coor]",
                        size: 17,
                        value: port_a_coor,
                        style: "margin-left: "+((10 + excess) / 2)+"px"
                });

                coorB = document.createElement("input");
                $(coorB).attr({
                        type: "hidden",
                        id: "line_"+line.id+"_port_b_coor",
                        name: "line["+line.id+"][port_b_coor]",
                        size: 17,
                        value: port_b_coor,
                        style: "margin-left: "+((10 + excess) / 2)+"px"
                });
        }

        if(port_a == null || port_a == "")
                port_a = "port a";

        if(port_b == null || port_b == "")
                port_b = "port b";

        var delCable = document.createElement("input");
        $(delCable).attr({
                type: "button",
                id: "delete_cable_"+line.id,
```

```
        class: "button",
        value: "Delete",
        style: "margin-left: "+((10 + excess) / 2)+"px"
})
.click(function() {
        if($("#line_"+line.id+"_id").val() != "") {
                var deletedID = document.createElement("input");
                $(deletedID).attr({
                        type: "hidden",
                        id: "deleted_"+line.id+"_id",
                        name: "deleted["+line.id+"][id]",
                        value: $("#line_"+line.id+"_id").val()
                });

                var deletedType = document.createElement("input");
                $(deletedType).attr({
                        type: "hidden",
                        id: "deleted_"+line.id+"_type",
                        name: "deleted["+line.id+"][type]",
                        value: "cable"
                });

                $("#topology-model-form").append(deletedID);
                $("#topology-model-form").append(deletedType);
        }

        deleteCable(line);
});

var cableID = document.createElement("input");
$(cableID).attr({
        type: "hidden",
        id: "line_"+line.id+"_id",
        name: "line["+line.id+"][id]",
        value: cable_id,
        style: "margin-left: "+((10 + excess) / 2)+"px",
        readonly: true
});

var typeForm = document.createElement("input");
$(typeForm).attr({
        type: "hidden",
        id: "line_"+line.id+"_type",
        name: "line["+line.id+"][type]",
        value: type,
        style: "margin-left: "+((10 + excess) / 2)+"px",
        readonly: true
});

var pathForm = document.createElement("input");
$(pathForm).attr({
        type: "hidden",
        id: "line_"+line.id+"_path",
        name: "line["+line.id+"][path]",
        value: path,
        style: "margin-left: "+((10 + excess) / 2)+"px",
        readonly: true
});

if(line.data("type") < 2) {
        var pts = line.attr("path");

        a = paper.text(pts[0][1], pts[0][2], port_a);
} else {
        var x = line.attr("x"),
        y = line.attr("y") + line.attr("height") / 2;

        a = paper.text(port_a_coor.split(",")[0], port_a_coor.split(",")[1], port_a);
}

var boxA = a.getBBox();

rectA = paper.rect(boxA.x - 2, boxA.y - 2, boxA.width + 4, boxA.height + 4)
.attr("fill","#ffffff").insertAfter(line);
a.insertAfter(rectA);

formA = document.createElement("input");
$(formA).attr({
        type: "hidden",
        id: "line_"+line.id+"_rect_a",
        name: "line["+line.id+"][rect_a]",
        value: rectA.attr("x")+","+rectA.attr("y")+","+rectA.attr("width")+","+rectA.
             attr("height")
});

line.data("portA", a.id)
.data("rectA", rectA.id);

var portA = document.createElement("input");
$(portA).attr({
        type: "text",
        id: "line_"+line.id+"_port_a",
        name: "line["+line.id+"][port_a]",
        size: 17,
        value: port_a,
        style: "margin-left: "+((10 + excess) / 2)+"px"
})
.focus(function() {
        if($(this).val() == "port a")
```

```
                $(this).val("");
})
.change(function() {
        $(this).val($(this).val().trim());

        if($(this).val() == "")
                $(this).val("port a");

        a.attr("text", $(this).val());

        var ba = a.getBBox();

        var ra = paper.getById(line.data("rectA"));

        ra.attr({
                x: ba.x - 2,
                y: ba.y - 2,
                width: ba.width + 4,
                height: ba.height + 4
        });

        $("#line_"+line.id+"_rect_a").val(ra.attr("x")+","+ra.attr("y")+","+ra.attr("
             width")+","+ra.attr("height"));
});

if(line.data("type") == 0) {
        var pts = line.attr("path");

        b = paper.text(pts[1][1], pts[1][2], port_b);
} else if(line.data("type") == 1) {
        var pts = line.attr("path");

        b = paper.text(pts[1][5], pts[1][6], port_b);
} else {
        var x = line.attr("x") + line.attr("width"),
        y = line.attr("y") + line.attr("height") / 2;

        b = paper.text(port_b_coor.split(",")[0], port_b_coor.split(",")[1], port_b);
}

var boxB = b.getBBox();

rectB = paper.rect(boxB.x - 2, boxB.y - 2, boxB.width + 4, boxB.height + 4)
.attr("fill","#ffffff").insertAfter(line);
b.insertAfter(rectB);

formB = document.createElement("input");
$(formB).attr({
        type: "hidden",
        id: "line_"+line.id+"_rect_b",
        name: "line["+line.id+"][rect_b]",
        value: rectB.attr("x")+","+rectB.attr("y")+","+rectB.attr("width")+","+rectB.
             attr("height")
});

line.data("portB", b.id)
.data("rectB", rectB.id);

var portB = document.createElement("input");
$(portB).attr({
        type: "text",
        id: "line_"+line.id+"_port_b",
        name: "line["+line.id+"][port_b]",
        size: 17,
        value: port_b,
        style: "margin-left: "+((10 + excess) / 2)+"px",
})
.focus(function() {
        if($(this).val() == "port b")
                $(this).val("");
})
.change(function() {
        $(this).val($(this).val().trim());

        if($(this).val() == "")
                $(this).val("port b");

        b.attr("text", $(this).val());

        var bb = b.getBBox();

        var rb = paper.getById(line.data("rectB"));

        rb.attr({
                x: bb.x - 2,
                y: bb.y - 2,
                width: bb.width + 4,
                height: bb.height + 4
        });

        $("#line_"+line.id+"_rect_b").val(rb.attr("x")+","+rb.attr("y")+","+rb.attr("
             width")+","+rb.attr("height"));
});

$("#cables-tab").append(delCable);
$("#cables-tab").append(cableID);
$("#cables-tab").append(typeForm);
$("#cables-tab").append(pathForm);
$("#cables-tab").append(portA);
```

```
        $("#cables-tab").append(portB);
        $("#cables-tab").append(formA);
        $("#cables-tab").append(formB);

        if(type < 2) {
                $("#cables-tab").append(colorForm);
        } else {
                $("#cables-tab").append(rotator);
                $("#cables-tab").append(transForm);
                $("#cables-tab").append(cableForm);
                $("#cables-tab").append(coorA);
                $("#cables-tab").append(coorB);
        }

        $("#TopologyModel_cable").prop("selectedIndex", 0);

        toggleCableSelect(line);
}

function addCable(cable_id, type, path, port_a, port_b, color, cable_name, transform,
    port_a_coor, port_b_coor) {
        if(cable_id == null) {
                type = $("#TopologyModel_cable").prop("selectedIndex");

                if(type == 0) {
                        path = "M 3 103 L 103 3";
                        color = "#FF0000";
                        drawCable(cable_id, type, path, port_a, port_b, color);
                } else if(type == 1) {
                        path = "M 3 48 C 53 3 53 113 3 68";
                        color = "#FF0000";
                        drawCable(cable_id, type, path, port_a, port_b, color);
                } else {
                        transform = 0;
                        var cableImg = $("#TopologyModel_cable").val();
                        cable_name = cableImg.substring(cableImg.lastIndexOf("/")+1,cableImg.
                            lastIndexOf("."));
                        var img = new Image();
                        img.src = cableImg;
                        document.body.appendChild(img);
                        $(img).load(function() {
                                path = "3,3,"+img.width+","+img.height;
                                port_a_coor = 3+","+(3 + img.height / 2);
                                port_b_coor = (3 + img.width)+","+(3 + img.height / 2);
                                document.body.removeChild(img);

                                drawCable(cable_id, type, path, port_a, port_b, color,
                                    cableImg, cable_name, transform, port_a_coor, port_b_coor)
                                    ;
                        });
                }
        } else {
                var cableImg = cable_name;
                cable_name = cableImg.substring(cableImg.lastIndexOf("/")+1,cableImg.
                    lastIndexOf("."));
                drawCable(cable_id, type, path, port_a, port_b, color, cableImg, cable_name,
                    transform, port_a_coor, port_b_coor);
        }

        return false;
}

function toggleCableSelect(line) {
        var pts, i, handles = new Array();

        deselectAllCables();
        deselectAllDevices();
        activate("cables");

        line.toFront();

        paper.getById(line.data("portA")).hide();
        paper.getById(line.data("rectA")).hide();
        paper.getById(line.data("portB")).hide();
        paper.getById(line.data("rectB")).hide();

        if(line.data("type") < 2) {
                pts = line.attr("path");
                $("#line_"+line.id+"_color").show();
        } else {
                pts = $("#line_"+line.id+"_path").val().split(",");

                for(i = 0; i < pts.length; i++)
                        pts[i] = parseFloat(pts[i]);

                line.animate({transform: "r0"}, 100);

                $("#rotator_"+line.id).show();
                //$("#line_"+line.id+"_transform").show();
                $("#line_"+line.id+"_device_name").show();
                $("#line_"+line.id+"_port_a_coor").show();
                $("#line_"+line.id+"_port_b_coor").show();
        }

        $("#delete_cable_"+line.id).show();
        $("#line_"+line.id+"_id").show();
        $("#line_"+line.id+"_type").show();
        $("#line_"+line.id+"_path").show();
        $("#line_"+line.id+"_port_a").show();
```

```
        $("#line_"+line.id+"_port_b").show();

        if(line.data("type") == 0) {
                handles[0] = paper.circle(pts[0][1], pts[0][2], 3).data("id", 0);
                handles[1] = paper.circle(pts[1][1], pts[1][2], 3).data("id", 1);
                handles[2] = paper.circle((pts[0][1] + pts[1][1]) / 2,(pts[0][2] + pts[1][2])
                    / 2, 3).data("id", 2);
        } else if(line.data("type") == 1) {
                handles[0] = paper.circle(pts[0][1], pts[0][2], 3).data("id", 0);
                handles[1] = paper.circle(pts[1][1], pts[1][2], 3).data("id", 1);
                handles[2] = paper.circle(pts[1][3], pts[1][4], 3).data("id", 3);
                handles[3] = paper.circle(pts[1][5], pts[1][6], 3).data("id", 5);

                //paper.path("M "+pts[0][1]+" "+pts[0][2]+" L "+pts[1][3]+" "+pts[1][6]);
                //paper.path("M "+pts[1][5]+" "+pts[1][6]+" L "+pts[1][1]+" "+pts[0][2]);

                var inter = Raphael.pathIntersection("M "+pts[0][1]+" "+pts[0][2]+" L "+pts
                    [1][3]+" "+pts[1][6],
                        "M "+pts[1][5]+" "+pts[1][6]+" L "+pts[1][1]+" "+pts[0][2]);

                handles[4] = paper.circle(inter[0].x, inter[0].y, 3).data("id", 7);
        } else {
                var x = pts[0], y = pts[1], w = pts[2], h = pts[3], r = pts[3] / 2;

                handles[0] = paper.circle(x, y, 3).data("id", 0);
                handles[1] = paper.circle((2 * x + w) / 2, y, 3).data("id", 1);
                handles[2] = paper.circle((x + w), y, 3).data("id", 2);
                handles[3] = paper.circle((x + w), (2 * y + h) / 2, 3).data("id", 3);
                handles[4] = paper.circle((x + w), (y + h), 3).data("id", 4);
                handles[5] = paper.circle((2 * x + w) / 2, (y + h), 3).data("id", 5);
                handles[6] = paper.circle(x, (y + h), 3).data("id", 6);
                handles[7] = paper.circle(x, (2 * y + h) / 2, 3).data("id", 7);

                var inter = Raphael.pathIntersection("M "+x+" "+y+" L "+(x + w)+" "+(y + h), "
                    M "+x+" "+(y + h)+" L "+(x + w)+" "+y);

                handles[8] = paper.circle(inter[0].x, inter[0].y, 3).data("id", 8);
        }

        for(i = 0; i < handles.length; i++) {
                handles[i].attr("fill","#000000")
                .drag(move(line, handles), start(handles), stop(line))
                .mouseover(function(event) {
                        this.animate({r: 6}, 100);
                })
                .mouseout(function(event) {
                        this.animate({r: 3}, 100);
                });
        }
}

function move(line, handles) {
        var pts, i;

        if(line.data("type") < 2)
                pts = line.attr("path");

        return function(dx, dy){
                if(line.data("type") > 1 && this.data("id") % 2 > 0) {
                        if(this.data("id") == 1 || this.data("id") == 5)
                                this.attr({
                                        cy: this.oy + dy
                                });
                        else
                                this.attr({
                                        cx: this.ox + dx
                                });
                } else {
                        this.attr({
                                cx: this.ox + dx,
                                cy: this.oy + dy
                        });
                }

                if(this.id == handles[handles.length - 1].id)
                        for(i = 0; i < handles.length - 1; i++)
                                handles[i].attr({
                                        cx: handles[i].ox + dx,
                                        cy: handles[i].oy + dy
                                });

                if(line.data("type") == 0) {
                        if(this.data("id") == 2) {
                                for(i = 0; i < handles.length - 1; i++) {
                                        pts[i][1] = handles[i].attr("cx");
                                        pts[i][2] = handles[i].attr("cy");
                                }
                        } else {
                                pts[this.data("id")][1] = this.attr("cx");
                                pts[this.data("id")][2] = this.attr("cy");

                                handles[2].attr({
                                        cx: (pts[0][1] + pts[1][1]) / 2,
                                        cy: (pts[0][2] + pts[1][2]) / 2
                                });
                        }
                } else if(line.data("type") == 1) {
                        if(this.data("id") == 7) {
                                pts[0][1] = handles[0].attr("cx");
```

```
                pts[0][2] = handles[0].attr("cy");

                for(i = 1; i < handles.length - 1; i++) {
                        pts[1][handles[i].data("id")] = handles[i].attr("cx");
                        pts[1][handles[i].data("id") + 1] = handles[i].attr("
                             cy");
                }
        } else if(this.data("id") == 0) {
                handles[3].attr({
                        cx: handles[3].ox + dx,
                        cy: handles[3].oy - dy
                });

                pts[0][1] = handles[0].attr("cx");
                pts[0][2] = handles[0].attr("cy");
                pts[1][5] = handles[3].attr("cx");
                pts[1][6] = handles[3].attr("cy");
        } else if(this.data("id") == 1) {
                handles[2].attr({
                        cx: handles[2].ox + dx,
                        cy: handles[2].oy - dy
                });

                pts[1][1] = handles[1].attr("cx");
                pts[1][2] = handles[1].attr("cy");
                pts[1][3] = handles[2].attr("cx");
                pts[1][4] = handles[2].attr("cy");
        } else if(this.data("id") == 3) {
                handles[1].attr({
                        cx: handles[1].ox + dx,
                        cy: handles[1].oy - dy
                });

                pts[1][1] = handles[1].attr("cx");
                pts[1][2] = handles[1].attr("cy");
                pts[1][3] = handles[2].attr("cx");
                pts[1][4] = handles[2].attr("cy");
        } else if(this.data("id") == 5) {
                handles[0].attr({
                        cx: handles[0].ox + dx,
                        cy: handles[0].oy - dy
                });

                pts[0][1] = handles[0].attr("cx");
                pts[0][2] = handles[0].attr("cy");
                pts[1][5] = handles[3].attr("cx");
                pts[1][6] = handles[3].attr("cy");
        }

        var inter = Raphael.pathIntersection("M "+pts[0][1]+" "+pts[0][2]+" L
             "+pts[1][3]+" "+pts[1][6],
        "M "+pts[1][5]+" "+pts[1][6]+" L "+pts[1][1]+" "+pts[0][2]);

        handles[4].attr({
                cx: inter[0].x,
                cy: inter[0].y
        });
} else {
        if(this.data("id") == 8) {
                line.attr({
                        x: handles[0].attr("cx"),
                        y: handles[0].attr("cy")
                });
        } else if(this.data("id") == 0) {
                handles[1].attr({
                        cx: handles[1].ox + (dx / 2),
                        cy: handles[1].oy + dy
                });
                handles[2].attr({
                        cy: handles[2].oy + dy
                });
                handles[3].attr({
                        cy: handles[3].oy + (dy / 2)
                });
                handles[5].attr({
                        cx: handles[5].ox + (dx / 2)
                });
                handles[6].attr({
                        cx: handles[6].ox + dx
                });
                handles[7].attr({
                        cx: handles[7].ox + dx,
                        cy: handles[7].oy + (dy / 2)
                });
                console.log(handles[2].attr("cx")+" - "+this.attr("cx")+" =
                     "+(handles[2].attr("cx") - this.attr("cx"))+"\n"+
                        handles[6].attr("cy")+" - "+this.attr("cy")+" = "+(
                             handles[6].attr("cy") - this.attr("cy")));
                console.log("END");
        } else if(this.data("id") == 2) {
                handles[0].attr({
                        cy: handles[0].oy + dy
                });
                handles[1].attr({
                        cx: handles[1].ox + (dx / 2),
                        cy: handles[1].oy + dy
                });
                handles[3].attr({
                        cx: handles[3].ox + dx,
```

```
                cy: handles[3].oy + (dy / 2)
        });
        handles[4].attr({
                cx: handles[3].ox + dx
        });
        handles[5].attr({
                cx: handles[5].ox + (dx / 2)
        });
        handles[7].attr({
                cy: handles[7].oy + (dy / 2)
        });
} else if(this.data("id") == 4) {
        handles[1].attr({
                cx: handles[1].ox + (dx / 2)
        });
        handles[2].attr({
                cx: handles[2].ox + dx
        });
        handles[3].attr({
                cx: handles[3].ox + dx,
                cy: handles[3].oy + (dy / 2)
        });
        handles[5].attr({
                cx: handles[5].ox + (dx / 2),
                cy: handles[5].oy + dy
        });
        handles[6].attr({
                cy: handles[6].oy + dy
        });
        handles[7].attr({
                cy: handles[7].oy + (dy / 2)
        });
} else if(this.data("id") == 6) {
        handles[0].attr({
                cx: handles[0].ox + dx
        });
        handles[1].attr({
                cx: handles[1].ox + (dx / 2)
        });
        handles[3].attr({
                cy: handles[3].oy + (dy / 2)
        });
        handles[4].attr({
                cy: handles[4].oy + dy
        });
        handles[5].attr({
                cx: handles[5].ox + (dx / 2),
                cy: handles[5].oy + dy
        });
        handles[7].attr({
                cx: handles[7].ox + dx,
                cy: handles[7].oy + (dy / 2)
        });
} else if(this.data("id") % 2 > 0) {
        var a = this.data("id") - 2, b = this.data("id") + 2,
        c = this.data("id") - 1, d = this.data("id") + 1,
        max = handles.length - 1;

        a = a < 0 ? max + a : a;
        b = b > max ? b - max : b;
        d = d % max;

        if(this.data("id") == 1 || this.data("id") == 5) {
                handles[c].attr({
                        cy: handles[c].oy + dy
                });
                handles[d].attr({
                        cy: handles[d].oy + dy
                });
                handles[a].attr({
                        cy: handles[a].oy + (dy / 2)
                });
                handles[b].attr({
                        cy: handles[b].oy + (dy / 2)
                });
        } else if(this.data("id") == 3 || this.data("id") == 7) {
                handles[c].attr({
                        cx: handles[c].ox + dx
                });
                handles[d].attr({
                        cx: handles[d].ox + dx
                });
                handles[a].attr({
                        cx: handles[a].ox + (dx / 2)
                });
                handles[b].attr({
                        cx: handles[b].ox + (dx / 2)
                });
        }
}

line.attr({
        x: handles[0].attr("cx"),
        y: handles[0].attr("cy"),
        width: Math.abs(handles[0].attr("cx") - handles[4].attr("cx"))
            ,
        height: Math.abs(handles[0].attr("cy") - handles[4].attr("cy")
            )
});
```

```
                        var inter = Raphael.pathIntersection("M "+line.attr("x")+" "+line.attr
                            ("y")+
                                " L "+(line.attr("x") + line.attr("width"))+" "+(line.attr("y
                                    ") + line.attr("height")),
                                "M "+line.attr("x")+" "+(line.attr("y") + line.attr("height"))
                                    +
                                " L "+(line.attr("x") + line.attr("width"))+" "+line.attr("y")
                                    );

                        handles[8].attr({
                                cx: inter[0].x,
                                cy: inter[0].y
                        });
                }

                if(line.data("type") < 2)
                        line.attr("path", pts.join(" "));

        };
}

function start(handles) {
        var i;

        return function() {
                this.ox = this.attr("cx");
                this.oy = this.attr("cy");

                for(i = 0; i < handles.length - 1; i++) {
                        handles[i].ox = handles[i].attr("cx");
                        handles[i].oy = handles[i].attr("cy");
                }
        };
}

function stop(line) {
        var pts,
        portA = paper.getById(line.data("portA")),
        portB = paper.getById(line.data("portB")),
        rectA = paper.getById(line.data("rectA")),
        rectB = paper.getById(line.data("rectB"));

        if(line.data("type") < 2)
                pts = line.attr("path");

        return function() {
                if(line.data("type") < 2) {
                        $("#line_"+line.id+"_path").val(pts.join(" "));

                        if(line.data("type") == 0) {
                                portA.attr({
                                        x: pts[0][1],
                                        y: pts[0][2]
                                });

                                portB.attr({
                                        x: pts[1][1],
                                        y: pts[1][2]
                                });
                        } else if(line.data("type") == 1) {
                                portA.attr({
                                        x: pts[0][1],
                                        y: pts[0][2]
                                });

                                portB.attr({
                                        x: pts[1][5],
                                        y: pts[1][6]
                                });
                        }
                } else {
                        var circle = paper.path("M "+(line.attr("x") + line.attr("width") / 2)
                            +" "+(line.attr("y") + line.attr("height") / 2)+
                                " m "+(-line.attr("width") / 2)+", 0"+
                                " a "+(line.attr("width") / 2)+", "+(line.attr("width") / 2)+"
                                    0 1, 0 "+line.attr("width")+", 0"+
                                " a "+(line.attr("width") / 2)+", "+(line.attr("width") / 2)+"
                                    0 1, 0 "+(-line.attr("width"))+", 0")
                        .attr("opacity", 0);

                        $("#line_"+line.id+"_path").val(
                                line.attr("x")+","+
                                line.attr("y")+","+
                                line.attr("width")+","+
                                line.attr("height")
                        );

                        if($("#line_"+line.id+"_transform").val() > 0)
                                portA.attr({
                                        x: circle.getPointAtLength(circle.getTotalLength() *
                                            ((360 - $("#line_"+line.id+"_transform").val()) /
                                            360)).x,
                                        y: circle.getPointAtLength(circle.getTotalLength() *
                                            ((360 - $("#line_"+line.id+"_transform").val()) /
                                            360)).y
                                });
                        else
                                portA.attr({
```

```
                                        x: circle.getPointAtLength(circle.getTotalLength() *
                                             ((0 - $("#line_"+line.id+"_transform").val()) /
                                             360)).x,
                                        y: circle.getPointAtLength(circle.getTotalLength() *
                                             ((0 - $("#line_"+line.id+"_transform").val()) /
                                             360)).y
                                });

                        portB.attr({
                                x: circle.getPointAtLength(circle.getTotalLength() * ((180 - $
                                     ("#line_"+line.id+"_transform").val()) / 360)).x,
                                y: circle.getPointAtLength(circle.getTotalLength() * ((180 - $
                                     ("#line_"+line.id+"_transform").val()) / 360)).y
                        });

                        $("#line_"+line.id+"_port_a_coor").val(portA.attr("x")+","+portA.attr
                             ("y"));
                        $("#line_"+line.id+"_port_b_coor").val(portB.attr("x")+","+portB.attr
                             ("y"));

                        circle.remove();
                }

                var boxA = portA.getBBox();

                rectA.attr({
                        x: boxA.x - 2,
                        y: boxA.y - 2,
                        width: boxA.width + 4,
                        height: boxA.height + 4
                });

                var boxB = portB.getBBox();

                rectB.attr({
                        x: boxB.x - 2,
                        y: boxB.y - 2,
                        width: boxB.width + 4,
                        height: boxB.height + 4
                });

                $("#line_"+line.id+"_rect_a").val(rectA.attr("x")+","+rectA.attr("y")+","+
                     rectA.attr("width")+","+rectA.attr("height"));
                $("#line_"+line.id+"_rect_b").val(rectB.attr("x")+","+rectB.attr("y")+","+
                     rectB.attr("width")+","+rectB.attr("height"));
        };
}

function removeHandles() {
        var index, handles = new Array(), lines = new Array();

        paper.forEach(function (el) {
                if(el.type == "circle") {
                        handles[handles.length] = el.id;
                } else if(el.data("type") != null && (el.type == "path" || el.type == "image")
                     ) {
                        lines[lines.length] = el.id;

                        if(el.type == "image")
                                el.animate({transform: "r"+$("#line_"+el.id+"_transform").val
                                     ()}, 100);
                }
        });

        for(index = 0; index < handles.length; index++)
                paper.getById(handles[index]).remove();

        for(index = 0; index < lines.length; index++) {
                var line = paper.getById(lines[index]),
                portA = paper.getById(line.data("portA")),
                rectA = paper.getById(line.data("rectA")),
                portB = paper.getById(line.data("portB")),
                rectB = paper.getById(line.data("rectB"));

                line.toBack();

                if(portA.attr("text") != null &&
                portA.attr("text") != "" &&
                portA.attr("text") != "port a") {
                        portA.show();
                        rectA.show();
                }

                if(portB.attr("text") != null &&
                portB.attr("text") != "" &&
                portB.attr("text") != "port b") {
                        portB.show();
                        rectB.show();
                }
        }
}

function deselectAllCables() {
        removeHandles();

        $("#cables-tab input").each(function() {
                if(this.type == "color" || this.type == "text" || this.value == "Delete")
                        $(this).hide();
        });
```

```
        $("#cables-tab .ui-slider").each(function() {
                $(this).hide();
        });
}

function deleteCable(line) {
        $("#delete_cable_"+line.id).remove();
        $("#line_"+line.id+"_id").remove();
        $("#line_"+line.id+"_type").remove();
        $("#line_"+line.id+"_path").remove();
        $("#line_"+line.id+"_port_a").remove();
        $("#line_"+line.id+"_port_b").remove();
        $("#line_"+line.id+"_rect_a").remove();
        $("#line_"+line.id+"_rect_b").remove();

        if(line.data("type") < 2) {
                $("#line_"+line.id+"_color").remove();
        } else {
                $("#rotator_"+line.id).remove();
                $("#line_"+line.id+"_transform").remove();
                $("#line_"+line.id+"_device_name").remove();
                $("#line_"+line.id+"_port_a_coor").remove();
                $("#line_"+line.id+"_port_b_coor").remove();
        }

        paper.getById(line.id).remove();
        removeHandles();
}
```

Listing 3: Main JS

```
var excess;

function drawStage() {
        var width = 1024, height = 786, padding = 40, margin = 10;
        excess = window.innerWidth - ($("#control-div").outerWidth() + width + padding);

        if(excess > 0) {
                $("#control-div").width($("#control-div").width() + excess);
                $("#tab-contents div select, #tab-contents div input[type='button'], "+
                "#tab-contents div input[type='submit'], #tab-contents div input[type='text'],
                        "+
                "#tab-contents div input[type='color'], #control-div input[type='button'], "+
                "#control-div input[type='submit']")
                .css("margin-left", (margin + excess) / 2);
        }

        $("#topology-div").height(height + padding);
        $("#topology-div").width(width + $("#control-div").outerWidth() + padding);

        $("#canvas-div").height(height);
        $("#canvas-div").width(width);
        $("#canvas-div").css("left", $("#control-div").outerWidth());

        $("#topology-canvas").height(height);
        $("#topology-canvas").width(width);

        var canvas = Raphael(document.getElementById("topology-canvas"), $("#topology-canvas")
             .width(), $("#topology-canvas").height());

        for(var x = 50; x < canvas.width; x += 50)
                for(var y = 50; y < canvas.height; y += 50)
                        canvas.circle(x, y, 0.5);

        cableInit();
}
```

Listing 4: Pod JS

```
function configure() {
        var x = ($(window).width() - $("#topology-div").width()) / 2;
        $("#topology-div").css("left", x < 0 ? 0 : x);

        $("#close-button").css({
                "position": "absolute",
                "top": "0px",
                "left": ($("#canvas-div").width() + 20)+"px",
                "z-index": 5
        }).click(function() {
                window.location.assign(url);
        });

        grayscale($(".grayscale"));
}

function toggleSelected(topologyIndex, podIndex) {
        if($("#"+topologyIndex).hasClass("selected")) {
                $("#"+topologyIndex).removeClass("selected");

                var deletedTopologyID = document.createElement("input");
                $(deletedTopologyID).attr({
                        type: "hidden",
                        id: "deleted_"+topologyIndex+"_topology_id",
                        name: "deleted["+topologyIndex+"][topology_id]",
                        value: $("#pods_"+topologyIndex+"_topology_id").val()
```

```
                });

                var deletedPodID = document.createElement("input");
                $(deletedPodID).attr({
                        type: "hidden",
                        id: "deleted_"+topologyIndex+"_pod_id",
                        name: "deleted["+topologyIndex+"][pod_id]",
                        value: $("#pods_"+topologyIndex+"_pod_id").val()
                });

                $("#"+topologyIndex).append(deletedTopologyID);
                $("#"+topologyIndex).append(deletedPodID);

                $("#pods_"+topologyIndex+"_topology_id").remove();
                $("#pods_"+topologyIndex+"_pod_id").remove();
        } else {
                $("#"+topologyIndex).addClass("selected");

                var topologyID = document.createElement("input");
                $(topologyID).attr({
                        type: "hidden",
                        id: "pods_"+topologyIndex+"_topology_id",
                        name: "pods["+topologyIndex+"][topology_id]",
                        value: topologyIndex
                });

                var podID = document.createElement("input");
                $(podID).attr({
                        type: "hidden",
                        id: "pods_"+topologyIndex+"_pod_id",
                        name: "pods["+topologyIndex+"][pod_id]",
                        value: podIndex
                });

                $("#"+topologyIndex).append(topologyID);
                $("#"+topologyIndex).append(podID);

                if($("#deleted_"+topologyIndex+"_topology_id").attr("type") == "hidden" && $
                     ("#deleted_"+topologyIndex+"_pod_id").attr("type") == "hidden") {
                        $("#deleted_"+topologyIndex+"_topology_id").remove();
                        $("#deleted_"+topologyIndex+"_pod_id").remove();
                }
        }
}

function save(event) {
        window.onbeforeunload = function() {null}

        var unselected = 0, devices = 0;

        $("#canvas-div div").each(function(event) {
                devices++;
                if(!$(this).hasClass("selected"))
                        unselected++;
        });

        if(unselected == devices) {
                alert("No devices selected");
                window.onbeforeunload = function() {
                        return "Everything will be lost if you don't save your work."
                }
                event.preventDefault();
        }
}

function connect(id) {
        var request = new XMLHttpRequest();
        request.onreadystatechange = function() {
                if(request.readyState == 4 && request.status == 200) {
                        var object = new ActiveXObject("Scripting.FileSystemObject"), app,
                             params, command = request.responseText.split("|");
                        app = command[0];
                        params = command[1];

                        if(app == "putty")
                                if(object.FileExists("C:\\Program Files (x86)\\PuTTY\\putty.
                                     exe"))
                                        app = "C:\\Program Files (x86)\\PuTTY\\"+app;
                                else if(object.FileExists("C:\\Program Files\\PuTTY\\putty.exe
                                     "))
                                        app = "C:\\Program Files\\PuTTY\\"+app;
                                else
                                   alert("Please install PuTTY...");

                        var shell = new ActiveXObject("Shell.Application");
                        shell.ShellExecute(app, params, "", "open", "1");
                }
        }
        request.open("GET", url+"/index.php?r=class/connect&id="+id, true);
        request.send();
}
```

Listing 5: Topology JS

```
var devices = 0, deleted = 0, src;

function getSubjects(id, url) {
        var request = new XMLHttpRequest();
```

```
        request.onreadystatechange = function() {
                if(request.readyState == 4 && request.status == 200) {
                        $("#StudentForm_subj").html(request.responseText);
                }
        }
        request.open("GET", url+"&employee_id="+id, true);
        request.send();
}

function add(topology_id, device_name, src, left, top, width, height, type, ipAddr, host_name,
      display_name) {
        if(topology_id == null) {
                src = $("#TopologyModel_device").val();
                device_name = src.substring(src.lastIndexOf("/")+1,src.lastIndexOf("."));
                left = top = 20;
                ipAddr = "ip address";
                host_name = "host name";
                display_name = "display name";
        }

        if(src != 0) {
                var container = document.getElementById("container");

                var div = document.createElement("div");
                $(div).attr({
                        id: "draggable"+devices,
                        class: "draggable",
                        align: "center",
                        style: "position: absolute; top: "+top+"px; left: "+left+"px;"
                })
                .mousedown(function() {
                        toggleSelected(this);
                });

                var topologyid = document.createElement("input");
                $(topologyid).attr({
                        type: "hidden",
                        id: "draggable_"+devices+"_id",
                        name: "draggable["+devices+"][id]",
                        value: topology_id
                });

                var img = new Image();
                $(img).attr({
                        id: "img"+devices,
                        src: src
                });

                if(width != null || height !=null) {
                        img.width = width;
                        img.height = height;
                }

                $(img).on("load", function() {
                        var index = img.id.charAt(img.id.length - 1);

                        $("#draggable"+index).height(img.height);

                        $("#draggable_"+index+"_width").val(img.width);
                        $("#draggable_"+index+"_height").val(img.height);

                        $("#draggable"+index).resizable({
                                containment: container,
                                minWidth: 50,
                                minHeight: 50,
                                aspectRatio: img.width / img.height,
                                grid: 10,
                                handles: "n,ne,nw,e,w,s,se,sw",
                                resize: function() {
                                        $(img).attr({
                                                width: $(this).width(),
                                                height: $(this).height()
                                        });
                                },
                                stop: function(event, ui) {

                                        $("#draggable_"+index+"_x").val(this.offsetLeft);
                                        $("#draggable_"+index+"_y").val(this.offsetTop);

                                        if($("#draggable_"+index+"_x").val() < 20) {
                                                $("#draggable"+index).css("left", 20);
                                                $("#draggable_"+index+"_x").val(20);
                                        }

                                        if($("#draggable_"+index+"_y").val() < 20) {
                                                $("#draggable"+index).css("top", 20);
                                                $("#draggable_"+index+"_y").val(20);
                                        }

                                        $("#draggable_"+index+"_width").val($(this).width());
                                        $("#draggable_"+index+"_height").val($(this).height())
                                                ;
                                }
                        });
                });

                $(div).draggable({
                        containment: container,
                        grid: [10, 10],
```

```
        cursor: "move",
        stop: function(event, ui) {
                var index = this.id.substr(9, this.id.length);
                $("#draggable_"+index+"_x").val(this.offsetLeft);
                $("#draggable_"+index+"_y").val(this.offsetTop);
        },
});

var imgform = document.createElement("input");
$(imgform).attr({
        type: "hidden",
        id: "draggable_"+devices+"_device_name",
        name: "draggable["+devices+"][device_name]",
        value: device_name
});

var leftOffset = document.createElement("input");
$(leftOffset).attr({
        type: "hidden",
        id: "draggable_"+devices+"_x",
        name: "draggable["+devices+"][x]",
        value: left
});

var topOffset = document.createElement("input");
$(topOffset).attr({
        type: "hidden",
        id: "draggable_"+devices+"_y",
        name: "draggable["+devices+"][y]",
        value: top
});

var width = document.createElement("input");
$(width).attr({
        type: "hidden",
        id: "draggable_"+devices+"_width",
        name: "draggable["+devices+"][width]"
});

var height = document.createElement("input");
$(height).attr({
        type: "hidden",
        id: "draggable_"+devices+"_height",
        name: "draggable["+devices+"][height]"
});

var del = document.createElement("input");
$(del).attr({
        type: "button",
        id: "delete"+devices,
        class: "button",
        value: "Delete",
        style: "margin-left: "+((10 + excess) / 2)+"px"
})
.click(function() {
        rem(div.id);
});

$("#devices-tab").append(del);

var selectul = document.createElement("ul");
$(selectul).attr({
        type: "none",
        id: "ul"+devices,
        style: "margin: 0px; padding: 0px",
});

var select = document.createElement("select");
$(select).attr({
        id: "draggable_"+devices+"_type",
        name: "draggable["+devices+"][type]",
        style: "padding: 1px; margin-left: "+((10 + excess) / 2)+"px"
});
select.add(new Option("Choose Device Type", null), null);
select.add(new Option("Terminal Server", 1), null);
select.add(new Option("Router/Switch", 2), null);
select.add(new Option("PC/Server", 3), null);
$(select).change(function() {
        var index = this.id.substring(this.id.indexOf("_") + 1,this.id.
             lastIndexOf("_"));
        $("#draggable_"+index+"_ip").val("ip address");
        $("#draggable_"+index+"_host").val("host name");
        $("#draggable_"+index+"_display").val("display name");

        if(this.value == 1)
                $("#node"+index).text("TS");

        view(this.value, index);
});
var selectli = document.createElement("li");
$(selectli).append(select);

var ip = document.createElement("input");
$(ip).attr({
        type: "text",
        id: "draggable_"+devices+"_ip",
        name: "draggable["+devices+"][ip]",
        size: 17,
        style: "display: none; margin-left: "+((10 + excess) / 2)+"px",
```

```
                        value: ipAddr
                })
                .focus(function() {
                        if($(this).val() == "ip address")
                                $(this).val("");
                })
                .change(function() {
                        $(this).val($(this).val().trim());

                        if($(this).val() == "")
                                $(this).val("ip address");
                });
                var ipli = document.createElement("li");
                $(ipli).append(ip);

                var hostname = document.createElement("input");
                $(hostname).attr({
                        type: "text",
                        id: "draggable_"+devices+"_host",
                        name: "draggable["+devices+"][host]",
                        size: 17,
                        style: "display: none; margin-left: "+((10 + excess) / 2)+"px",
                        value: host_name
                })
                .focus(function() {
                        if($(this).val() == "host name")
                                $(this).val("");
                })
                .change(function() {
                        $(this).val($(this).val().trim());

                        if($(this).val() == "")
                                $(this).val("host name");
                });
                var hostnameli = document.createElement("li");
                $(hostnameli).append(hostname);

                var displayname = document.createElement("input");
                $(displayname).attr({
                        type: "text",
                        id: "draggable_"+devices+"_display",
                        name: "draggable["+devices+"][display]",
                        size: 17,
                        style: "display: none; margin-left: "+((10 + excess) / 2)+"px",
                        value: display_name
                })
                .focus(function() {
                        if($(this).val() == "display name")
                                $(this).val("");
                })
                .change(function() {
                        $(this).val($(this).val().trim());

                        var index = this.id.substring(this.id.indexOf("_") + 1,this.id.
                            lastIndexOf("_"));
                        $("#node"+index).text($(this).val().toUpperCase());

                        if($(this).val() == "")
                                $(this).val("display name");
                });
                var displaynameli = document.createElement("li");
                $(displaynameli).append(displayname);

                $(selectul).append(selectli);
                $(selectul).append(ipli);
                $(selectul).append(hostnameli);
                $(selectul).append(displaynameli);

                $("#devices-tab").append(selectul);

                $(div).append(img);
                $(div).append(imgform);
                $(div).append(leftOffset);
                $(div).append(topOffset);
                $(div).append(width);
                $(div).append(height);
                $(div).append(topologyid);
                $(div).append(document.createElement("br"));

                var span = document.createElement("span");
                $(span).attr("id","node"+devices)
                .append(document.createTextNode($(displayname).val()));

                $(div).append(span);

                $(container).append(div);

                $(select).prop("selectedIndex", type);

                $("#TopologyModel_device").prop("selectedIndex", 0);

                toggleSelected(div);

                devices++;
        }

        return false;
}
```

```
function rem(id) {
        var index = id.charAt(id.length - 1);

        if($("#draggable_"+index+"_id").val() != "") {
                var deletedID = document.createElement("input");
                $(deletedID).attr({
                        type: "hidden",
                        id: "deleted_"+deleted+"_id",
                        name: "deleted["+deleted+"][id]",
                        value: $("#draggable_"+index+"_id").val()
                });

                var deletedType = document.createElement("input");
                $(deletedType).attr({
                        type: "hidden",
                        id: "deleted_"+deleted+"_type",
                        name: "deleted["+deleted+"][type]",
                        value: "device"
                });

                $("#topology-model-form").append(deletedID);
                $("#topology-model-form").append(deletedType);

                deleted++;
        }

        $("#"+id).remove();

        $("#delete"+index).remove();
        $("#ul"+index).remove();

        return false;
}

function toggleSelected(element) {
        deselectAllDevices();
        deselectAllCables();
        activate("devices");

        $(element).addClass("selected");

        var index = element.id.substr(9, element.id.length);
        $("#delete"+index).show();
        $("#ul"+index).show();

        view($("#draggable_"+index+"_type").val(), index);
}

function view(value, index) {
        switch(value) {
                case '2':
                        $("#draggable_"+index+"_ip").hide();
                        $("#draggable_"+index+"_host").show();
                        $("#draggable_"+index+"_display").show();
                        break;
                case '3':
                        $("#draggable_"+index+"_ip").show();
                        $("#draggable_"+index+"_host").hide();
                        $("#draggable_"+index+"_display").show();
                        break;
                default:
                        $("#draggable_"+index+"_ip").hide();
                        $("#draggable_"+index+"_host").hide();
                        $("#draggable_"+index+"_display").hide();
                        break;
        }

        return false;
}

function save(event) {
        window.onbeforeunload = function() {null}

        removeHandles();

        var TSPresent = false, emptyField = null, id = null, invalidIP = false;

        $("#devices-tab ul li select").each(function(event) {
                if(this.value == 1) {
                        TSPresent = true;
                } else if(this.value == 2) {
                        var host = $("#"+this.id.replace("type","host")).val();
                        var display = $("#"+this.id.replace("type","display")).val();
                        if(host == "" || host == "host name") {
                                emptyField = "host";
                                id = this.id;
                        } else if(display == "" || display == "display name") {
                                emptyField = "rs display";
                                id = this.id;
                        }
                } else if(this.value == 3) {
                        var ip = $("#"+this.id.replace("type","ip")).val();
                        var display = $("#"+this.id.replace("type","display")).val();
                        var ipArray = ip.split(".");
                        var i;
                        if(ip == "" || ip == "ip address") {
                                emptyField = "ip";
                                id = this.id;
                        } else if(display == "" || display == "display name") {
```

```
                                emptyField = "ps display";
                                id = this.id;
                        }

                        if(ipArray.length != 4) {
                                invalidIP = true;
                                id = this.id;
                        } else {
                                for(i = 0; i < ipArray.length; i++) {
                                        ipArray[i] = ipArray[i].trim();
                                        if(!$.isNumeric(ipArray[i]) || ipArray[i] < 0 ||
                                             ipArray[i] > 255) {
                                                invalidIP = true;
                                                id = this.id;
                                        }
                                }
                                ip = ipArray.join(".");
                                $("#"+this.id.replace("type","ip")).val(ip);
                        }
                } else {
                        id = this.id;
                }
        });

        if(emptyField != null && id != null) {
                alert("Empty field(s)");
                var index = id.substring(id.indexOf("_") + 1,id.lastIndexOf("_"));
                toggleSelected(document.getElementById("draggable"+index));
                window.onbeforeunload = function() {
                        return "Everything will be lost if you don't save your work."
                }
                event.preventDefault();
        } else if(invalidIP) {
                alert("Invalid IP Address");
                var index = id.substring(id.indexOf("_") + 1,id.lastIndexOf("_"));
                toggleSelected(document.getElementById("draggable"+index));
                window.onbeforeunload = function() {
                        return "Everything will be lost if you don't save your work."
                }
                event.preventDefault();
        } else if(id != null) {
                alert("No Device Type");
                var index = id.substring(id.indexOf("_") + 1,id.lastIndexOf("_"));
                toggleSelected(document.getElementById("draggable"+index));
                window.onbeforeunload = function() {
                        return "Everything will be lost if you don't save your work."
                }
                event.preventDefault();
        } else if(!TSPresent) {
                alert("Terminal Server not found");
                window.onbeforeunload = function() {
                        return "Everything will be lost if you don't save your work."
                }
                event.preventDefault();
        }

        return;
}

function activate(tab) {
        $("#"+tab+"-tab").show().siblings().hide();
        $("#"+tab+"-link").addClass("active").siblings().removeClass("active");
}

function deselectAllDevices() {
        $("#container div").each(function() {
                $(this).removeClass("selected");
                var index = this.id.substr(9, this.id.length);
                $("#delete"+index).hide();
                $("#ul"+index).hide();
                view(null, this.id);
        });
}
```

D. Models

Listing 6: Administrator Setup Form

```
<?php

/**
 * AdminForm class.
 * AdminForm is the data structure for keeping
 * admin configuration form data. It is used by the 'admin' action of 'SetupController'.
 */
class AdminForm extends CFormModel
{
        public $employee_id, $pass, $retype, $first_name, $last_name;

        /**
         * Declares the validation rules.
         * The rules state that employee id, password, retype password, first name, and
             password are required,
         * and password needs to be authenticated.
         */
```

```
        public function rules()
        {
                return array(
                        // employee id, password, retype password, first name, and password
                            are required
                        array('employee_id, pass, retype, first_name, last_name', 'required'),
                        // password needs to be validated
                        array('pass', 'valid'),
                );
        }

        /**
         * Validates the password and retype.
         * This is the 'valid' validator as declared in rules().
         */
        public function valid($attribute, $params)
        {
                if(strlen($this->pass) < 6)
                        $this->addError('pass', 'Password must be at least six (6) characters
                            long.');

                if($this->pass !== $this->retype)
                        $this->addError('pass', 'Passwords did not match.');
        }

        /**
         * Declares attribute labels.
         */
        public function attributeLabels()
        {
                return array(
                        'employee_id'=>'Employee ID',
                        'pass'=>'Password',
                        'retype'=>'Retype Password',
                );
        }
}
```

Listing 7: Cable Model

```
<?php

/**
 * This is the model class for table "cable".
 *
 * The followings are the available columns in table 'cable':
 * @property integer $id
 * @property string $course_code
 * @property string $device_name
 * @property integer $type
 * @property string $path
 * @property string $transform
 * @property string $port_a
 * @property string $port_b
 * @property string $port_a_coor
 * @property string $port_b_coor
 * @property string $color
 *
 * The followings are the available model relations:
 * @property Course $courseCode
 * @property Device $deviceName
 */
class CableModel extends CActiveRecord
{
        /**
         * @return string the associated database table name
         */
        public function tableName()
        {
                return 'cable';
        }

        /**
         * @return array validation rules for model attributes.
         */
        public function rules()
        {
                // NOTE: you should only define rules for those attributes that
                // will receive user inputs.
                return array(
                        array('course_code, type, path', 'required'),
                        array('type', 'numerical', 'integerOnly'=>true),
                        array('course_code, device_name, transform', 'length', 'max'=>64),
                        array('path, port_a_coor, port_b_coor', 'length', 'max'=>128),
                        array('port_a, port_b', 'length', 'max'=>16),
                        array('color', 'length', 'max'=>32),
                        // The following rule is used by search().
                        // @todo Please remove those attributes that should not be searched.
                        array('id, course_code, device_name, type, path, transform, port_a,
                            port_b, port_a_coor, port_b_coor, color', 'safe', 'on'=>'search'),
                );
        }

        /**
         * @return array relational rules.
         */
        public function relations()
        {
```

```
                // NOTE: you may need to adjust the relation name and the related
                // class name for the relations automatically generated below.
                return array(
                        'courseModel' => array(self::BELONGS_TO, 'CourseModel', 'course_code')
                            ,
                        'deviceModel' => array(self::BELONGS_TO, 'DeviceModel', 'device_name')
                            ,
                );
        }

        /**
         * @return array customized attribute labels (name=>label)
         */
        public function attributeLabels()
        {
                return array(
                        'id' => 'ID',
                        'course_code' => 'Course Code',
                        'device_name' => 'Device Name',
                        'type' => 'Type',
                        'path' => 'Path',
                        'transform' => 'Transform',
                        'port_a' => 'Port A',
                        'port_b' => 'Port B',
                        'port_a_coor' => 'Port A Coor',
                        'port_b_coor' => 'Port B Coor',
                        'color' => 'Color',
                );
        }

        /**
         * Retrieves a list of models based on the current search/filter conditions.
         *
         * Typical usecase:
         * - Initialize the model fields with values from filter form.
         * - Execute this method to get CActiveDataProvider instance which will filter
         * models according to data in model fields.
         * - Pass data provider to CGridView, CListView or any similar widget.
         *
         * @return CActiveDataProvider the data provider that can return the models
         * based on the search/filter conditions.
         */
        public function search()
        {
                // @todo Please modify the following code to remove attributes that should not
                        be searched.

                $criteria=new CDbCriteria;

                $criteria->compare('id',$this->id);
                $criteria->compare('course_code',$this->course_code,true);
                $criteria->compare('device_name',$this->device_name,true);
                $criteria->compare('type',$this->type);
                $criteria->compare('path',$this->path,true);
                $criteria->compare('transform',$this->transform,true);
                $criteria->compare('port_a',$this->port_a,true);
                $criteria->compare('port_b',$this->port_b,true);
                $criteria->compare('port_a_coor',$this->port_a_coor,true);
                $criteria->compare('port_b_coor',$this->port_b_coor,true);
                $criteria->compare('color',$this->color,true);

                return new CActiveDataProvider($this, array(
                        'criteria'=>$criteria,
                ));
        }

        /**
         * Returns the static model of the specified AR class.
         * Please note that you should have this exact method in all your CActiveRecord
            descendants!
         * @param string $className active record class name.
         * @return CableModel the static model class
         */
        public static function model($className=__CLASS__)
        {
                return parent::model($className);
        }
}
```

Listing 8: Change Password Form

```
<?php

/**
 * ChangePasswordForm class.
 * ChangePasswordForm is the data structure for keeping
 * user change password form data. It is used by the 'changePass' action of 'UserController'.
 */
class ChangePasswordForm extends CFormModel
{
        public $currentPass, $newPass, $retype;

        /**
         * Declares the validation rules.
         * The rules state that username and password are required,
         * and password needs to be authenticated.
         */
        public function rules()
```

```
        {
                return array(
                        // current password, new password, and  retype password are required
                        array('currentPass, newPass, retype', 'required'),
                        // current password needs to be authenticated
                        array('currentPass', 'authenticate'),
                        // new password needs to be validated
                        array('newPass', 'valid'),

                );
        }

        /**
         * Authenticates the current password.
         * This is the 'authenticate' validator as declared in rules().
         */
        public function authenticate($attribute, $params)
        {
                $user = UserModel::model()->findByPk(Yii::app()->user->id);

                if(!password_verify($this->currentPass, $user->pass))
                        $this->addError('currentPass', 'Invalid Password.');
        }

        /**
         * Validates the new password.
         * This is the 'authenticate' validator as declared in rules().
         */
        public function valid($attribute, $params)
        {
                if(strlen($this->newPass) < 6)
                        $this->addError('newPass', 'Password must be at least six (6)
                             characters long.');

                if($this->retype !== $this->newPass)
                        $this->addError('retype', 'Passwords did not match');
        }

        /**
         * Declares attribute labels.
         */
        public function attributeLabels()
        {
                return array(
                        'currentPass'=>'Current Password',
                        'newPass'=>'New Password',
                        'retype'=>'Retype New Password',
                );
        }
}
```

Listing 9: Class Model

```
<?php

/**
 * This is the model class for table "class".
 *
 * The followings are the available columns in table 'class':
 * @property integer $id
 * @property string $course_code
 * @property string $pass
 * @property string $prof
 * @property string $ts_ip
 * @property string $ts_user
 * @property string $ts_pass
 * @property integer $num_of_students
 * @property string $has_pods
 * @property string $exp
 *
 * The followings are the available model relations:
 * @property User $professor
 * @property Course $courseCode
 * @property Pod[] $pods
 */
class ClassModel extends CActiveRecord
{
        /**
         * @return string the associated database table name
         */
        public function tableName()
        {
                return 'class';
        }

        /**
         * @return array validation rules for model attributes.
         */
        public function rules()
        {
                // NOTE: you should only define rules for those attributes that
                // will receive user inputs.
                return array(
                        array('course_code, pass, prof, ts_ip, ts_user, ts_pass,
                             num_of_students, exp', 'required'),
                        array('num_of_students', 'numerical', 'integerOnly'=>true),
                        array('course_code', 'length', 'max'=>64),
                        array('pass, prof, ts_ip, ts_user, ts_pass', 'length', 'max'=>32),
```

```
                array('has_pods', 'length', 'max'=>3),
                array('course_code', 'nonZero'),
                array('ts_ip', 'validIP'),
                array('exp', 'future'),
                // The following rule is used by search().
                // @todo Please remove those attributes that should not be searched.
                array('id, course_code, pass, prof, ts_ip, ts_user, ts_pass,
                    num_of_students, has_pods, exp', 'safe', 'on'=>'search'),
        );
}

/**
 * Checks whether the course code is empty.
 * This is the 'nonZero' validator as declared in rules().
 */
public function nonZero($attribute, $params)
{
        if($this->course_code === '0')
                $this->addError('course_code', 'Course Code cannot be blank');
}

/**
 * Checks whether the terminal server IP address is valid.
 * This is the 'validIP' validator as declared in rules().
 */
public function validIP($attribute, $params)
{
        $ipaddr = explode(".", $this->ts_ip);
        if(count($ipaddr) != 4) {
                $this->addError('ts_ip', 'Invalid IP Address length');
        } else {
                foreach($ipaddr as $ip) {
                        $ip = trim($ip);
                        if(!is_numeric($ip) || $ip < 0 || $ip > 255)
                                $this->addError('ts_ip', 'Invalid IP Address');
                        else
                                $ip = intval($ip);
                }

                $this->ts_ip = implode(".", $ipaddr);
        }
}

/**
 * Checks whether the Expiration Date is set in the future.
 * This is the 'future' validator as declared in rules().
 */
public function future($attribute, $params)
{
        if(date($this->exp) < date("Y-m-d"))
                $this->addError('exp', 'Expiration Date must be set in the future');
}

/**
 * @return array relational rules.
 */
public function relations()
{
        // NOTE: you may need to adjust the relation name and the related
        // class name for the relations automatically generated below.
        return array(
                'professor' => array(self::BELONGS_TO, 'UserModel', 'prof'),
                'course' => array(self::BELONGS_TO, 'CourseModel', 'course_code'),
                'pods' => array(self::HAS_MANY, 'PodModel', 'class_id'),
        );
}

/**
 * @return array customized attribute labels (name=>label)
 */
public function attributeLabels()
{
        return array(
                'id' => 'ID',
                'course_code' => 'Course Code',
                'pass' => 'Password',
                'prof' => 'Professor',
                'ts_ip' => 'Terminal Server IP Address',
                'ts_user' => 'Terminal Server Username',
                'ts_pass' => 'Terminal Server Password',
                'num_of_students' => 'Number Of Students',
                'has_pods' => 'Has Pods',
                'exp' => 'Expiration Date',
        );
}

/**
 * Retrieves a list of models based on the current search/filter conditions.
 *
 * Typical usecase:
 * - Initialize the model fields with values from filter form.
 * - Execute this method to get CActiveDataProvider instance which will filter
 * models according to data in model fields.
 * - Pass data provider to CGridView, CListView or any similar widget.
 *
 * @return CActiveDataProvider the data provider that can return the models
 * based on the search/filter conditions.
 */
public function search()
```

```
        {
                // @todo Please modify the following code to remove attributes that should not
                    be searched.

                $criteria=new CDbCriteria;

                $criteria->compare('id',$this->id);
                $criteria->compare('course_code',$this->course_code,true);
                $criteria->compare('pass',$this->pass,true);
                $criteria->compare('prof',Yii::app()->user->id,true);
                $criteria->compare('ts_ip',$this->ts_ip,true);
                $criteria->compare('ts_user',$this->ts_user,true);
                $criteria->compare('ts_pass',$this->ts_pass,true);
                $criteria->compare('num_of_students',$this->num_of_students);
                $criteria->compare('has_pods',$this->has_pods,true);
                $criteria->compare('exp',$this->exp,true);

                return new CActiveDataProvider($this, array(
                        'criteria'=>$criteria,
                ));
        }

        /**
         * Returns the static model of the specified AR class.
         * Please note that you should have this exact method in all your CActiveRecord
            descendants!
         * @param string $className active record class name.
         * @return ClassModel the static model class
         */
        public static function model($className=__CLASS__)
        {
                return parent::model($className);
        }
}
```

Listing 10: Course Model

```
<?php

/**
 * This is the model class for table "course".
 *
 * The followings are the available columns in table 'course':
 * @property string $code
 * @property string $name
 * @property integer $students_per_pod
 * @property string $paired
 * @property string $has_topology
 *
 * The followings are the available model relations:
 * @property Class[] $classes
 * @property Topology[] $topologies
 */
class CourseModel extends CActiveRecord
{
        /**
         * @return string the associated database table name
         */
        public function tableName()
        {
                return 'course';
        }

        /**
         * @return array validation rules for model attributes.
         */
        public function rules()
        {
                // NOTE: you should only define rules for those attributes that
                // will receive user inputs.
                return array(
                        array('code, name, students_per_pod', 'required'),
                        array('students_per_pod', 'numerical', 'integerOnly'=>true),
                        array('code', 'length', 'max'=>64),
                        array('paired, has_topology', 'length', 'max'=>3),
                        array('code', 'unique'),
                        // The following rule is used by search().
                        // @todo Please remove those attributes that should not be searched.
                        array('code, name, students_per_pod, paired, has_topology', 'safe', '
                            on'=>'search'),
                );
        }

        /**
         * @return array relational rules.
         */
        public function relations()
        {
                // NOTE: you may need to adjust the relation name and the related
                // class name for the relations automatically generated below.
                return array(
                        'classes' => array(self::HAS_MANY, 'ClassModel', 'course_code'),
                        'topologies' => array(self::HAS_MANY, 'TopologyModel', 'course_code'),
                );
        }

        /**
         * @return array customized attribute labels (name=>label)
```

```
         */
        public function attributeLabels()
        {
                return array(
                        'code' => 'Course Code',
                        'name' => 'Course Name',
                        'students_per_pod' => 'Students Per Pod',
                        'paired' => 'Paired',
                        'has_topology' => 'Has Topology',
                );
        }

        /**
         * Retrieves a list of models based on the current search/filter conditions.
         *
         * Typical usecase:
         * - Initialize the model fields with values from filter form.
         * - Execute this method to get CActiveDataProvider instance which will filter
         * models according to data in model fields.
         * - Pass data provider to CGridView, CListView or any similar widget.
         *
         * @return CActiveDataProvider the data provider that can return the models
         * based on the search/filter conditions.
         */
        public function search()
        {
                // @todo Please modify the following code to remove attributes that should not
                        be searched.

                $criteria=new CDbCriteria;

                $criteria->compare('code',$this->code,true);
                $criteria->compare('name',$this->name,true);
                $criteria->compare('students_per_pod',$this->students_per_pod);
                $criteria->compare('paired',$this->paired,true);
                $criteria->compare('has_topology',$this->has_topology,true);

                return new CActiveDataProvider($this, array(
                        'criteria'=>$criteria,
                ));
        }

        /**
         * Returns the static model of the specified AR class.
         * Please note that you should have this exact method in all your CActiveRecord
             descendants!
         * @param string $className active record class name.
         * @return CourseModel the static model class
         */
        public static function model($className=__CLASS__)
        {
                return parent::model($className);
        }
}
```

Listing 11: Database Setup Form

```
<?php

/**
 * DatabaseForm class.
 * DatabaseForm is the data structure for keeping
 * database configuration form data. It is used by the 'database' action of 'SetupController'.
 */
class DatabaseForm extends CFormModel
{
        public $host, $user, $pass;

        /**
         * Declares the validation rules.
         * The rules state that hostname, username, and password are required,
         * and hostname needs to be accessible.
         */
        public function rules()
        {
                return array(
                        // hostname and username are required
                        array('host, user', 'required'),
                        // password needs to be accessible
                        array('host', 'connect'),
                );
        }

        /**
         * Validates the host name.
         * This is the 'connect' validator as declared in rules().
         */
        public function connect($attribute, $params) {
                try {
                        new PDO('mysql:host='.$this->host,$this->user,$this->pass);
                        $connection = mysqli_connect($this->host,$this->user,$this->pass);
                        $sql = 'CREATE DATABASE IF NOT EXISTS trends_rla';

                        if(!mysqli_query($connection,$sql))
                                $this->addError('user', mysqli_error($connection));
                } catch(PDOException $e) {
                        $this->addError('host', 'Invalid Host Name');
                }
```

```
        }

        /**
         * Declares attribute labels.
         */
        public function attributeLabels()
        {
                return array(
                        'host'=>'Host Name',
                        'user'=>'Username',
                        'pass'=>'Password',
                );
        }
}
```

Listing 12: Device Model

```
<?php

/**
 * This is the model class for table "device".
 *
 * The followings are the available columns in table 'device':
 * @property string $name
 * @property string $file
 * @property string $connectable
 *
 * The followings are the available model relations:
 * @property Cable[] $cables
 * @property Topology[] $topologies
 */
class DeviceModel extends CActiveRecord
{
        /**
         * @return string the associated database table name
         */
        public function tableName()
        {
                return 'device';
        }

        /**
         * @return array validation rules for model attributes.
         */
        public function rules()
        {
                // NOTE: you should only define rules for those attributes that
                // will receive user inputs.
                return array(
                        array('name, file', 'required'),
                        array('name', 'length', 'max'=>64),
                        array('file', 'length', 'max'=>128),
                        array('connectable', 'length', 'max'=>3),
                        array('name', 'unique'),
                        // The following rule is used by search().
                        // @todo Please remove those attributes that should not be searched.
                        array('name, file, connectable', 'safe', 'on'=>'search'),
                );
        }

        /**
         * @return array relational rules.
         */
        public function relations()
        {
                // NOTE: you may need to adjust the relation name and the related
                // class name for the relations automatically generated below.
                return array(
                        'cables' => array(self::HAS_MANY, 'CableModel', 'device_name'),
                        'topologies' => array(self::HAS_MANY, 'TopologyModel', 'device_name'),
                );
        }

        /**
         * @return array customized attribute labels (name=>label)
         */
        public function attributeLabels()
        {
                return array(
                        'name' => 'Device Name',
                        'file' => 'Image File',
                        'connectable' => 'Connectable',
                );
        }

        /**
         * Retrieves a list of models based on the current search/filter conditions.
         *
         * Typical usecase:
         * - Initialize the model fields with values from filter form.
         * - Execute this method to get CActiveDataProvider instance which will filter
         * models according to data in model fields.
         * - Pass data provider to CGridView, CListView or any similar widget.
         *
         * @return CActiveDataProvider the data provider that can return the models
         * based on the search/filter conditions.
         */
        public function search()
```

```
        {
                // @todo Please modify the following code to remove attributes that should not
                        be searched.

                $criteria=new CDbCriteria;

                $criteria->compare('name',$this->name,true);
                $criteria->compare('file',$this->file,true);
                $criteria->compare('connectable',$this->connectable,true);

                return new CActiveDataProvider($this, array(
                        'criteria'=>$criteria,
                ));
        }

        /**
         * Returns the static model of the specified AR class.
         * Please note that you should have this exact method in all your CActiveRecord
             descendants!
         * @param string $className active record class name.
         * @return DeviceModel the static model class
         */
        public static function model($className=__CLASS__)
        {
                return parent::model($className);
        }
}
```

Listing 13: Login Form

```
<?php

/**
 * LoginForm class.
 * LoginForm is the data structure for keeping
 * user login form data. It is used by the 'login' action of 'SiteController'.
 */
class LoginForm extends CFormModel
{
        public $username;
        public $password;

        private $_identity;

        /**
         * Declares the validation rules.
         * The rules state that username and password are required,
         * and password needs to be authenticated.
         */
        public function rules()
        {
                return array(
                        // username and password are required
                        array('username, password', 'required'),
                        // password needs to be authenticated
                        array('password', 'authenticate'),
                );
        }

        /**
         * Declares attribute labels.
         */
        public function attributeLabels()
        {
                return array(
                        'username'=>'Employee ID'
                );
        }

        /**
         * Authenticates the password.
         * This is the 'authenticate' validator as declared in rules().
         */
        public function authenticate($attribute, $params)
        {
                if(!$this->hasErrors())
                {
                        $this->_identity=new UserIdentity($this->username,$this->password);
                        if(!$this->_identity->authenticate())
                                $this->addError('password','Incorrect username or password.');
                }
        }

        /**
         * Logs in the user using the given username and password in the model.
         * @return boolean whether login is successful
         */
        public function login()
        {
                if($this->_identity===null)
                {
                        $this->_identity=new UserIdentity($this->username,$this->password);
                        $this->_identity->authenticate();
                }
                if($this->_identity->errorCode===UserIdentity::ERROR_NONE)
                {
                        Yii::app()->user->login($this->_identity,0);
                        return true;
```

```
                }
                else
                        return false;
        }
}
```

Listing 14: Pod Device Model

```
<?php

/**
 * This is the model class for table "pod_device".
 *
 * The followings are the available columns in table 'pod_device':
 * @property integer $id
 * @property integer $pod_id
 * @property integer $topology_id
 *
 * The followings are the available model relations:
 * @property Pod $pod
 * @property Topology $topology
 */
class PodDeviceModel extends CActiveRecord
{
        /**
         * @return string the associated database table name
         */
        public function tableName()
        {
                return 'pod_device';
        }

        /**
         * @return array validation rules for model attributes.
         */
        public function rules()
        {
                // NOTE: you should only define rules for those attributes that
                // will receive user inputs.
                return array(
                        array('pod_id, topology_id', 'required'),
                        array('pod_id, topology_id', 'numerical', 'integerOnly'=>true),
                        // The following rule is used by search().
                        // @todo Please remove those attributes that should not be searched.
                        array('id, pod_id, topology_id', 'safe', 'on'=>'search'),
                );
        }

        /**
         * @return array relational rules.
         */
        public function relations()
        {
                // NOTE: you may need to adjust the relation name and the related
                // class name for the relations automatically generated below.
                return array(
                        'pod' => array(self::BELONGS_TO, 'PodModel', 'pod_id'),
                        'topology' => array(self::BELONGS_TO, 'TopologyModel', 'topology_id'),
                );
        }

        /**
         * @return array customized attribute labels (name=>label)
         */
        public function attributeLabels()
        {
                return array(
                        'id' => 'ID',
                        'pod_id' => 'Pod',
                        'topology_id' => 'Topology',
                );
        }

        /**
         * Retrieves a list of models based on the current search/filter conditions.
         *
         * Typical usecase:
         * - Initialize the model fields with values from filter form.
         * - Execute this method to get CActiveDataProvider instance which will filter
         * models according to data in model fields.
         * - Pass data provider to CGridView, CListView or any similar widget.
         *
         * @return CActiveDataProvider the data provider that can return the models
         * based on the search/filter conditions.
         */
        public function search()
        {
                // @todo Please modify the following code to remove attributes that should not
                        be searched.

                $criteria=new CDbCriteria;

                $criteria->compare('id',$this->id);
                $criteria->compare('pod_id',$this->pod_id);
                $criteria->compare('topology_id',$this->topology_id);

                return new CActiveDataProvider($this, array(
                        'criteria'=>$criteria,
```

```
                ));
        }

        /**
         * Returns the static model of the specified AR class.
         * Please note that you should have this exact method in all your CActiveRecord
            descendants!
         * @param string $className active record class name.
         * @return PodDeviceModel the static model class
         */
        public static function model($className=__CLASS__)
        {
                return parent::model($className);
        }
}
```

Listing 15: Pod Login Form

```
<?php

/**
 * PodForm class.
 * PodForm is the data structure for keeping
 * pod login form data. It is used by the 'index' action of 'SiteController'.
 */
class PodForm extends CFormModel
{
        public $podNum, $pass, $classID;

        /**
         * Declares the validation rules.
         * The rules state that pod number and password are required,
         * and password needs to be authenticated.
         */
        public function rules()
        {
                return array(
                        // username and password are required
                        array('podNum, pass, classID', 'required'),
                        // password needs to be authenticated
                        array('pass', 'authenticate'),
                );
        }

        /**
         * Declares attribute labels.
         */
        public function attributeLabels()
        {
                return array(
                        'podNum'=>'Pod Number',
                        'pass'=>'Password',
                        'classID'=>'Class ID',
                );
        }

        /**
         * Authenticates the password.
         * This is the 'authenticate' validator as declared in rules().
         */
        public function authenticate($attribute, $params)
        {
                $pod = PodModel::model()->findByPk($this->podNum);

                if($pod->pass !== $this->pass)
                        $this->addError('pass', 'Incorrect Password.');
        }

        /**
         * Logs in the user using the given pod number and password in the model.
         * @return boolean whether login is successful
         */
        public function login()
        {
                $pod = PodModel::model()->findByPk($this->podNum);

                if($pod->pass === $this->pass)
                        return true;

                return false;
        }
}
```

Listing 16: Pod Model

```
<?php

/**
 * This is the model class for table "pod".
 *
 * The followings are the available columns in table 'pod':
 * @property integer $id
 * @property integer $pod_num
 * @property integer $class_id
 * @property string $pass
 *
```

```
 * The followings are the available model relations:
 * @property Class $class
 * @property PodDevice[] $podDevices
 */
class PodModel extends CActiveRecord
{
	/**
	 * @return string the associated database table name
	 */
	public function tableName()
	{
		return 'pod';
	}

	/**
	 * @return array validation rules for model attributes.
	 */
	public function rules()
	{
		// NOTE: you should only define rules for those attributes that
		// will receive user inputs.
		return array(
			array('pod_num, class_id, pass', 'required'),
			array('pod_num, class_id', 'numerical', 'integerOnly'=>true),
			array('pass', 'length', 'max'=>32),
			array('pod_num', 'doesNotExist'),
			// The following rule is used by search().
			// @todo Please remove those attributes that should not be searched.
			array('id, pod_num, class_id, pass', 'safe', 'on'=>'search'),
		);
	}

	/**
	 * Checks whether the pod number/class id combination already exists.
	 * This is the 'doesNotExist' validator as declared in rules().
	 */
	public function doesNotExist($attribute, $params)
	{
		$pod = PodModel::model()->findByAttributes(array(
			'pod_num'=>$this->pod_num,
			'class_id'=>$this->class_id,
		));
		if(isset($pod))
			$this->addError('pod_num', 'Pod Number "'.$this->pod_num.'" has
			    already been taken');
	}

	/**
	 * @return array relational rules.
	 */
	public function relations()
	{
		// NOTE: you may need to adjust the relation name and the related
		// class name for the relations automatically generated below.
		return array(
			'class' => array(self::BELONGS_TO, 'ClassModel', 'class_id'),
			'podDevices' => array(self::HAS_MANY, 'PodDeviceModel', 'pod_id'),
		);
	}

	/**
	 * @return array customized attribute labels (name=>label)
	 */
	public function attributeLabels()
	{
		return array(
			'id' => 'ID',
			'pod_num' => 'Pod Number',
			'class_id' => 'Class ID',
			'pass' => 'Password',
		);
	}

	/**
	 * Retrieves a list of models based on the current search/filter conditions.
	 *
	 * Typical usecase:
	 * - Initialize the model fields with values from filter form.
	 * - Execute this method to get CActiveDataProvider instance which will filter
	 * models according to data in model fields.
	 * - Pass data provider to CGridView, CListView or any similar widget.
	 *
	 * @return CActiveDataProvider the data provider that can return the models
	 * based on the search/filter conditions.
	 */
	public function search($class_id)
	{
		// @todo Please modify the following code to remove attributes that should not
		    be searched.

		$criteria=new CDbCriteria;

		$criteria->compare('id',$this->id);
		$criteria->compare('pod_num',$this->pod_num);
		$criteria->compare('class_id',$class_id);
		$criteria->compare('pass',$this->pass,true);

		return new CActiveDataProvider($this, array(
			'criteria'=>$criteria,
```

```
                ));
        }

        /**
         * Returns the static model of the specified AR class.
         * Please note that you should have this exact method in all your CActiveRecord
             descendants!
         * @param string $className active record class name.
         * @return PodModel the static model class
         */
        public static function model($className=__CLASS__)
        {
                return parent::model($className);
        }
}
```

Listing 17: Student Login Form

```
<?php

/**
 * StudentForm class.
 * StudentForm is the data structure for keeping
 * student login form data. It is used by the 'index' action of 'SiteController'.
 */
class StudentForm extends CFormModel
{
        public $prof, $subj, $pass;

        private $_identity;

        /**
         * Declares the validation rules.
         * The rules state that professor, subject, and password are required,
         * and password needs to be authenticated.
         */
        public function rules()
        {
                return array(
                        // password is required
                        array('prof, subj, pass', 'required'),
                        // professor and subject are required
                        array('prof, subj', 'nonZero'),
                        // password needs to be authenticated
                        array('pass', 'authenticate'),
                );
        }

        /**
         * Checks whether the professor or subject is empty.
         * This is the 'nonZero' validator as declared in rules().
         */
        public function nonZero($attribute, $params)
        {
                if($this->prof === '0')
                        $this->addError('prof', 'Professor cannot be blank.');

                if($this->subj === '0')
                        $this->addError('subj', 'Subject cannot be blank.');
        }

        /**
         * Declares attribute labels.
         */
        public function attributeLabels()
        {
                return array(
                        'prof'=>'Professor',
                        'subj'=>'Subject',
                        'pass'=>'Password',
                );
        }

        /**
         * Authenticates the password.
         * This is the 'authenticate' validator as declared in rules().
         */
        public function authenticate($attribute,$params)
        {
                if(!$this->hasErrors())
                {
                        $this->_identity=new UserIdentity($this->prof,$this->pass);
                        $this->_identity->setSubj($this->subj);
                        if(!$this->_identity->authenticate())
                                $this->addError('pass','Incorrect password.');
                }
        }

        /**
         * Logs in the user using the given professor, subject, and password in the model.
         * @return boolean whether login is successful
         */
        public function login()
        {
                if($this->_identity===null)
                {
                        $this->_identity=new UserIdentity($this->prof,$this->pass);
                        $this->_identity->setSubj($this->subj);
```

```
                $this->_identity->authenticate();
            }
            if($this->_identity->errorCode===UserIdentity::ERROR_NONE)
            {
                Yii::app()->user->login($this->_identity,0);
                return true;
            }
            else
                return false;
        }
}
```

Listing 18: Topology Model

```
<?php

/**
 * This is the model class for table "topology".
 *
 * The followings are the available columns in table 'topology':
 * @property integer $id
 * @property string $course_code
 * @property string $device_name
 * @property integer $x
 * @property integer $y
 * @property integer $width
 * @property integer $height
 * @property string $type
 * @property string $ip
 * @property string $host
 * @property string $display
 *
 * The followings are the available model relations:
 * @property PodDevice[] $podDevices
 * @property Device $deviceName
 * @property Course $courseCode
 */
class TopologyModel extends CActiveRecord
{
        public $device, $cable;
        /**
         * @return string the associated database table name
         */
        public function tableName()
        {
                return 'topology';
        }

        /**
         * @return array validation rules for model attributes.
         */
        public function rules()
        {
                // NOTE: you should only define rules for those attributes that
                // will receive user inputs.
                return array(
                        array('course_code, device_name, x, y, width, height, type, ip, host,
                             display', 'required'),
                        array('x, y, width, height', 'numerical', 'integerOnly'=>true),
                        array('course_code, device_name', 'length', 'max'=>64),
                        array('type', 'length', 'max'=>1),
                        array('ip', 'length', 'max'=>32),
                        array('host, display', 'length', 'max'=>16),
                        // The following rule is used by search().
                        // @todo Please remove those attributes that should not be searched.
                        array('id, course_code, device_name, x, y, width, height, type, ip,
                             host, display', 'safe', 'on'=>'search'),
                );
        }

        /**
         * @return array relational rules.
         */
        public function relations()
        {
                // NOTE: you may need to adjust the relation name and the related
                // class name for the relations automatically generated below.
                return array(
                        'podDevices' => array(self::HAS_MANY, 'PodDeviceModel', 'topology_id')
                             ,
                        'deviceModel' => array(self::BELONGS_TO, 'DeviceModel', 'device_name')
                             ,
                        'courseModel' => array(self::BELONGS_TO, 'CourseModel', 'course_code')
                             ,
                );
        }

        /**
         * @return array customized attribute labels (name=>label)
         */
        public function attributeLabels()
        {
                return array(
                        'id' => 'ID',
                        'course_code' => 'Course Code',
                        'device_name' => 'Device Name',
                        'x' => 'X',
                        'y' => 'Y',
```

```
                        'width' => 'Width',
                        'height' => 'Height',
                        'type' => 'Type',
                        'ip' => 'IP Address',
                        'host' => 'Host Name',
                        'display' => 'Display Name',
                );
        }

        /**
         * Retrieves a list of models based on the current search/filter conditions.
         *
         * Typical usecase:
         * - Initialize the model fields with values from filter form.
         * - Execute this method to get CActiveDataProvider instance which will filter
         * models according to data in model fields.
         * - Pass data provider to CGridView, CListView or any similar widget.
         *
         * @return CActiveDataProvider the data provider that can return the models
         * based on the search/filter conditions.
         */
        public function search()
        {
                // @todo Please modify the following code to remove attributes that should not
                        be searched.

                $criteria=new CDbCriteria;

                $criteria->compare('id',$this->id);
                $criteria->compare('course_code',$this->course_code,true);
                $criteria->compare('device_name',$this->device_name,true);
                $criteria->compare('x',$this->x);
                $criteria->compare('y',$this->y);
                $criteria->compare('width',$this->width);
                $criteria->compare('height',$this->height);
                $criteria->compare('type',$this->type,true);
                $criteria->compare('ip',$this->ip,true);
                $criteria->compare('host',$this->host,true);
                $criteria->compare('display',$this->display,true);

                return new CActiveDataProvider($this, array(
                        'criteria'=>$criteria,
                ));
        }

        /**
         * Returns the static model of the specified AR class.
         * Please note that you should have this exact method in all your CActiveRecord
             descendants!
         * @param string $className active record class name.
         * @return TopologyModel the static model class
         */
        public static function model($className=__CLASS__)
        {
                return parent::model($className);
        }
}
```

Listing 19: User Model

```
<?php

/**
 * This is the model class for table "user".
 *
 * The followings are the available columns in table 'user':
 * @property string $employee_id
 * @property string $pass
 * @property string $user_type
 * @property string $first_name
 * @property string $last_name
 *
 * The followings are the available model relations:
 * @property Class[] $classes
 */
class UserModel extends CActiveRecord
{
        public $name, $retype;

        /**
         * @return string the associated database table name
         */
        public function tableName()
        {
                return 'user';
        }

        /**
         * @return array validation rules for model attributes.
         */
        public function rules()
        {
                // NOTE: you should only define rules for those attributes that
                // will receive user inputs.
                return array(
                        array('employee_id, pass, retype, user_type, first_name, last_name', '
                            required'),
                        array('employee_id', 'length', 'max'=>32),
```

```
                array('pass', 'length', 'max'=>128),
                array('user_type', 'length', 'max'=>13),
                array('first_name, last_name', 'length', 'max'=>64),
                array('employee_id', 'unique'),
                array('pass', 'valid'),
                array('user_type', 'nonZero'),
                // The following rule is used by search().
                // @todo Please remove those attributes that should not be searched.
                array('employee_id, pass, user_type, first_name, last_name, name', '
                    safe', 'on'=>'search'),
        );
}

/**
 * Validates the password and retype.
 * This is the 'valid' validator as declared in rules().
 */
public function valid($attribute, $params)
{
        if(strlen($this->pass) < 6)
                $this->addError('pass', 'Password must be at least six (6) characters
                    long.');

        $this->pass = password_hash($this->pass, PASSWORD_DEFAULT);
        if(!password_verify($this->retype, $this->pass))
                $this->addError('retype', 'Passwords did not match');
}

/**
 * Checks whether the user type is empty.
 * This is the 'nonZero' validator as declared in rules().
 */
public function nonZero($attribute, $params)
{
        if($this->user_type === '0')
                $this->addError('user_type', 'User Type cannot be blank');
}

/**
 * @return array relational rules.
 */
public function relations()
{
        // NOTE: you may need to adjust the relation name and the related
        // class name for the relations automatically generated below.
        return array(
                'classes' => array(self::HAS_MANY, 'ClassModel', 'prof'),
        );
}

/**
 * @return array customized attribute labels (name=>label)
 */
public function attributeLabels()
{
        return array(
                'employee_id' => 'Employee ID',
                'pass' => 'Password',
                'retype' => 'Retype Password',
                'user_type' => 'User Type',
                'first_name' => 'First Name',
                'last_name' => 'Last Name',
        );
}

/**
 * Retrieves a list of models based on the current search/filter conditions.
 *
 * Typical usecase:
 * - Initialize the model fields with values from filter form.
 * - Execute this method to get CActiveDataProvider instance which will filter
 * models according to data in model fields.
 * - Pass data provider to CGridView, CListView or any similar widget.
 *
 * @return CActiveDataProvider the data provider that can return the models
 * based on the search/filter conditions.
 */
public function search()
{
        // @todo Please modify the following code to remove attributes that should not
              be searched.

        $criteria=new CDbCriteria;

        $criteria->compare('employee_id',$this->employee_id,true);
        $criteria->compare('pass',$this->pass,true);
        $criteria->compare('user_type',$this->user_type,true);
        $criteria->compare('first_name',$this->first_name,true);
        $criteria->compare('last_name',$this->last_name,true);
        $criteria->compare("concat(last_name,', ',first_name)",$this->name,true);

        return new CActiveDataProvider($this, array(
                'criteria'=>$criteria,
        ));
}

/**
 * Returns the static model of the specified AR class.
 * Please note that you should have this exact method in all your CActiveRecord
```

```
            descendants!
         * @param string $className active record class name.
         * @return UserModel the static model class
         */
        public static function model($className=__CLASS__)
        {
                return parent::model($className);
        }
}
```

E. Views

E..1 Class View

Listing 20: Class Admin View

```
<?php
/* @var $this ClassController */
/* @var $model ClassModel */

$this->breadcrumbs=array(
        'Classes'=>array('index'),
        'Manage',
);

$this->menu=array(
        array('label'=>'List Classes', 'url'=>array('index')),
        array('label'=>'Create Class', 'url'=>array('create')),
);

Yii::app()->clientScript->registerScript('search', "
$('.search-button').click(function(){
        $('.search-form').toggle();
        return false;
});
$('.search-form form').submit(function(){
        $('#class-model-grid').yiiGridView('update', {
                data: $(this).serialize()
        });
        return false;
});
");
?>

<h1>Manage Classes</h1>

<p>
You may optionally enter a comparison operator (<b>&lt;</b>, <b>&lt;=</b>, <b>&gt;</b>, <b>&gt
    ;=</b>, <b>&lt;&gt;</b>
or <b>=</b>) at the beginning of each of your search values to specify how the comparison
    should be done.
</p>

<?php echo CHtml::link('Advanced Search','#',array('class'=>'search-button')); ?>
<div class="search-form" style="display:none">
<?php $this->renderPartial('_search',array(
        'model'=>$model,
)); ?>
</div><!-- search-form -->

<?php $this->widget('zii.widgets.grid.CGridView', array(
        'id'=>'class-model-grid',
        'dataProvider'=>$model->search(),
        'filter'=>$model,
        'columns'=>array(
                'course_code',
                //'pass',
                'ts_ip',
                /*
                'ts_user',
                'ts_pass',
                */
                'num_of_students',
                array(
                        'header'=>'Options',
                        'class'=>'CButtonColumn',
                ),
        ),
)); ?>
```

Listing 21: Create Class View

```
<?php
/* @var $this ClassController */
/* @var $model ClassModel */

$this->breadcrumbs=array(
        'Classes'=>array('index'),
        'Create',
);

$this->menu=array(
        array('label'=>'List Classes', 'url'=>array('index')),
```

```
        array('label'=>'Manage Classes', 'url'=>array('admin')),
);
?>

<h1>Create Class</h1>

<?php $this->renderPartial('_form', array('model'=>$model)); ?>
```

Listing 22: Class Form View

```
<?php
/* @var $this ClassController */
/* @var $model ClassModel */
/* @var $form CActiveForm */
?>

<div class="form">

<?php
        $form=$this->beginWidget('CActiveForm', array(
                'id'=>'class-model-form',
                // Please note: When you enable ajax validation, make sure the corresponding
                // controller action is handling ajax validation correctly.
                // There is a call to performAjaxValidation() commented in generated
                    controller code.
                // See class documentation of CActiveForm for details on this.
                'enableAjaxValidation'=>false,
        ));

        $courses = CHtml::listData(CourseModel::model()->findAll(),'code','code');

        $courseArray = array('Select Course');
        if(count($courses) > 0) {
                foreach($courses as $key => $value)
                        $courseArray[$key] = $value;
        } else {
                Yii::app()->user->setFlash('error','No courses found. Please contact your
                    system administrator.');
                $this->redirect(array('/class/admin'));
        }
?>

        <p class="note">Fields with <span class="required">*</span> are required.</p>

        <?php echo $form->errorSummary($model); ?>

        <div class="row">
                <?php echo $form->labelEx($model,'course_code'); ?>
                <?php echo $form->dropDownList($model,'course_code',$courseArray); ?>
                <?php echo $form->error($model,'course_code'); ?>
        </div>

        <div class="row">
                <?php echo $form->labelEx($model,'pass'); ?>
                <?php echo $form->textField($model,'pass',array('size'=>32,'maxlength'=>32));
                    ?>
                <?php echo $form->error($model,'pass'); ?>
        </div>

        <div class="row">
                <?php echo $form->labelEx($model,'ts_ip'); ?>
                <?php echo $form->textField($model,'ts_ip',array('size'=>32,'maxlength'=>32));
                    ?>
                <?php echo $form->error($model,'ts_ip'); ?>
        </div>

        <div class="row">
                <?php echo $form->labelEx($model,'ts_user'); ?>
                <?php echo $form->textField($model,'ts_user',array('size'=>32,'maxlength'=>32)
                    ); ?>
                <?php echo $form->error($model,'ts_user'); ?>
        </div>

        <div class="row">
                <?php echo $form->labelEx($model,'ts_pass'); ?>
                <?php echo $form->textField($model,'ts_pass',array('size'=>32,'maxlength'=>32)
                    ); ?>
                <?php echo $form->error($model,'ts_pass'); ?>
        </div>

        <div class="row">
                <?php echo $form->labelEx($model,'num_of_students'); ?>
                <?php echo $form->numberField($model,'num_of_students',array('min'=>1)); ?>
                <?php echo $form->error($model,'num_of_students'); ?>
        </div>

        <div class="row">
                <?php echo $form->labelEx($model,'exp'); ?>
                <?php echo $form->dateField($model,'exp',array('min'=>date('Y-m-d'))); ?>
                <?php echo $form->error($model,'exp'); ?>
        </div>

        <div class="row buttons">
                <?php echo CHtml::submitButton($model->isNewRecord ? 'Create' : 'Save'); ?>
        </div>

<?php $this->endWidget(); ?>
```

```
</div><!-- form -->
```

Listing 23: Class Index View

```
<?php
/* @var $this ClassController */
/* @var $dataProvider CActiveDataProvider */

$this->breadcrumbs=array(
        'Classes',
);

$this->menu=array(
        array('label'=>'Create Class', 'url'=>array('create')),
        array('label'=>'Manage Classes', 'url'=>array('admin')),
);
?>

<h1>Classes</h1>

<?php $this->widget('zii.widgets.CListView', array(
        'dataProvider'=>$model->search(),
        'itemView'=>'_view',
)); ?>
```

Listing 24: Class Search View

```
<?php
/* @var $this ClassController */
/* @var $model ClassModel */
/* @var $form CActiveForm */
?>

<div class="wide form">

<?php $form=$this->beginWidget('CActiveForm', array(
        'action'=>Yii::app()->createUrl($this->route),
        'method'=>'get',
)); ?>

        <div class="row">
                <?php echo $form->label($model,'id'); ?>
                <?php echo $form->numberField($model,'id',array('min'=>0)); ?>
        </div>

        <div class="row">
                <?php echo $form->label($model,'course_code'); ?>
                <?php echo $form->textField($model,'course_code',array('size'=>32,'maxlength
                        '=>64)); ?>
        </div>

        <div class="row">
                <?php echo $form->label($model,'pass'); ?>
                <?php echo $form->passwordField($model,'pass',array('size'=>32,'maxlength
                        '=>32)); ?>
        </div>

        <div class="row">
                <?php echo $form->label($model,'prof'); ?>
                <?php echo $form->textField($model,'prof',array('size'=>32,'maxlength'=>32));
                        ?>
        </div>

        <div class="row">
                <?php echo $form->label($model,'ts_ip'); ?>
                <?php echo $form->textField($model,'ts_ip',array('size'=>32,'maxlength'=>32));
                         ?>
        </div>

        <div class="row">
                <?php echo $form->label($model,'ts_user'); ?>
                <?php echo $form->textField($model,'ts_user',array('size'=>32,'maxlength'=>32)
                        ); ?>
        </div>

        <div class="row">
                <?php echo $form->label($model,'ts_pass'); ?>
                <?php echo $form->textField($model,'ts_pass',array('size'=>32,'maxlength'=>32)
                        ); ?>
        </div>

        <div class="row">
                <?php echo $form->label($model,'num_of_students'); ?>
                <?php echo $form->numberField($model,'num_of_students',array('min'=>1)); ?>
        </div>

        <div class="row buttons">
                <?php echo CHtml::submitButton('Search'); ?>
        </div>

<?php $this->endWidget(); ?>

</div><!-- search-form -->
```

Listing 25: Update Class View

```
<?php
/* @var $this ClassController */
/* @var $model ClassModel */

$this->breadcrumbs=array(
        'Classes'=>array('index'),
        $model->id=>array('view','id'=>$model->id),
        'Update',
);

$this->menu=array(
        array('label'=>'List Classes', 'url'=>array('index')),
        array('label'=>'Create Class', 'url'=>array('create')),
        array('label'=>'View Class', 'url'=>array('view', 'id'=>$model->id)),
        array('label'=>'Manage Classes', 'url'=>array('admin')),
        array('label'=>'Pods', 'url'=>array('/pod/admin', 'class_id'=>$model->id)),
);
?>

<h1>Update Class #<?php echo $model->id; ?></h1>

<?php $this->renderPartial('_form', array('model'=>$model)); ?>
```

Listing 26: View Class Widget

```
<?php
/* @var $this ClassController */
/* @var $model ClassModel */

$this->breadcrumbs=array(
        'Classes'=>array('index'),
        $model->id,
);

$this->menu=array(
        array('label'=>'List Classes', 'url'=>array('index')),
        array('label'=>'Create Class', 'url'=>array('create')),
        array('label'=>'Update Class', 'url'=>array('update', 'id'=>$model->id)),
        array('label'=>'Delete Class', 'url'=>'#', 'linkOptions'=>array('submit'=>array('
            delete','id'=>$model->id),'confirm'=>'Are you sure you want to delete this class
            ?')),
        array('label'=>'Manage Classes', 'url'=>array('admin')),
        array('label'=>'Pods', 'url'=>array('/pod/admin', 'class_id'=>$model->id)),
);
?>

<h1>View Class #<?php echo $model->id; ?></h1>

<?php $this->widget('zii.widgets.CDetailView', array(
        'data'=>$model,
        'attributes'=>array(
                'id',
                'course_code',
                'pass',
                'prof',
                'ts_ip',
                'ts_user',
                'ts_pass',
                'num_of_students',
        ),
)); ?>
```

Listing 27: View Class Layout

```
<?php
/* @var $this ClassController */
/* @var $data ClassModel */
?>

<div class="view">

        <b><?php echo CHtml::encode($data->getAttributeLabel('id')); ?>:</b>
        <?php echo CHtml::link(CHtml::encode($data->id), array('view', 'id'=>$data->id)); ?>
        <br />

        <b><?php echo CHtml::encode($data->getAttributeLabel('course_code')); ?>:</b>
        <?php echo CHtml::encode($data->course_code); ?>
        <br />

        <b><?php echo CHtml::encode($data->getAttributeLabel('pass')); ?>:</b>
        <?php echo CHtml::encode($data->pass); ?>
        <br />

        <b><?php echo CHtml::encode($data->getAttributeLabel('prof')); ?>:</b>
        <?php echo CHtml::encode($data->prof); ?>
        <br />

        <b><?php echo CHtml::encode($data->getAttributeLabel('ts_ip')); ?>:</b>
        <?php echo CHtml::encode($data->ts_ip); ?>
        <br />

        <b><?php echo CHtml::encode($data->getAttributeLabel('ts_user')); ?>:</b>
        <?php echo CHtml::encode($data->ts_user); ?>
        <br />
```

```
        <b><?php echo CHtml::encode($data->getAttributeLabel('ts_pass')); ?>:</b>
        <?php echo CHtml::encode($data->ts_pass); ?>
        <br />

        <?php /*
        <b><?php echo CHtml::encode($data->getAttributeLabel('num_of_students')); ?>:</b>
        <?php echo CHtml::encode($data->num_of_students); ?>
        <br />

        <b><?php echo CHtml::encode($data->getAttributeLabel('has_pods')); ?>:</b>
        <?php echo CHtml::encode($data->has_pods); ?>
        <br />

        */ ?>

</div>
```

E..2 Course View

Listing 28: Course Admin View

```
<?php
/* @var $this CourseController */
/* @var $model CourseModel */

$this->breadcrumbs=array(
        'Courses'=>array('index'),
        'Manage',
);

$this->menu=array(
        array('label'=>'List Courses', 'url'=>array('index')),
        array('label'=>'Create Course', 'url'=>array('create')),
);

Yii::app()->clientScript->registerScript('search', "
$('.search-button').click(function(){
        $('.search-form').toggle();
        return false;
});
$('.search-form form').submit(function(){
        $('#course-model-grid').yiiGridView('update', {
                data: $(this).serialize()
        });
        return false;
});
");
?>

<h1>Manage Courses</h1>

<p>
You may optionally enter a comparison operator (<b>&lt;</b>, <b>&lt;=</b>, <b>&gt;</b>, <b>&gt
    ;=</b>, <b>&lt;&gt;</b>
or <b>=</b>) at the beginning of each of your search values to specify how the comparison
    should be done.
</p>

<?php echo CHtml::link('Advanced Search','#',array('class'=>'search-button')); ?>
<div class="search-form" style="display:none">
<?php $this->renderPartial('_search',array(
        'model'=>$model,
)); ?>
</div><!-- search-form -->

<?php $this->widget('zii.widgets.grid.CGridView', array(
        'id'=>'course-model-grid',
        'dataProvider'=>$model->search(),
        'filter'=>$model,
        'columns'=>array(
                'code',
                'name',
                'students_per_pod',
                'paired',
                array(
                        'header'=>'Options',
                        'class'=>'CButtonColumn',
                ),
        ),
)); ?>
```

Listing 29: Create Course View

```
<?php
/* @var $this CourseController */
/* @var $model CourseModel */

$this->breadcrumbs=array(
        'Courses'=>array('index'),
        'Create',
);

$this->menu=array(
        array('label'=>'List Courses', 'url'=>array('index')),
```

```
        array('label'=>'Manage Courses', 'url'=>array('admin')),
);
?>

<h1>Create Course</h1>

<?php $this->renderPartial('_form', array('model'=>$model)); ?>
```

Listing 30: Course Form View

```
<?php
/* @var $this CourseController */
/* @var $model CourseModel */
/* @var $form CActiveForm */
?>

<div class="form">

<?php $form=$this->beginWidget('CActiveForm', array(
        'id'=>'course-model-form',
        // Please note: When you enable ajax validation, make sure the corresponding
        // controller action is handling ajax validation correctly.
        // There is a call to performAjaxValidation() commented in generated controller code.
        // See class documentation of CActiveForm for details on this.
        'enableAjaxValidation'=>false,
)); ?>

        <p class="note">Fields with <span class="required">*</span> are required.</p>

        <?php echo $form->errorSummary($model); ?>

        <div class="row">
                <?php echo $form->labelEx($model,'code'); ?>
                <?php echo $form->textField($model,'code',array('size'=>60,'maxlength'=>64));
                    ?>
                <?php echo $form->error($model,'code'); ?>
        </div>

        <div class="row">
                <?php echo $form->labelEx($model,'name'); ?>
                <?php echo $form->textArea($model,'name',array('rows'=>6, 'cols'=>46)); ?>
                <?php echo $form->error($model,'name'); ?>
        </div>

        <div class="row">
                <?php echo $form->labelEx($model,'students_per_pod'); ?>
                <?php echo $form->numberField($model,'students_per_pod',array('min'=>1)); ?>
                <?php echo $form->error($model,'students_per_pod'); ?>
        </div>

        <div class="row">
                <?php echo $form->labelEx($model,'paired'); ?>
                <?php echo $form->dropDownList($model,'paired',array(
                        'Yes'=>'Yes',
                        'No'=>'No'
                )); ?>
                <?php echo $form->error($model,'paired'); ?>
        </div>

        <div class="row buttons">
                <?php echo CHtml::submitButton($model->isNewRecord ? 'Create' : 'Save'); ?>
        </div>

<?php $this->endWidget(); ?>

</div><!-- form -->
```

Listing 31: Course Index View

```
<?php
/* @var $this CourseController */
/* @var $dataProvider CActiveDataProvider */

$this->breadcrumbs=array(
        'Courses',
);

$this->menu=array(
        array('label'=>'Create Course', 'url'=>array('create')),
        array('label'=>'Manage Courses', 'url'=>array('admin')),
);
?>

<h1>Courses</h1>

<?php $this->widget('zii.widgets.CListView', array(
        'dataProvider'=>$dataProvider,
        'itemView'=>'_view',
)); ?>
```

Listing 32: Course Search View

```
<?php
/* @var $this CourseController */
```

```
/* @var $model CourseModel */
/* @var $form CActiveForm */
?>

<div class="wide form">

<?php $form=$this->beginWidget('CActiveForm', array(
        'action'=>Yii::app()->createUrl($this->route),
        'method'=>'get',
)); ?>

        <div class="row">
                <?php echo $form->label($model,'code'); ?>
                <?php echo $form->textField($model,'code',array('size'=>60,'maxlength'=>64));
                    ?>
        </div>

        <div class="row">
                <?php echo $form->label($model,'name'); ?>
                <?php echo $form->textArea($model,'name',array('rows'=>6, 'cols'=>46)); ?>
        </div>

        <div class="row">
                <?php echo $form->label($model,'students_per_pod'); ?>
                <?php echo $form->numberField($model,'students_per_pod',array('min'=>0)); ?>
        </div>

        <div class="row buttons">
                <?php echo CHtml::submitButton('Search'); ?>
        </div>

<?php $this->endWidget(); ?>

</div><!-- search-form -->
```

Listing 33: Course Logical Connection View

```
<?php
/* @var $this CourseController */
/* @var $model TopologyModel */
/* @var $form CActiveForm */
?>

<link rel="stylesheet" type="text/css" href="<?php echo Yii::app()->request->baseUrl; ?>/css/
    jquery-ui.css"></link>
<link rel="stylesheet" type="text/css" href="<?php echo Yii::app()->request->baseUrl; ?>/css/
    topology.css"></link>

<script type="text/javascript" src="<?php echo Yii::app()->request->baseUrl; ?>/js/jquery
    -2.1.1.js"></script>
<script type="text/javascript" src="<?php echo Yii::app()->request->baseUrl; ?>/js/jquery-ui.
    js"></script>
<script type="text/javascript" src="<?php echo Yii::app()->request->baseUrl; ?>/js/raphael.js
    "></script>
<script type="text/javascript" src="<?php echo Yii::app()->request->baseUrl; ?>/js/main.js"></
    script>
<script type="text/javascript" src="<?php echo Yii::app()->request->baseUrl; ?>/js/topology.js
    "></script>
<script type="text/javascript" src="<?php echo Yii::app()->request->baseUrl; ?>/js/cable.js
    "></script>
<script type="text/javascript">
        window.addEventListener("load",function() {
                if(paper == null)
                        drawStage();
        },false);
        window.onbeforeunload = function() {
                return "Everything will be lost if you don't save your work."
        }

        var handlePath = "<?php echo Yii::app()->request->baseUrl; ?>/images/rotate.png";
</script>

<div id="main-div">

<?php
        $model = new TopologyModel;

        $form=$this->beginWidget('CActiveForm', array(
                'id'=>'topology-model-form',
                // Please note: When you enable ajax validation, make sure the corresponding
                // controller action is handling ajax validation correctly.
                // See class documentation of CActiveForm for details on this,
                // you need to use the performAjaxValidation()-method described there.
                'enableAjaxValidation'=>false,
        ));

        echo '<div id="topology-div">';
                echo '<div id="control-div">';

                        echo '<h1>'.$course->code.'</h1>';

                        echo '<h4>Controls</h4>';

                        echo '<div id="tab-controls">';
                                echo '<a id="devices-link" class="active" onclick="activate(\'
                                    devices\')">Devices</a>';
                                echo '<a id="cables-link" onclick="activate(\'cables\')">Misc
                                    </a>';
```

```
                        echo '</div>';

                        echo '<div id="tab-contents">';
                                echo '<div id="devices-tab" class="tab-content active">';
                                        $array = array("Choose Device");
                                        $devices = DeviceModel::model()->findAll();
                                        foreach($devices as $device) {
                                                if($device->connectable == "Yes") {
                                                        $file = Yii::app()->basePath.'/devices
                                                            /'.$device->file;
                                                        $url = Yii::app()->assetManager->
                                                            publish($file);
                                                        $array[$url] = $device->name;
                                                }
                                        }

                                        echo $form->dropDownList($model,'device',$array,array(
                                                'style'=>'padding: 1px'
                                        ));

                                        echo CHtml::button('Add',array(
                                                'onclick'=>'return add()',
                                        ));

                                        echo CHtml::submitButton('Save',array(
                                                'onclick'=>'return save(event)'
                                        ));

                                        echo CHtml::button('Back',array(
                                                'onclick'=>'window.location.assign("'.Yii::app
                                                    ()->createUrl('/course/admin').'")'
                                        ));
                                echo '</div>';

                                echo '<div id="cables-tab" class="tab-content">';
                                        $cableArray = array("Default Cable", "Loopback Cable")
                                            ;
                                        $cables = DeviceModel::model()->findAll();
                                        foreach($cables as $cable) {
                                                if($cable->connectable == "No") {
                                                        $cableFile = Yii::app()->basePath.'/
                                                            devices/'.$cable->file;
                                                        $cableUrl = Yii::app()->assetManager->
                                                            publish($cableFile);
                                                        $cableArray[$cableUrl] = $cable->name;
                                                }
                                        }

                                        echo $form->dropDownList($model,'cable',$cableArray,
                                            array(
                                                'style'=>'padding: 1px'
                                        ));

                                        echo CHtml::button('Add',array(
                                                'onclick'=>'return addCable()',
                                        ));

                                        echo CHtml::submitButton('Save',array(
                                                'onclick'=>'return save(event)'
                                        ));

                                        echo CHtml::button('Back',array(
                                                'onclick'=>'window.location.assign("'.Yii::app
                                                    ()->createUrl('/course/admin').'")'
                                        ));
                                echo '</div>';
                        echo '</div>';

                echo '</div>';
?>
                <div id="canvas-div">
                        <div id="container">
                                <svg id="topology-canvas">
                                        Your browser does not support Scalable Vector Graphics
                                </svg>
                        </div>
                </div>
        </div>

<?php
        $this->endWidget();

        if($course->has_topology == "Yes") {
                $lines = CableModel::model()->findAllByAttributes(array(
                        'course_code'=>$course->code
                ));

                echo "<script type='text/javascript'>
                        drawStage();";

                foreach($lines as $line) {
                        $src = null;
                        if(isset($line->device_name)) {
                                $device = DeviceModel::model()->findByPk($line->device_name);
                                $file = Yii::app()->basePath.'/devices/'.$device->file;
                                $src = Yii::app()->assetManager->publish($file);
                        }
                        echo "addCable('".$line->id."',
                                '".$line->type."',
```

```
                                        '".$line->path."',
                                        '".$line->port_a."',
                                        '".$line->port_b."',
                                        '".$line->color."',
                                        '".$src."',
                                        '".$line->transform."',
                                        '".$line->port_a_coor."',
                                        '".$line->port_b_coor."');";
                        }

                        echo "</script>";

                        $draggables = TopologyModel::model()->findAllByAttributes(array(
                                'course_code'=>$course->code
                        ));

                        foreach($draggables as $draggable) {
                                if(!strpos($draggable->device_name,"Cable")) {
                                        //$device = DeviceModel::model()->findByPk($draggable->
                                            device_name);
                                        $file = Yii::app()->basePath.'/devices/'.$draggable->
                                            deviceModel->file;
                                        $src = Yii::app()->assetManager->publish($file);
                                        echo "<script type='text/javascript'>
                                                add('".$draggable->id."',
                                                        '".$draggable->device_name."',
                                                        '".$src."',
                                                        '".$draggable->x."',
                                                        '".$draggable->y."',
                                                        '".$draggable->width."',
                                                        '".$draggable->height."',
                                                        '".$draggable->type."',
                                                        '".$draggable->ip."',
                                                        '".$draggable->host."',
                                                        '".$draggable->display."');
                                        </script>";
                                }
                        }
                }
?>

</div><!-- main-div -->
```

Listing 34: Update Course View

```
<?php
/* @var $this CourseController */
/* @var $model CourseModel */

$this->breadcrumbs=array(
        'Courses'=>array('index'),
        $model->code=>array('view','id'=>$model->code),
        'Update',
);

$this->menu=array(
        array('label'=>'List Courses', 'url'=>array('index')),
        array('label'=>'Create Course', 'url'=>array('create')),
        array('label'=>'View Course', 'url'=>array('view', 'id'=>$model->code)),
        array('label'=>'Manage Courses', 'url'=>array('admin')),
        array('label'=>'Topology', 'url'=>array('topology','id'=>$model->code)),
);
?>

<h1>Update <?php echo strtoupper($model->code); ?></h1>

<?php $this->renderPartial('_form', array('model'=>$model)); ?>
```

Listing 35: View Course Widget

```
<?php
/* @var $this CourseController */
/* @var $model CourseModel */

$this->breadcrumbs=array(
        'Courses'=>array('index'),
        $model->code,
);

$this->menu=array(
        array('label'=>'List Courses', 'url'=>array('index')),
        array('label'=>'Create Course', 'url'=>array('create')),
        array('label'=>'Update Course', 'url'=>array('update', 'id'=>$model->code)),
        array('label'=>'Delete Course', 'url'=>'#', 'linkOptions'=>array('submit'=>array('
            delete','id'=>$model->code),'confirm'=>'Are you sure you want to delete this
            course?')),
        array('label'=>'Manage Courses', 'url'=>array('admin')),
        array('label'=>'Topology', 'url'=>array('topology','id'=>$model->code)),
);
?>

<h1>View <?php echo strtoupper($model->code); ?></h1>

<?php $this->widget('zii.widgets.CDetailView', array(
        'data'=>$model,
        'attributes'=>array(
```

```
                'code',
                'name',
                'students_per_pod',
                'paired',
        ),
)); ?>
```

Listing 36: View Course Layout

```
<?php
/* @var $this CourseController */
/* @var $data CourseModel */
?>

<div class="view">

        <b><?php echo CHtml::encode($data->getAttributeLabel('code')); ?>:</b>
        <?php echo CHtml::link(CHtml::encode($data->code), array('view', 'id'=>$data->code));
             ?>
        <br />

        <b><?php echo CHtml::encode($data->getAttributeLabel('name')); ?>:</b>
        <?php echo CHtml::encode($data->name); ?>
        <br />

        <b><?php echo CHtml::encode($data->getAttributeLabel('students_per_pod')); ?>:</b>
        <?php echo CHtml::encode($data->students_per_pod); ?>
        <br />

        <b><?php echo CHtml::encode($data->getAttributeLabel('paired')); ?>:</b>
        <?php echo CHtml::encode($data->paired); ?>
        <br />


</div>
```

E..3 Device View

Listing 37: Device Admin View

```
<?php
/* @var $this DeviceController */
/* @var $model DeviceModel */

$this->breadcrumbs=array(
        'Devices'=>array('index'),
        'Manage',
);

$this->menu=array(
        array('label'=>'List Devices', 'url'=>array('index')),
        array('label'=>'Create Device', 'url'=>array('create')),
);

Yii::app()->clientScript->registerScript('search', "
$('.search-button').click(function(){
        $('.search-form').toggle();
        return false;
});
$('.search-form form').submit(function(){
        $('#device-model-grid').yiiGridView('update', {
                data: $(this).serialize()
        });
        return false;
});
");
?>

<h1>Manage Devices</h1>

<p>
You may optionally enter a comparison operator (<b>&lt;</b>, <b>&lt;=</b>, <b>&gt;</b>, <b>&gt
    ;=</b>, <b>&lt;&gt;</b>
or <b>=</b>) at the beginning of each of your search values to specify how the comparison
    should be done.
</p>

<?php echo CHtml::link('Advanced Search','#',array('class'=>'search-button')); ?>
<div class="search-form" style="display:none">
<?php $this->renderPartial('_search',array(
        'model'=>$model,
)); ?>
</div><!-- search-form -->

<?php $this->widget('zii.widgets.grid.CGridView', array(
        'id'=>'device-model-grid',
        'dataProvider'=>$model->search(),
        'filter'=>$model,
        'columns'=>array(
                'name',
                'file',
                'connectable',
                array(
                        'header'=>'Options',
                        'class'=>'CButtonColumn',
```

```
                ),
        ),
)); ?>
```

Listing 38: Create Device View

```
<?php
/* @var $this DeviceController */
/* @var $model DeviceModel */

$this->breadcrumbs=array(
        'Devices'=>array('index'),
        'Create',
);

$this->menu=array(
        array('label'=>'List Devices', 'url'=>array('index')),
        array('label'=>'Manage Devices', 'url'=>array('admin')),
);
?>

<h1>Create Device</h1>

<?php $this->renderPartial('_form', array('model'=>$model)); ?>
```

Listing 39: Device Form View

```
<?php
/* @var $this DeviceController */
/* @var $model DeviceModel */
/* @var $form CActiveForm */
?>

<div class="form">

<?php $form=$this->beginWidget('CActiveForm', array(
        'id'=>'device-model-form',
        // Please note: When you enable ajax validation, make sure the corresponding
        // controller action is handling ajax validation correctly.
        // There is a call to performAjaxValidation() commented in generated controller code.
        // See class documentation of CActiveForm for details on this.
        'enableAjaxValidation'=>false,
        'htmlOptions' =>array('enctype' => 'multipart/form-data'),
)); ?>

        <p class="note">Fields with <span class="required">*</span> are required.</p>

        <?php echo $form->errorSummary($model); ?>

        <div class="row">
                <?php echo $form->labelEx($model,'name'); ?>
                <?php echo $form->textField($model,'name',array('size'=>60,'maxlength'=>64));
                    ?>
                <?php echo $form->error($model,'name'); ?>
        </div>

        <div class="row">
                <?php echo $form->labelEx($model,'file'); ?>
                <?php
                        $file=Yii::app()->basePath.'/devices/'.$model->file;
                        if(!$model->isNewRecord && file_exists($file)) {
                                $src=Yii::app()->assetManager->publish($file);
                                echo '<div class="rows">
                                        <img src="'.$src.'" style="padding: 10px" />
                                </div>';
                        } else {
                                echo $form->fileField($model,'file');
                        }
                ?>
                <?php echo $form->error($model,'file'); ?>
        </div>

        <div class="row">
                <?php echo $form->labelEx($model,'connectable'); ?>
                <?php echo $form->dropDownList($model,'connectable',array(
                        'Yes'=>'Yes',
                        'No'=>'No'
                )); ?>
                <?php echo $form->error($model,'connectable'); ?>
        </div>

        <div class="row buttons">
                <?php echo CHtml::submitButton($model->isNewRecord ? 'Create' : 'Save'); ?>
        </div>

<?php $this->endWidget(); ?>

</div><!-- form -->
```

Listing 40: Device Index View

```
<?php
/* @var $this DeviceController */
/* @var $dataProvider CActiveDataProvider */
```

```
$this->breadcrumbs=array(
        'Devices',
);

$this->menu=array(
        array('label'=>'Create Device', 'url'=>array('create')),
        array('label'=>'Manage Devices', 'url'=>array('admin')),
);
?>

<h1>Devices</h1>

<?php $this->widget('zii.widgets.CListView', array(
        'dataProvider'=>$dataProvider,
        'itemView'=>'_view',
)); ?>
```

Listing 41: Device Search View

```
<?php
/* @var $this DeviceController */
/* @var $model DeviceModel */
/* @var $form CActiveForm */
?>

<div class="wide form">

<?php $form=$this->beginWidget('CActiveForm', array(
        'action'=>Yii::app()->createUrl($this->route),
        'method'=>'get',
)); ?>

        <div class="row">
                <?php echo $form->label($model,'name'); ?>
                <?php echo $form->textField($model,'name',array('size'=>60,'maxlength'=>64));
                    ?>
        </div>

        <div class="row">
                <?php echo $form->label($model,'file'); ?>
                <?php echo $form->textField($model,'file',array('size'=>60,'maxlength'=>128));
                    ?>
        </div>

        <div class="row buttons">
                <?php echo CHtml::submitButton('Search'); ?>
        </div>

<?php $this->endWidget(); ?>

</div><!-- search-form -->
```

Listing 42: Update Device View

```
<?php
/* @var $this DeviceController */
/* @var $model DeviceModel */

$this->breadcrumbs=array(
        'Devices'=>array('index'),
        $model->name=>array('view','id'=>$model->name),
        'Update',
);

$this->menu=array(
        array('label'=>'List Devices', 'url'=>array('index')),
        array('label'=>'Create Device', 'url'=>array('create')),
        array('label'=>'View Device', 'url'=>array('view', 'id'=>$model->name)),
        array('label'=>'Manage Devices', 'url'=>array('admin')),
);
?>

<h1>Update <?php echo strtoupper($model->name); ?></h1>

<?php $this->renderPartial('_form', array('model'=>$model)); ?>
```

Listing 43: View Device Widget

```
<?php
/* @var $this ClassController */
/* @var $model ClassModel */

$this->breadcrumbs=array(
        'Classes'=>array('index'),
        $model->id,
);

$this->menu=array(
        array('label'=>'List Classes', 'url'=>array('index')),
        array('label'=>'Create Class', 'url'=>array('create')),
        array('label'=>'Update Class', 'url'=>array('update', 'id'=>$model->id)),
        array('label'=>'Delete Class', 'url'=>'#', 'linkOptions'=>array('submit'=>array('
            delete','id'=>$model->id),'confirm'=>'Are you sure you want to delete this class
            ?')),
```

```
        array('label'=>'Manage Classes', 'url'=>array('admin')),
        array('label'=>'Pods', 'url'=>array('/pod/admin', 'class_id'=>$model->id)),
);
?>

<h1>View Class #<?php echo $model->id; ?></h1>

<?php $this->widget('zii.widgets.CDetailView', array(
        'data'=>$model,
        'attributes'=>array(
                'id',
                'course_code',
                'pass',
                'prof',
                'ts_ip',
                'ts_user',
                'ts_pass',
                'num_of_students',
        ),
)); ?>
```

Listing 44: View Device Layout

```
<?php
/* @var $this ClassController */
/* @var $data ClassModel */
?>

<div class="view">

        <b><?php echo CHtml::encode($data->getAttributeLabel('id')); ?>:</b>
        <?php echo CHtml::link(CHtml::encode($data->id), array('view', 'id'=>$data->id)); ?>
        <br />

        <b><?php echo CHtml::encode($data->getAttributeLabel('course_code')); ?>:</b>
        <?php echo CHtml::encode($data->course_code); ?>
        <br />

        <b><?php echo CHtml::encode($data->getAttributeLabel('pass')); ?>:</b>
        <?php echo CHtml::encode($data->pass); ?>
        <br />

        <b><?php echo CHtml::encode($data->getAttributeLabel('prof')); ?>:</b>
        <?php echo CHtml::encode($data->prof); ?>
        <br />

        <b><?php echo CHtml::encode($data->getAttributeLabel('ts_ip')); ?>:</b>
        <?php echo CHtml::encode($data->ts_ip); ?>
        <br />

        <b><?php echo CHtml::encode($data->getAttributeLabel('ts_user')); ?>:</b>
        <?php echo CHtml::encode($data->ts_user); ?>
        <br />

        <b><?php echo CHtml::encode($data->getAttributeLabel('ts_pass')); ?>:</b>
        <?php echo CHtml::encode($data->ts_pass); ?>
        <br />

        <?php /*
        <b><?php echo CHtml::encode($data->getAttributeLabel('num_of_students')); ?>:</b>
        <?php echo CHtml::encode($data->num_of_students); ?>
        <br />

        <b><?php echo CHtml::encode($data->getAttributeLabel('has_pods')); ?>:</b>
        <?php echo CHtml::encode($data->has_pods); ?>
        <br />

        */ ?>

</div>
```

E..4 Pod View

Listing 45: Pod Admin View

```
<?php
/* @var $this PodController */
/* @var $model PodModel */

$this->breadcrumbs=array(
        'Pods'=>array('index','class_id'=>$class_id),
        'Manage',
);

$this->menu=array(
        array('label'=>'List Pods', 'url'=>array('index', 'class_id'=>$class_id)),
        array('label'=>'Create Pod', 'url'=>array('create', 'class_id'=>$class_id)),
);

Yii::app()->clientScript->registerScript('search', "
$('.search-button').click(function(){
        $('.search-form').toggle();
        return false;
});
$('.search-form form').submit(function(){
```

```
        $('#pod-model-grid').yiiGridView('update', {
                data: $(this).serialize()
        });
        return false;
});
");
?>

<h1>Manage Pods</h1>

<p>
You may optionally enter a comparison operator (<b>&lt;</b>, <b>&lt;=</b>, <b>&gt;</b>, <b>&gt
    ;=</b>, <b>&lt;&gt;</b>
or <b>=</b>) at the beginning of each of your search values to specify how the comparison
    should be done.
</p>

<?php echo CHtml::link('Advanced Search','#',array('class'=>'search-button')); ?>
<div class="search-form" style="display:none">
<?php $this->renderPartial('_search',array(
        'model'=>$model,
)); ?>
</div><!-- search-form -->

<?php $this->widget('zii.widgets.grid.CGridView', array(
        'id'=>'pod-model-grid',
        'dataProvider'=>$model->search($class_id),
        'filter'=>$model,
        'columns'=>array(
                'pod_num',
                //'class_id',
                'pass',
                array(
                        'class'=>'CButtonColumn',
                ),
        ),
)); ?>
```

Listing 46: Create Pod View

```
<?php
/* @var $this PodController */
/* @var $model PodModel */

$this->breadcrumbs=array(
        'Pods'=>array('index','class_id'=>$model->class_id),
        'Create',
);

$this->menu=array(
        array('label'=>'List Pods', 'url'=>array('index', 'class_id'=>$model->class_id)),
        array('label'=>'Manage Pods', 'url'=>array('admin', 'class_id'=>$model->class_id)),
);
?>

<h1>Create Pod</h1>

<?php $this->renderPartial('_form', array('model'=>$model)); ?>
```

Listing 47: Pod Form View

```
<?php
/* @var $this PodController */
/* @var $model PodModel */
/* @var $form CActiveForm */
?>

<div class="form">

<?php
        $form=$this->beginWidget('CActiveForm', array(
                'id'=>'pod-model-form',
                // Please note: When you enable ajax validation, make sure the corresponding
                // controller action is handling ajax validation correctly.
                // There is a call to performAjaxValidation() commented in generated
                    controller code.
                // See class documentation of CActiveForm for details on this.
                'enableAjaxValidation'=>false,
        ));

        if($model->isNewRecord) {
                $num_of_pods = ceil($model->class->num_of_students / $model->class->course->
                    students_per_pod);
                $num_of_pods *= $model->class->course->paired === "Yes" ? 2 : 1;

                if(count($model->class->pods) == $num_of_pods) {
                        Yii::app()->user->setFlash('error','Maximum number of pods reached. To
                            add more pods, please add more students.');
                        $this->redirect(array('/pod/admin', 'class_id' => $model->class_id));
                }
        }
?>

        <p class="note">Fields with <span class="required">*</span> are required.</p>

        <?php echo $form->errorSummary($model); ?>
```

```
        <div class="row">
                <?php echo $form->labelEx($model,'pod_num'); ?>
                <?php echo $form->numberField($model,'pod_num',array('min'=>1)); ?>
                <?php echo $form->error($model,'pod_num'); ?>
        </div>

        <div class="row">
                <?php echo $form->labelEx($model,'pass'); ?>
                <?php echo $form->textField($model,'pass',array('size'=>32,'maxlength'=>32));
                    ?>
                <?php echo $form->error($model,'pass'); ?>
        </div>

        <div class="row buttons">
                <?php echo CHtml::submitButton($model->isNewRecord ? 'Create' : 'Save'); ?>
        </div>

<?php $this->endWidget(); ?>

</div><!-- form -->
```

Listing 48: Pod Index View

```
<?php
/* @var $this PodController */
/* @var $dataProvider CActiveDataProvider */

$this->breadcrumbs=array(
        'Pods',
);

$this->menu=array(
        array('label'=>'Create Pod', 'url'=>array('create', 'class_id'=>$class_id)),
        array('label'=>'Manage Pods', 'url'=>array('admin', 'class_id'=>$class_id)),
);
?>

<h1>Pods</h1>

<?php $this->widget('zii.widgets.CListView', array(
        'dataProvider'=>$model->search($class_id),
        'itemView'=>'_view',
)); ?>
```

Listing 49: Pod Search View

```
<?php
/* @var $this PodController */
/* @var $model PodModel */
/* @var $form CActiveForm */
?>

<div class="wide form">

<?php $form=$this->beginWidget('CActiveForm', array(
        'action'=>Yii::app()->createUrl($this->route),
        'method'=>'get',
)); ?>

        <div class="row">
                <?php echo $form->label($model,'pod_num'); ?>
                <?php echo $form->textField($model,'pod_num'); ?>
        </div>

        <div class="row">
                <?php echo $form->label($model,'pass'); ?>
                <?php echo $form->passwordField($model,'pass',array('size'=>32,'maxlength
                    '=>32)); ?>
        </div>

        <div class="row buttons">
                <?php echo CHtml::submitButton('Search'); ?>
        </div>

<?php $this->endWidget(); ?>

</div><!-- search-form -->
```

Listing 50: Update Pod View

```
<?php
/* @var $this PodController */
/* @var $model PodModel */

$this->breadcrumbs=array(
        'Pods'=>array('index','class_id'=>$model->class_id),
        $model->id=>array('view','id'=>$model->id),
        'Update',
);

$this->menu=array(
        array('label'=>'List Pods', 'url'=>array('index', 'class_id'=>$model->class_id)),
        array('label'=>'Create Pod', 'url'=>array('create', 'class_id'=>$model->class_id)),
```

```
        array('label'=>'View Pod', 'url'=>array('view', 'id'=>$model->id)),
        array('label'=>'Manage Pods', 'url'=>array('admin', 'class_id'=>$model->class_id)),
        array('label'=>'Pod Devices', 'url'=>array('pods', 'id'=>$model->id)),
);
?>

<h1>Update Pod #<?php echo $model->pod_num; ?></h1>

<?php $this->renderPartial('_form', array('model'=>$model)); ?>
```

Listing 51: View Pod Widget

```
<?php
/* @var $this PodController */
/* @var $model PodModel */

$this->breadcrumbs=array(
        'Pods'=>array('index','class_id'=>$model->class_id),
        $model->id,
);

$this->menu=array(
        array('label'=>'List Pods', 'url'=>array('index', 'class_id'=>$model->class_id)),
        array('label'=>'Create Pod', 'url'=>array('create', 'class_id'=>$model->class_id)),
        array('label'=>'Update Pod', 'url'=>array('update', 'id'=>$model->id)),
        array('label'=>'Delete Pod', 'url'=>'#', 'linkOptions'=>array('submit'=>array('delete
            ','id'=>$model->id),'confirm'=>'Are you sure you want to delete this pod?')),
        array('label'=>'Manage Pods', 'url'=>array('admin', 'class_id'=>$model->class_id)),
        array('label'=>'Pod Devices', 'url'=>array('pods', 'id'=>$model->id)),
);
?>

<h1>View Pod #<?php echo $model->pod_num; ?></h1>

<?php $this->widget('zii.widgets.CDetailView', array(
        'data'=>$model,
        'attributes'=>array(
                'pod_num',
                //'class_id',
                'pass',
        ),
)); ?>
```

Listing 52: View Pod Layout

```
<?php
/* @var $this PodController */
/* @var $data PodModel */
?>

<div class="view">

        <b><?php echo CHtml::encode($data->getAttributeLabel('pod_num')); ?>:</b>
        <?php echo CHtml::link($data->pod_num, array('view', 'id'=>$data->id)); ?>
        <br />

        <b><?php echo CHtml::encode($data->getAttributeLabel('pass')); ?>:</b>
        <?php echo CHtml::encode($data->pass); ?>
        <br />


</div>
```

E..5 Setup View

Listing 53: Administrator Setup View

```
<?php
/* @var $this AdminFormController */
/* @var $model AdminForm */
/* @var $form CActiveForm */
?>

<h1>Create Administrator Account</h1>

<div class="form">

<?php $form=$this->beginWidget('CActiveForm', array(
        'id'=>'admin-form',
        // Please note: When you enable ajax validation, make sure the corresponding
        // controller action is handling ajax validation correctly.
        // See class documentation of CActiveForm for details on this,
        // you need to use the performAjaxValidation()-method described there.
        'enableAjaxValidation'=>false,
)); ?>

        <p class="note">Fields with <span class="required">*</span> are required.</p>

        <?php echo $form->errorSummary($model); ?>

        <div class="row">
                <?php echo $form->labelEx($model,'employee_id'); ?>
                <?php echo $form->textField($model,'employee_id'); ?>
```

```
                <?php echo $form->error($model,'employee_id'); ?>
        </div>

        <div class="row">
                <?php echo $form->labelEx($model,'pass'); ?>
                <?php echo $form->passwordField($model,'pass',array('value'=>'')); ?>
                <?php echo $form->error($model,'pass'); ?>
        </div>

        <div class="row">
                <?php echo $form->labelEx($model,'retype'); ?>
                <?php echo $form->passwordField($model,'retype',array('value'=>'')); ?>
                <?php echo $form->error($model,'retype'); ?>
        </div>

        <div class="row">
                <?php echo $form->labelEx($model,'first_name'); ?>
                <?php echo $form->textField($model,'first_name'); ?>
                <?php echo $form->error($model,'first_name'); ?>
        </div>

        <div class="row">
                <?php echo $form->labelEx($model,'last_name'); ?>
                <?php echo $form->textField($model,'last_name'); ?>
                <?php echo $form->error($model,'last_name'); ?>
        </div>


        <div class="row buttons">
                <?php echo CHtml::submitButton('Create'); ?>
        </div>

<?php $this->endWidget(); ?>

</div><!-- form -->
```

Listing 54: Database Setup View

```
<?php
/* @var $this DatabaseFormController */
/* @var $model DatabaseForm */
/* @var $form CActiveForm */
?>

<h1>Configure Database</h1>

<div class="form">

<?php $form=$this->beginWidget('CActiveForm', array(
        'id'=>'database-form',
        // Please note: When you enable ajax validation, make sure the corresponding
        // controller action is handling ajax validation correctly.
        // See class documentation of CActiveForm for details on this,
        // you need to use the performAjaxValidation()-method described there.
        'enableAjaxValidation'=>false,
)); ?>

        <p class="note">Fields with <span class="required">*</span> are required.</p>

        <?php echo $form->errorSummary($model); ?>

        <div class="row">
                <?php echo $form->labelEx($model,'host'); ?>
                <?php echo $form->textField($model,'host'); ?>
                <?php echo $form->error($model,'host'); ?>
        </div>

        <div class="row">
                <?php echo $form->labelEx($model,'user'); ?>
                <?php echo $form->textField($model,'user'); ?>
                <?php echo $form->error($model,'user'); ?>
        </div>

        <div class="row">
                <?php echo $form->labelEx($model,'pass'); ?>
                <?php echo $form->textField($model,'pass'); ?>
                <?php echo $form->error($model,'pass'); ?>
        </div>


        <div class="row buttons">
                <?php echo CHtml::submitButton('Connect'); ?>
        </div>

<?php $this->endWidget(); ?>

</div><!-- form -->
```

E..6 Site View

Listing 55: Site Index View

```
<?php
/* @var $this SiteController */
```

```
$this->pageTitle=Yii::app()->name;
?>

<script type="text/javascript" src="<?php echo Yii::app()->request->baseUrl; ?>/js/topology.js
    "></script>
<script type="text/javascript" src="<?php echo Yii::app()->request->baseUrl; ?>/js/jquery.
    cycle2.js"></script>
<script type="text/javascript" src="<?php echo Yii::app()->request->baseUrl; ?>/js/jquery.
    cycle2.center.js"></script>

<script type="text/javascript">
        var url = "<?php echo Yii::app()->getBaseUrl(true); ?>/index.php?r=site/subjects";

        $("#student-form").ready(function() {
                getSubjects("<?php echo Yii::app()->user->isGuest && isset($model) ? $model->
                    prof : 0; ?>", url);
        });
</script>

<?php if(!Yii::app()->user->isGuest) :?>

        <h1>
                Welcome to <i>
                        <?php
                                if(Yii::app()->user->isStdnt()) {
                                        $class = ClassModel::model()->findByPk(Yii::app()->
                                            user->getState('class'));
                                        echo CHtml::encode($class->course->name);
                                } else {
                                        echo CHtml::encode(Yii::app()->name);
                                }
                        ?>
                </i>
        </h1>

        <?php if(Yii::app()->user->isStdnt()) : ?>

                <h3>
                        by <?php
                                $prof = UserModel::model()->findByPk(Yii::app()->user->prof);
                                echo $prof->user_type.' '.$prof->first_name.' '.$prof->
                                    last_name;
                        ?>
                </h3>

                <?php
                /* @var $this SiteController */
                /* @var $model PodForm */
                /* @var $form CActiveForm */

                $podArray = array('Select Pod');
                $pods = CHtml::listData(
                        PodModel::model()->findAllByAttributes(array('class_id'=>Yii::app()->
                            user->getState('class'))),
                        'id',
                        'pod_num'
                );

                if(count($pods) > 0) {
                        foreach($pods as $key => $value)
                                $podArray[$key] = $value;
                } else {
                        Yii::app()->user->setFlash('error','This class has no pods yet.');
                }
                ?>

                <p>Please choose a pod & enter the pod's password:</p>

                <div class="form login">

                <?php $form=$this->beginWidget('CActiveForm', array(
                        'id'=>'pod-form',
                        // Please note: When you enable ajax validation, make sure the
                            corresponding
                        // controller action is handling ajax validation correctly.
                        // See class documentation of CActiveForm for details on this,
                        // you need to use the performAjaxValidation()-method described there.
                        'enableAjaxValidation'=>false,
                )); ?>

                        <p class="note">Fields with <span class="required">*</span> are
                            required.</p>

                        <div class="row">
                                <?php echo $form->labelEx($model,'podNum'); ?>
                                <?php echo $form->dropDownList($model,'podNum',$podArray); ?>
                                <?php echo $form->error($model,'podNum'); ?>
                        </div>

                        <div class="row">
                                <?php echo $form->labelEx($model,'pass'); ?>
                                <?php echo $form->passwordField($model,'pass',array('value
                                    '=>''')); ?>
                                <?php echo $form->error($model,'pass'); ?>
                        </div>


                        <div class="row buttons">
                                <?php echo CHtml::submitButton('Submit'); ?>
```

```
                        </div>

                <?php $this->endWidget(); ?>

                </div><!-- form -->

        <?php else : ?>
                <center>
                <!-- tileBlind -->
                <div class="cycle-slideshow"
                        data-cycle-speed=1000
                        data-cycle-timeout=1500
                        data-cycle-center-horz=true
                >
                        <img src="<?php echo Yii::app()->request->baseUrl; ?>/images/troom1.
                            png">
                        <img src="<?php echo Yii::app()->request->baseUrl; ?>/images/troom2.
                            png">
                        <img src="<?php echo Yii::app()->request->baseUrl; ?>/images/troom3.
                            png">
                </div>
                </center>
        <?php endif; ?>

<?php else : ?>

        <?php
        /* @var $this SiteController */
        /* @var $model StudentForm */
        /* @var $form CActiveForm */

        $profArray = array('Select Professor');
        $profs = CHtml::listData(
                UserModel::model()->findAllByAttributes(array('user_type'=>'Professor')),
                'employee_id',
                'first_name'
        );

        if(count($profs) > 0) {
                foreach($profs as $key => $value) {
                        $user = UserModel::model()->findByPk($key);
                        $profArray[$key] = $value.' '.$user->last_name;
                }
        } else {
                Yii::app()->user->setFlash('error','No professors found.');
        }
        ?>

        <h1>Student Login</h1>

        <p>Please fill out the following form with your login credentials:</p>

        <div class="form login">

        <?php $form=$this->beginWidget('CActiveForm', array(
                'id'=>'student-form',
                // Please note: When you enable ajax validation, make sure the corresponding
                // controller action is handling ajax validation correctly.
                // See class documentation of CActiveForm for details on this,
                // you need to use the performAjaxValidation()-method described there.
                'enableAjaxValidation'=>false,
        )); ?>

                <p class="note">Fields with <span class="required">*</span> are required.</p>

                <div class="row">
                        <?php echo $form->labelEx($model,'prof'); ?>
                        <?php echo $form->dropDownList($model,'prof',$profArray,array(
                                'onChange'=>'getSubjects(this.value, url)'
                        )); ?>
                        <?php echo $form->error($model,'prof'); ?>
                </div>

                <div class="row">
                        <?php echo $form->labelEx($model,'subj'); ?>
                        <?php echo $form->dropDownList($model,'subj',array()); ?>
                        <?php echo $form->error($model,'subj'); ?>
                </div>

                <div class="row">
                        <?php echo $form->labelEx($model,'pass'); ?>
                        <?php echo $form->passwordField($model,'pass',array('value'=>'')); ?>
                        <?php echo $form->error($model,'pass'); ?>
                </div>


                <div class="row buttons">
                        <?php echo CHtml::submitButton('Login'); ?>
                </div>

        <?php $this->endWidget(); ?>

        </div><!-- form -->

<?php endif; ?>
```

Listing 56: Site Login View

```
<?php
/* @var $this SiteController */
/* @var $model LoginForm */
/* @var $form CActiveForm  */

$this->pageTitle=Yii::app()->name . ' - Login';
?>

<h1>Login</h1>

<p>Please fill out the following form with your login credentials:</p>

<div class="form login">
<?php $form=$this->beginWidget('CActiveForm', array(
        'id'=>'login-form',
        'enableClientValidation'=>true,
        'clientOptions'=>array(
                'validateOnSubmit'=>true,
        ),
)); ?>

        <p class="note">Fields with <span class="required">*</span> are required.</p>

        <div class="row">
                <?php echo $form->labelEx($model,'username'); ?>
                <?php echo $form->textField($model,'username'); ?>
                <?php echo $form->error($model,'username'); ?>
        </div>

        <div class="row">
                <?php echo $form->labelEx($model,'password'); ?>
                <?php echo $form->passwordField($model,'password',array('value'=>'')); ?>
                <?php echo $form->error($model,'password'); ?>
        </div>

        <div class="row buttons">
                <?php echo CHtml::submitButton('Login'); ?>
        </div>

<?php $this->endWidget(); ?>
</div><!-- form -->
```

E..7 User View

Listing 57: User Admin View

```
<?php
/* @var $this UserController */
/* @var $model UserModel */

$this->breadcrumbs=array(
        'Users'=>array('index'),
        'Manage',
);

$this->menu=array(
        array('label'=>'List Users', 'url'=>array('index')),
        array('label'=>'Create User Account', 'url'=>array('create')),
);

Yii::app()->clientScript->registerScript('search', "
$('.search-button').click(function(){
        $('.search-form').toggle();
        return false;
});
$('.search-form form').submit(function(){
        $('#user-model-grid').yiiGridView('update', {
                data: $(this).serialize()
        });
        return false;
});
");
?>

<h1>Manage Users</h1>

<p>
You may optionally enter a comparison operator (<b>&lt;</b>, <b>&lt;=</b>, <b>&gt;</b>, <b>&gt
    ;=</b>, <b>&lt;&gt;</b>
or <b>=</b>) at the beginning of each of your search values to specify how the comparison
    should be done.
</p>

<?php echo CHtml::link('Advanced Search','#',array('class'=>'search-button')); ?>
<div class="search-form" style="display:none">
<?php $this->renderPartial('_search',array(
        'model'=>$model,
)); ?>
</div><!-- search-form -->

<?php $this->widget('zii.widgets.grid.CGridView', array(
        'id'=>'user-model-grid',
        'dataProvider'=>$model->search(),
        'filter'=>$model,
        'columns'=>array(
```

```
                'employee_id',
                'user_type',
                array(
                        'header'=>$model->getAttributeLabel('name'),
                        'type'=>'raw',
                        'value'=>'CHtml::tag("center",array(),$data->last_name.", ".$data->
                            first_name)',
                        'filter' => CHtml::activeTextField($model, 'name')
                ),
                array(
                        'header'=>'Options',
                        'class'=>'CButtonColumn',
                ),
        ),
)); ?>
```

Listing 58: Change User Password View

```
<?php
/* @var $this UserController */
/* @var $model ChangePasswordForm */
/* @var $form CActiveForm */

$this->menu=array(
        array('label'=>'List Users', 'url'=>array('index')),
        array('label'=>'Create User Account', 'url'=>array('create')),
        array('label'=>'Manage Users', 'url'=>array('admin')),
);
?>

<h1>Change Password</h1>

<div class="form">

<?php $form=$this->beginWidget('CActiveForm', array(
        'id'=>'change-password-form',
        // Please note: When you enable ajax validation, make sure the corresponding
        // controller action is handling ajax validation correctly.
        // See class documentation of CActiveForm for details on this,
        // you need to use the performAjaxValidation()-method described there.
        'enableAjaxValidation'=>false,
)); ?>

        <p class="note">Fields with <span class="required">*</span> are required.</p>

        <?php echo $form->errorSummary($model); ?>

        <div class="row">
                <?php echo $form->labelEx($model,'currentPass'); ?>
                <?php echo $form->passwordField($model,'currentPass',array('value'=>'')); ?>
                <?php echo $form->error($model,'currentPass'); ?>
        </div>

        <div class="row">
                <?php echo $form->labelEx($model,'newPass'); ?>
                <?php echo $form->passwordField($model,'newPass',array('value'=>'')); ?>
                <?php echo $form->error($model,'newPass'); ?>
        </div>

        <div class="row">
                <?php echo $form->labelEx($model,'retype'); ?>
                <?php echo $form->passwordField($model,'retype',array('value'=>'')); ?>
                <?php echo $form->error($model,'retype'); ?>
        </div>


        <div class="row buttons">
                <?php echo CHtml::submitButton('Submit'); ?>
        </div>

<?php $this->endWidget(); ?>

</div><!-- form -->
```

Listing 59: Create User Account View

```
<?php
/* @var $this UserController */
/* @var $model UserModel */

$this->breadcrumbs=array(
        'Users'=>array('index'),
        'Create',
);

$this->menu=array(
        array('label'=>'List Users', 'url'=>array('index')),
        array('label'=>'Manage Users', 'url'=>array('admin')),
);
?>

<h1>Create User Account</h1>

<?php $this->renderPartial('_form', array('model'=>$model)); ?>
```

Listing 60: User Form View

```
<?php
/* @var $this UserController */
/* @var $model UserModel */
/* @var $form CActiveForm */
?>

<div class="form">

<?php $form=$this->beginWidget('CActiveForm', array(
        'id'=>'user-model-form',
        // Please note: When you enable ajax validation, make sure the corresponding
        // controller action is handling ajax validation correctly.
        // There is a call to performAjaxValidation() commented in generated controller code.
        // See class documentation of CActiveForm for details on this.
        'enableAjaxValidation'=>false,
)); ?>

        <p class="note">Fields with <span class="required">*</span> are required.</p>

        <?php echo $form->errorSummary($model); ?>

        <div class="row">
                <?php echo $form->labelEx($model,'employee_id'); ?>
                <?php echo $form->textField($model,'employee_id',array('size'=>32,'maxlength
                    '=>32,'readonly'=>$model->isNewRecord?false:true)); ?>
                <?php echo $form->error($model,'employee_id'); ?>
        </div>

        <div class="row">
                <?php echo $form->labelEx($model,'pass'); ?>
                <?php echo $form->passwordField($model,'pass',array('size'=>32,'maxlength
                    '=>32,'value'=>'')); ?>
                <?php echo $form->error($model,'pass'); ?>
        </div>

        <div class="row">
                <?php echo $form->labelEx($model,'retype'); ?>
                <?php echo $form->passwordField($model,'retype',array('size'=>32, 'maxlength
                    '=>32,'value'=>'')); ?>
                <?php echo $form->error($model,'retype'); ?>
        </div>

        <div class="row">
                <?php echo $form->labelEx($model,'user_type'); ?>
                <?php echo $form->dropDownList($model,'user_type',array(
                        '0'=>'Select User Type',
                        'Administrator'=>'Administrator',
                        'Professor'=>'Professor'
                )); ?>
                <?php echo $form->error($model,'user_type'); ?>
        </div>

        <div class="row">
                <?php echo $form->labelEx($model,'first_name'); ?>
                <?php echo $form->textField($model,'first_name',array('size'=>60,'maxlength
                    '=>64)); ?>
                <?php echo $form->error($model,'first_name'); ?>
        </div>

        <div class="row">
                <?php echo $form->labelEx($model,'last_name'); ?>
                <?php echo $form->textField($model,'last_name',array('size'=>60,'maxlength
                    '=>64)); ?>
                <?php echo $form->error($model,'last_name'); ?>
        </div>

        <div class="row buttons">
                <?php echo CHtml::submitButton($model->isNewRecord ? 'Create' : 'Save'); ?>
        </div>

<?php $this->endWidget(); ?>

</div><!-- form -->
```

Listing 61: User Index View

```
<?php
/* @var $this UserController */
/* @var $dataProvider CActiveDataProvider */

$this->breadcrumbs=array(
        'Users',
);

$this->menu=array(
        array('label'=>'Create User Account', 'url'=>array('create')),
        array('label'=>'Manage Users', 'url'=>array('admin')),
);
?>

<h1>Users</h1>

<?php $this->widget('zii.widgets.CListView', array(
        'dataProvider'=>$dataProvider,
        'itemView'=>'_view',
)); ?>
```

Listing 62: User Search View

```
<?php
/* @var $this UserController */
/* @var $model UserModel */
/* @var $form CActiveForm */
?>

<div class="wide form">

<?php $form=$this->beginWidget('CActiveForm', array(
        'action'=>Yii::app()->createUrl($this->route),
        'method'=>'get',
)); ?>

        <div class="row">
                <?php echo $form->label($model,'employee_id'); ?>
                <?php echo $form->textField($model,'employee_id',array('size'=>32,'maxlength
                    '=>32)); ?>
        </div>

        <div class="row">
                <?php echo $form->label($model,'user_type'); ?>
                <?php echo $form->textField($model,'user_type',array('size'=>32,'maxlength
                    '=>13)); ?>
        </div>

        <div class="row">
                <?php echo $form->label($model,'first_name'); ?>
                <?php echo $form->textField($model,'first_name',array('size'=>32,'maxlength
                    '=>64)); ?>
        </div>

        <div class="row">
                <?php echo $form->label($model,'last_name'); ?>
                <?php echo $form->textField($model,'last_name',array('size'=>32,'maxlength
                    '=>64)); ?>
        </div>

        <div class="row buttons">
                <?php echo CHtml::submitButton('Search'); ?>
        </div>

<?php $this->endWidget(); ?>

</div><!-- search-form -->
```

Listing 63: Update User Account View

```
<?php
/* @var $this UserController */
/* @var $model UserModel */

$this->breadcrumbs=array(
        'Users'=>array('index'),
        $model->employee_id=>array('view','id'=>$model->employee_id),
        'Update',
);

$this->menu=array(
        array('label'=>'List Users', 'url'=>array('index')),
        array('label'=>'Create User Account', 'url'=>array('create')),
        array('label'=>'View User', 'url'=>array('view', 'id'=>$model->employee_id)),
        array('label'=>'Manage Users', 'url'=>array('admin')),
);
?>

<h1>Update <?php echo strtoupper($model->last_name.', '.$model->first_name); ?></h1>

<?php $this->renderPartial('_form', array('model'=>$model)); ?>
```

Listing 64: View User Account Widget

```
<?php
/* @var $this UserController */
/* @var $model UserModel */

$this->breadcrumbs=array(
        'Users'=>array('index'),
        $model->employee_id,
);

$this->menu=array(
        array('label'=>'List Users', 'url'=>array('index')),
        array('label'=>'Create User Account', 'url'=>array('create')),
        array('label'=>'Update User', 'url'=>array('update', 'id'=>$model->employee_id)),
        array('label'=>'Delete User', 'url'=>'#', 'linkOptions'=>array('submit'=>array('delete
            ','id'=>$model->employee_id),'confirm'=>'Are you sure you want to delete this user
            ?')),
        array('label'=>'Manage Users', 'url'=>array('admin')),
);
?>

<h1>View <?php echo strtoupper($model->last_name.', '.$model->first_name); ?></h1>

<?php $this->widget('zii.widgets.CDetailView', array(
```

```
        'data'=>$model,
        'attributes'=>array(
                'employee_id',
                'user_type',
                'first_name',
                'last_name',
        ),
)); ?>
```

Listing 65: View User Layout

```
<?php
/* @var $this UserController */
/* @var $data UserModel */
?>

<div class="view">

        <b><?php echo CHtml::encode($data->getAttributeLabel('employee_id')); ?>:</b>
        <?php echo CHtml::link(CHtml::encode($data->employee_id), array('view', 'id'=>$data->
             employee_id)); ?>
        <br />

        <b><?php echo CHtml::encode($data->getAttributeLabel('user_type')); ?>:</b>
        <?php echo CHtml::encode($data->user_type); ?>
        <br />

        <b><?php echo CHtml::encode($data->getAttributeLabel('first_name')); ?>:</b>
        <?php echo CHtml::encode($data->first_name); ?>
        <br />

        <b><?php echo CHtml::encode($data->getAttributeLabel('last_name')); ?>:</b>
        <?php echo CHtml::encode($data->last_name); ?>
        <br />


</div>
```

F. Controllers

Listing 66: Class Controller

```
<?php

class ClassController extends Controller
{
        /**
         * @var string the default layout for the views. Defaults to '//layouts/column2',
              meaning
         * using two-column layout. See 'protected/views/layouts/column2.php'.
         */
        public $layout='//layouts/column2';

        /**
         * @return array action filters
         */
        public function filters()
        {
                return array(
                        'accessControl', // perform access control for CRUD operations
                        'postOnly + delete', // we only allow deletion via POST request
                );
        }

        /**
         * Specifies the access control rules.
         * This method is used by the 'accessControl' filter.
         * @return array access control rules
         */
        public function accessRules()
        {
                return array(
                        array('allow',  // allow prof to perform all actions
                                'actions'=>array('index','view','create','update','admin','
                                     delete'),
                                'expression'=>'Yii::app()->user->isProf()',
                        ),
                        array('allow',  //allow student to perform 'loadClass' and 'connect'
                             actions
                                'actions'=>array('loadClass', 'connect'),
                                'expression'=>'Yii::app()->user->isStdnt()'
                        ),
                        array('deny',  // deny all users
                                'users'=>array('*'),
                        ),
                );
        }

        /**
         * Displays a particular model.
         * @param integer $id the ID of the model to be displayed
         */
        public function actionView($id)
        {
```

```
        $this->render('view',array(
                'model'=>$this->loadModel($id),
        ));
}

/**
 * Creates a new model.
 * If creation is successful, the browser will be redirected to the 'view' page.
 */
public function actionCreate()
{
        $model=new ClassModel;

        // Uncomment the following line if AJAX validation is needed
        // $this->performAjaxValidation($model);

        if(isset($_POST['ClassModel']))
        {
                $model->attributes=$_POST['ClassModel'];
                $model->prof=Yii::app()->user->id;
                if($model->save())
                        $this->redirect(array('view','id'=>$model->id));
        }

        $this->render('create',array(
                'model'=>$model,
        ));
}

/**
 * Updates a particular model.
 * If update is successful, the browser will be redirected to the 'view' page.
 * @param integer $id the ID of the model to be updated
 */
public function actionUpdate($id)
{
        $model=$this->loadModel($id);

        // Uncomment the following line if AJAX validation is needed
        // $this->performAjaxValidation($model);

        if(isset($_POST['ClassModel']))
        {
                $model->attributes=$_POST['ClassModel'];
                if($model->save())
                        $this->redirect(array('view','id'=>$model->id));
        }

        $this->render('update',array(
                'model'=>$model,
        ));
}

/**
 * Deletes a particular model.
 * If deletion is successful, the browser will be redirected to the 'admin' page.
 * @param integer $id the ID of the model to be deleted
 */
public function actionDelete($id)
{
        $class = $this->loadModel($id);
        if($class->exp <= date("Y-m-d"))
                $class->delete();

        // if AJAX request (triggered by deletion via admin grid view), we should not
            redirect the browser
        if(!isset($_GET['ajax']))
                $this->redirect(isset($_POST['returnUrl']) ? $_POST['returnUrl'] :
                     array('admin'));
}

/**
 * Lists all models.
 */
public function actionIndex()
{
        $model=new ClassModel('search');
        $model->unsetAttributes();  // clear any default values
        if(isset($_GET['ClassModel']))
                $model->attributes=$_GET['ClassModel'];

        $this->render('index',array(
                'model'=>$model,
        ));
}

/**
 * Manages all models.
 */
public function actionAdmin()
{
        $model=new ClassModel('search');
        $model->unsetAttributes();  // clear any default values
        if(isset($_GET['ClassModel']))
                $model->attributes=$_GET['ClassModel'];

        $this->render('admin',array(
                'model'=>$model,
        ));
```

```
	}

	/**
	 * Returns the data model based on the primary key given in the GET variable.
	 * If the data model is not found, an HTTP exception will be raised.
	 * @param integer $id the ID of the model to be loaded
	 * @return ClassModel the loaded model
	 * @throws CHttpException
	 */
	public function loadModel($id)
	{
		$model=ClassModel::model()->findByPk($id);
		if($model===null)
			throw new CHttpException(404,'The requested page does not exist.');
		return $model;
	}

	/**
	 * Performs the AJAX validation.
	 * @param ClassModel $model the model to be validated
	 */
	protected function performAjaxValidation($model)
	{
		if(isset($_POST['ajax']) && $_POST['ajax']==='class-model-form')
		{
			echo CActiveForm::validate($model);
			Yii::app()->end();
		}
	}

	/**
	 * Loads class for Student access.
	 * @param integer $pod_id the pod id
	 */
	public function actionLoadClass($pod_id) {
		$model = PodModel::model()->findByPk($pod_id);

		$this->render($model->class->course_code, array(
			'model'=>$model
		));
	}

	/**
	 * Connects to the device clicked.
	 * @param string $id the id of the device
	 */
	public function actionConnect($id) {
		$topology = TopologyModel::model()->findByPk($id);

		if(isset($topology)) {
			if($topology->type == 3) {
				$command = 'mstsc|/v:'.$topology->ip;
			} else {
				$class = ClassModel::model()->findByPk(Yii::app()->user->
					getState('class'));
				$command = 'putty|-ssh '.$class->ts_ip.' -l '.$class->ts_user
					.' -pw '.$class->ts_pass;
				$command .= $topology->display === 'TS' ? "" : ' -m "'.Yii::
					app()->basePath.'\views\class\\'.$class->course_code.'\\'.
					$topology->display.'.txt"';
			}
		}

		//exec($command, $result, $status);

		//echo $command."\n".$result."\n".$status;

		echo $command;
	}
}
```

Listing 67: Course Controller

```
<?php

class CourseController extends Controller
{
	/**
	 * @var string the default layout for the views. Defaults to '//layouts/column2',
		meaning
	 * using two-column layout. See 'protected/views/layouts/column2.php'.
	 */
	public $layout='//layouts/column2';

	/**
	 * @return array action filters
	 */
	public function filters()
	{
		return array(
			'accessControl', // perform access control for CRUD operations
			'postOnly + delete', // we only allow deletion via POST request
		);
	}

	/**
	 * Specifies the access control rules.
	 * This method is used by the 'accessControl' filter.
```

```
	 * @return array access control rules
	 */
	public function accessRules()
	{
		return array(
			array('allow',  // allow admin to perform all actions
				'actions'=>array('index','view','create','update','admin','
					delete','topology'),
				'expression'=>'Yii::app()->user->isAdmin()',
			),
			array('deny',  // deny all users
				'users'=>array('*'),
			),
		);
	}

	/**
	 * Displays a particular model.
	 * @param integer $id the ID of the model to be displayed
	 */
	public function actionView($id)
	{
		$this->render('view',array(
			'model'=>$this->loadModel($id),
		));
	}

	/**
	 * Creates a new model.
	 * If creation is successful, the browser will be redirected to the 'view' page.
	 */
	public function actionCreate()
	{
		$model=new CourseModel;

		// Uncomment the following line if AJAX validation is needed
		// $this->performAjaxValidation($model);

		if(isset($_POST['CourseModel']))
		{
			$model->attributes=$_POST['CourseModel'];
			if($model->save())
				$this->redirect(array('view','id'=>$model->code));
		}

		$this->render('create',array(
			'model'=>$model,
		));
	}

	/**
	 * Updates a particular model.
	 * If update is successful, the browser will be redirected to the 'view' page.
	 * @param integer $id the ID of the model to be updated
	 */
	public function actionUpdate($id)
	{
		$model=$this->loadModel($id);

		// Uncomment the following line if AJAX validation is needed
		// $this->performAjaxValidation($model);

		if(isset($_POST['CourseModel']))
		{
			$model->attributes=$_POST['CourseModel'];
			if($model->save())
				$this->redirect(array('view','id'=>$model->code));
		}

		$this->render('update',array(
			'model'=>$model,
		));
	}

	/**
	 * Deletes a particular model.
	 * If deletion is successful, the browser will be redirected to the 'admin' page.
	 * @param integer $id the ID of the model to be deleted
	 */
	public function actionDelete($id)
	{
		$model = $this->loadModel($id);
		$class = ClassModel::model()->findAllByAttributes(array('course_code'=>$model
			->id));
		if(count($class) == 0) {
			$path = Yii::app()->basePath.'\views\class\\';

			if(is_dir($path.$id))
				exec('rd /s /q '.$path.$id, $return, $status);

			if(file_exists($path.$id.".php"))
				unlink($path.$id.".php");

			$model->delete();
		}

		// if AJAX request (triggered by deletion via admin grid view), we should not
			redirect the browser
		if(!isset($_GET['ajax']))
```

```
                $this->redirect(isset($_POST['returnUrl']) ? $_POST['returnUrl'] :
                    array('admin'));
}

/**
 * Lists all models.
 */
public function actionIndex()
{
        $dataProvider=new CActiveDataProvider('CourseModel');
        $this->render('index',array(
                'dataProvider'=>$dataProvider,
        ));
}

/**
 * Manages all models.
 */
public function actionAdmin()
{
        $model=new CourseModel('search');
        $model->unsetAttributes();  // clear any default values
        if(isset($_GET['CourseModel']))
                $model->attributes=$_GET['CourseModel'];

        $this->render('admin',array(
                'model'=>$model,
        ));
}

/**
 * Returns the data model based on the primary key given in the GET variable.
 * If the data model is not found, an HTTP exception will be raised.
 * @param integer $id the ID of the model to be loaded
 * @return CourseModel the loaded model
 * @throws CHttpException
 */
public function loadModel($id)
{
        $model=CourseModel::model()->findByPk($id);
        if($model===null)
                throw new CHttpException(404,'The requested page does not exist.');
        return $model;
}

/**
 * Performs the AJAX validation.
 * @param CourseModel $model the model to be validated
 */
protected function performAjaxValidation($model)
{
        if(isset($_POST['ajax']) && $_POST['ajax']==='course-model-form')
        {
                echo CActiveForm::validate($model);
                Yii::app()->end();
        }
}

/**
 * Builds Topology.
 * @param string $id the code of the course of the topology
 */
 public function actionTopology($id)
 {
        $course = $this->loadModel($id);

        if(isset($_POST['TopologyModel'])) {
                /*
                echo "<pre>";
                print_r($_POST);
                echo "</pre>";

                die("---Nothing else follows---");
                */
                $path = Yii::app()->basePath.'\views\class\\';

                if(is_dir($path.$id))
                        exec('rmdir /s /q '.$path.$id, $return, $status);

                mkdir($path.$id, 0777);

                $errors = '';
                $filename = $path.$id.'.php';
                $filehandle = fopen($filename, 'w');
                $contents = '<?php'.PHP_EOL
                .'/* @var $this ClassController */'.PHP_EOL
                .'/* @var $model PodModel */'.PHP_EOL
                .'?>'.PHP_EOL.PHP_EOL
                .'<link rel="stylesheet" type="text/css" href="<?php echo Yii::app()->
                    request->baseUrl; ?>/css/topology.css"></link>'.PHP_EOL.PHP_EOL
                .'<script type="text/javascript" src="<?php echo Yii::app()->request->
                    baseUrl; ?>/js/jquery-2.1.1.js"></script>'.PHP_EOL
                .'<script type="text/javascript" src="<?php echo Yii::app()->request->
                    baseUrl; ?>/js/raphael.js"></script>'.PHP_EOL
                .'<script type="text/javascript" src="<?php echo Yii::app()->request->
                    baseUrl; ?>/js/main.js"></script>'.PHP_EOL
                .'<script type="text/javascript" src="<?php echo Yii::app()->request->
                    baseUrl; ?>/js/pod.js"></script>'.PHP_EOL
                .'<script type="text/javascript" src="<?php echo Yii::app()->request->
```

```
    baseUrl; ?>/js/cable.js"></script>'.PHP_EOL
.'<script type="text/javascript" src="<?php echo Yii::app()->request->
    baseUrl; ?>/js/grayscale.js"></script>'.PHP_EOL
.'<script type="text/javascript">'.PHP_EOL
.'      window.addEventListener("load",function() {'.PHP_EOL
.'              drawStage();'.PHP_EOL
.'              configure();'.PHP_EOL
.'      },false);'.PHP_EOL
.'      $(window).resize(function() {'.PHP_EOL
.'              configure();'.PHP_EOL
.'      })'.PHP_EOL
.'      if(<?php echo Yii::app()->user->isProf() ? \'true\' : \'false
    \'; ?>)'.PHP_EOL
.'              window.onbeforeunload = function() {'.PHP_EOL
.'                      return "Everything will be lost if you don\'t
    save your work."'.PHP_EOL
.'              }'.PHP_EOL
.'      var url = "<?php echo Yii::app()->getBaseUrl(true); ?>",'.
    PHP_EOL
.'      code = "'.$id.'",'.PHP_EOL
.'      name = "'.$course->name.'";'.PHP_EOL
.'</script>'.PHP_EOL.PHP_EOL
.'<div id="main-div">'.PHP_EOL
.'      <?php'.PHP_EOL
.'              if(Yii::app()->user->isProf()) {'.PHP_EOL
.'                      $form=$this->beginWidget(\'CActiveForm\',
    array('.PHP_EOL
.'                              \'id\'=>\'pod-model-form\','.PHP_EOL
.'                              \'enableAjaxValidation\'=>false'.
    PHP_EOL
.'                      ));'.PHP_EOL
.'              }'.PHP_EOL
.'      ?>'.PHP_EOL.PHP_EOL
.'      <div id="topology-div">'.PHP_EOL.PHP_EOL
.'      <?php if(Yii::app()->user->isProf()) {'.PHP_EOL
.'              echo \'<div id="control-div">\';'.PHP_EOL.PHP_EOL
.'                      echo \'<h1>Class \'.$model->class_id.\'</h4
    >\';'.PHP_EOL.PHP_EOL
.'                      echo \'<h4>Pod \'.$model->pod_num.\'</h4>\';'.
    PHP_EOL.PHP_EOL
.'                      echo $form->hiddenField($model, \'id\');'.
    PHP_EOL.PHP_EOL
.'                      echo $form->hiddenField($model, \'pod_num\')
    ;'.PHP_EOL.PHP_EOL
.'                      echo $form->hiddenField($model, \'class_id\')
    ;'.PHP_EOL.PHP_EOL
.'                      echo $form->hiddenField($model, \'pass\');'.
    PHP_EOL.PHP_EOL
.'                      echo CHtml::submitButton(\'Save\', array('.
    PHP_EOL
.'                              \'onclick\'=>\'return save(event)\''.
    PHP_EOL
.'                      ));'.PHP_EOL.PHP_EOL
.'                      echo CHtml::button(\'Back\', array('.PHP_EOL
.'                              \'onclick\'=>\'window.location.assign
    ("\'.Yii::app()->createUrl(\'/pod/admin\', array(\'class_id\'=>
    $model->class->id)).\'")\''.PHP_EOL
.'                      ));'.PHP_EOL
.'              echo \'</div>\';'.PHP_EOL
.'      } else {'.PHP_EOL
.'              echo \'<img src="'.Yii::app()->request->baseUrl.'/
    images/close.png" id="close-button" width="20px" height="20px"
    />\';'.PHP_EOL
.'      } ?>'.PHP_EOL.PHP_EOL
.'              <div id="canvas-div">'.PHP_EOL
.'                      <svg id="topology-canvas">'.PHP_EOL;

echo "<hr/>";

if(isset($_POST['line']))
        foreach($_POST['line'] as $line) {
                $cable = !empty($line['id']) ?
                        CableModel::model()->findByPk($line['id']) :
                        new CableModel;

                $cable->attributes = $line;
                $cable->course_code = $id;

                if(!$cable->save())
                        $errors .= CHtml::errorSummary($cable)."<hr
                             />";

                list($rax, $ray, $raw, $rah) = explode(",", $line['
                     rect_a']);
                list($rbx, $rby, $rbw, $rbh) = explode(",", $line['
                     rect_b']);

                //$rectA = explode(",", $line['rect_a']);
                //$rectB = explode(",", $line['rect_b']);

                if($cable->type < 2) {
                        $d = str_replace(",", " ", $cable->path);
                        $pts = explode(" ", $d);
                        $contents .='                                <path
                             d="'.$d.'" fill="none" stroke="'.$cable->
                             color.'" stroke-width="3" />'.PHP_EOL;

                        if(!empty($cable->port_a) && $cable->port_a !=
                             "port a") {
```

```
                $contents .= '
                                <rect x="'.$rax.'" y="'.
                    $ray.'" width="'.$raw.'" height
                    ="'.$rah.'" style="fill: #fff;
                    stroke: #000" />'.PHP_EOL
                .'                              <text
                    x="'.$pts[1].'" y="'.$pts[2].'"
                    style="text-anchor: middle; font-
                    size: 10px; font-family: Arial
                    ;">'.PHP_EOL
                .'
                                <tspan dy="3.5">'.$cable->
                    port_a.'</tspan>'.PHP_EOL
                .'                              </text
                    >'.PHP_EOL;
            }

            if(!empty($cable->port_b) && $cable->port_b !=
                  "port b") {
                    if($cable->type == 0)
                            $contents .= '
                                                <rect x
                                ="'.$rbx.'" y="'.$rby.'"
                                width="'.$rbw.'" height
                                ="'.$rbh.'" style="fill: #
                                fff; stroke: #000" />'.
                                PHP_EOL
                            .'
                                        <text x="'.$pts
                                [4].'" y="'.$pts[5].'"
                                style="text-anchor: middle
                                ; font-size: 10px; font-
                                family: Arial;">'.PHP_EOL
                            .'
                                                <tspan dy
                                ="3.5">'.$cable->port_b
                                .'</tspan>'.PHP_EOL
                            .'
                                        </text>'.PHP_EOL;
                    else
                            $contents .= '
                                                <rect x
                                ="'.$rbx.'" y="'.$rby.'"
                                width="'.$rbw.'" height
                                ="'.$rbh.'" style="fill: #
                                fff; stroke: #000" />'.
                                PHP_EOL
                            .'
                                        <text x="'.$pts
                                [8].'" y="'.$pts[9].'"
                                style="text-anchor: middle
                                ; font-size: 10px; font-
                                family: Arial;">'.PHP_EOL
                            .'
                                                <tspan dy
                                ="3.5">'.$cable->port_b
                                .'</tspan>'.PHP_EOL
                            .'
                                        </text>'.PHP_EOL;
            }

        } else {
            $dim = explode(",", $cable->path);

            list($cax, $cay) = explode(",", $cable->
                port_a_coor);
            list($cbx, $cby) = explode(",", $cable->
                port_b_coor);

            //$coorA = explode(",", $cable->port_a_coor);
            //$coorB = explode(",", $cable->port_b_coor);

            $imgfile = Yii::app()->basePath.'/devices/'.
                $cable->deviceModel->file;

            if(file_exists($imgfile))
                    $imgsrc = Yii::app()->assetManager->
                        publish($imgfile);

            $contents .='                               <image
                 x="'.$dim[0].'" y="'.$dim[1].'" width="'.
                $dim[2].'" height="'.$dim[3].'"
                preserveAspectRatio="none" xlink:href="'.
                $imgsrc.'" transform="rotate('.$cable->
                transform.' '.($dim[0] + $dim[2] / 2)
                .','.($dim[1] + $dim[3] / 2).')"/>'.
                PHP_EOL;

            if(!empty($cable->port_a) && $cable->port_a !=
                  "port a") {
                    $contents .= '
                                <rect x="'.$rax.'" y="'.
                        $ray.'" width="'.$raw.'" height
                        ="'.$rah.'" style="fill: #fff;
                        stroke: #000" />'.PHP_EOL
                    .'                              <text
                        x="'.$cax.'" y="'.$cay.'" style="
                        text-anchor: middle; font-size: 10
                        px; font-family: Arial;">'.PHP_EOL
```

```
                                        .'
                                                <tspan dy="3.5">'.$cable->
                                            port_a.'</tspan>'.PHP_EOL
                                        .'                                  </text
                                            >'.PHP_EOL;
                                }

                                if(!empty($cable->port_b) && $cable->port_b !=
                                      "port b") {
                                        $contents .= '
                                                <rect x="'.$rbx.'" y="'.
                                            $rby.'" width="'.$rbw.'" height
                                            ="'.$rbh.'" style="fill: #fff;
                                            stroke: #000" />'.PHP_EOL
                                        .'                                  <text
                                            x="'.$cbx.'" y="'.$cby.'" style="
                                            text-anchor: middle; font-size: 10
                                            px; font-family: Arial;">'.PHP_EOL
                                        .'
                                                <tspan dy="3.5">'.$cable->
                                            port_b.'</tspan>'.PHP_EOL
                                        .'                                  </text
                                            >'.PHP_EOL;
                                }
                        }
                }


$contents .= '                          Your browser does not support
     Scalable Vector Graphics'.PHP_EOL
.'                      </svg>'.PHP_EOL;

foreach($_POST['draggable'] as $draggable) {
        if(empty($draggable['device_name'])) {
                $topology = TopologyModel::model()->findByPk(
                    $draggable['id']);
                if(isset($topology))
                        $topology->delete();
        } else {
                $topology = !empty($draggable['id']) ?
                        TopologyModel::model()->findByPk($draggable['
                            id']) :
                        new TopologyModel;
                $topology->course_code = $id;
                $topology->attributes = $draggable;
                $topology->host = $topology->host == "host name" ?
                        $topology->host :
                        strtoupper($topology->host);
                $topology->display = $topology->display == "display
                    name" ?
                        ($topology->type == 1 ? "TS" : $topology->
                            display) :
                        strtoupper($topology->display);
                if(!isset($topology->device_name))
                        TopologyModel::model()->findByPk($topology->id
                            )->delete();
                else if(!$topology->save())
                        $errors .= CHtml::errorSummary($topology)."<hr
                            />";

                if($topology->type != 1) {
                        $confighandle = fopen($path.$id.'/'.$topology
                            ->display.'.txt', 'w');
                        fwrite($confighandle, 'connect '.$topology->
                            host);
                        fclose($confighandle);
                }

                $file = Yii::app()->basePath.'/devices/'.$topology->
                    deviceModel->file;

                if(file_exists($file))
                        $src = Yii::app()->assetManager->publish($file
                            );

                $contents .= '                  <?php if(Yii::app()->
                    user->isProf()) { ?>'.PHP_EOL
                .'                              <div id="'.$topology->
                    id.'" onClick="toggleSelected('.$topology->id.',<?
                    php echo $model->id; ?>)" style="top: '.$topology
                    ->y.'px; left: '.$topology->x.'px; position:
                    absolute">'.PHP_EOL
                .'                          <?php'.PHP_EOL
                .'                              } else {'.PHP_EOL
                .'                                  $podDevice =
                    PodDeviceModel::model()->findByAttributes(array('.
                    PHP_EOL
                .'                                      \'
                    pod_id\'=>$model->id,'.PHP_EOL
                .'                                      \'
                    topology_id\'=>'.$topology->id.PHP_EOL
                .'                                  ));'.PHP_EOL
                .'                          ?>'.PHP_EOL
                .'                              <div <?php echo isset(
                    $podDevice) ? \'ondblclick="connect(\\\''.
                    $topology->id.'\\\')"\' : \'\' ; ?> style="top: '.
                    $topology->y.'px; left: '.$topology->x.'px;
                    position: absolute;">'.PHP_EOL
                .'                          <?php } ?>'.PHP_EOL
```

```
                                        .'                                                  <ul type="none
                                            " align="center" style="margin: 0px; padding: 0px
                                            ">'.PHP_EOL
                                        .'                                                      <li><
                                            img src="'.$src.'" <?php echo (!isset($podDevice)
                                            && Yii::app()->user->isStdnt()) ? \'class="
                                            grayscale"\' : \'\' ; ?> style="width: '.$topology
                                            ->width.'px; height: '.$topology->height.'px;"
                                            />'.PHP_EOL
                                        .'                                                      <li>'.
                                            $topology->display.PHP_EOL
                                        .'                                                  </ul>'.PHP_EOL
                                        .'                                              </div>'.PHP_EOL.
                                            PHP_EOL;
                                }
                        }

                        if($course->has_topology == "No") {
                                $course->has_topology = "Yes";
                                if(!$course->validate() || !$course->save()) {
                                        $errors .= CHtml::errorSummary($course)."<hr/>";
                                }
                        }

                        $contents .= '          </div>'.PHP_EOL
                        .'              <?php'.PHP_EOL
                        .'                      if(Yii::app()->user->isProf()) {'.PHP_EOL
                        .'                              $this->endWidget();'.PHP_EOL.PHP_EOL
                        .'                              if($model->class->has_pods) {'.PHP_EOL
                        .'                                      $devices = PodDeviceModel::
                            model()->findAllByAttributes(array('.PHP_EOL
                        .'                                              \'pod_id\'=>$model->id
                            '.PHP_EOL
                        .'                                      ));'.PHP_EOL
                        .'                                      foreach($devices as $device)
                            {'.PHP_EOL
                        .'                                              echo \'<script type="
                            text/javascript">\''.PHP_EOL
                        .'                                                      .\'
                            toggleSelected(\'.$device->topology_id.\',\'.$model->id.\')\''.
                            PHP_EOL
                        .'                                              .\'</script>\';'.
                            PHP_EOL
                        .'                                      }'.PHP_EOL
                        .'                              }'.PHP_EOL
                        .'                      }'.PHP_EOL
                        .'              ?>'.PHP_EOL
                        .'      </div>'.PHP_EOL
                        .'</div>';

                        fwrite($filehandle, $contents);
                        fclose($filehandle);

                        if(isset($_POST['deleted']))
                                foreach($_POST['deleted'] as $deleted) {
                                        if($deleted['type'] == "device")
                                                TopologyModel::model()->findByPk($deleted['id
                                                    '])->delete();
                                        else if($deleted['type'] == "cable")
                                                CableModel::model()->findByPk($deleted['id'])
                                                    ->delete();
                                }

                        //die($errors);

                        $this->redirect(array('/course/admin'));
                }

                $this->render('topology',array(
                        'course'=>$course,
                ));
        }
}
```

Listing 68: Device Controller

```
<?php

class DeviceController extends Controller
{
        /**
         * @var string the default layout for the views. Defaults to '//layouts/column2',
             meaning
         * using two-column layout. See 'protected/views/layouts/column2.php'.
         */
        public $layout='//layouts/column2';

        /**
         * @return array action filters
         */
        public function filters()
        {
                return array(
                        'accessControl', // perform access control for CRUD operations
                        'postOnly + delete', // we only allow deletion via POST request
                );
        }
```

```
/**
 * Specifies the access control rules.
 * This method is used by the 'accessControl' filter.
 * @return array access control rules
 */
public function accessRules()
{
        return array(
                array('allow',  // allow admin to perform all actions
                        'actions'=>array('index','view','create','update','admin','
                            delete'),
                        'expression'=>'Yii::app()->user->isAdmin()',
                ),
                array('deny',  // deny all users
                        'users'=>array('*'),
                ),
        );
}

/**
 * Displays a particular model.
 * @param integer $id the ID of the model to be displayed
 */
public function actionView($id)
{
        $this->render('view',array(
                'model'=>$this->loadModel($id),
        ));
}

/**
 * Creates a new model.
 * If creation is successful, the browser will be redirected to the 'view' page.
 */
public function actionCreate()
{
        $model=new DeviceModel;

        // Uncomment the following line if AJAX validation is needed
        // $this->performAjaxValidation($model);

        if(isset($_POST['DeviceModel']))
        {
                $model->attributes=$_POST['DeviceModel'];
                $file=CUploadedFile::getInstance($model,'file');
                if(isset($file) && $file->error == 0) {
                        $model->file=$model->name.'.'.$file->extensionName;
                        $file->saveAs(Yii::app()->basePath."/devices/".$model->file);
                }
                if($model->save())
                        $this->redirect(array('view','id'=>$model->name));
        }

        $this->render('create',array(
                'model'=>$model,
        ));
}

/**
 * Updates a particular model.
 * If update is successful, the browser will be redirected to the 'view' page.
 * @param integer $id the ID of the model to be updated
 */
public function actionUpdate($id)
{
        $model=$this->loadModel($id);

        // Uncomment the following line if AJAX validation is needed
        // $this->performAjaxValidation($model);

        if(isset($_POST['DeviceModel']))
        {
                $model->attributes=$_POST['DeviceModel'];
                $file = Yii::app()->basePath."/devices/".$model->file;

                list($filename, $ext) = explode(".", $model->file);

                $model->file = $model->name.".".$ext;

                if(rename($file, Yii::app()->basePath."/devices/".$model->file))
                        if($model->save())
                                $this->redirect(array('view','id'=>$model->name));
        }

        $this->render('update',array(
                'model'=>$model,
        ));
}

/**
 * Deletes a particular model.
 * If deletion is successful, the browser will be redirected to the 'admin' page.
 * @param integer $id the ID of the model to be deleted
 */
public function actionDelete($id)
{
        $model = $this->loadModel($id);
        $file = Yii::app()->basePath."/devices/".$model->file;
        if(file_exists($file))
```

```
                        unlink($file);

                $model->delete();

                // if AJAX request (triggered by deletion via admin grid view), we should not
                     redirect the browser
                if(!isset($_GET['ajax']))
                        $this->redirect(isset($_POST['returnUrl']) ? $_POST['returnUrl'] :
                             array('admin'));
        }

        /**
         * Lists all models.
         */
        public function actionIndex()
        {
                $dataProvider=new CActiveDataProvider('DeviceModel');
                $this->render('index',array(
                        'dataProvider'=>$dataProvider,
                ));
        }

        /**
         * Manages all models.
         */
        public function actionAdmin()
        {
                $model=new DeviceModel('search');
                $model->unsetAttributes();  // clear any default values
                if(isset($_GET['DeviceModel']))
                        $model->attributes=$_GET['DeviceModel'];

                $this->render('admin',array(
                        'model'=>$model,
                ));
        }

        /**
         * Returns the data model based on the primary key given in the GET variable.
         * If the data model is not found, an HTTP exception will be raised.
         * @param integer $id the ID of the model to be loaded
         * @return DeviceModel the loaded model
         * @throws CHttpException
         */
        public function loadModel($id)
        {
                $model=DeviceModel::model()->findByPk($id);
                if($model===null)
                        throw new CHttpException(404,'The requested page does not exist.');
                return $model;
        }

        /**
         * Performs the AJAX validation.
         * @param DeviceModel $model the model to be validated
         */
        protected function performAjaxValidation($model)
        {
                if(isset($_POST['ajax']) && $_POST['ajax']==='device-model-form')
                {
                        echo CActiveForm::validate($model);
                        Yii::app()->end();
                }
        }
}
```

Listing 69: Pod Controller

```
<?php

class PodController extends Controller
{
        /**
         * @var string the default layout for the views. Defaults to '//layouts/column2',
              meaning
         * using two-column layout. See 'protected/views/layouts/column2.php'.
         */
        public $layout='//layouts/column2';

        /**
         * @return array action filters
         */
        public function filters()
        {
                return array(
                        'accessControl', // perform access control for CRUD operations
                        'postOnly + delete', // we only allow deletion via POST request
                );
        }

        /**
         * Specifies the access control rules.
         * This method is used by the 'accessControl' filter.
         * @return array access control rules
         */
        public function accessRules()
        {
                return array(
```

```
                array('allow',  // allow prof to perform all actions
                        'actions'=>array('index','view','create','update','admin','
                            delete','pods'),
                        'expression'=>'Yii::app()->user->isProf()',
                ),
                array('deny',  // deny all users
                        'users'=>array('*'),
                ),
        );
}

/**
 * Displays a particular model.
 * @param integer $id the ID of the model to be displayed
 */
public function actionView($id)
{
        $this->render('view',array(
                'model'=>$this->loadModel($id),
        ));
}

/**
 * Creates a new model.
 * If creation is successful, the browser will be redirected to the 'view' page.
 */
public function actionCreate($class_id)
{
        $model=new PodModel;
        $model->class_id = $class_id;

        // Uncomment the following line if AJAX validation is needed
        // $this->performAjaxValidation($model);

        if(isset($_POST['PodModel']))
        {
                $model->attributes=$_POST['PodModel'];
                if($model->save())
                        $this->redirect(array('view','id'=>$model->id));
        }

        $this->render('create',array(
                'model'=>$model,
        ));
}

/**
 * Updates a particular model.
 * If update is successful, the browser will be redirected to the 'view' page.
 * @param integer $id the ID of the model to be updated
 */
public function actionUpdate($id)
{
        $model=$this->loadModel($id);

        // Uncomment the following line if AJAX validation is needed
        // $this->performAjaxValidation($model);

        if(isset($_POST['PodModel']))
        {
                $model->attributes=$_POST['PodModel'];
                if($model->save())
                        $this->redirect(array('view','id'=>$model->id));
        }

        $this->render('update',array(
                'model'=>$model,
        ));
}

/**
 * Deletes a particular model.
 * If deletion is successful, the browser will be redirected to the 'admin' page.
 * @param integer $id the ID of the model to be deleted
 */
public function actionDelete($id)
{
        $this->loadModel($id)->delete();

        // if AJAX request (triggered by deletion via admin grid view), we should not
            redirect the browser
        if(!isset($_GET['ajax']))
                $this->redirect(isset($_POST['returnUrl']) ? $_POST['returnUrl'] :
                    array('admin'));
}

/**
 * Lists all models.
 */
public function actionIndex($class_id)
{
        $model=new PodModel('search');
        $model->unsetAttributes();  // clear any default values
        if(isset($_GET['PodModel']))
                $model->attributes=$_GET['PodModel'];

        $this->render('index',array(
                'model'=>$model,
                'class_id'=>$class_id,
```

```
        ));
}

/**
 * Manages all models.
 */
public function actionAdmin($class_id)
{
        $model=new PodModel('search');
        $model->unsetAttributes();  // clear any default values
        if(isset($_GET['PodModel']))
                $model->attributes=$_GET['PodModel'];

        $this->render('admin',array(
                'model'=>$model,
                'class_id'=>$class_id,
        ));
}

/**
 * Returns the data model based on the primary key given in the GET variable.
 * If the data model is not found, an HTTP exception will be raised.
 * @param integer $id the ID of the model to be loaded
 * @return PodModel the loaded model
 * @throws CHttpException
 */
public function loadModel($id)
{
        $model=PodModel::model()->findByPk($id);
        if($model===null)
                throw new CHttpException(404,'The requested page does not exist.');
        return $model;
}

/**
 * Performs the AJAX validation.
 * @param PodModel $model the model to be validated
 */
protected function performAjaxValidation($model)
{
        if(isset($_POST['ajax']) && $_POST['ajax']==='pod-model-form')
        {
                echo CActiveForm::validate($model);
                Yii::app()->end();
        }
}

/**
 * Lets the Professor select pods.
 * @param integer $id the pod id
 */
public function actionPods($id) {
        $model = PodModel::model()->findByPk($id);

        if(isset($_POST['PodModel'])) {
                $errors = '';
                foreach($_POST['pods'] as $pod) {
                        $device = PodDeviceModel::model()->findByAttributes(array(
                                'topology_id'=>$pod['topology_id'],
                                'pod_id'=>$pod['pod_id']
                        ));

                        if(!isset($device)) {
                                $device = new PodDeviceModel;
                                $device->attributes = $pod;
                                if(!$device->save())
                                        $errors .= CHtml::errorSummary($device)."<hr
                                             />";
                        }
                }

                if($model->class->has_pods == "No") {
                        $class = ClassModel::model()->findByPk($model->class_id);
                        $class->has_pods = "Yes";
                        if(!$class->save())
                                $errors .= CHtml::errorSummary($class)."<hr/>";
                }

                if(isset($_POST['deleted']))
                        foreach($_POST['deleted'] as $deleted) {
                                $deletedPodDevice = PodDeviceModel::model()->
                                     findByAttributes(array(
                                         'topology_id'=>$deleted['topology_id'],
                                         'pod_id'=>$deleted['pod_id']
                                ));
                                if(isset($deletedPodDevice))
                                        $deletedPodDevice->delete();
                        }

                //die($errors);

                $this->redirect(array('/pod/admin', 'class_id'=>$model->class->id));
        }

        if($model->class->course->has_topology === "Yes") {
                $this->render('/class/'.$model->class->course_code, array(
                        'model'=>$model,
                ));
        } else {
```

```
                        Yii::app()->user->setFlash('error','This course has no topology yet.
                            Please contact your system administrator.');
                        CController::redirect(array('/pod/admin', 'class_id' => $model->
                            class_id));
                }
        }
}
```

Listing 70: Setup Controller

```
<?php

class SetupController extends Controller
{
        public function actionAdmin()
        {
                $model = new AdminForm;

                if(isset($_POST['AdminForm'])) {
                        $model->attributes = $_POST['AdminForm'];

                        if($model->validate()) {
                                $admin = new UserModel;
                                $admin->attributes = $model->attributes;

                                $admin->user_type = 'Administrator';
                                if($admin->save())
                                        $this->redirect(array('/site/login'));
                        }
                }

                $this->render('admin', array(
                        'model'=>$model,
                ));
        }

        public function actionDatabase()
        {
                if(!Yii::app()->user->isGuest)
                        Yii::app()->user->logout();

                $model = new DatabaseForm;

                if(isset($_POST['DatabaseForm'])) {
                        $model->attributes = $_POST['DatabaseForm'];

                        if($model->validate()) {
                                $filename = Yii::app()->basePath.'\config\config.txt';
                                $filehandle = fopen($filename, 'w');
                                $config = $model->host.PHP_EOL.$model->user.PHP_EOL.$model->
                                    pass;

                                fwrite($filehandle, $config);
                                fclose($filehandle);

                                $file = Yii::app()->basePath.'\data\trends_rla.sql';
                                $commands = explode(';', file_get_contents($file));

                                foreach($commands as $command) {
                                        if(trim($command) !== '') {
                                                Yii::app()->db->createCommand($command)->
                                                    execute();
                                        }
                                }

                                $this->redirect(array('/setup/admin'));
                        }
                }

                $this->render('database', array(
                        'model'=>$model,
                ));
        }
}
```

Listing 71: Site Controller

```
<?php

class SiteController extends Controller
{
        /**
         * Declares class-based actions.
         */
        public function actions()
        {
                return array(
                        // captcha action renders the CAPTCHA image displayed on the contact
                            page
                        'captcha'=>array(
                                'class'=>'CCaptchaAction',
                                'backColor'=>0xFFFFFF,
                        ),
                        // page action renders "static" pages stored under 'protected/views/
                            site/pages'
                        // They can be accessed via: index.php?r=site/page&view=FileName
```

```
                'page'=>array(
                        'class'=>'CViewAction',
                ),
        );
}

/**
 * This is the default 'index' action that is invoked
 * when an action is not explicitly requested by users.
 */
public function actionIndex()
{
        try {
                Yii::app()->db;
        } catch(CDbException $e) {
                Yii::app()->controller->redirect(array('/setup/Database'));
        }

        $admin = UserModel::model()->findByAttributes(array('user_type'=>'
            Administrator'));
        if(!isset($admin))
                Yii::app()->controller->redirect(array('/setup/Admin'));

        $classes = Yii::app()->db->createCommand()
                ->select("*")
                ->from("class")
                ->queryAll();
        $ongoing = array();
        foreach($classes as $class)
                if($class["exp"] < date('Y-m-d'))
                        ClassModel::model()->findByPk($class["id"])->delete();

        // collect user input data
        if(Yii::app()->user->isGuest) {
                $model = new StudentForm;

                if(isset($_POST['StudentForm'])) {
                        $model->attributes=$_POST['StudentForm'];
                        // validate user input and redirect to the previous page if
                            valid
                        if($model->validate() && $model->login())
                                $this->redirect(Yii::app()->user->returnUrl);
                }
        } else if(Yii::app()->user->isStdnt()) {
                $model = new PodForm;

                if(isset($_POST['PodForm'])) {
                        $model->attributes=$_POST['PodForm'];
                        $model->classID = Yii::app()->user->getState('class');
                        // validate user input and redirect to the previous page if
                            valid
                        if($model->validate() && $model->login())
                                Yii::app()->controller->redirect(array('class/
                                    loadClass',
                                        'pod_id'=>$model->podNum
                                ));
                }
        }

        // renders the view file 'protected/views/site/index.php'
        // using the default layout 'protected/views/layouts/main.php'
        if(isset($model))
                $this->render('index', array(
                        'model'=>$model
                ));
        else
                $this->render('index');
}

/**
 * This is the action to handle external exceptions.
 */
public function actionError()
{
        if($error=Yii::app()->errorHandler->error)
        {
                if(Yii::app()->request->isAjaxRequest)
                        echo $error['message'];
                else
                        $this->render('error', $error);
        }
}

/**
 * Displays the contact page
 */
public function actionContact()
{
        $model=new ContactForm;
        if(isset($_POST['ContactForm']))
        {
                $model->attributes=$_POST['ContactForm'];
                if($model->validate())
                {
                        $name='=?UTF-8?B?'.base64_encode($model->name).'?=';
                        $subject='=?UTF-8?B?'.base64_encode($model->subject).'?=';
                        $headers="From: $name <{$model->email}>\r\n".
                                "Reply-To: {$model->email}\r\n".
                                "MIME-Version: 1.0\r\n".
```

```
                                        "Content-Type: text/plain; charset=UTF-8";

                                mail(Yii::app()->params['adminEmail'],$subject,$model->body,
                                     $headers);
                                Yii::app()->user->setFlash('contact','Thank you for contacting
                                      us. We will respond to you as soon as possible.');
                                $this->refresh();
                        }
                }
                $this->render('contact',array('model'=>$model));
        }

        /**
         * Displays the login page
         */
        public function actionLogin()
        {
                try {
                        Yii::app()->db;
                } catch(CDbException $e) {
                        Yii::app()->controller->redirect(array('/setup/Database'));
                }

                $admin = UserModel::model()->findByAttributes(array('user_type'=>'
                     Administrator'));
                if(!isset($admin))
                        Yii::app()->controller->redirect(array('/setup/Admin'));

                if(!Yii::app()->user->isGuest)
                        $this->redirect(array('index'));

                $model=new LoginForm;

                // if it is ajax validation request
                if(isset($_POST['ajax']) && $_POST['ajax']==='login-form')
                {
                        echo CActiveForm::validate($model);
                        Yii::app()->end();
                }

                // collect user input data
                if(isset($_POST['LoginForm']))
                {
                        $model->attributes=$_POST['LoginForm'];
                        // validate user input and redirect to the previous page if valid
                        if($model->validate() && $model->login())
                                $this->redirect(Yii::app()->user->returnUrl);
                }
                // display the login form
                $this->render('login',array('model'=>$model));
        }

        /**
         * Logs out the current user and redirect to homepage.
         */
        public function actionLogout()
        {
                Yii::app()->user->logout();
                $this->redirect(Yii::app()->homeUrl);
        }

        /**
         * Generates options for the student form.
         * @param integer $employee_id the Employee ID of the Professor holding the class
         */
        public function actionSubjects($employee_id) {
                $classes = ClassModel::model()->findAllByAttributes(array('prof'=>$employee_id
                     ));
                $return = '<option value="0">Select Subject</option>';
                foreach($classes as $class)
                        $return .= "<option value=".$class->id.">".$class->course_code." Class
                             ".$class->id."</option>";
                echo $return;
        }
}
```

Listing 72: User Controller

```
<?php

class SiteController extends Controller
{
        /**
         * Declares class-based actions.
         */
        public function actions()
        {
                return array(
                        // captcha action renders the CAPTCHA image displayed on the contact
                             page
                        'captcha'=>array(
                                'class'=>'CCaptchaAction',
                                'backColor'=>0xFFFFFF,
                        ),
                        // page action renders "static" pages stored under 'protected/views/
                             site/pages'
                        // They can be accessed via: index.php?r=site/page&view=FileName
                        'page'=>array(
```

```
                        'class'=>'CViewAction',
                ),
        );
}

/**
 * This is the default 'index' action that is invoked
 * when an action is not explicitly requested by users.
 */
public function actionIndex()
{
        try {
                Yii::app()->db;
        } catch(CDbException $e) {
                Yii::app()->controller->redirect(array('/setup/Database'));
        }

        $admin = UserModel::model()->findByAttributes(array('user_type'=>'
            Administrator'));
        if(!isset($admin))
                Yii::app()->controller->redirect(array('/setup/Admin'));

        $classes = Yii::app()->db->createCommand()
                ->select("*")
                ->from("class")
                ->queryAll();
        $ongoing = array();
        foreach($classes as $class)
                if($class["exp"] < date('Y-m-d'))
                        ClassModel::model()->findByPk($class["id"])->delete();

        // collect user input data
        if(Yii::app()->user->isGuest) {
                $model = new StudentForm;

                if(isset($_POST['StudentForm'])) {
                        $model->attributes=$_POST['StudentForm'];
                        // validate user input and redirect to the previous page if
                            valid
                        if($model->validate() && $model->login())
                                $this->redirect(Yii::app()->user->returnUrl);
                }
        } else if(Yii::app()->user->isStdnt()) {
                $model = new PodForm;

                if(isset($_POST['PodForm'])) {
                        $model->attributes=$_POST['PodForm'];
                        $model->classID = Yii::app()->user->getState('class');
                        // validate user input and redirect to the previous page if
                            valid
                        if($model->validate() && $model->login())
                                Yii::app()->controller->redirect(array('class/
                                    loadClass',
                                        'pod_id'=>$model->podNum
                                ));
                }
        }

        // renders the view file 'protected/views/site/index.php'
        // using the default layout 'protected/views/layouts/main.php'
        if(isset($model))
                $this->render('index', array(
                        'model'=>$model
                ));
        else
                $this->render('index');
}

/**
 * This is the action to handle external exceptions.
 */
public function actionError()
{
        if($error=Yii::app()->errorHandler->error)
        {
                if(Yii::app()->request->isAjaxRequest)
                        echo $error['message'];
                else
                        $this->render('error', $error);
        }
}

/**
 * Displays the contact page
 */
public function actionContact()
{
        $model=new ContactForm;
        if(isset($_POST['ContactForm']))
        {
                $model->attributes=$_POST['ContactForm'];
                if($model->validate())
                {
                        $name='=?UTF-8?B?'.base64_encode($model->name).'?=';
                        $subject='=?UTF-8?B?'.base64_encode($model->subject).'?=';
                        $headers="From: $name <{$model->email}>\r\n".
                                "Reply-To: {$model->email}\r\n".
                                "MIME-Version: 1.0\r\n".
                                "Content-Type: text/plain; charset=UTF-8";
```

```
                                mail(Yii::app()->params['adminEmail'],$subject,$model->body,
                                    $headers);
                                Yii::app()->user->setFlash('contact','Thank you for contacting
                                    us. We will respond to you as soon as possible.');
                                $this->refresh();
                        }
                }
                $this->render('contact',array('model'=>$model));
        }

        /**
         * Displays the login page
         */
        public function actionLogin()
        {
                try {
                        Yii::app()->db;
                } catch(CDbException $e) {
                        Yii::app()->controller->redirect(array('/setup/Database'));
                }

                $admin = UserModel::model()->findByAttributes(array('user_type'=>'
                    Administrator'));
                if(!isset($admin))
                        Yii::app()->controller->redirect(array('/setup/Admin'));

                if(!Yii::app()->user->isGuest)
                        $this->redirect(array('index'));

                $model=new LoginForm;

                // if it is ajax validation request
                if(isset($_POST['ajax']) && $_POST['ajax']==='login-form')
                {
                        echo CActiveForm::validate($model);
                        Yii::app()->end();
                }

                // collect user input data
                if(isset($_POST['LoginForm']))
                {
                        $model->attributes=$_POST['LoginForm'];
                        // validate user input and redirect to the previous page if valid
                        if($model->validate() && $model->login())
                                $this->redirect(Yii::app()->user->returnUrl);
                }
                // display the login form
                $this->render('login',array('model'=>$model));
        }

        /**
         * Logs out the current user and redirect to homepage.
         */
        public function actionLogout()
        {
                Yii::app()->user->logout();
                $this->redirect(Yii::app()->homeUrl);
        }

        /**
         * Generates options for the student form.
         * @param integer $employee_id the Employee ID of the Professor holding the class
         */
        public function actionSubjects($employee_id) {
                $classes = ClassModel::model()->findAllByAttributes(array('prof'=>$employee_id
                    ));
                $return = '<option value="0">Select Subject</option>';
                foreach($classes as $class)
                        $return .= "<option value=".$class->id.">".$class->course_code." Class
                            ".$class->id."</option>";
                echo $return;
        }
}
```

XII. Acknowledgement

First off, I give thanks to the Creator who made me what I am and helped me all the way to where I am.

Secondly, I would like to thank my parents from whom I owe everything I had, have, and will have; for things too vast to enumerate... no words will be enough to express my gratitude.

Next, I would like to express my thanks to those who had (and have) financed my college education: the Department of Science and Technology; the University of the Philippines Manila - Association of Parents and Councilors; my father's aunt, Lola Paring, and her family who have given me hope when all else failed.

I would also like to thank Trends.Net Education Centre Inc. (particularly Sir Jonathan P. Oira) for letting me work on this project and for the all support they have provided me, technically and legally, to finish it; and my adviser, Prof. Gregorio Baes for checking my papers and assisting me during the proposal and henceforth.

To my mother's sister, Tita Lorna, her husband, Tito John, and their sons: Kuya Jaspher, Kuya Claudine, (and specially) Kuya Lester who are my cousins in blood but my brothers in bond. I give my thanks.

I would also like to thank ComSci Batch '09 and '10, and UPM-OMAKE, who was my family at the university, for all the Ongaku, (perplexingly sinister) Manga, (psychologically deranging) Anime (which has kept me sane during my stay in UP despite of the ominous words I attached to it), Kosupure (which I never do), and Eiga; for Master Az; for having awesome Kokkais and members... for the hitsuzen... for the time and space...

...And finally, for Ms. Dianne Lyneth C. Alavado who has provided me (still providing) with the 99 % of this success, the desparation, the cerebration, the pseudo-randomized attitude fluctuation, the distraction, the colligation, and the enigmatic coruscation.

"*If you're going through hell, keep going.*"

- Sir Winston Leonard Spencer-Churchill

I also thank Sir Winston Churchill for the quote.