Eastern Michigan University
DigitalCommons@EMU

Master's Theses and Doctoral Dissertations

Master's Theses, and Doctoral Dissertations, and Graduate Capstone Projects

2012

Development techniques for android platform mobile device application

Ivan Njunjic

Follow this and additional works at: http://commons.emich.edu/theses

Part of the OS and Networks Commons

Recommended Citation

Njunjic, Ivan, "Development techniques for android platform mobile device application" (2012). *Master's Theses and Doctoral Dissertations*. 394.
http://commons.emich.edu/theses/394

This Open Access Thesis is brought to you for free and open access by the Master's Theses, and Doctoral Dissertations, and Graduate Capstone Projects at DigitalCommons@EMU. It has been accepted for inclusion in Master's Theses and Doctoral Dissertations by an authorized administrator of DigitalCommons@EMU. For more information, please contact lib-ir@emich.edu.

Development Techniques for Android Platform Mobile Device Application

by

Ivan Njunjic

Thesis

Submitted to the Department of Computer Information Systems

Eastern Michigan University

in partial fulfillment of the requirements

for the degree of

MASTER OF SCIENCE

in

Information Systems

Thesis Committee:

Stevan Mrdalj, Ph.D., Chair

Hung-Lian Tang, Ph.D.

Robert Goffeney, MBA

March 12, 2012

Ypsilanti, Michigan

ABSTRACT

This thesis focuses on Android application development techniques needed to implement a mobile application portal that consists of features used at Eastern Michigan University. Since there is not a single source available to developers that explains such techniques, this thesis represents a unique manual for such development. Based on the similarity of features, mainly in terms of data nature and access, five techniques are defined in a step-by-step procedural manner. This is accomplished by outlining the development techniques and presenting them "in action" with coding examples from a fully developed demo application. As a result, the demo application demonstrates functional solutions to research problems that are able to operate on an actual Android device. This thesis provides a unique approach to Android development due to its single focus on the data and IT environment of Eastern Michigan University.

Table of Contents

Abstract ii

Chapter 1: Introduction 1

Chapter 2: Literature Review 5

Android Fundamentals 6

Prior Research 7

Existing Literature Deficiencies 8

Chapter 3: Android Application Environment 10

Android Environment Setup Instructions 10

Android Application file structure 14

Activities 15

Intents 17

Threads 18

XML Layouts 19

ListView objects 20

Spinner objects 22

Chapter 4: Development Techniques 25

Unsecured Web Access 27

Secured Web Access 33

Accessing APIs 40

Embedding Web views 44

Programmable Components 47

Chapter 5: Implementation of an Android Mobile EMU Portal 48

Application AndroidManifest.xml 48
Main Menu 50
Library Catalog 51
Tuition & Fees Calculator 53
Online Directory 54
Weekly Schedule 56
Course Lookup 61
Final Grades 62
Twitter Updates 64
Athletics News 66
Dining Locations 67
Campus Map 68
Bus Schedule 69
Contact EMU 70
GPA Calculator 71
Chapter 6: Conclusion 73
Thesis Contributions 73
Thesis Limitations 74
Future Opportunities for Research 75
Final Remarks 76
References 77
Appendix A: ScheduleParser.txt 82
Appendix B: Script Injection FAQ 89
Appendix C: AndroidManifest.xml code 91

Appendix D: Main Menu code 93
Appendix E: Library Catalog code 97
Appendix F: Tution & Fees Calculator code 101
Appendix G: Online Directory code 110
Appendix H: Weekly Schedule code 120
Appendix I: Course Lookup code 132
Appendix J: Final Grades code 145
Appendix K: Twitter Updates code 152
Appendix L: Athletics News code 164
Appendix M: Dining Locations code 165
Appendix N: Campus Map code 171
Appendix O: Bus Schedule code 172
Appendix P: Contact EMU code 175
Appendix Q: GPA Calculator code 177

List of Tables

Table		Page
1	Device Setup for Development	13
2	Activity States	16
3	EMU Portal features and data sources	25
4	Development Techniques	26
5	Oracle Supported Client Platforms	34
6	Banner Self-Services portal packages	38

List of Figures

Figure		Page
1	Android SDK Manager	11
2	Android Emulator Screen	12
3	Android Device Chooser	13
4	Android App File Structure	14
5	Activity Lifecycle	16
6	Graphical Layout Window	20
7	ListView	20
8	ListView Optimization issue	21
9	AsyncTask Class Implementation	28
10	Architecture for Device with a Mobile Client and Client Database	34
11	WebView Screenshot	46
12	Android Mobile EMU Portal file structure	48
13	“All Apps” Menu	49
14	Main Menu Screenshot	51
15	Library Catalog Screenshots	52
16	Tuition & Fees Calculator Screenshots	54
17	Online Directory Screenshots	55
18	Online Directory Error Messages Screenshots	55
19	Email Composition Window Screenshot	56
20	Successful Login for Weekly Schedule Screenshots	57
21	Weekly Schedule List of Classrooms and Directions	58

22 Unsuccessful Login for Weekly Schedule Screenshots 58
23 Course Lookup Screenshots 61
24 Unsuccessful Course Lookup and Final Grades Login 62
25 Final Grades Screenshots 63
26 Twitter Updates Screenshots 64
27 Athletics News Screenshots 66
28 Dining Locations Screenshots 68
29 Dining Menus Screenshots 68
30 Campus Map Screenshots 69
31 Bus Schedule Screenshots 70
32 Contact EMU Screenshots 71
33 GPA Calculator Screenshots 72

Chapter 1: Introduction

Mobile applications (commonly referred as “apps”), are considered to be one of the fastest growing trends in Information Systems industry (Eddy, 2011). Users enjoy the variety of features that mobile apps can provide quickly and without introducing unnecessary complexity into their designs. As a result, mobile apps present a more popular interface for interaction with business systems than using web applications via Web Browser.

Eastern Michigan University (EMU) introduced its official applications for iPhone and Android, which are powered by Straxis, LLC., a company located in Tulsa, Oklahoma. Since the contract is signed with an external vendor, all code and methods implemented for these applications are owned by Straxis, LLC. Official apps provide content from www.emich.edu website by following its formatting styles. This presentation style is not too practical as data are condensed to fit the small mobile device screens. In addition to this, information available at www.emich.edu does not include individual student records like weekly class schedules or available courses per semester.

This research starts with a desire to develop Android Mobile EMU Portal, which provides the information unavailable through the official EMU Android app and presents the available content in a more user friendly manner. Based on the literature review, preliminary findings show that there is no single resource that combines mobile application development techniques that are needed to create an Android Mobile EMU Portal. This thesis attempts to determine different techniques used in the Android application development process that need to be implemented in order to access wide range of sources specific to EMU. It provides a strategy on how to determine which programming technique to use for which application

feature. It combines stepwise descriptions of programming techniques into a single document with coding explanations. As a result, this thesis illustrates the successful implementation of an Android Mobile EMU Portal, which incorporates features normally accessed via bannerweb.emich.edu portal, www.emich.edu website, and others.

This thesis intends to define Android mobile app programming techniques that are needed to implement the following features:

- Library Catalog
- Tuition & Fees Calculator
- Online Directory
- Weekly Schedule
- Course Lookup
- Final Grades
- Twitter Social Media Updates
- Athletics News
- Dining Locations
- Campus Map
- Bus Schedule
- Contact EMU
- GPA Calculator

These features cover a representative set of EMU-related data sources and connection types needed to effectively define programming techniques. They also help to illustrate the implementation of identified techniques and provide a functional set of instructions. In

addition to this, a majority of features listed is incorporated into existing Android apps designed for other universities.

By analyzing various data sources and connection types required for above selected features, programming techniques needed to access information can be categorized in five different groups:

- Unsecured Web Access
- Secured Web Access
- Accessing APIs
- Embedding Web views
- Programmable Components

By defining the technique determination process, this thesis provides a framework that selects which development technique should be utilized for any feature, if a particular data source and connection type are known. Development technique determination strategy and step-by-step instructions for their implementation are the main contributions of this thesis. Successful implementation of Android Mobile EMU Portal supports this strategy and instructions with a functional product.

The audience that benefits from information in this thesis includes anyone attempting to develop applications that utilize defined techniques. If the Information Technology Department wants to produce an in-house solution for an Android app in the future, this thesis would significantly reduce the research time for it. This thesis can encourage students' interest in Android application development as well.

Following the introduction, Chapter 2 includes the Literature Review, which points out the differences among the available development resources and further clarifies the importance of this research. Chapter 3 describes how to set up the Android programming environment and demonstrates the basics needed to understand Android app functionality. Chapter 4 explains Android development techniques that are needed for EMU-related data access with step-by-step instructions. Chapter 5 illustrates the implementation of Android Mobile EMU Portal, while Chapter 6 consists of conclusions, observations, and limitations of this research.

Chapter 2: Literature Review

Android is a relatively new platform. It is produced by Google, Inc., and its first release was presented in 2007 (Meier, 2010). Android is installed on many different mobile devices and its users can download Android apps and other content through Google Play service, which replaced the old Android Market (Bishop, 2012).

This thesis discusses technologies incorporated in Android application development and how they apply to the research problem. As the official Android website describes this platform, "Android is a software stack for mobile devices that includes an operating system, middleware and key applications" ("What is Android," 2012). Android provides the "core set of applications including an email client, SMS program, calendar, maps, browser, contacts, and others" ("What is Android," 2012), while additional applications can be downloaded through Google Play service (Bishop, 2012).

Google (n.d.) claims that "Android powers millions of phones, tablets and other devices." Phones and tablets are mobile devices that can have Android applications installed on them. These applications are written in Java programming language ("What is Android," 2012) and they are called mobile device applications or apps. Development techniques for apps are structured sets of Java code focused on implementing particular task that provides content for a mobile device application. Although Java programming language includes a broad variety of topics, this thesis focuses on development techniques required for successful implementation of Android Mobile EMU Portal. The following paragraphs analyze research efforts that addressed these techniques in the past.

Android Fundamentals

Many authors described Android application development fundamentals, which include setting up Android development environment on the machine, AndroidManifest.xml file, Activities, Intents, and XML layouts. Jackson (2011) outlines "three major components of an Android development environment: Java, Eclipse, Android" and provides instructions on how to download and install necessary files to establish this environment. Felker (2011) does not explicitly state the components but rather points out that Java JDK, Android SDK, Eclipse IDE, and Android ADT need to be installed and configured on a machine. The steps provided by these two authors are standard. They appear in many books written on Android development and are also presented on official Android website ("Installing the SDK", n.d.).

Ableson, King, and Sen (2011) present "four primary components of Android applications": Activity, Service, BroadcastReceiver, and ContentProvider. It is noted that "a particular Android application might not contain all of these elements, but will have at least one of these elements" (Ableson, King, and Sen, 2011). Since Activity "displays a UI (user interface) and responds to system and user initiated events" (Ableson, King, and Sen, 2011), it is used very frequently for Android applications. These Activities are declared in AndroidManifest.xml file, which provides "the foundation for any Android application" (Murphy, 2010). Activities present their views through XML layouts and "communicate" with each other through Intents. Clear understanding of these concepts and Java programming language is a prerequisite to start implementing the development techniques used in Android applications.

Prior Research

Contributions of prior research efforts provide useful information for successful implementation of Android Mobile EMU Portal. This thesis analyzes how to determine which development technique to use for a particular feature, what are the steps to implement each technique, and whether they can be applied for EMU-related data. Testing of the official Android EMU app has shown that it provides some of the pre-selected features for Android Mobile EMU portal like Twitter Updates, Online Directory, Athletics News, and Campus Map.

Implementation steps for development techniques required are explored in various Android development books and Internet tutorials. Since a majority of Android Mobile EMU Portal content is obtained from the Web, this section reflects existing development techniques that enable gathering online data. Steele (2011) provides "recipes" for using web content, such as "Customizing a Web Browser," "Using an HTTP GET," or "Using HTTP POST." Steele (2011) suggests that "a separate thread should be spawned anytime network data is required" and he uses *AsyncTask* class for this. Murphy (2011) describes the steps on how to use *AsyncTask* class, which include creation of an *AsyncTask* subclass, overriding one or more of its methods, creating an instance of *AsyncTask* subclass, and executing it. These steps can be implemented anytime *AsyncTask* class is needed, and Vogel (2010b) provides a working example of *AsyncTask* class implementation in his tutorial.

Jordan and Greyling (2011) combine some ideas presented in previous paragraph. For their instructions, they first describe a particular Android problem, briefly illustrate the solution (development technique), and finally explain in detail how it works. They provide

"recipes" on "Displaying Web Information" and many others (Jordan and Greyling, 2011). This approach generates step-by-step instructions for specific development techniques and shows working examples of them.

Collecting information from Twitter uses some of the techniques mentioned, such as Steele's (2011) "Using an HTTP GET" and Murphy's (2011) instructions for *AsyncTask* class. With *AsyncTask* class and HTTP GET request already in place, retrieving Twitter data can be accomplished through Twitter API (Bradby, 2011). Data returned from the Twitter API are in JSON format (Bradby, 2011), which can be parsed to store desired information.

Existing Literature Deficiencies

Although the majority of the research completed on Android development techniques is very comprehensive, existing literature still lacks support for thesis problems. These deficiencies are explored throughout the research for this thesis and explained in the following paragraphs.

First, the methodology that explicitly determines which development technique to use for which feature does not exist. Available instructions would require a certain degree of customization to adjust them for all data sources and connection types. This is especially evident for data sources that require a secure connection. Since some of the Android Mobile EMU Portal features use EMU Banner Self Services portal for data retrieval, there is a need for establishing a development technique determination process to provide guidelines.

Second, techniques currently available do not provide all necessary step-by-step instructions. Systems change in different environments, and development techniques cannot be applied the same way every time. In addition to this, security policies vary as well. Step-

by-step instructions for accessing data through EMU Banner Self-Services portal, emich.edu, and others would provide directions for successful implementation of pre-selected features.

Third, it cannot be stated with certainty that all Android Mobile EMU Portal features can be implemented. The documentation on the official Android EMU app is unavailable. An Internet search does not provide examples of Android EMU applications that exploit pre-selected features not implemented in official EMU app. Therefore, it can be concluded that while some of the Android Mobile EMU Portal features can be created for EMU, the successful completion of all of them is still uncertain.

These three problems provide research topics for this thesis. The following chapters investigate these problems and provide solutions to them.

Chapter 3: Android Application Environment

This chapter provides the information necessary for a reader to become familiar with an Android application environment in order to follow solutions given in the subsequent chapters. It outlines the steps needed to set up an environment, explains the fundamental concepts of Android application development, and describes a few common techniques frequently used for a Android Mobile EMU Portal implementation.

Android Environment Setup instructions

The steps on how to set up Android development environment are acquired from official Android website and are outlined below ("Installing the SDK," n.d.).

Install Android Java Development Kit. The first step in setting up an Android development environment is to install the official Java Development Kit (JDK) from the Sun/Oracle website:

http://www.oracle.com/technetwork/java/javase/downloads/index.html

Install Android Software Development Kit. The second step in the process involves installing the Android Software Development Kit (SDK) from Android Developers website:

http://developer.android.com/sdk/index.html

This site provides the tools and resources needed to test Android applications. Upon installation, users should verify that all the necessary components are installed via SDK Manager. The window may look like the one in Figure 1. Users need to make sure that "Tools" section, desired versions of Android APIs and possible extras are installed. SDK Manager is used to check for Android environment updates in the future. It is recommended

to install different versions of Android APIs is because not all Android devices on the market support all SDK versions, but the application should (Murphy, 2010).

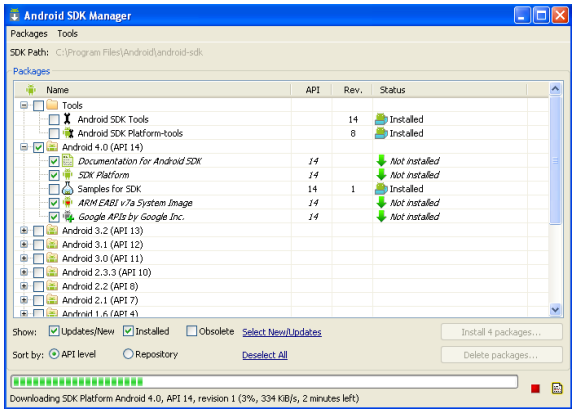


Figure 1. Android SDK Manager

Install Eclipse. The third step is to install Eclipse IDE for Java Development. Installation files can be downloaded from the official Eclipse website:

http://www.eclipse.org/downloads/

Install Android Developer Tools plug-in. After starting the Eclipse software through the executable icon, the Android Developer Tools (ADT) plug-in needs to be installed. In order to do this, click on "Help," then "Install new software." In the installation dialog box, click on "Add" and paste the following URL:

https://dl-ssl.google.com/android/eclipse

Name the instance "Android" or any way preferred. Eclipse will automatically load the list of available packages, where "Developer Tools" should be checked. Continue with the installation, click "Finish" at the end, and restart Eclipse software.

Set up testing environment. The last step in the installation process is setting up an Android Emulator for testing purposes. Felker (2011) explains how to set up an Emulator. First, start AVD Manager and create a new Android Virtual Device. In an opened window, click “New” and give this instance a name. The target option should be set up to Google APIs. Virtual memory card space can be specified that an application will operate with. Allocate enough memory to meet your application needs. When these tasks are completed, click on “Create AVD” and an emulator will be created.

To start an Emulator, select the created instance and click on “Start” from the AVD Manager window. This can be accomplished through the Eclipse window as well by clicking on Window from the main menu and then selecting AVD Manager. It will take some time until the Emulator is initialized for the first time. The process will go through two different windows until it reaches the final screen when your application can be tested. This screen looks like the one presented in Figure 2.

Figure 2. Android Emulator Screen

There is another way to test applications. It involves using a personal Android device as a debugging tool, which is an easier and faster way to navigate through an app (Steele, 2011). Android devices are preset to accept apps installed only from the Android Market, so Table 1 describes a few steps that need to be taken care of before using this feature.

Table 1

Device Setup for Development (based on "Using Hardware Devices," 2012)

Step	Action
1	Declare your application as "debuggable" in your Android Manifest: In Eclipse, you can do this from the Application tab when viewing the Manifest (on the right side, set "Debuggable" to true). Otherwise, in theAndroidManifest.xml file, add android:debuggable="true" to the <application> element.
2	Set up your device to allow installation of non-Market applications: On the device, go to Settings > Applications and enable Unknown sources (on an Android 4.0 device, the setting is located in Settings > Security).
3	Turn on "USB Debugging" on your device: On the device, go to Settings > Applications > Development and enable USB debugging (on an Android 4.0 device, the setting is located in Settings > Developer options).
4	Set up your system to detect your device: If you're developing on Windows, you need to install a USB driver for adb. If you're using an Android Developer Phone (ADP), Nexus One, or Nexus S, see the Google Windows USB Driver (http://developer.android.com/sdk/win-usb.html). Otherwise, you can find a link to the appropriate OEM driver from http://developer.android.com/sdk/oem-usb.html. If you're developing on Mac OS X, it just works. Skip this step.

Once the Android device drivers are installed on the computer, Eclipse will recognize the phone when it is plugged in via USB connection. When the developer runs an application, he or she will be presented with the screen that looks like the one in Figure 3.

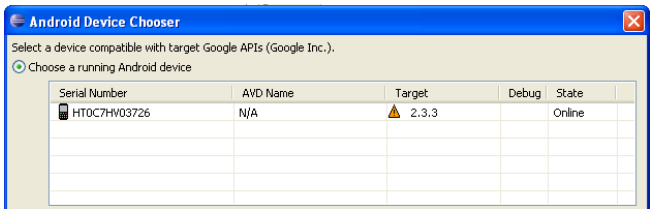


Figure 3. Android Device Chooser

After selecting Android device, Eclipse will install the files on it, and the developer will be able to test the application directly on the phone.

Android Application file structure

Android apps have a set of folders with resources that they use. Start an Android project in Eclipse by clicking “File,” then “New,” and then “Project” (Murphy, 2010). After “Android Project” is selected from Android group, follow the wizard steps to enter project, package, and main Activity names, and upon completion start developing. Each app is structured the same way, and Figure 4 gives a general overview of it.

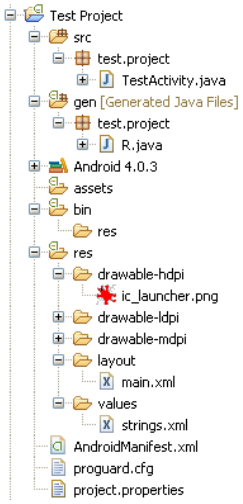


Figure 4. Android App File Structure

On top of this hierarchy is the project folder, which provides the workspace for an application. Automatic debugging in Eclipse will provide real time updates on errors that appear in the application and drill into the structure further to point out a specific resource containing errors. Those will be marked with a red “X” icon. Felker (2011) outlines the file folder structure, which can be described like this:

- “src” folder contains classes and packages. This is where the AndroidManifest.xml file will reference them from.

- “gen” folder stores automatically generated identification data about all views and widgets, which provide information that can be seen on the screen. The content of this folder should not be manually changed.
- “Android” folder contains the data about the Target Android library being used for the app.
- “assets” folder stores any additional data that an app is using.
- “bin” folder contains the data created after debugging, like project .apk file that gets installed on an actual device and information about resources used.
- “res” folder stores different app resources. Most commonly, this folder is used to reference XML layouts from the “layout” sub-folder or different images for icons.

AndroidManifest.xml file is the most important file in the Android app file structure. Felker (2011) states that the file “keeps track of everything your application needs, requests and has to use to run.” Some of its functions include providing information about available activities, services, and permissions used, or Android API level requirements.

Activities

Activities are classes within packages that interact with the user (“Activity,” 2012). These classes extend an “Activity” type and thus inherit methods and other related information needed for successful Android app implementation. Table 2 presents states that an activity can be in, depending on the user’s interaction with it. Figure 5 describes the state paths an activity can take. According to the official Android Developers website, in this figure “square rectangles represent callback methods you can implement to perform operations when the Activity moves between states (“Activity,” 2012). The colored ovals are

major states the Activity can be in ("Activity," 2012). Steps to initialize Activities are described below.

Table 2

Activity States (based on "Activity", 2012)

State	Description
1	If an activity in the foreground of the screen (at the top of the stack), it is *active* or *running*.
2	If an activity has lost focus but is still visible (that is, a new non-full-sized or transparent activity has focus on top of your activity), it is *paused*. A paused activity is completely alive (it maintains all state and member information and remains attached to the window manager), but can be killed by the system in extreme low memory situations.
3	If an activity is completely obscured by another activity, it is *stopped*. It still retains all state and member information, however, it is no longer visible to the user so its window is hidden and it will often be killed by the system when memory is needed elsewhere.
4	If an activity is paused or stopped, the system can drop the activity from memory by either asking it to finish, or simply killing its process. When it is displayed again to the user, it must be completely restarted and restored to its previous state.

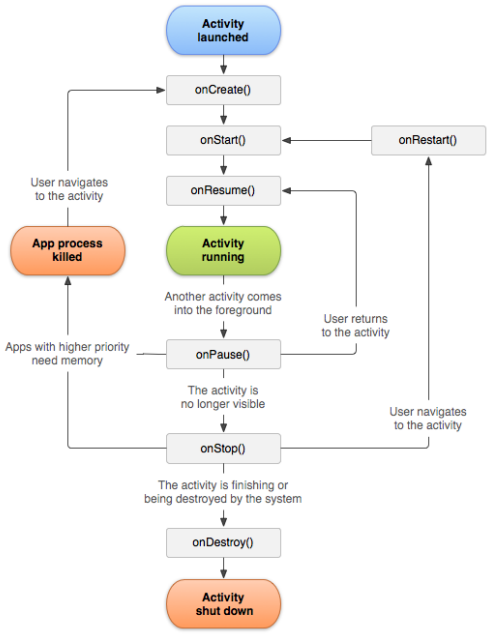


Figure 5. Activity Lifecycle ("Activity," 2012)

Define Activities. Assuming a new Android project if initialized by following the steps in first paragraph of "Android Application File Structure" section, Activities have to be defined within AndroidManifest.xml ("Activity," 2012). An Activity can be introduced by copying following code inside the <*application*> tags in the Manifest:

```
<activity android:label="@string/app_name"
android:name="test.project.TestActivity" ></activity>
```

Setting up the Activity that launches when application starts. One of activities is a "launcher" activity that opens up when the application starts. This information is generated automatically with the first Activity created. If it is desired to have a different "launcher" Activity, copy the following code within that Activity's <*activity*> tags:

```
<intent-filter><action android:name="android.intent.action.MAIN" />
<category android:name="android.intent.category.LAUNCHER" /></intent-filter>
```

Intents

Activities can use Intents to communicate with each other. Intents are "messages" shared between Activities (Felker, 2011). Each Intent can store various types of information and pass it to another Activity. Besides sending variables, developers can control current activity action by shutting it down or leaving it open.

Steps below will present the methods that Activities use in order to communicate with each other via Intents.

Initiate an Intent. In order to be able to initiate an Intent, the following line of code can be implemented within a class sending the information (Felker, 2011):

```
Intent i = new Intent(this, Target_Activity.class);
i.putExtra("token_name_1", variable_1);
```

```
i.putExtra("token_name_2", variable_2);
startActivity(i);
```

Developer can attach as many token-variable pairs as he or she wants to the Intent. To kill the previous activity, add a *finish( )* command to the method above (Meier, 2010).

Receive an Intent. Target Activity accepts the variables and makes them available for further manipulation. This is done by typing the following (Felker, 2011):

```
Intent j =  getIntent();
String var_1 = j.getStringExtra("token_name_1");
Int var_2 = j.getIntExtra("token_name_2);
```

Eclipse will suggest modifying the "get" method with a different one in case the variable being passed is an array or has a different type. To find out more, start typing "get" to invoke a method on a particular Intent and the entire list of available methods will come up on the screen.

Threads

Besides Intents, Activities are also using different Threads to perform their actions. Felker (2011) defines a Thread as "a process that runs separately from and simultaneously with everything else that's happening." Good examples of these are Asynchronous calls made during Activity lifetime which are used to develop some of discussed app features. Any time some task takes a long time to be processed, a new Thread should be launched to perform a task in the background (Vogel, 2010b). For example, if an app needs to download data from the Internet, an Asynchronous call can be made to perform this. Users may experience different Internet speeds on their devices, and without an Asynchronous call, it may appear to them that the application is "frozen" or has stopped working.

XML Layouts

XML layouts define what you a user on the screen (Murphy, 2011). Each XML layout file should be stored inside res/layout folder of your application (Murphy, 2011). XML layouts can be modified in two different ways explained below.

Manual XML layout setup. Developers can manually enter each individual object and manipulate its attributes by typing it into the file. The structure looks like this:

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
android:orientation="vertical"
android:layout_width="match_parent"
android:layout_height="match_parent" >
<object object_attributes />
</LinearLayout>
```

Objects within *LinearLayout* tags can have various object attributes. These can be Button, TextView, EditText, and so on. To reference a particular object from a class, use the following syntax (Murphy, 2011):

```
Object_Type some_object = (Object_Type) findViewById(R.id.xml_object_id);
```

Developers can also use different types of layouts instead of Linear. One that is particularly implemented in this thesis is *RelativeLayout*. This layout allows the objects to be positioned on the screen however the developer desires (Murphy, 2010). If an object needs to display the information that is taking more space than available, enclose that object in its XML file with <*ScrollView*> tags (Murphy, 2010). This way a user can scroll through the data by sliding down on its mobile device touch screen.

Automatic XML layout setup. Another way of creating XML layouts is by dragging and dropping objects inside a work area. Its advantage is that Eclipse will generate the XML code, but its disadvantage lies in the fact that it introduces only a few attributes for them. Objects are structured by their type in the Palette on the left side. Figure 6 will present a screenshot of the view provided for automatically generating XML layout files.

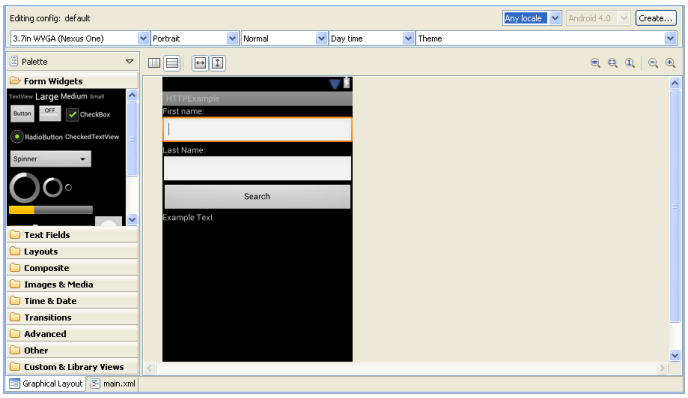


Figure 6. Graphical Layout Window

ListView objects

ListView object creates a scrollable list that can be clickable and leads to invoking other actions or can be used for simple presentation. Figure 7 shows what a typical ListView object looks like. Steps to initialize a ListView object are outlined below.

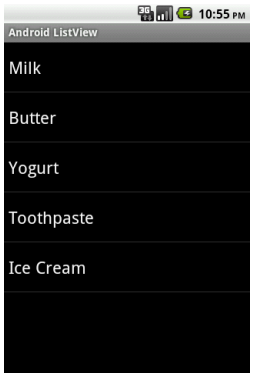


Figure 7. ListView (Anwar, 2011)

Entering ListView object into XML layout. The first step to initializing these objects is creating them in an XML layout by placing following code within <*LinearLayout*> tags (Anwar, 2011):

```
<ListView android:id="@+id/listView1"
android:layout_width="match_parent"
android:layout_height="wrap_content"></ListView>
```

A developer can enter other details to describe the ListView, like background information, rendering help, and so on. If a custom background to a ListView is set up, a user may experience a certain "flickering" effect while scrolling through the list. On the official Android Developers website, it is suggested that this issue "is caused by an optimization of the Android framework enabled by default on all instances of ListView" ("ListView Backgrounds," n.d.). This will cause a ListView to perform as displayed in Figure 8.

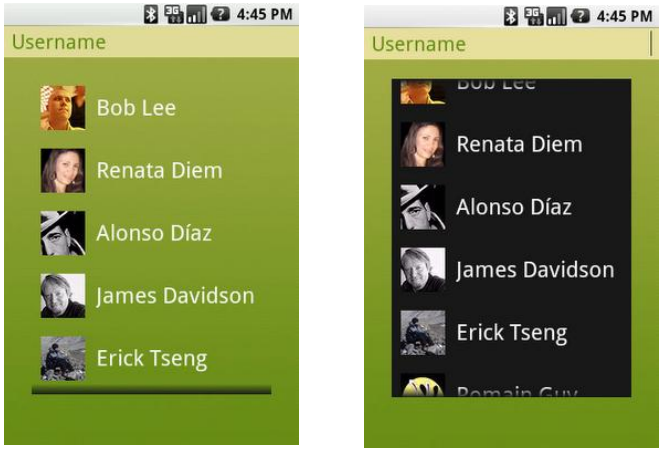


Figure 8. ListView Optimization issue ("ListView Backgrounds," n.d.)

Fortunately, there is an easy solution to this problem. The website is suggesting that "all you have to do is either disable the cache color hint optimization, if you use a non-solid color background, or set the hint to the appropriate solid color value" ("ListView

Backgrounds," n.d.). Inside an XML file's ListView tags, set the following attribute and the problem disappears:

```
android:cacheColorHint="#00000000"
```

Attaching a ListView to an Activity object. Anwar (2011) explains the basic setup of a ListView. This is the code used within an Activity class:

```
ListView listView1 = (ListView) findViewById(R.id.listView1);
String[] items = {"Milk", "Butter", "Yogurt", "Tootpaste", "Ice Cream"};
ArrayAdapter<String> adapter = new ArrayAdapter<String>(this,android.R.layout.simple_list_item_1,
items);
listView1.setAdapter(adapter);
```

This code will create a generic list like the one presented in Figure 7. The "android.R.layout.simple_list_item_1" line is a reference to a generic style defined in an Android package that will be applied to each field. This can be changed and applied on a custom layout to the fields by specifying it instead of *simple_list_item_1*.

Defining ListView "onClick" Listener. In case the desire is to make the fields clickable, a developer should introduce the following code right after setting up an adapter to a ListView (Anwar, 2011):

```
listView1.setOnItemClickListener(new OnItemClickListener() {
    public void onItemClick(AdapterView<?> parent, View view, int position, long id) {
    /* some action; position determines the location of a view within a ListView */ } });
```

Spinner objects

Spinner objects are used for dropdown menus (Murphy, 2011). For the purpose of this thesis, Spinner objects will present the data, and upon selection they will allow the

system to process them by receiving selected inputs. Steps to create Spinner objects are presented below.

Entering Spinner object into XML layout. Spinner objects are defined in an XML layout file the same way all other views are:

```
<Spinner android:id="@+id/spinner"
android:layout_width="match_parent"
android:layout_height="wrap_content"></Spinner>
```

Attaching a Spinner to an Activity object. Murphy (2011) presents basic instructions for setting up a Spinner object. The general steps are to create a String array, attach a view to the Spinner variable, define an adapter, apply it to a Spinner object, and manipulate what happens upon selection.

This is how Murphy (2011) coded his solution:

```
Spinner spin=(Spinner)findViewById(R.id.spinner);
spin.setOnItemSelectedListener(this);
ArrayAdapter<String> aa=new ArrayAdapter<String>(this, android.R.layout.simple_spinner_item,
items);
aa.setDropDownViewResource(android.R.layout.simple_spinner_dropdown_item);
spin.setAdapter(aa);
```

Defining Spinner "onClick" actions. To set up an event-based action launching when a certain item is selected or when nothing is selected, place the code within following two methods (Murphy, 2011):

```
public void onItemSelected(AdapterView<?> parent, View v, int position, long id)
{ /* some actions */ }
public void onNothingSelected(AdapterView<?> parent)
{ /* some actions */ }
```

The "parent" variable provides the identification for a specific Spinner object if many of them are used in the same Activity, while "position" gives the location of a choice inside a

Spinner. In order for Spinner objects to work the way described, the Activity needs to implement *AdapterView.OnItemSelectedListener*. This is included in the initial declaration of the class which extends the Activity (Murphy, 2011):

```
public class SpinnerDemo extends Activity implements AdapterView.OnItemSelectedListener{ /* */ }
```

Chapter 4: Development Techniques

The process of defining programming techniques needed to implement given features starts with data access and connection type analysis of the preselected features. These features do not cover every segment of information that Eastern Michigan University provides, but they provide a sufficient variety of data sources and connection types. Such analysis is given as follows, and different data sources for Android Mobile EMU Portal's features are summarized in Table 3.

Table 3

EMU Portal features and data sources

Feature	**Data Source**	**Connection Type**
Library Catalog	portal.emich.edu	Unsecured
Tuition & Fees Calculator	emich.edu	Unsecured
Online Directory	emich.edu	Unsecured
Weekly Schedule	bannerweb.emich.edu	Secured
Course Lookup	bannerweb.emich.edu	Secured
Final Grades	bannerweb.emich.edu	Secured
Twitter Updates	twitter.com	Unsecured
Athletics News	emueagles.com	Unsecured
Dining Locations	emich.edu	Unsecured
Campus Map	emich.edu	Unsecured
Bus Schedule	theride.org	Unsecured
Contact EMU	programmable	Not applicable
GPA Calculator	programmable	Not applicable

Accessing Library Catalog, Tuition & Fees Calculator, and Online Directory requires filling out PHP form for data retrieval, which does not require storing persistent state information about individual sessions.

Course Lookup, Weekly Schedule, and Final Grades use Banner for data access. Banner is EMU Enterprise Resource Planning (ERP) Suite. Banner can be accessed through

Banner Self-Services and its bannerweb.emich.edu portal. It is a secure web-based system, which requires saving cookies on the client side to bypass the verification mode.

Twitter Updates use Application Programming Interface (API) to access data without involving too much code in the application. Twitter already has its "search" API in place. Banner Self-Services also uses APIs for data retreival.

Athletics News, Dining Locations, Campus Map, and Bus Schedule use information obtained from web pages. This process involves embedding full web page views or HTML code to display information.

Two of the above mentioned features solely involve programming logic to display the results based on users' inputs. GPA Calculator operates by performing the logic that can be researched online, and it is only a matter of programming preference how this can be designed. Contact EMU is used to display a list of useful EMU-related contacts and start a preset dialer for each of them.

The data sources and connection types listen in Table 3 can be next classified in five groups. Each group requires a unique programming technique to be used for implementation. Table 4 summarizes an approach in selecting which programming technique to use for any given feature if the data sources and connection types needed for that feature are known.

Table 4

Development Techniques

Data Source	**Connection Type**	**Development Technique**
emich.edu, portal.emich.edu	Unsecured	Unsecured Web Access
bannerweb.emich.edu	Secured	Secured Web Access
twitter.com, bannerweb.emich.edu	Unsecured/Secured	Accessing APIs
emueagles.com, theride.org, emich.edu	Unsecured	Embedding Web views
All	Not applicable	Programmable Components

This methodology can be used to determine applicable development techniques for included features, as well as any additional feature. It requires that data source and connection type are determined. Based on that, development technique can be selected and used for feature implementation.

Unsecured Web Access

Web forms present a common way of communicating with the server and receiving results from it. GET and POST methods can be utilized to send requests from an Android app, based on the custom interface designed. By using existing java APIs in Eclipse, a developer can create HTTP GET and POST requests and parse the results to display them on the screen (Vogel, 2010a).

From features mentioned in the previous section, Library Catalog, Tuition & Fees Calculator, and Online Directory are the ones requiring the implementation of this technique.

There are five steps necessary to complete in order to obtain the unsecured web access, and these steps are described in detail in the following sections.

Setting up AndroidManifest.xml Internet permission. Any kind of interaction with the Internet requires setting up special permission in the application's AndroidManifest.xml file (Vogel, 2010a). This is done by adding the following code before the opening <*application*> tag:

```
<uses-permission android:name="android.permission.INTERNET" />
```

AsyncTask class implementation. Processing HTTP requests can occasionally take a long time, depending on the user's Internet connection, because mobile carriers offer

different Internet speeds. Android app has a Main Thread called the UI thread, which should contain functionality that can be quickly processed (Vogel, 2010b). If HTTP requests are placed in UI Thread, it can cause the application to crash due to unresponsiveness. Therefore, it is suggested that this code is placed in a separate Thread, and the app will perform correctly (Vogel, 2010b).

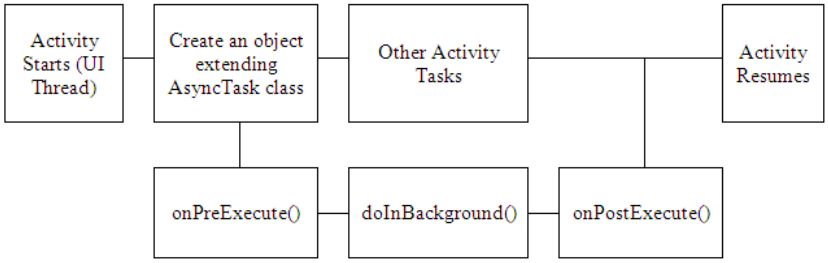


Figure 9. AsyncTask Class Implementation

As Figure 9 suggests, this can be accomplished by implementing *AsyncTask* class. On his blog, Lars Vogel (2010b) presents many valuable tutorials on how to implement different Android techniques. Vogel (2010b) explains how to use the *AsyncTask* class and provides a working example of it. A majority of the code mentioned in this section is referenced from his site. The inner class within the Activity should extend the *AsyncTask* class, and this is the syntax that is used:

```
private class Example_Class extends AsyncTask<Type_1, Type_2, Type_3>{ }
```

Each of the types in the above syntax has a specific meaning. They can be strings, integers, or any other type of data usually implemented. The first type is the type of the parameter being passed to a *doInBackground( )* method, which is required to be invoked within a class (Vogel, 2010b). The second type is used for progress information, while the third type

is the type of the result that *doInBackground( )* method will return (Vogel, 2010b). This result will be passed to *onPostExecute( )* method as a parameter (Vogel, 2010b).

An object of this class can be initiated like this:

```
Example_Class test = new Example_Class();
test.execute(Type_1 sample_variable);
```

The *execute()* method can have a parameter that needs be the same as *Type_1* from the earlier syntax. The class will run the tasks within *onPreExecute( )* method and start working on the main task defined in *doInBackground( )* method. *onPreExecute( )* method is a good place to initialize the private *ProgressDialog* object that should be declared at the beginning of *Example_Class* (Vogel, 2010b). *ProgressDialog* object displays the progress bar until this object is dismissed. In essence, it tells the user that the application is currently working. This is how it can be initialized:

```
progress = ProgressDialog.show(Parent_Class.this,"","Loading Message",true);
```

By using the *AsyncTask,* an action is performed "in the background" asynchronously (Vogel, 2010b). A class is created that extends the features of an *AsyncTask*, executes the request, parses, and presents the results. All of the following code is placed within a *doInBackground( )* method, which passes a *Type_1* variable as a parameter and returns *Type_3* variable:

```
protected Type_3 doInBackground(Type_1 sample_variable)
{ /* code below goes here */ return result_variable; }
```

After *doInBackground( )* method completes the task, it returns the *Type_3* value to *onPostExecute( )* method. Since the main chunk of the work is done, this is where *ProgressDialog* object should be dismissed. Use the *onPostExecute( )* method like this:

```
protected void onPostExecute(Type_3 result_variable){ progress.dismiss(); }
```

A developer should place the next action that an Activity takes within *onPostExecute( )* method. Acquiring data from the Web can be a lengthy process. By waiting until the initial task is completed and then invoking the next action, a developer can avoid runtime errors.

Executing HTTP request. After an *AsyncTask* class object is in place, HTTP request needs to be prepared and added to the *doInBackground( )* method. Therefore, this method contains all the code presented in this section.

Vogel (2008) describes the process of utilizing HttpClient for HTTP requests. *DefaultHttpClient* object needs to be created to interact with the Web:

```
DefaultHttpClient client = new DefaultHttpClient();
```

After this, *HttpPost* object contains URL where an app submits the POST request as a parameter:

```
HttpPost httpost = new HttpPost("url");
```

In order to submit the inputs, the *List<NameValuePair>* array needs to be created. This array stores the pair of form input names with the corresponding values:

```
List<NameValuePair> pairs = new ArrayList<NameValuePair>(number_of_pairs);
```

Each pair is added to this array through an "add" method as a *BasicValueNamePair*, where first value is the input name string, while the second is an input value string:

```
pairs.add(new BasicValueNamePair(name, value));
```

These data are attached to the HttpPost object via *setEntity( )* method, which takes *UrlEncodedFormEntity* object as a parameter. This object can be initialized earlier and uses *List<NameValuePair>* as a parameter:

```
httppost.setEntity(new UrlEncodedForEntity(pairs));
```

HttpPost object is ready for execution. Use a *HttpResponse* object to store the response from this request:

```
HttpResponse results = client.execute(httppost);
```

Converting HttpResponse object to a String. The variable "results" from previous section stores HTML code that would have been displayed in a browser. This information can be converted into a string for easier manipulation. The response can be converted to a string like this:

```
InputStream content = results.getEntity().getContent();
BufferedReader input_reader = new BufferedReader(new InputStreamReader(content));
String s = ""; String string_response = "";
while ((s = input_reader.readLine()) != null) { string_response += s; }
```

Parsing the results with internal/external parsers. In the previous step, a string variable named "string_response" is stored. This variable can be used to extract text, attributes, or links from any HTML tag. This is called parsing, and it can be performed by using internal *XmlPullParser* and *SAXParser,* or external parsing libraries like *Jsoup*.

Using the internal XmlPullParser parser. The following code illustrates how to utilize XmlPullParser ("XmlPullParser," 2012). First, initiate the *XmlPullParser* object:

```
XmlPullParserFactory factory = XmlPullParserFactory.newInstance();
factory.setNamespaceAware(true);
XmlPullParser xpp = factory.newPullParser();
```

With the code above, the parser is initialized and can be used to review the response string and parse the information:

```
xpp.setInput(new StringReader(string_response));
 int eventType = xpp.getEventType();
while (eventType != XmlPullParser.END_DOCUMENT)
{ /* read data */  eventType = xpp.next(); }
```

The "while" loop is used to scroll through the results and select what is needed. Different methods can be used to extract data, but explaining this gets complicated, because parsing is a topic for itself and not a primary focus of this research.

Using the external Jsoup parser. Jsoup is a popular parsing library that works well with Android apps (Houston, 2012). Library JAR file can be downloaded from official Jsoup website *http://jsoup.org*. In Eclipse, a developer will have to right click on the project, select "Properties," then "Java Build Path," and import the library from an external location. Once this is done, apply the methods and implement Jsoup parser in the application like this:

```
Document doc = Jsoup.parseBodyFragment(string_response);
```

A particular section from a page can be extracted in many ways:

```
Element body = doc.body();                          //selects everything within <body> tags
Element specific_section = doc.select("table").first();     //selects first <table> tag
Element specific_section = doc.select("a").get(n);          //select (n+1)-th <a> tag
```

More information on Jsoup parsing can be found on *http://jsoup.org*. This page provides downloads and resources on how to use this package.

Parsing is a method that can be used, assuming that a developer is already familiar with concepts of HTML and XML.

Secured Web Access

Banner is an Oracle-based system used at EMU as an ERP solution. It is a complicated system with thousands of tables that include confidential data about students, faculty, staff, or certain university processes.

Oracle applications are housed behind the firewall, which makes them very secure, but also difficult to access from the outside (M. Howell, personal communication, January 11, 2012). Banner is a source of information needed for Course Lookup, Weekly Schedule, and Final Grades. Both are available in Web form through Banner Self-Services web portal, but its usage is not exactly suitable for small screens on mobile phones.

There are two solutions provided for enabling secured Web access to these applications, and both are the described in the sections that follow.

Oracle Database Mobile Server Implementation. The first method of accessing the Oracle Database is by using Oracle Database Mobile Server. The official Oracle website suggests that this is "the best way to securely connect embedded devices and mobile applications to Oracle Database" (Oracle, 2011a). Since EMU has an established enterprise environment, this technology "is well suited for mission critical applications or any

application where high performance and reliability are required" (Oracle, 2011a). The key benefits of using Orable Database Mobile Server include:

- Secure, efficient, resilient mobile synchronization with Oracle Database.
- Remote application, user, and device management.
- Standards-based encryption for remote data, in both storage and transit.
- Robust and reliable mobile data synchronization over unreliable networks.
- Highly scalable server configuration, supporting large and growing mobile or remote deployments.

This technology also supports cross-platform development and "familiar data access interfaces such as ODBC, JDBC, and ADO.NET" (Oracle, 2011a). Table 5 presents the supported client platforms, whereas Figure 10 illustrates the implementation of Oracle Database Mobile Server.

Table 5

Oracle Supported Client Platforms (Oracle, 2011a)

O/S	ODBC	JDBC	ADO.Net
Java	N/A	Yes	N/A
Android	N/A	Yes	N/A
Blackberry	N/A	Yes	N/A
Windows Desktop and Mobile	Yes	Yes	Yes
Linux	Yes	Yes	N/A

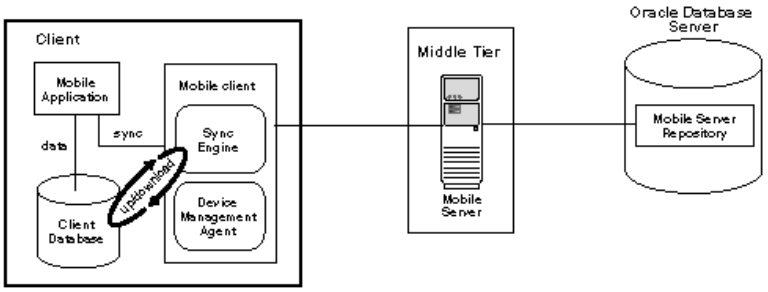


Figure 10. Architecture for Device with a Mobile Client and Client Database (Oracle, 2011b)

An existing mobile app can have SQLLite database designed and implemented on the front end mobile device (Oracle, 2011a). The SQLite Mobile Client on the device would perform SQLite database synchronization with the Oracle Database on the back-end via its Sync Engine and through the Oracle Database Mobile Server (Oracle, 2011a).

There are many inconveniences with utilizing this technique, and that is why its implementation is not described in more detail. First of all, it requires an installation of the Mobile Server, which includes acquiring the product and necessary training. Second, Banner contains enormous amounts of data, so syncing it with a local database can take time and, most important, a significantly large memory space, which is not always available (M. Howell, personal communication, November 2, 2011). Together, both the installation and performance issues do not present Oracle Database Mobile Server as a best solution to the problem. However, it presents the necessity of utilizing the "Middle Tier," and this concept is very useful.

Utilization of Banner Self-Services portal. A preferable method for developing Weekly Schedule, Course Lookup, and Final Grades features is by using the existing infrastructure in form of Banner Self-Services portal. This is a web application providing the interface for users' interaction with Banner data in a secure manner (M. Howell, personal communication, November 2, 2011). Banner Self-Services portal has packages already installed, which make the application easier to use.

Literature does not provide a significant amount of information on how to combine smaller features to retrieve information from this type of a secured site. Research in this thesis showed that Banner Self-Services portal is programmed to place a user in the verification mode once the initial HTTP request is sent. Any time a user logs into Banner

Self-Services, the system will store cookies on its machine with EMU-specific data to maintain the state. Once cookies exist, the user can utilize packages within the domain.

However, further online research exposed a Java class that performs what is needed. It utilizes the Banner Self-Services portal customized for Dowling College located in Oakdale, Long Island, NY, and makes a connection to the system based on the user's credentials (Dowling College, 2005). Since this portal is the same for all educational institutions that are using Banner Self-Services, pieces of the code provided in the class can be implemented for EMU-specific solution and customized to achieve the same goal. This class is attached in Appendix A.

Analysis of the class attached in Appendix A derived steps for successful secured Web access implementation. Before a detailed description of each step is introduced, a developer should enable Internet permission in AndroidManifest.xml file and create an *AsyncTask* class object to execute these steps. This can be accomplished by following the procedures from "Setting up AndroidManifest.xml Internet permission" and "AsyncTask class implementation" sections of this thesis. All steps should be executed in *doInBackground()* method, and they are described below.

Initialization of HttpsURLConnection object and Login Cookies String. The first step to obtaining a secure Web access to Banner Self-Service via mobile app is creating *HttpsURLConnection* variable (Dowling College, 2005). This variable is used to connect to the Banner Self-Services, maintain the cookies on the client, and extract data from Banner. The *HttpsURLConnection* variable is private, and it is used throughout this Activity's life cycle:

```
private HttpsURLConnection connection;
```

Another important variable that needs to be initialized is the "login cookies" string, which holds the information stored in cookies required by Banner Self-Services portal packages. The syntax mimics the name/value pair structure as follows, with the pairs modified to satisfy particular system's needs:

```
String LOGIN_COOK = "NAME_1=VALUE_1; NAME_2=VALUE_2 ";
```

If there is more than one value being passed around, each of them is separated by the semicolon and a space (Dowling College, 2005). These cookies function under a certain domain specified with its creation. Research in this thesis showed that the domain used by Banner Self-Services portal is bannerweb.emich.edu. The cookies needed for this connection are available in browser cookie history upon access and are outlined below:

- BW_COOKIE: R2158265430
- TESTID: set
- SESSID: *generated_value*
- accessibility: false

Enabling Certificates. The second step involves making sure that an application is set to accept incoming certificates. The "Enabling Certificates" step came in effect upon unsuccessful application testing on the mobile device. The Apache Software Foundation (n.d.) website states that "HttpClient makes use of SSLSocketFactory to create SSL connections. It can take an instance of javax.net.ssl.SSLContext as a parameter and use it to create custom configured SSL connections" (The Apache, n.d.). The code provided suggests the implementation of *X509TrustManager* object for managing certificates as follows:

```
TrustManager easyTrustManager = new X509TrustManager() {
```

```
@Override
public void checkClientTrusted(X509Certificate[] chain, String authType) throws CertificateException{ }
@Override
public void checkServerTrusted(X509Certificate[] chain, String authType) throws CertificateException{ }
@Override
public X509Certificate[] getAcceptedIssuers() { return null; }};
SSLContext sslcontext = SSLContext.getInstance("TLS");
sslcontext.init(null, new TrustManager[] { easyTrustManager }, null);
```

For the sake of Android Mobile EMU Portal, the "TLS" parameter of *getInstance( )* method needs to be modified to "SSL." Finally, a developer should set the *DefualtSSLSocketFactory* object *HttpsURLConnection* by adding a following line to the example suggested above (The Apache, n.d.):

```
HttpsURLConnection.setDefaultSSLSocketFactory(sslcontext.getSocketFactory());
```

All certificates involved in the process of logging into the Banner Self-Services system will be collected, which is acceptable for this connection. The customization of this solution should be considered, which requires a deeper understanding of SSL/TLS protocol.

Executing HttpsURLConnection object. The third step involves setting up parameters for *HttpsURLConnection* and executing an object of this type. The packages involved in retrieving the data from EMU Banner Self-Services portal are researched by accessing the Banner Self-Services system and are outlined in Table 6.

Table 6

Banner Self-Services portal packages

Package	**Description**
twbkwbis.P_ValLogin	Validates the Login
bwskfshd.P_CrseSchd	Displays Weekly Course Schedule
bwskfcls.P_GetCrse	Displays the Search results for Class Lookup
bwskogrd.P_ViewTermGrde	Displays terms for Final Grades
bwckogrd.P_ViewGrde	Displays Final Grades for a particular semester

The packages in Table 6 are used for appropriate actions. As an example, the full address of *twbkwbis.P_ValLogin* package is used in place of a "package_url" variable described with the code below. To access these packages, a class attached in Appendix A suggests opening a secure connection and setting up the following parameters for executing the secure HTTP request to Banner Self-Services (Dowling College, 2005):

```
url = new URL("package_url");
connection = (HttpsURLConnection) url.openConnection();
connection.setRequestMethod("POST");
connection.setDoInput(true);
connection.setDoOutput(true);
connection.setUseCaches(true);
connection.setRequestProperty("Cookie", LOGIN_COOK);
```

The greatest chunk of the login request is prepared for the execution. In order to retrieve the information upon the verification of the user's credentials, a *DataOutputStream* object should be used to write the bytes of information into the stream (Oracle, 2010a). For converting the bytes of information from a *HttpsURLConnection* object that has a *DataOutputStream* attached, a *BufferedReader* object uses an *InputStreamReader* (Oracle, 2010b). This sequence of events executes the secure HTTP request, and the code from Appendix A is copied here (Dowling College, 2005):

```
DataOutputStream out = new DataOutputStream(connection.getOutputStream());
String content = "sid=" + URLEncoder.encode(userId, "UTF-8") + "&pin=" +
                 URLEncoder.encode(password, "UTF-8");
out.writeBytes(content);
out.flush(); out.close();
parseCookie();
BufferedReader input = new BufferedReader(new InputStreamReader(connection.getInputStream()));
String alllines = ""; String line = "";
while ((line = input.readLine()) != null) { alllines += line + "\n"; }
input.close();
```

A *DataOutputStream* object attaches the "sid" and "pin," which are the user's EMU ID and PIN numbers, flushes the request, and gets the bytes of resulting data from it. A *parseCookie( )* method is used to retreive the new cookie string that now contains the session information upon the successful login. This string is important if the desire is to further maintain the connection to Banner Self-Services and exploit its packages for data retrieval (Dowling College, 2005). This is the code for *parseCookie( )* method:

```
private void parseCookie() { cookie = connection.getHeaderField("Set-Cookie"); }
```

With the code above, an "alllines" variable stores an HTML page, which generates a main menu within Banner Self-Services or shows an error message due to an unsuccessful login. If the goal is to access specific data, an another secure HTTP request should be initialized with the new cookie string that is saved through the *parseCookie( )* method (Dowling College, 2005). This is done in the same manner as stated in the explanation above, except this time a developer does not need to use the *writeBytes( )* method because the user is already within a system, nor the *parseCookie( )* method because cookies are already set (Dowling College, 2005).

Parsing the results. This can be accomplished by following the procedure described in "Parsing the results with internal/external parsers" section from this thesis. An "alllines" variable stored in the previous step is the one used for parsing.

Accessing APIs

This technique is essentially using the steps for unsecured Web access described earlier, but it adds an additional attribute to it and requires an API to be in place in order to perform the intended actions. An API stores the logic and provides the user with the ability to

effectively use its power without much coding involved. These are particularly well suited for introducing Twitter updates regarding EMU community in Android Mobile EMU Portal.

Before introducing steps to access a Twitter API, it should be noted how a developer can send an HTTP request to a Banner package API and deliver filtered data to the audience. This can be accomplished by adding the name/value pairs of data to a web address string:

bwskfshd.P_CrseSchd?start_date_in=01/09/2012

This way, the "Course Schedule" package is filtered to jump directly to the week starting on January 9, 2012. Various data can be added by separating the name/value pairs with an ampersand to specifically meet the needs. In order to use this, certain procedures need to be placed in the rewrite rules for the packages (M. Howell, personal communication, January 11, 2012). These procedures are enabled by the EMU Information Technology Department.

However, due to a certain vulnerability, Android Mobile EMU Portal is only allowed to view the data from Banner and cannot write information to its tables (M. Howell, personal communication, January 11, 2012). For example, this means that the user can view the list of the available classes but cannot register for them. SunGard Higher Education, Banner provider for EMU, is currently working on fixing this issue. The Information Technology Department provided more information about it in a document attached in the Appendix B.

The steps to enable interaction with Twitter API are introduced in the following sections. Before a detailed description of each step is shown, a developer should enable an Internet permission in AndroidManifest.xml file and create an *AsyncTask* class object to execute these steps by following the procedures from "Setting up AndroidManifest.xml

Internet permission" and "AsyncTask class implementation" sections of this thesis. All steps should be executed in *doInBackground( )* method and they are described below.

Executing HTTP request. Bradby (2011) suggests that developers can create a "simple Android app to display a list of tweets coming from the JSON based search API provided by Twitter." This API retrieves the search results from Twitter, which can be used to display "tweets" from specific users. The first step to accessing Twitter API is executing an HTTP request like this:

```
HttpClient hc = new DefaultHttpClient();
HttpGet get = new HttpGet();
get.setURI(new URI("http://search.twitter.com/search.json?q=from%3acnnbrk+OR+from%3anpratc"));
HttpResponse rp = hc.execute(get);
```

The first line creates a *HttpClient* that is needed for the connection. Then, after setting up an *HttpGet* object, its function needs to be specified. Notice how the string in parentheses is created. After adding "?q=" characters, a developer can specify the users an application should receive updates from. A keyword "OR" is introduced to combine the tweets from all users included in the syntax. This API will automatically sort the results chronologically. In the example above, the search will display the tweets from CNN Breaking News and NPR's evening news magazine.

Storing individual "tweets." Bradby (2011) suggests creating a "helper" class to store the information for each individual "tweet" about its content and the author. This class is created by right-clicking on the package within the project, selecting "New," and choosing "Class." The following constructor is used for this class:

```
public class Tweet { String author; String content; }
```

A *Tweet* class can be modified, depending on what other information is intended for display. In the variable declaration section of the parent Activity, an object to hold the "tweets" needs to be initialized like this:

```
private ArrayList<Tweet> tweets = new ArrayList<Tweet>();
```

The previous step yielded an *HttpResponse* object "rp," which stored the data about individual tweets received. Right after an *HttpResponse* object, copy the following code:

```
if(rp.getStatusLine().getStatusCode() == HttpStatus.SC_OK)
{
	String result = EntityUtils.toString(rp.getEntity());
	JSONObject root = new JSONObject(result);
	JSONArray sessions = root.getJSONArray("results");
	for (int i = 0; i < sessions.length(); i++) {
		JSONObject session = sessions.getJSONObject(i);
		Tweet tweet = new Tweet();
		tweet.content = session.getString("text");
		tweet.author = session.getString("from_user");
		tweets.add(tweet);
	}
}
```

By executing the code Bradby (2011) implemented in his tutorial, a List object "tweets" is populated to contain all Twitter API search results, which can be displayed as preferred. One of the easier ways to do it is by implementing a ListView object with an adapter by following the procedure from a "ListView" section of this thesis.

Twitter updates are easy to retrieve and display if only text data like username, messages, or dates posted are captured. A Twitter API also provides the web locations of images associated with each tweet. Downloading these images can take a long time. Presenting the images in a generic ListView involves a redownload of the same images as the

users are scrolling through the list, which is inefficient and slow (Cois, 2011). Optimized custom solution is provided in Chapter 5.

Embedding Web views

In order to embed the views from the Web, there needs to be a way to capture the full web pages, or pieces of HTML code within an application. According to many different sources, a WebView is the preferred way of presenting web pages and other online resources if their content can be displayed so the users have no issues in reading it. The features embedding data from the Web include Athletics News, Dining Locations, Campus Map, and Bus Routes and Schedules.

A WebView is an Android app layout view, and it behaves like a browser within an app. A developer can present both the internal and external files in a WebView, and they can have different types. A WebView can reference the Web URLs or files from the resource folder, which is a part of Android app structure.

Implementing this technique is quick and efficient and requires the steps outlined below. Before a detailed description of each step is introduced, a developer should enable an Internet permission in AndroidManifest.xml file by following the procedures from "Setting up AndroidManifest.xml Internet permission" section of this thesis.

Initializing a WebView in XML layout. The first step is to introduce a WebView element in an XML layout file ("Building Web Apps," 2012):

```
<WebView android:id="@+id/element_name" />
```

Within these tags, a developer can set up the preferences for a WebView, such as screen size, and many others ("Building Web Apps," 2012). The screen can be manipulated as preferred to make sure an application is viewable on the small cell phone displays.

Attaching a WebView object to an XML layout view. The second step is to initialize the WebView object in its native Activity class. If a user tries to open a link within a WebView, the default behavior is to send the link address to the phone's Internet browser and open it there. This requires a user to switch between the applications. This can be avoided by setting up a WebViewClient to a WebView ("Building Web Apps," 2012):

```
WebView testView = (WebView) findViewById(R.id.element_name);
testView.setWebViewClient(new WebViewClient());
```

The last part involves entering the URL of a webpage or a resource that needs to be displayed within a WebView:

```
testView.loadUrl(some_url);
```

These are the steps required for a basic WebView implementation. It is a simple and very useful technique. Figure 11 displays a typical WebView. This object is presenting the pages but does not have a navigation bar to enter addresses and should not be confused with an Internet browser.

A WebView object is not limited to presenting only URLs from the Web. A string variable in form of an HTML page can be displayed as well (Murphy, 2011):

```
testView.loadDataWithBaseURL("", html_string, "text/html", "utf-8", "");
```

Figure 11. WebView Screenshot

Customizing WebView settings. This step is optional. However, if the page displayed requres special attention, a developer should modify the WebView settings accordingly. On Android Developer website ("Building Web Apps," 2012), they suggest adding the following code after initializing a WebView object to enable JavaScript:

```
WebSettings test_settings = testView.getSettings();
test_settings.setJavaScriptEnabled(true);
```

A WebView object is not limited to presenting only web pages. If the PDF files need to be embedded in a WebView, there is a specific trick used to display the external PDF files, and it involves the usage of an online service provided by Google. In the "Android – Load PDF / PDF Viewer" discussion thread on stackoverflow.com, Mayani (2010) suggested using the following:

```
testView.loadUrl("http://docs.google.com/gview?embedded=true&url=pdf_path");
```

It works great and provides the necessary functionality to display a PDF file, zoom in/out, and scroll through it. This trick requires JavaScript to be enabled.

To disallow clicks within a WebView, a developer needs to disable opening the new windows automatically and choose to handle the clicks personally by overwriting a WebViewClient (Nuetzmann, 2009). Use the following code:

```
testView.getSettings().setJavaScriptCanOpenWindowsAutomatically(false);
testView.setWebViewClient(new WebViewClient() {
    @Override
    public boolean shouldOverrideUrlLoading(WebView view, String url) {
    // handle the click yourself
    return true; } });
```

Programmable Components

All Java programming techniques can be implemented for Android app development. Discussing this particular topic is a work for itself. The major concepts that are used involve passing the parameters in the class methods, declaring the private and public variables, and creating the inner classes within Activities.

From the list of features introduced earlier, GPA Calculator and Contact EMU are the only ones that fully belong to this group, because they do not require online resources for their implementation. Users can plug the values in the application dialog box and calculate the result based on the logic or click on an object to instantiate a desired action.

Chapter 5: Implementation of the Android Mobile EMU Portal

Android Mobile EMU Portal is fully developed and tested on an actual device. The application's file structure is presented in Figure 12.

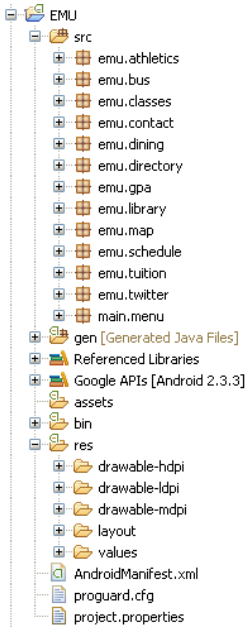


Figure 12. Android Mobile EMU Portal file structure

Most of the packages contain classes corresponding to one specific app feature. A package named "emu.classes" contains the classes used for Course Lookup and Final Grades, while the "main.menu" package holds the classes used for Main Menu. The following sections contain descriptions of AndroidManifest.xml file, Main Menu, and all the application features.

Application AndroidManifest.xml

The AndroidManifest.xml file is the most important file within the Android application structure. This is where all Activities, permissions, and target libraries are

defined. The AndroidManifest.xml file content for the Android Mobile EMU Portal is provided in the Appendix C.

For understanding, a few concepts need to be pointed out. Since the application uses Internet resources and GPS location for its features, the permission section of the AndroidManifest.xml file looks like this (Vogel, 2010c):

```
<uses-permission android:name="android.permission.INTERNET" />
<uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION" />
<uses-permission android:name="android.permission.ACCESS_FINE_LOCATION" />
```

The application is set to prefer an external location for "installLocation," which means that an app will attempt to install itself on the device's memory card if available. The minimum SDK version is 10, but this can be changed to a lower one to match the other target build libraries for older mobile devices.

An "icon" attribute provides the location of an icon that will be displayed on the device for the application and "label" provides the name which will show up in the "Applications" section within phone's settings. A "label" attribute determines the name that the users see in the "All Apps" menu displayed in Figure 13. The individual labels for each Activity define the title displayed in the particular Activity window.

Figure 13. "All Apps" Menu

EMUActivity has a "singleTop" *launchMode* attribute, which controls the Activity stack when the user is interacting with this Activity (Klumpp, 2011). Most Activities are "killed" upon exit which removes them from an Activity stack. If the same was done with EMUActivity, a user would not be able to see the application's main menu after he/she leaves it the first time. However, if a user wanted to return to this Activity, Android would create another instance of the same Activity, causing two of them to be open at the same time. A user would have to exit the application as many times as there are EMUActivity instances, which is a bad practice. What solves the issue, is introducing the "singleTop" attribute and supporting it with a "flag" attached to an Intent used to migrate from a particular Activity (Klumpp, 2011).

Main Menu

Main Menu is the application's dashboard to the other Activities. Figure 14 presents the layout used for Main Menu. This layout uses icons that are stored in the "drawable" folder. The complete code for the main.xml file and the EMUActivity which are used for Main Menu is provided in the Appendix D.

Each of the buttons in the layout has an *onClick* action attached to it. These are defined inside the class using the layout, otherwise, the application would crash.

The methods *btn_2( )* and *btn_13( )* attach an "ACTIVITY_CHECK" variable to an Intent. This is needed because Class Lookup and Final Grades features use the same Classes_Login class for verification.

Before introducing the individual features, it should be mentioned that all of them contain a white "Eastern Michigan University" button on the bottom of the screen. When

pressed, this button takes a user back to the Main Menu, and is common to all layouts other than the Main Menu layout.

Figure 14. Main Menu Screenshot

Library Catalog

Library Catalog presents a user with a search field to look for a particular Halle Library resource and then displays the list of results in the next window. Figure 15 shows the screens that users will be presented with.

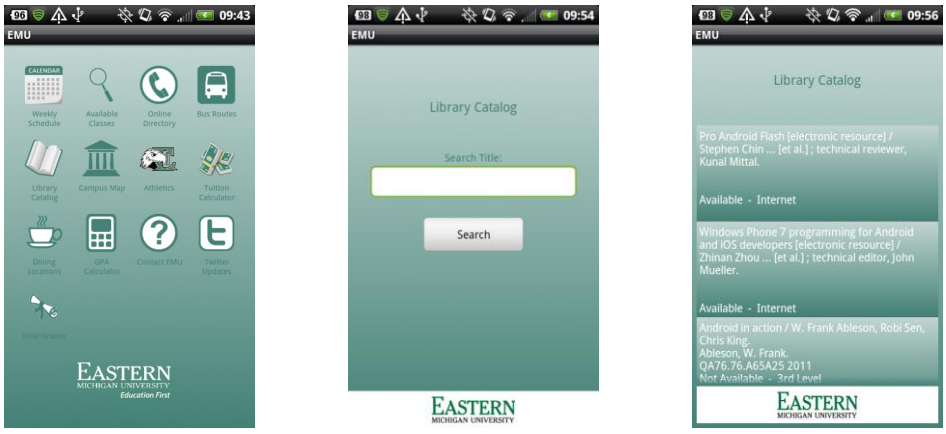
If a user clicks on the "Library Catalog" icon in Figure 15.a, he/she is presented with the search view. By entering the search item in Figure 15.b, this feature will execute an unsecured HTTP request and display the results in the Figure 15.c. This request is formed within *doInBackground( )* method of an *AsyncTask* by following the instructions like this:

```
DefaultHttpClient client = new DefaultHttpClient();
HttpGet httpget = new HttpGet(url);
try {
       HttpResponse execute = client.execute(httpget);
       InputStream content = execute.getEntity().getContent();
      BufferedReader buffer = new BufferedReader(new InputStreamReader(content));
```

```
        String s = "";
        while ((s = buffer.readLine()) != null) { response += s; }
        Document doc = Jsoup.parseBodyFragment(response);
        Elements results_content = doc.select("div.resultListTextCell");
        response = "";
        for (Element book : results_content) {
            String line_1 = book.select("div.line1Link").first().text();
            String line_2 = book.select("div.line2Link").first().text();
            String line_3 = book.select("div.line4Link").first().text();
            String line_4 = book.select("div.line5Link").first().text();
            response = line_1 + "\n" + line_2 + "\n" + line_3 + "\n" + line_4 + "\n";
            search_results_items.add(response);
            }
}
```

Jsoup external parser is used to save the data from the page after the execution. Notice in the example above that the "search_result_items" variable stores individual result information. This list is used for a ListView adapter to present detailed results as shown in Figure 15.c. The code is executed in onPostExecute() method of an *AsyncTask* like this:

```
adapter = new ArrayAdapter<String>(myContext, R.layout.library_item, search_results_items);
library_results.setAdapter(adapter);
```



a. Main Menu b. Library Search c. Catalog Results

Figure 15. Library Catalog Screenshots

The variable “library_results” is a ListView object initialized in the Library_Catalog Activity. Library Catalog uses library_search.xml, library_item.xml and library_catalog.xml for layouts and Library_Search and Library_Catalog classes as its Activities. The complete code is provided in the Appendix E.

Tuition & Fees Calculator

Tuition & Fees Calculator computes the user’s predicted tuition costs based on inputs entered. Figure 16 presents the sequence of events that happens in this feature.

After a user clicks on the “Tuition Calculator” icon from Figure 16.a, he/she would have to enter the inputs in the screen from Figure 16.b, and the Tuition Calculator would then compute the results and display them in the layout presented in Figure 16.c. For its implementation, Tuition & Fees Calculator uses tuition_calculator.xml and tuition_display.xml for XML layouts and Tuition_Calculator and Tution_Display as its Activities. This HTTP request is formed within *doInBackground( )* method of an *AsyncTask*. The *BasicNameValuePair* is used like this:

```
List<NameValuePair> nameValuePairs = new ArrayList<NameValuePair>(16);
nameValuePairs.add(new BasicNameValuePair("residency", residency_value));
nameValuePairs.add(new BasicNameValuePair("credits0", n1));
nameValuePairs.add(new BasicNameValuePair("credits1", n2));
nameValuePairs.add(new BasicNameValuePair("credits2", n3));
nameValuePairs.add(new BasicNameValuePair("credits3", n4));
nameValuePairs.add(new BasicNameValuePair("credits4", n5));
nameValuePairs.add(new BasicNameValuePair("level0", level0));
nameValuePairs.add(new BasicNameValuePair("level1", level1));
nameValuePairs.add(new BasicNameValuePair("level2", level2));
nameValuePairs.add(new BasicNameValuePair("level3", level3));
nameValuePairs.add(new BasicNameValuePair("level4", level4));
nameValuePairs.add(new BasicNameValuePair("grouping0", grouping0));
nameValuePairs.add(new BasicNameValuePair("grouping1", grouping1));
nameValuePairs.add(new BasicNameValuePair("grouping2", grouping2));
nameValuePairs.add(new BasicNameValuePair("grouping3", grouping3));
```

```
nameValuePairs.add(new BasicNameValuePair("grouping4", grouping4));
httppost.setEntity(new UrlEncodedFormEntity(nameValuePairs));
```

The code above illustrates how the specific data is submitted for a query. The full code for this feature is provided in the Appendix F.

a. Main Menu b. Tuition Calculator c. Tuition Display

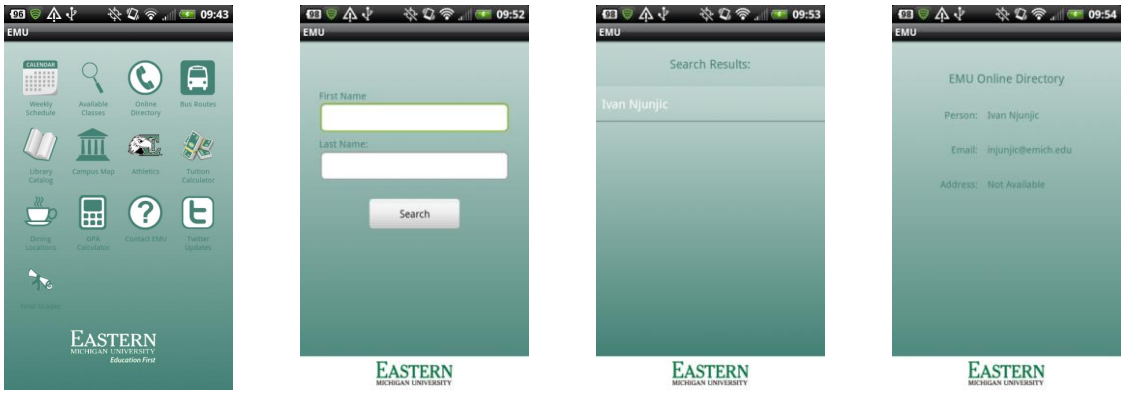
Figure 16. Tuition & Fees Calculator Screenshots

Online Directory

Online Directory enables a user to search through the directory of students, faculty, and staff at EMU and retrieve particular information about them. Figure 17 provides the typical events that take place if a user intends to perform this action.

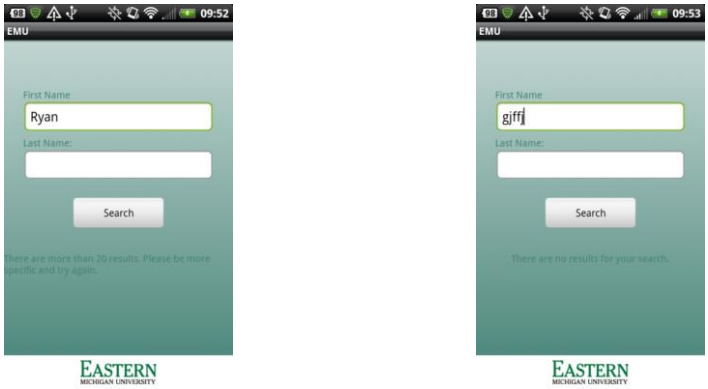
If a user clicks on the "Online Directory" icon, as shown in Figure 17.a, he/she will be presented with the search options as shown in Figure 17.b. Clicking on the Search button leads to three possible outcomes. A successful search yields the results as presented in Figure 17.c, while the unsuccessful search results in the messages shown in Figures 18.a and 18.b. These can happen if there are more than 20, or no results available for the searched inputs.

If a search is conducted successfully and the user clicks on a particular result shown in Figure 17.c, a new window opens which looks like the one presented in Figure 17.d. At this point, a user can review the information displayed or decide to send an email to a person. By clicking on the email data, the new Activity opens as shown in Figure 19 with the preset address information. This is useful for contacting people through this application, which is convenient and eliminates the necessity of copying the email address before using the default mail client.



a. Main Menu b. Directory Search c. Search Results d. Person Info

Figure 17. Online Directory Screenshots



a. Over 20 results error b. No results found

Figure 18. Online Directory Error Messages Screenshots

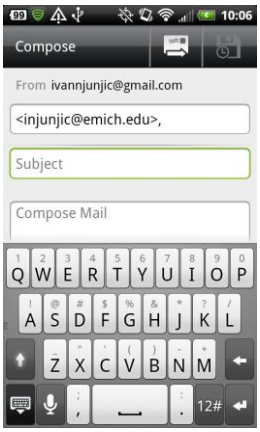


Figure 19. Email Composition Window Screenshot

Online Directory uses an unsecured HTTP request and a ListView object. What is new, and not used in previously described features, is the way that this app invokes an email window. The following code for "email_btn" within an Intent defines an *onClick* handler for the email field ("Start Email-Activity," 2009):

```
public void send_email(View view) {
        Intent email_btn = new Intent(Intent.ACTION_SEND);
        email_btn.setType("plain/text");
        email_btn.putExtra(Intent.EXTRA_EMAIL, new String[]{email.getText().toString()});
        startActivity(email_btn);
}
```

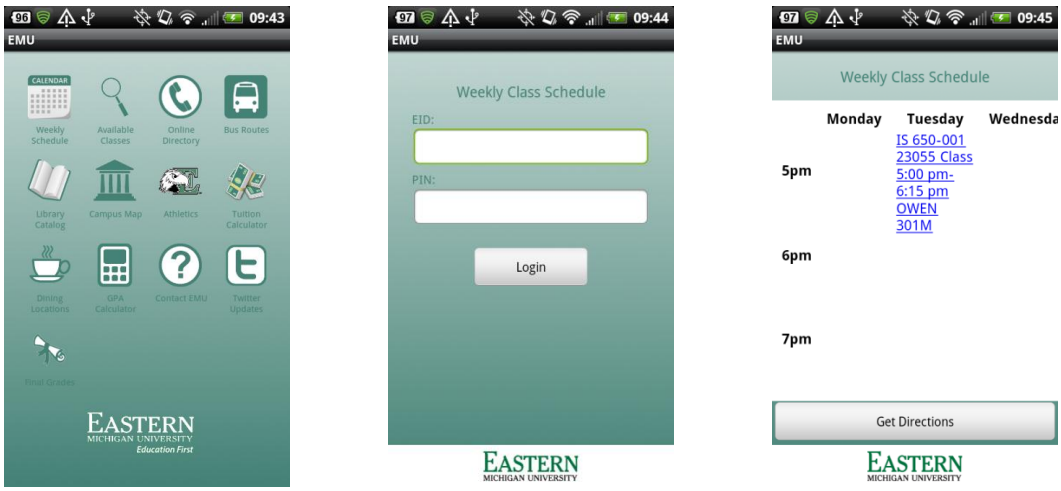
Online Directory uses directory_search.xml, directory_results.xml, and directory_person.xml for layouts and Directory_Search, Directory_Results, and Directory_Person as its Activities. The full code is provided in the Appendix G.

Weekly Schedule

Weekly Schedule provides the registered students with their weekly class schedule and directions to buildings where each class is held. Figure 20 presents the sequence of views during a successful login in this feature.

After clicking on the “Weekly Schedule” icon in Main Menu from Figure 20.a, the users are verified with the system in Figure 20.b, and upon a successful login, they can see their current weekly class schedule as shown in Figure 20.c. By clicking on the “View Directions” button, the application generates the list of the distinct buildings where student’s classes are held as presented in Figure 21.a.

Based on the user’s selection, the application provides the directions from his/her current location to the desired building by using the Google Maps Mobile Web solution. In case of an unsuccessful login, an error message is presented as shown in Figure 22 and the windows from Figures 20.c, 21.a and 21.b are unavailable.

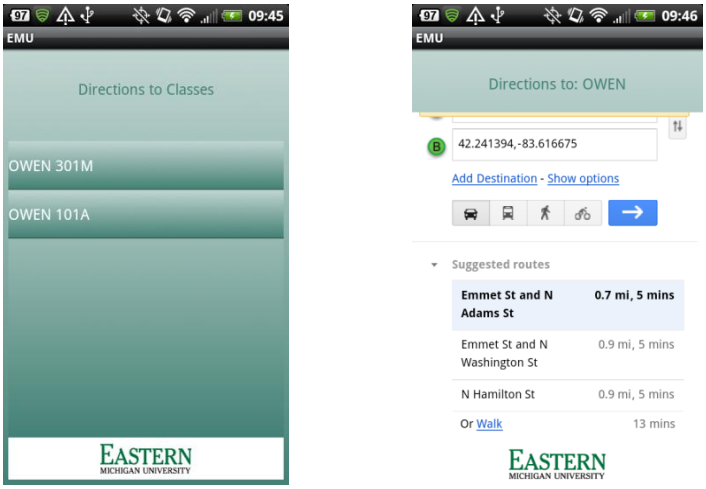


a. Main Menu b. Login c. Schedule

Figure 20. Successful Login for Weekly Schedule Screenshots

Weekly Schedule implements a secure HTTP request technique and embedding Web views. For information display, it uses the ListView and WebView objects. If the users are successfully logged in, they are not able to instantiate any clicks within the layout presented in Figure 20.c. This is done in order to prohibit the navigation outside of this window.

However, a user can click on a Map icon within a WebView to see the visual directions instead of reading them in a text format.



a. List of Classrooms b. Directions

Figure 21. Weekly Schedule List of Classrooms and Directions

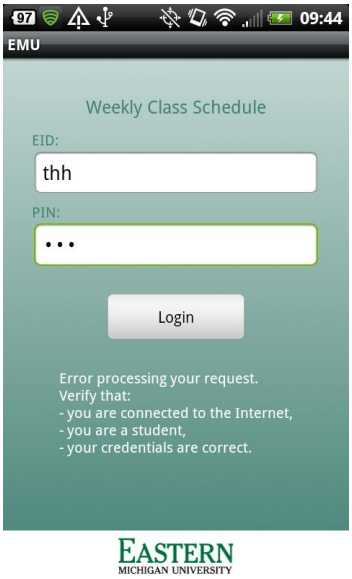


Figure 22. Unsuccessful Login for Weekly Schedule Screenshot

A secured HTTP request in Weekly Schedule feature is used like this:

```
private void parseCookie()
{
        cookie = connection.getHeaderField("Set-Cookie");
}
private void login(String userId, String password, String pass_url)  throws MalformedURLException,
IOException {
        url = new URL(pass_url);
        TrustManager[] trustAllCerts = new TrustManager[] { new X509TrustManager() {
        public java.security.cert.X509Certificate[] getAcceptedIssuers() {
           return null;
        }
        public void checkClientTrusted(
              java.security.cert.X509Certificate[] certs, String authType) {
        }
        public void checkServerTrusted(
              java.security.cert.X509Certificate[] certs, String authType) {
        } } };
        try {
                SSLContext sc = SSLContext.getInstance("SSL");
                sc.init(null, trustAllCerts, new java.security.SecureRandom());
                HttpsURLConnection.setDefaultSSLSocketFactory(sc.getSocketFactory());
        }
        catch (Exception e) {
        e.printStackTrace();
        }
        connection = (HttpsURLConnection) url.openConnection();
        connection.setRequestMethod("POST");
        connection.setDoInput(true);
        connection.setDoOutput(true);
        connection.setUseCaches(true);
        connection.setRequestProperty("Cookie", LOGIN_COOK);
        DataOutputStream out = new DataOutputStream(connection.getOutputStream());
        String content = "sid=" + URLEncoder.encode(userId, "UTF-8") + "&pin=" +
URLEncoder.encode(password, "UTF-8");
        out.writeBytes(content);
        out.flush();
        out.close();
        parseCookie();

        BufferedReader input = new BufferedReader(new InputStreamReader(
connection.getInputStream()));
          input.close();
}
```

The code snippet above performs the login function by following the instructions described in the "Secured Web Access" section of this thesis. This is used in the Schedule_Login Activity.

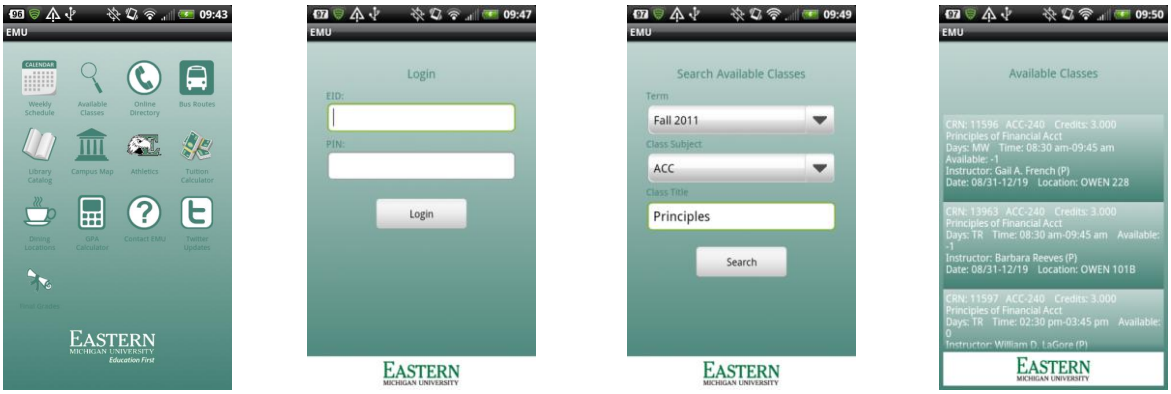
Another interesting concept used in this feature is the GPS location tracking which provides directions from the user's current location to the requested building where a class is held. The Schedule_Directions Activity makes use of an Android Location API for this. Vogel (2010c) explains that "you can register a 'LocationListener' with the 'LocationManager' and will receive periodic updates about the geoposition. The class 'LocationProvider' is the superclass of the different location providers which deliver the information about the current location." Here is the full code that accomplishes this:

```
LocationManager locationManager;
locationManager = (LocationManager) getSystemService(Context.LOCATION_SERVICE);
Criteria criteria = new Criteria();
provider = locationManager.getBestProvider(criteria, false);
Location location = locationManager.getLastKnownLocation(provider);
if (location != null) {
        double lat = (double) (location.getLatitude());
        double lng = (double) (location.getLongitude()); }
else {}
```

For its implementation, Weekly Schedule uses schedule_login_layout.xml, schedule_view_success.xml, schedule_locations.xml, schedule_locations_item.xml, schedule_directions.xml for XML layouts and Schedule_Login, Schedule_View, Schedule_Locations, and Schedule_Directions as its Activities. The full code for Weekly Schedule feature is provided in the Appendix H.

Course Lookup

Course Lookup provides a list of classes for a particular semester, as well as specific information regarding class CRN, class abbreviation, credits, class title, days and times the class is held, class availability, teacher, dates, and the class location. Figure 23 displays the screens that a user is presented with upon a successful login.



a. Main Menu b. Login c. Look up Courses d. Available Courses

Figure 23. Course Lookup Screenshots

After a user clicks on the "Available Classes" icon, as shown in Figure 23.a, he/she needs to log into the system through a screen presented in Figure 23.b. If a login is completed successfully, the search fields are displayed as shown in Figure 23.c. When a user clicks on "Search," the feature executes a request and displays the results as shown in Figure 23.d. Figure 24 shows an error message due to an unsuccessful login.

Course Lookup executes a secured HTTP request and obtains the login in the similar fashion as Weekly Schedule feature. One method that has not been described in practice in the previous sections is how Spinner objects are set up:

```
class_subject = (Spinner) findViewById(R.id.classes_search_subject);
```

```
class_subject.setOnItemSelectedListener(this);
class_term = (Spinner) findViewById(R.id.classes_search_term);
class_term.setOnItemSelectedListener(this);
ArrayAdapter<String> srch_term = new ArrayAdapter<String>(this,
android.R.layout.simple_spinner_item, term_strings);
ArrayAdapter<String> srch_subj = new ArrayAdapter<String>(this,
android.R.layout.simple_spinner_item, class_subjects);
srch_term.setDropDownViewResource(android.R.layout.simple_spinner_dropdown_item);
srch_subj.setDropDownViewResource(android.R.layout.simple_spinner_dropdown_item);
class_term.setAdapter(srch_term);
class_subject.setAdapter(srch_subj);
```

In the example above, predefined string arrays “term_strings” and “class_subjects” are adapted to Spinner objects. For its implementation, Course Lookup uses classes_login.xml, classes_search.xml, classes_view.xml, and classes_item.xml for XML layouts and Classes_Login, Classes_Search, and Classes_View as its Activities. The full code for Course Lookup is provided in the Appendix I.

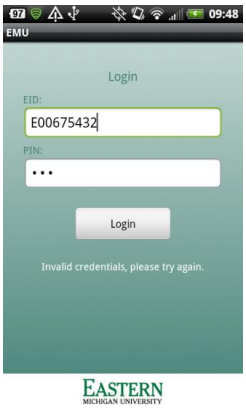
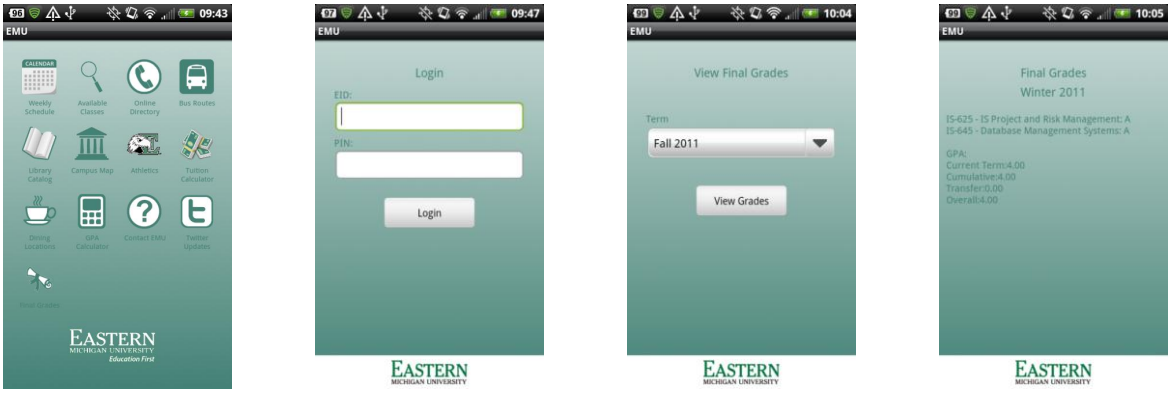


Figure 24. Unsuccessful Course Lookup and Final Grades Login

Final Grades

Final Grades displays the student’s final grades for a particular semester. This feature provides a list of available terms and upon selection displays the final semester grades per class along with the current, cummulative, transfer, and overall grade point averages. For

login, Final Grades shares the same classes_login.xml XML layout and Classes_Login Activity as Course Lookup, while the other views and classes are unique. Figure 25 displays the sequence of events following a successful login into the system.



a. Main Menu b. Login c. Available Terms d. Grades Display

Figure 25. Final Grades Screenshots

After a user clicks on Final Grades icon in Figure 25.a, he/she needs to log into the system through the screen presented in Figure 25.b. If a login is completed successfully, the search fields are displayed as shown in Figure 25.c. When a user clicks on "View Grades," the feature executes a request and displays the results as shown in Figure 25.d. Figure 24 shows an error message due to an unsuccessful login.

Final Grades is very similar in functionality as Course Lookup. For a login, this feature utilizes classes_login.xml for its XML layout and Classes_Login as its Activity, which are available in the Appendix I. For the rest of its implementation, it uses classes_terms.xml and classes_grades.xml for XML layouts and Classes_Terms and Classes_Grades for its Activities, which are all provided in the Appendix J.

Twitter Updates

Twitter Updates provide "tweets" from EMU-affiliated accounts. Figure 26 displays the functionality of this feature.

a. Main Menu b. Categories c. "Tweets"

Figure 26. Twitter Updates Screenshots

After the users click on the "Twitter Updates" icon in Figure 26.a, they need to select a category from Figure 26.b to retreive the specific "tweets" as displayed in Figure 26.c.

By following the "Accessing APIs" procedure in this thesis, the information about "tweets" is stored in the individual objects. Here is the method implemented that executes an HTTP request that completes this task:

```
public ArrayList<Tweet> getTweets(String url) {
        String searchUrl = url;
        ArrayList<Tweet> tweets = new ArrayList<Tweet>();
        try {
                HttpClient hc = new DefaultHttpClient();
                HttpGet get = new HttpGet();
                get.setURI(new URI(searchUrl));
                HttpResponse rp = hc.execute(get);
                 if(rp.getStatusLine().getStatusCode() == HttpStatus.SC_OK)
```

```
                {
                        String result = EntityUtils.toString(rp.getEntity());
                        JSONObject root = new JSONObject(result);
                        JSONArray sessions = root.getJSONArray("results");
                        for (int i = 0; i < sessions.length(); i++) {
                                JSONObject session = sessions.getJSONObject(i);
                                Tweet tweet = new Tweet(session.getString("from_user"),
                                                        session.getString("text"),
                                                        session.getString("profile_image_url"),
                                                        session.getString("created_at"));
                                tweets.add(tweet);
                        }
                }
        }
        catch (Exception e) {
                e.printStackTrace();
        }
        return tweets;
}
```

The code above uses a "url" parameter which is supposed to retrieve the "tweets" from the specific accounts. Notice that a *Tweet* object "tweet" uses the information about a Twitter profile image url for its constructor. By doing so, a Tweet object is able to store a web location of an account's profile image.

If it is desired to combine the "tweets" with account profile images for display, certain optimization issues are encountered. Cois (2011) describes caching of the bitmap images from the web in order to make a ListView perform quicker. The complete code for Twitter Updates is provided in the Appendix K.

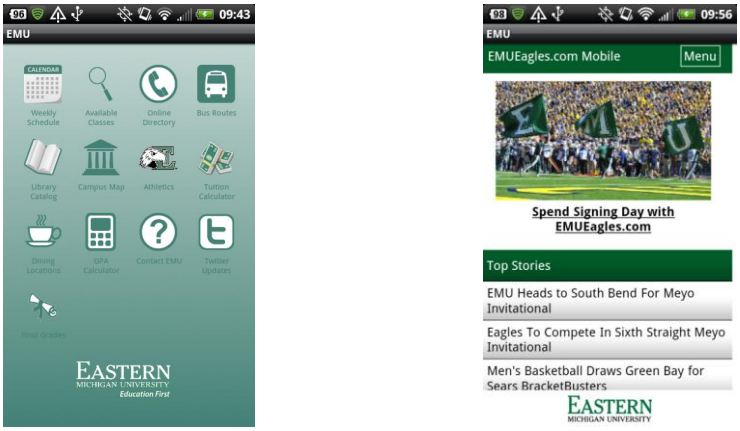
If presenting the account profile images is unnecessary, it is sufficient to use twitter_options.xml, twitter_feed.xml, and twitter_tweet.xml (without ImageView object) for XML layouts, Twitter_Options and Twitter_Feed as its Activities, and Tweet as a custom class. A Tweet class contains *twitterHumanFriendlyDate( )* method which Jaglale (2010)

provides on his website. The Twitter_Feed and Tweet classes should be modified accordingly to remove any image references or uses of non existing classes.

However, if the account profile images are desired for presentation, the full code from the Appendix J should be used. Along with the already mentioned XML files and classes, the custom Twitter_Image_Manager and Twitter_Adapter classes should be implemented as well (Cois, 2011).

Athletics News

Athletics News provides news about the EMU Athletics, rosters, and schedules. The users are allowed to browse through the most recent stories uploaded on the mobile website and explore the interactive menu with all EMU Atletics teams listed. Figure 27 shows a typical user interaction with this feature.



a. Main Menu b. Mobile Site

Figure 27. Athletics News Screenshot

When a user clicks on the "Athletics" icon from Main Menu in Figure 27.a, he/she will be able to access the EMU Athletics mobile site as shown in Figure 27.b. For its

implementation, Athletics News is embedding Web views in its layout by following the procedure from “Embedding Web views” section of this thesis:

```
page = (WebView) findViewById(R.id.athletics_view);
page.setWebViewClient(new WebViewClient());
page.getSettings().setJavaScriptEnabled(true);
page.loadUrl("http://emueagles.com/mobile/?");
```

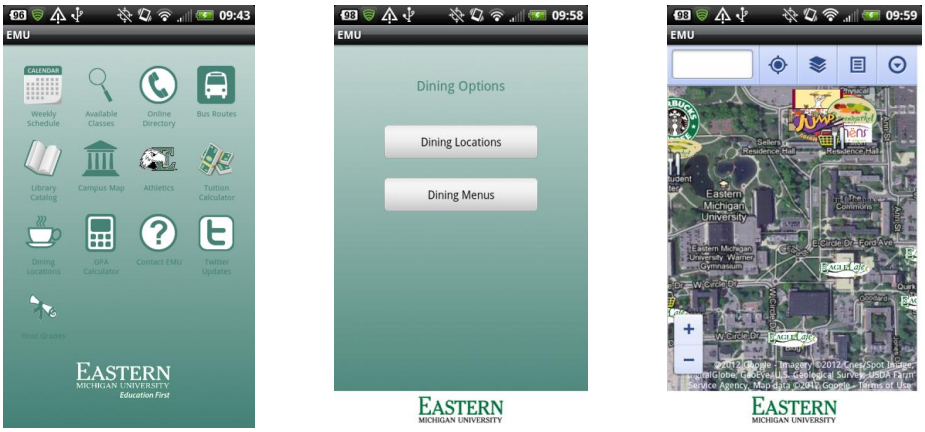
In the code above, JavaScript is enabled, which allows the navigation through the mobile site. Athletics News uses athletics.xml for XML layout and Athletics as its Activity. The full code is provided in Appendix L.

Dining Locations

Dining Locations provides the map of the dining locations on campus and displays menus for each of them. For both of its functionalities, it embeds the web views from the online resources. Figures 28 and 29 present the possible scenarios if a user takes advantage of this feature.

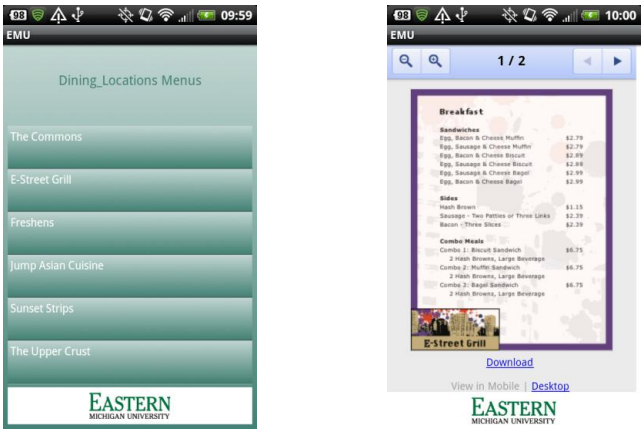
When a user clicks on the “Dining Locations” icon in Figure 28.a, he/she is presented with two options as shown in Figure 28.b. After selecting “Dining Locations” option, a user is presented with the dining locations map as given in Figure 28.c. If a user selects “Dining Menus” in Figure 28.b, he/she will be presented with a list of available dining locations as shown in Figure 29.a. By clicking on an individual item, a specific menu for that location is displayed as provided in Figure 29.b.

Dining Locations uses dining_main.xml, dining_menus.xml, dining_item.xml, and dining_view.xml for XML layouts and Dining_Main, Dining_Menus, and Dining_View as its Activities. The complete code for this feature is attached in the Appendix M.



a. Main Menu b. Options c. Locations

Figure 28. Dining Locations Screenshots



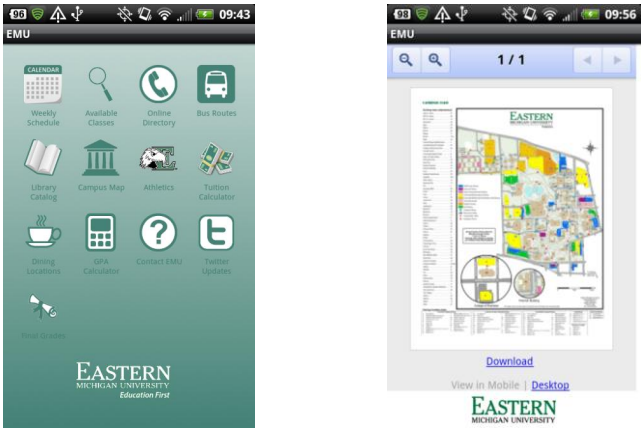
a. Dining Menus b. Store Menu

Figure 29. Dining Menus Screenshots

Campus Map

Campus Map is a simple feature that embeds the view of the EMU campus map. Figure 30 illustrates the Campus Map functionality. After a user clicks on the “Campus Map” icon in Figure 30.a, he/she is presented with a Campus Map view as shown in Figure 30.b.

Campus Map uses map.xml for XML layout and Map as its Activity. The complete code for Campus Map is provided in the Appendix N.



a. Main Menu b. Map

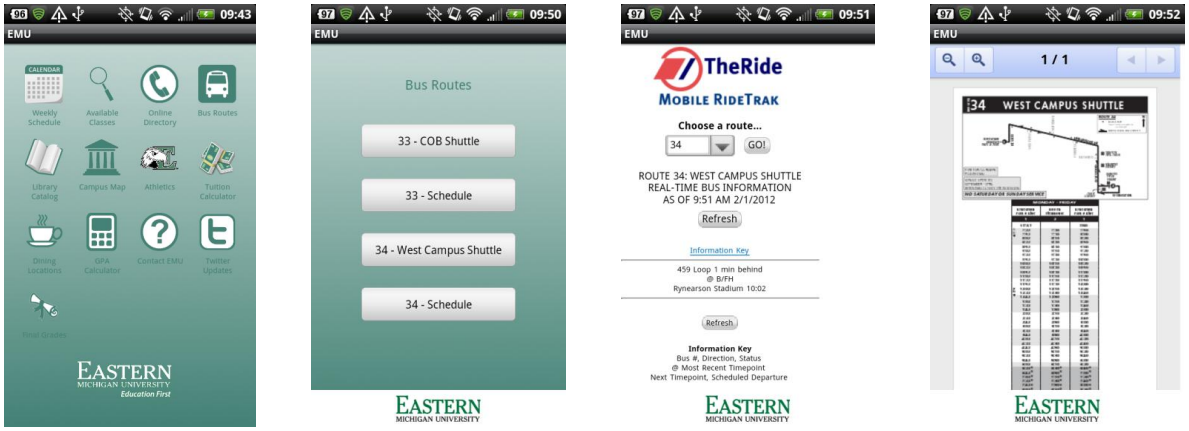
Figure 30. Campus Map Screenshots

Bus Schedule

Bus Schedule gives the current locations of buses operating on EMU campus, as well as their daily schedules and hours of operation. Figure 31 shows a typical interaction with the Bus Schedule feature.

If a user clicks on the "Bus Routes" icon in Figure 31.a, he/she is presented with a menu as given in Figure 31.b. After selecting "CC – COB Shuttle" or "34 – West Campus Shuttle," this feature displays the current location of a chosen route as illustrated in Figure 31.c. If a user selects "33 – Schedule" or "34 – Schedule" options, Bus Schedule displays a daily schedule for a corresponding route as given in Figure 31.d.

For its implementation, Bus Schedule embeds Web views. It uses bus_menu.xml and bus_display.xml for XML layouts and Bus_Menu and Bus_Display as its Activities. The complete code for this feature is attached in the Appendix O.



a. Main Menu b. Menu c. Current Location d. Schedule

Figure 31. Bus Schedule Screenshots

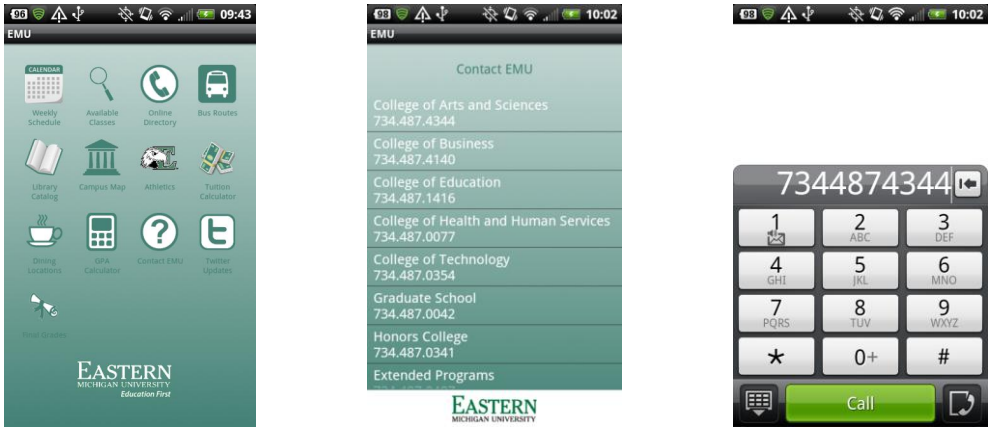
Contact EMU

Contact EMU displays a quick reference list with the phone numbers of offices that can be contacted. Figure 32 illustrates the Contact EMU functionality.

When a user clicks on the "Contact EMU" icon on the screen from Figure 32.a, he/she can see the phone numbers associated with the EMU offices as given in Figure 32.b. By clicking on a particular item, the dialer application is started as presented in Figure 32.c. Guidry (n.d.) introduces the following code for initiating the dialer application:

```
String numberToDial = "tel:"+number;
startActivity(new Intent(Intent.ACTION_DIAL, Uri.parse(numberToDial)));
```

By specifying the Intent action, “Android Dialer” Activity is selected to start with a predetermined parameter. Contact EMU uses contact.xml for XML layout and Contact as its Activity. The full code is provided in the Appendix P.



a. Main Menu b. List of contacts c. Dialer

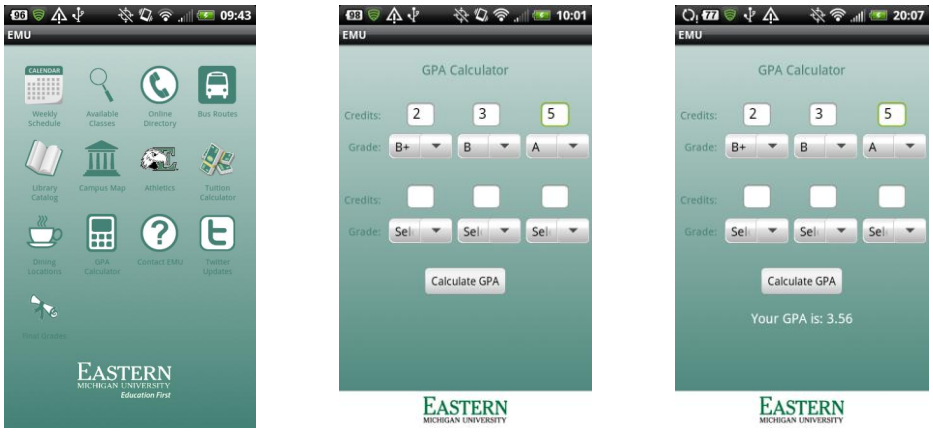
Figure 32. Contact EMU Screenshots

GPA Calculator

GPA Calculator allows a user to calculate his/her grade point average based on the grades received and the number of credit hours. Figure 33 presents a user’s interaction with the GPA Calculator feature.

When a user clicks on the “GPA Calculator” icon on the screen displayed in Figure 33.a, he/she can enter the number of credits for 6 or less different classes and the respective grades received as shown in Figure 33.b. After entering the inputs and clicking on the “Calculate GPA” button, the grade point average is displayed as shown in Figure 33.c.

GPA Calculator uses gpa_calculator.xml for XML Layout and GPA_Calculator as its Activity. The complete code for this feature is provided in the Appendix Q.



a. Main Menu b. Inputs for GPA c. GPA

Figure 33. GPA Calculator Screenshots

Chapter 6: Conclusion

This thesis describes a development process for the Android Mobile EMU Portal application. It begins with the definition of the intended features for this application and continues with the descriptions of the fundamental Android development concepts. Based on the analysis of those features, the development techniques are derived by grouping the mutual similarities and the common data access points. This thesis groups the development techniques in five separate categories that are classified based on methods they use for the information access:

- Unsecured Web access
- Secured Web access
- Accessing APIs
- Embedding Web views
- Programmable Components

This thesis provides step-by-step instructions on how to implement each of the techniques tested on an actual Android device. The implementation of these techniques is illustrated through the Android Mobile EMU Portal and described in detail with the complete code. Therefore, this thesis results in a successful definition of the instructions for the development techniques and a functional Android Mobile EMU Portal application.

Thesis Contributions

The main contribution of this thesis is the methodology that can be used to determine an applicable Android development technique, based on the required data source and a connection type of a desired feature.

This thesis contributes to the Android development knowledge literature by providing a distinct approach due to its single focus on EMU data. For each development technique, various implementation approaches are considered using the available resources, leading to a creation of step-by-step instructions for each of the techniques.

Android Mobile EMU Portal presents a unique Android app solution. This application provides a working prototype of a "mobile portal" that incorporates certain features currently not available in existing EMU Android applications.

Thesis Limitations

The research for this thesis has been limited mainly with three constraints. The first constraint involves the inability to directly access Banner and Twitter databases. The information from these databases is stored behind the secured systems and access to them is not publicly allowed. In order to access them, acquisition of additional software is required, permissions need to be granted, and other security issues occur that do not make this feasible. Accessing data through Banner Self-Services and Twitter API causes the application to work slower than its optimized version could.

Due to same security issues, Android Mobile EMU Portal is not able to "write" information into the Banner database. This is why users can only view the information behind the firewall regarding classes, grades, and schedules, but cannot register for courses. Therefore, the logical choice was to use the Banner Self-Services portal.

The third constraint involves the availability of literature and time resources. Accessing secure information from the university's ERP system is a confidential topic, and

this is why this thesis does not include any reference to the topic. This thesis is also bound by the available timeframe for completion which imposes a limitation on the research time.

Future Opportunities for Research

This thesis presents useful techniques for accessing EMU-specific data by tackling a variety of information to produce complete and diverse Android development instructions. However, other approaches can be considered.

First, thesis can focus on a single data source. By attempting to access the information from the Banner ERP Suite, the research would start from a data access source and then produce the specific features. Topics to focus on would involve the security and integration issues, as well as data presentation.

Second, the research can be more optimization driven. From the Twitter Updates example, it is clear that downloading images causes delays and slow performance. Besides this, executing a Library Catalog search takes time to execute HTTP requests. This research would attempt to assess these issues and derive performance boosting techniques for the desired features.

Third, more time can be allocated for conducting meetings with the responsible professionals on campus and gathering information for logic implementation behind services. This would combine the efforts from multiple individuals and would potentially encourage better project management. With more people involved in this project, other possible ideas may occur over time that could direct the research in a better direction.

Final Remarks

Step-by-step instructions of the development techniques described in this thesis are applied to the EMU information. Android Mobile EMU Portal provides a foundation to build up from by including more EMU-related data. Application performance optimization and additional features can be considered to improve its quality. The features that should be included can be further investigated and could lead to producing an EMU Android application appealing to the EMU student community. Additional new features may lead into discovery of new programming techniques as well.

References

Ableson, W.F., Sen, R. and King, C. (2011). *Android in Action*. Stamford, CT. Manning Publications Co.

Activitiy. (2012). Retrieved January 28, 2012 from http://developer.android.com/reference/android/app/Activity.html

Anwar, W. (2011, August 10). "Binding Android ListView with String Array using ArrayAdapter". Retrieved from http://www.ezzylearning.com/tutorial.aspx?tid=1659127&q=binding-android-listview-with-string-array-using-arrayadapter

Bishop, T. (2012, March 6). "Google Play replaces Android Market, cosolidates Google's media marketplaces". Retrieved from http://www.geekwire.com/2012/google-play-replaces-android-market-consolidates-googles-media-marketplaces

Bradby, D. (2011, February 22) "Loading Twitter Data into Android with Lists". Retrieved from http://www.sitepoint.com/loading-twitter-data-into-android-with-lists/

Building Web Apps in WebView. (2012). Retrieved February 3, 2012 from http://developer.android.com/guide/webapps/webview.html

Cois, A. (2011, June 15). "Android Development Tutorial: Asynchronous Lazy Loading and Caching of ListView". Retrieved from http://codehenge.net/blog/2011/06/android-development-tutorial-asynchronous-lazy-loading-and-caching-of-listview-images

Dowling College. (2005)."ScheduleParser.java". Retrieved from http://arcib.dowling.edu/~csc4175s5b/source/ScheduleParser.java

Eddy, N. (2011, December 8). “Cloud Computing: Cloud, Mobile Apps, Public Storage Are Top IT Trends for 2012 and Beyond: Gartner”. Retrieved from http://www.eweek.com/c/a/Cloud-Computing/Cloud-Mobile-Apps-Public-Storage-Are-Top-IT-Trends-for-2012-and-Beyond-Gartner-130060/

Felker, D. (2011). *Android Application Development For Dummies*. Hoboken, NJ. Wiley Publishing, Inc.

Google (n.d.). “Android – Discover Android”. Retrieved March 10, 2012 from http://www.android.com/about/

Guidry, M. (n.d.). “Simple Dialer Application”. Retrieved December 23, 2011 from http://eagle.phys.utk.edu/guidry/android/simpleDialer.html

Houston, P. (2012, February 4). “Android XML Adventure – Parsing HTML using Jsoup”. Retreived from http://xjaphx.wordpress.com/2012/02/04/android-xml-adventure-parsing-html-using-jsoup/

Installing the SDK. (n.d.). Retrieved October 28, 2011 from http://developer.android.com/sdk/installing.html

Jackson, W. (2011). *Android Apps for Absolute Beginners*. New York, NY. Apress.

Jaglale, J. (2010). ” Convert Twitter date to human friendly string”. Retrieved from http://maestric.com/doc/java/twitter

Jordan, L. and Greyling, P. (2011). *Practical Android Projects*. New York, NY. Apress.

Klumpp, S. (2011, March 11). “Android: new Intent() starts new instance with android:launchMode=’sinleTop’”. Message posted to http://stackoverflow.com/questions/2424488/android-new-intent-starts-new-instance-with-androidlaunchmode-singletop

ListView Backgrounds: An Optimization. (n.d.). Retrieved February 10, 2012 from http://developer.android.com/resources/articles/listview-backgrounds.html

Mayani, P. (2010, December 24). “Android – Load PDF / PDF Viewer”. Message posted to http://stackoverflow.com/questions/4468621/android-load-pdf-pdf-viewer/4525717#4525717

Meier, R. (2010). *Professional Android 2 Application Development*. Indianapolis, IN. Wiley Publishing, Inc.

Murphy, M. (2009). *Beginning Android*. New York, NY. Apress.

Murphy, M. (2010). *Beginning Android 2*. New York, NY. Apress.

Murphy, M. (2011). *Beginning Android 3*. New York, NY. Apress.

Nuetzmann, M. (2009, February 2). “Extending the WebView”. Message posted to http://groups.google.com/group/android-developers/browse_thread/thread/f14e899b953f333f#

Oracle (2010a). “DataOutputStream”. Retrieved February 2, 2012 from http://docs.oracle.com/javase/1.4.2/docs/api/java/io/DataOutputStream.html

Oracle (2010b). “InputStreamReader”. Retrieved February 2, 2012 from http://docs.oracle.com/javase/1.4.2/docs/api/java/io/InputStreamReader.html

Oracle (2011a). “Oracle Database Mobile Server 11g”. Retrieved from http://www.oracle.com/technetwork/database/database-mobile-server/dms-11g-datasheet-512117.pdf

Oracle. (2011b). “Oracle Mobile Server Documentation”. Retrieved from http://docs.oracle.com/cd/E22663_01/doc.11100/e22681.pdf

Start Email-Activity with Preset Data (via Intents). (2009). Retrieved December 23, 2011 from http://www.androidsnippets.com/start-email-activity-with-preset-data-via-intents

Steele, J. (2011). *The Android Developer's Cookbook*. Boston, MA. Pearson Education, Inc.

The Apache Software Foundation. (n.d.). “Chapter 2. Connection management”. Retrieved from http://hc.apache.org/httpcomponents-client-ga/tutorial/html/connmgmt.html

Twitter. (2012, February 1). “GET search”. Retrieved February 13, 2012 from https://dev.twitter.com/docs/api/1/get/search

Using Hardware Devices. (2012). Retrieved January 26, 2012 from http://developer.android.com/guide/developing/device.html

Van Emery, S. (2009). *Pro Android Media: Developing Graphics, Music, Video and RichMedia Apps for Smartphones and Tablets*. New York, NY. Apress

Vogel, L. (2008). “Apache HttpClient - Tutorial”. Retrieved November 21, 2011 from http://www.vogella.de/articles/ApacheHttpClient/article.html

Vogel, L. (2010a). “Android HTTP Access - Tutorial”. Retrieved November 14, 2011 from http://www.vogella.de/articles/AndroidNetworking/article.html

Vogel, L. (2010b). “Android Threads, Handlers and AsyncTask - Tutorial”. Retrieved January 10, 2012 from http://www.vogella.de/articles/AndroidPerformance/article.html

Vogel, L. (2010c). “Location API and Google Maps in Android - Tutorial”. Retrieved January 18, 2012 from http://www.vogella.de/articles/AndroidLocationAPI/article.html

What is Android? (2012). Retrieved March 10, 2012 from http://developer.android.com/guide/basics/what-is-android.html

XmlPullParser. (2012). Retrieved January 10, 2012 from http://developer.android.com/reference/org/xmlpull/v1/XmlPullParser.html

Zechner, M. (2011). *Beginning Android Games*. New York, NY. Apress.

Appendix A: ScheduleParser.txt

```
/**
*THIS CLASS PARSES THE STUDENT'S SCHEDULE FROM BANNER
*You should call getSchedule(String username, String password) from the jsp page.
*http://arcib.dowling.edu/~csc4175s5b/source/ScheduleParserBackup.java
*/
//Uncomment on deployment
//package admin;
import java.util.*;
import java.text.*;
import java.net.*;
import java.io.*;
import javax.net.ssl.*;
//For regex
import java.util.regex.*;
//For SSL
import com.sun.net.ssl.internal.www.protocol.https.*;
import java.security.Security;
public class ScheduleParser {
	private static final String LOGIN_ADDR =
"https://bannerweb.dowling.edu/pls/PROD/twbkwbis.P_WWWLogin";
	private static final String VALDN_ADDR =
"https://bannerweb.dowling.edu/pls/PROD/twbkwbis.P_ValLogin";
	private static final String TRANS_ADDR =
"https://bannerweb.dowling.edu/pls/PROD/bwskotrn.P_ViewTran";
	private static final String SCHED_ADDR =
"https://bannerweb.dowling.edu/pls/PROD/bwskfshd.P_CrseSchdDetl";

	private static final String LOGIN_COOK = "SESSID=; TESTID=set";
	private static final String TRANS_POST = "levl=&tprt=STDT";
	private static final String SCHED_POST = "term_in=200502"; //Wtr/Spr 2005 (View Only)
	private LinkedList events; //These are the events that have to be added to the db
	private URL url;
	private HttpsURLConnection connection;
	private String cookie;
	private LinkedList rawData; //raw schedule in HTML as obtained from bannerweb
	private void parseCookie()
	{
		cookie = connection.getHeaderField("Set-Cookie");
	}
	//This method is called by getSchedule()
	private void login(String userId, String password)  throws MalformedURLException,
IOException {
		url = new URL(VALDN_ADDR);
		connection = (HttpsURLConnection) url.openConnection();
```

```
                connection.setRequestMethod("POST");
                connection.setDoInput(true);
                connection.setDoOutput(true); // true for POST, false for GET
                connection.setUseCaches(true);
                connection.setRequestProperty("Cookie", LOGIN_COOK);
                DataOutputStream out = new DataOutputStream(connection.getOutputStream());
                String content = "sid=" + URLEncoder.encode(userId, "UTF-8") + "&pin=" +
URLEncoder.encode(password, "UTF-8");
                out.writeBytes(content);
                out.flush();
                out.close();
                parseCookie();
                BufferedReader input = new BufferedReader(new InputStreamReader(
connection.getInputStream()));
                String line = "";
                while ((line = input.readLine()) != null)
                {
                        //System.out.println(line);
                }
                input.close();
        }
        //Call after you've called login()!
        public String getSchedule(String username, String password) throws IOException,
MalformedURLException {
                login(username, password);
                url = new URL(SCHED_ADDR);
                connection = (HttpsURLConnection) url.openConnection();
                connection.setRequestMethod("POST");
                connection.setDoInput(true);
                connection.setDoOutput(true);
                connection.setUseCaches(true);
                connection.setRequestProperty("Cookie", cookie);
                connection.setRequestProperty("Content-Type", "text/html");
                DataOutputStream out = new DataOutputStream(connection.getOutputStream());
                out.writeBytes(SCHED_POST);
                out.flush();
                out.close();
                BufferedReader input = new BufferedReader(new InputStreamReader(
connection.getInputStream()));
            String alllines = "";
                String line = "";
                //Put the lines of html code into the rawData list
                rawData = new LinkedList();
                while ((line = input.readLine()) != null)
                {
                        rawData.add(line);
                        alllines += line + "\n";
```

```
                        System.out.println(line);
                }
                input.close();
                return alllines;
        }
        private void parseSchedule() {
                Pattern pattern = Pattern.compile("This layout table is used to present the schedule
course detail");;
                Matcher matcher;
                System.out.println(rawData.size());
                events = new LinkedList();
                while(rawData.size() > 0) {
                        matcher = pattern.matcher((String)rawData.getFirst());
                        if(matcher.find()) { //We found the beginning of a course description. Parse it.
                                //System.out.println("Found beginning of a course");
                                parseCourse();
                                //rawData.removeFirst();
                        }
                        else {
                                rawData.removeFirst();
                        }
                }
        }
        private static String PTRN_COURSE = "(This layout table is used to present the schedule
course detail\"><CAPTION class=\"captiontext\">)(.*)(</CAPTION>)";
        private static String PTRN_MEETING = "This table lists the scheduled meeting times and
assigned instructors for this class";
        //parse course parses the info and adds it into the database
        private void parseCourse() {
                String courseName = "";
                //System.out.println(rawData.getFirst());
                Pattern pattern = Pattern.compile(PTRN_COURSE);
                Matcher matcher = pattern.matcher((String)rawData.getFirst());
                if(matcher.find()) {
                        System.out.println(matcher.group(2)); //This is the course name
**********
                        courseName = matcher.group(2);
                        //remove the course name and the remaining
                        rawData.removeFirst();
                        //pattern = Pattern.compile("This table lists the scheduled meeting times and
assigned instructors for this class");
                        pattern = Pattern.compile(PTRN_MEETING);
                        matcher = pattern.matcher((String)rawData.getFirst());
                        while(!matcher.find()) {
                                matcher = pattern.matcher((String)rawData.getFirst());
                                rawData.removeFirst();
                        }
```

```
                        //System.out.println("Found beginning of meeting times desc");
                        rawData.removeFirst(); //remove beginning of meeting times line
                for(int i=0; i<9; i++) { //remove 9 lines of useless data. these are the upper row of the
meeting times table
                                rawData.removeFirst();
                        }
                        //System.out.println(rawData.getFirst() + "\n");
                        boolean intable = true;
                        //This should match the meeting times
                        //Each iteration will spawn a new Event
                        while(intable) {
                                Event curevent = new Event(); //At the end of the while loop we add
curevent to the events list
                                curevent.setName(courseName);
                                rawData.removeFirst();

                                pattern = Pattern.compile("(<TD CLASS=\"dddefault\">)(.*)(</TD>)");
                                matcher = pattern.matcher((String)rawData.getFirst());
                                if(matcher.find()) {
                                        System.out.println( "These are the meegin dates: " +
matcher.group(2)); //These are the meeting time range ********************
                                        rawData.removeFirst();
                                        String unfTime = matcher.group(2);
                                        pattern = Pattern.compile("(.*)( - )(.*)");
                                        matcher = pattern.matcher(unfTime);
                                        if(matcher.find()) {
                                                //System.out.println(matcher.group(1) + " " +
matcher.group(3)); //This is the extracted time range. Sweeeet.
                                                Date date = new Date("12 Aug 1995 " +
matcher.group(1));
                                                //System.out.println(date.getMinutes());
                                                //System.out.println(Integer.toString(date.getHours())
+ ":" + Integer.toString(date.getMinutes()));

        curevent.setStartTime(Integer.toString(date.getHours()) + ":" +
Integer.toString(date.getMinutes()) + ":00");
                                        }
                                        else {
                                                System.out.println("Could not parse meeting times.
However, i extracted them.");
                                        }
                                }
                                else {
                                        System.out.println("ERROR. Could not get course time. The
string was " + rawData.getFirst());
                                        rawData.removeFirst();
                                }
```

```
                                        //This should match the meeting days
                                        pattern = Pattern.compile("(<TD CLASS=\"dddefault\">)(.*)(</TD>)");
                                        matcher = pattern.matcher((String)rawData.getFirst());
                                        if(matcher.find()) {
                                                System.out.println(matcher.group(2)); //These are the
meeting days ********************
                                                rawData.removeFirst();
                                        }
                                        else {
                                                System.out.println("ERROR. Could not get meeting days. The
string was " + rawData.getFirst());
                                                rawData.removeFirst();
                                        }
                                        //This should match the meeting location
                                        pattern = Pattern.compile("(<TD CLASS=\"dddefault\">)(.*)(</TD>)");
                                        matcher = pattern.matcher((String)rawData.getFirst());
                                        if(matcher.find()) {
                                                System.out.println(matcher.group(2)); //This is the meeting
location ********************
                                                rawData.removeFirst();
                                                curevent.setLocation(matcher.group(2));
                                        }
                                        else {
                                                System.out.println("ERROR. Could not get meeting location.
The string was " + rawData.getFirst());
                                                rawData.removeFirst();
                                        }
                                        //This should match the meeting date period
                                        pattern = Pattern.compile("(<TD CLASS=\"dddefault\">)(.*)(</TD>)");
                                        matcher = pattern.matcher((String)rawData.getFirst());
                                        if(matcher.find()) {
                                                System.out.println(matcher.group(2)); //This is the meeting
date period ********************

                                                String unfDates = matcher.group(2);
                                                pattern = Pattern.compile("(.*)( - )(.*)");
                                                matcher = pattern.matcher(unfDates);
                                                if(matcher.find()) {
                                                        //NOTE THIS DOES NOT ADD END DATE
                                                        System.out.println(matcher.group(1) + " " +
matcher.group(3)); //These are the extracted dates. Sweeeet.
                                                        DateFormat df = new SimpleDateFormat("MMM dd,
yyyy");
                                                        try {
                                                                Date date =
(Date)df.parse(matcher.group(1));
                                                                //Date date = (Date)df.parse("Feb 15, 2005");
```

```
                                                //curevent.setDate(date.toString());
                                                //System.out.println(date.toString());
                                        //System.out.println(Integer.toString(date.getYear()+1900) + "-
" + Integer.toString(date.getMonth()+1) + "-" + Integer.toString(date.getDate()));

                                                /*NOTE that you have to add 1900 to the
year to get the true year (say, 2005, not 105)
                                                and you have to add 1 to the month, since
jan is 0*/
                                                curevent.setDate((String)
Integer.toString(date.getYear()+1900) + "-" + Integer.toString(date.getMonth()+1) + "-" +
Integer.toString(date.getDate()));
                                                //System.out.println("HAHAHA: " +
(date.getYear()+1900));
                                                //System.out.println("HAHAHA: " +
(date.getMonth()+1));
                                                //System.out.println("HAHAHA: " +
(date.getDate()));
                                                //curevent.setMonth(date.getMonth());
                                                //curevent.setDay(date.getDay());
                                                //curevent.setYear(date.getYear());
                                        }
                                        catch(ParseException e) {
                                                System.out.println("Could not parse date to
add to curevent");
                                        }
                                }
                                else {
                                        System.out.println("Could not parse meeting date
period. However, i extracted them.");
                                }

                                        rawData.removeFirst();
                        }
                        else {
                                System.out.println("ERROR. Could not get meeting date
period. The string was " + rawData.getFirst());
                                rawData.removeFirst();
                        }
                        //remove the type of meeting and instructor name
                        for(int i=0; i<3; i++) {
                                rawData.removeFirst();
                        }
                        //System.out.println(rawData.getFirst());
                        pattern = Pattern.compile("</TABLE>");
                        matcher = pattern.matcher((String)rawData.getFirst()); //see if this is
</TABLE> and we have no more times to read
```

```
                                        if(matcher.find()) {
                                                //System.out.println(matcher.group(2)); //This is the meeting
date period ********************
                                                //System.out.println("Exiting table...");
                                                intable = false;
                                                rawData.removeFirst();
                                        }
                                        else {
                                                //System.out.println("More meeting times to read...\n");
                                                rawData.removeFirst();
                                        }
                                        //Adds an event
                                        events.add(curevent);
                                }//END OF while(intable)
                                System.out.println("");
                        }
                        else {
                                System.out.println("Something went wrong. Could not get course title");
                                rawData.removeFirst();
                        }
                        System.out.println("Number of events to add: " + events.size());
                }
                /*public static void main( String argv[] ) throws IOException {
                        ScheduleParser test = new ScheduleParser();
                        test.getSchedule("litc1383", "5557605");
                        test.parseSchedule();
                }*/
}
```

Appendix B: Script Injection FAQ

1. What vulnerability was found?

It was discovered that a Banner Self-Service user could use standard browser features to copy a Banner Self-Service generated HTML page. The data values in this copied page could be modified, the page could be saved and hosted on a malicious "hacker" website and could then be used to update records.

2. How could this be used to maliciously update information?

A legitimate user would need to be logged into Banner Self-Service. The user would need to be enticed by a hacker via email, IM, etc. to click on a malicious link while they were still logged into Banner Self-Service. The link would execute the saved HTML page described above, updating information with the altered values.

3. What data could be changed?

Only data that the legitimate user has authorized access to change could be updated. The more complex the page in self-service, the more difficult it would be to script all the actions necessary to compromise the page.

4. What pages in Banner Self-Service could be used to do this?

Any page in Banner Self-Service could be compromised in this way. As the legitimate user would need to initiate the first step, only fairly simple pages would be easy to compromise.

5. What would the user see in their browser if they clicked this link?

Any update would generate the normal acknowledgement from Banner Self-Service. However, the hacker could quickly clear the acknowledgement page so the user may not see it.

6. Why does this happen? Is it unique to Banner Self-Service?

This type of vulnerability is inherent to most Internet applications, including banks, ecommerce, and other websites that require you to login to perform actions. It relies on the user to first login to the application, then, while still logged in, bring up an email, instant message, or other communication, and click on a link designed to inherit the cookies and session objects of the browser that is logged in. This issue is not unique to Banner Self-Service.

7. What is the best way to avoid this vulnerability?

The best way to avoid the issue from occurring is to instruct users to never click email or instant messenger links when they are logged into applications in their browser.

8. What is SunGard Higher Education doing to make Banner Self-Service less susceptible to this type of attack?

We are delivering server configuration values that use inherent features of the Apache web server to increase the level security. These values verify that all pages originated from the SSB server. These configuration values will need to be tailored to the needs of each institution. This is due to the unique nature of the implementation at each site.

See FAQ 1-2PE6V7 for details of the Apache change.

Appendix C: AndroidManifest.xml code

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android=”http://schemas.android.com/apk/res/android”
package="main.menu"
android:installLocation="preferExternal"
android:versionCode="1" android:versionName="1.0" >
<uses-sdk android:minSdkVersion="10" />
<uses-permission android:name="android.permission.INTERNET" />
<uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION" />
<uses-permission android:name="android.permission.ACCESS_FINE_LOCATION" />
<application
android:icon="@drawable/emu" android:label="Eastern Michigan University" >
<activity android:label="@string/app_name" android:name=".EMUActivity"
android:launchMode="singleTop" android:screenOrientation="portrait" >
<intent-filter ><action android:name="android.intent.action.MAIN" />
<category android:name="android.intent.category.LAUNCHER" /></intent-filter>
</activity>
<activity android:label="@string/app_name" android:name="emu.schedule.Schedule_Login"
android:screenOrientation="portrait" ></activity>
<activity android:label="@string/app_name" android:name="emu.schedule.Schedule_View"
android:screenOrientation="portrait" ></activity>
<activity android:label="@string/app_name" android:name="emu.schedule.Schedule_Locations"
android:screenOrientation="portrait" >
</activity>
<activity android:label="@string/app_name" android:name="emu.schedule.Schedule_Directions"
android:screenOrientation="portrait" >
</activity>
<activity android:label="@string/app_name" android:name="emu.directory.Directory_Search"
android:screenOrientation="portrait" >
</activity>
<activity android:label="@string/app_name"
android:name="emu.directory.Directory_Results" android:screenOrientation="portrait" >
</activity>
<activity android:label="@string/app_name"
android:name="emu.directory.Directory_Person" android:screenOrientation="portrait" >
</activity>
<activity android:label="@string/app_name" android:name="emu.athletics.Athletics"
android:screenOrientation="portrait" ></activity>
<activity android:label="@string/app_name" android:name="emu.bus.Bus_Menu"
android:screenOrientation="portrait" ></activity>
<activity android:label="@string/app_name" android:name="emu.bus.Bus_Display"
android:screenOrientation="portrait" ></activity>
<activity android:label="@string/app_name" android:name="emu.tuition.Tuition_Calculator"
android:screenOrientation="portrait" >
</activity>
```

```
<activity android:label="@string/app_name" android:name="emu.tuition.Tuition_Display"
android:screenOrientation="portrait" ></activity>
<activity android:label="@string/app_name" android:name="emu.map.Map"
android:screenOrientation="portrait" ></activity>
<activity android:label="@string/app_name" android:name="emu.gpa.GPA_Calculator"
android:screenOrientation="portrait" ></activity>
<activity android:label="@string/app_name" android:name="emu.twitter.Twitter_Options"
android:screenOrientation="portrait" ></activity>
<activity android:label="@string/app_name" android:name="emu.twitter.Twitter_Feed"
android:screenOrientation="portrait" ></activity>
<activity android:label="@string/app_name" android:name="emu.library.Library_Search"
android:screenOrientation="portrait" ></activity>
<activity android:label="@string/app_name" android:name="emu.library.Library_Catalog"
android:screenOrientation="portrait" ></activity>
<activity android:label="@string/app_name" android:name="emu.contact.Contact"
android:screenOrientation="portrait" ></activity>
<activity android:label="@string/app_name" android:name="emu.classes.Classes_Login"
android:screenOrientation="portrait" ></activity>
<activity android:label="@string/app_name" android:name="emu.classes.Classes_Search"
android:screenOrientation="portrait" ></activity>
<activity android:label="@string/app_name" android:name="emu.classes.Classes_View"
android:screenOrientation="portrait" ></activity>
<activity android:label="@string/app_name" android:name="emu.classes.Classes_Terms"
android:screenOrientation="portrait" ></activity>
<activity android:label="@string/app_name" android:name="emu.classes.Classes_Grades"
android:screenOrientation="portrait" ></activity>
<activity android:label="@string/app_name" android:name="emu.dining.Dining_Main"
android:screenOrientation="portrait" ></activity>
<activity android:label="@string/app_name" android:name="emu.dining.Dining_Menus"
android:screenOrientation="portrait" ></activity>
<activity android:label="@string/app_name" android:name="emu.dining.Dining_View"
android:screenOrientation="portrait" ></activity>
</application></manifest>
```

Appendix D: Main Menu code

main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android=”http://schemas.android.com/apk/res/android”
android:id="@+id/relativeLayout1" android:layout_width="fill_parent" android:layout_height="fill_parent"
android:orientation="vertical" android:layout_weight="0.37" >
<Button android:id="@+id/button1" android:layout_width="wrap_content"
android:layout_height="wrap_content" android:background="@drawable/calendar"
android:onClick="btn_1" android:layout_marginLeft="40px" android:layout_marginTop="40px" />
<TextView android:id="@+id/textView1" android:layout_width="90px"
android:gravity="center_vertical|center_horizontal" android:layout_height="wrap_content"
android:text="Weekly Schedule" android:layout_marginLeft="32px" android:layout_marginTop="120px"
android:textSize="15px" android:textColor="#448275" />
<Button android:id="@+id/button2" android:layout_width="wrap_content"
android:layout_height="wrap_content" android:background="@drawable/lookup"
android:onClick="btn_2" android:layout_marginLeft="150px" android:layout_marginTop="40px" />
<TextView android:id="@+id/textView2" android:layout_width="90px"
android:gravity="center_vertical|center_horizontal" android:layout_height="wrap_content"
android:text="Available Classes" android:layout_marginLeft="142px"
android:layout_marginTop="120px" android:textSize="15px" android:textColor="#448275" />
<Button android:id="@+id/button3" android:layout_width="wrap_content"
android:layout_height="wrap_content" android:background="@drawable/directory"
android:onClick="btn_3" android:layout_marginLeft="260px" android:layout_marginTop="40px" />
<TextView android:id="@+id/textView3" android:layout_width="90px"
android:gravity="center_vertical|center_horizontal" android:layout_height="wrap_content"
android:text="Online Directory" android:layout_marginLeft="252px" android:layout_marginTop="120px"
android:textSize="15px" android:textColor="#448275" />
<Button android:id="@+id/button4" android:layout_width="wrap_content"
android:layout_height="wrap_content" android:background="@drawable/bus" android:onClick="btn_4"
android:layout_marginLeft="370px" android:layout_marginTop="40px" />
<TextView android:id="@+id/textView4" android:layout_width="90px"
android:gravity="center_vertical|center_horizontal" android:layout_height="wrap_content"
android:text="Bus Routes" android:layout_marginLeft="362px" android:layout_marginTop="120px"
android:textSize="15px" android:textColor="#448275" />
<Button android:id="@+id/button5" android:layout_width="wrap_content"
android:layout_height="wrap_content" android:background="@drawable/library"
android:onClick="btn_5" android:layout_marginLeft="40px" android:layout_marginTop="180px" />
<TextView android:id="@+id/textView5" android:layout_width="90px"
android:gravity="center_vertical|center_horizontal" android:layout_height="wrap_content"
android:text="Library Catalog" android:layout_marginLeft="32px" android:layout_marginTop="260px"
android:textSize="15px" android:textColor="#448275" />
```

```
<Button android:id="@+id/button6" android:layout_width="wrap_content"
android:layout_height="wrap_content" android:background="@drawable/building"
android:onClick="btn_6" android:layout_marginLeft="150px" android:layout_marginTop="180px" />
<TextView android:id="@+id/textView6" android:layout_width="90px"
android:gravity="center_vertical|center_horizontal" android:layout_height="wrap_content"
android:text="Campus Map" android:layout_marginLeft="142px" android:layout_marginTop="260px"
android:textSize="15px" android:textColor="#448275" />
<Button android:id="@+id/button7" android:layout_width="wrap_content"
android:layout_height="wrap_content" android:background="@drawable/athletics"
android:onClick="btn_7" android:layout_marginLeft="260px" android:layout_marginTop="180px" />
<TextView android:id="@+id/textView7" android:layout_width="90px"
android:gravity="center_vertical|center_horizontal" android:layout_height="wrap_content"
android:text="Athletics" android:layout_marginLeft="252px" android:layout_marginTop="260px"
android:textSize="15px" android:textColor="#448275" />
<Button android:id="@+id/button8" android:layout_width="wrap_content"
android:layout_height="wrap_content" android:background="@drawable/tuition_calc"
android:onClick="btn_8" android:layout_marginLeft="370px" android:layout_marginTop="180px" />
<TextView android:id="@+id/textView8"  android:layout_width="90px"
android:gravity="center_vertical|center_horizontal" android:layout_height="wrap_content"
android:text="Tuition Calculator" android:layout_marginLeft="362px"
android:layout_marginTop="260px" android:textSize="15px" android:textColor="#448275" />
<Button android:id="@+id/button9" android:layout_width="wrap_content"
android:layout_height="wrap_content" android:background="@drawable/housing_dining"
android:onClick="btn_9" android:layout_marginLeft="40px" android:layout_marginTop="320px" />
<TextView android:id="@+id/textView9" android:layout_width="90px"
android:gravity="center_vertical|center_horizontal" android:layout_height="wrap_content"
android:text="Dining Locations" android:layout_marginLeft="32px" android:layout_marginTop="400px"
android:textSize="15px" android:textColor="#448275" />
<Button android:id="@+id/button10" android:layout_width="wrap_content"
android:layout_height="wrap_content" android:background="@drawable/gpa_calc"
android:onClick="btn_10" android:layout_marginLeft="150px" android:layout_marginTop="320px" />
<TextView android:id="@+id/textView10" android:layout_width="90px"
android:gravity="center_vertical|center_horizontal" android:layout_height="wrap_content"
android:text="GPA Calculator" android:layout_marginLeft="142px" android:layout_marginTop="400px"
android:textSize="15px" android:textColor="#448275" />
<Button android:id="@+id/button11" android:layout_width="wrap_content"
android:layout_height="wrap_content" android:background="@drawable/contact_emu"
android:onClick="btn_11" android:layout_marginLeft="260px" android:layout_marginTop="320px" />
<TextView android:id="@+id/textView11" android:layout_width="90px"
android:gravity="center_vertical|center_horizontal" android:layout_height="wrap_content"
android:text="Contact EMU" android:layout_marginLeft="252px" android:layout_marginTop="400px"
android:textSize="15px" android:textColor="#448275" />
<Button android:id="@+id/button12" android:layout_width="wrap_content"
android:layout_height="wrap_content" android:background="@drawable/twitter"
android:onClick="btn_12" android:layout_marginLeft="370px" android:layout_marginTop="320px" />
<TextView android:id="@+id/textView12" android:layout_width="90px"
android:gravity="center_vertical|center_horizontal" android:layout_height="wrap_content"
```

```
android:text="Twitter Updates"  android:layout_marginLeft="362px" android:layout_marginTop="400px"
android:textSize="15px" android:textColor="#448275" />
<Button android:id="@+id/button13" android:layout_width="wrap_content"
android:layout_height="wrap_content" android:background="@drawable/final_grades"
android:onClick="btn_13" android:layout_marginLeft="40px" android:layout_marginTop="460px" />
<TextView android:id="@+id/textView13" android:layout_width="90px"
android:gravity="center_vertical|center_horizontal" android:layout_height="wrap_content"
android:text="Final Grades" android:layout_marginLeft="32px" android:layout_marginTop="540px"
android:textSize="15px" android:textColor="#448275" />
<ImageView android:id="@+id/imageView1" android:layout_width="wrap_content"
android:layout_height="wrap_content" android:layout_marginLeft="140px"
android:layout_marginTop="600px" android:src="@drawable/emu_logo" />
</RelativeLayout>
```

EMUActivity

```
package main.menu;
import emu.athletics.Athletics; import emu.bus.Bus_Menu; import emu.classes.Classes_Login; import
emu.contact.Contact; import emu.dining.Dining_Main;
import emu.directory.Directory_Search; import emu.gpa.GPA_Calculator;
import emu.library.Library_Search; import emu.map.Map; import emu.schedule.Schedule_Login; import
emu.tuition.Tuition_Calculator;
import emu.twitter.Twitter_Options; import android.app.Activity; import android.content.Intent; import
android.os.Bundle; import android.view.View;
public class EMUActivity extends Activity {
   @Override
   public void onCreate(Bundle savedInstanceState) {
      super.onCreate(savedInstanceState);
      setContentView(R.layout.main);
   }
   public void btn_1(View view) {
         Intent btn = new Intent(this, Schedule_Login.class);
         startActivity(btn);
   }
   public void btn_2(View view) {
         Intent btn = new Intent(this, Classes_Login.class);
         btn.putExtra("ACTIVITY_CHECK", 1);
         startActivity(btn);
   }
   public void btn_3(View view) {
         Intent btn = new Intent(this, Directory_Search.class);
         startActivity(btn);
   }
   public void btn_4(View view) {
         Intent btn = new Intent(this, Bus_Menu.class);
```

```
        startActivity(btn);
    }
    public void btn_5(View view) {
        Intent btn = new Intent(this, Library_Search.class);
        startActivity(btn);
    }
    public void btn_6(View view) {
        Intent btn = new Intent(this, Map.class);
        startActivity(btn);
    }
    public void btn_7(View view) {
        Intent btn = new Intent(this, Athletics.class);
        startActivity(btn);
    }
    public void btn_8(View view) {
        Intent btn = new Intent(this, Tuition_Calculator.class);
        startActivity(btn);
    }
    public void btn_9(View view) {
        Intent btn = new Intent(this, Dining_Main.class);
        startActivity(btn);
    }
    public void btn_10(View view) {
        Intent btn = new Intent(this, GPA_Calculator.class);
        startActivity(btn);
    }
    public void btn_11(View view) {
        Intent btn = new Intent(this, Contact.class);
        startActivity(btn);
    }
    public void btn_12(View view) {
        Intent btn = new Intent(this, Twitter_Options.class);
        startActivity(btn);
    }
    public void btn_13(View view) {
        Intent btn = new Intent(this, Classes_Login.class);
        btn.putExtra("ACTIVITY_CHECK", 2);
        startActivity(btn);
    }
}
```

Appendix E: Library Catalog code

library_search.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android=”http://schemas.android.com/apk/res/android”
android:layout_width="fill_parent" android:layout_height="fill_parent" android:orientation="vertical"
android:background="@drawable/background" >
<TextView android:id="@+id/library_search_title" android:layout_width="wrap_content"
android:layout_height="wrap_content" android:textColor="#448275" android:textSize="25px"
android:layout_centerHorizontal="true" android:layout_marginTop="100px" android:text="Library
Catalog" />
<TextView android:id="@+id/search_title" android:layout_width="wrap_content"
android:layout_height="wrap_content" android:textColor="#448275" android:textSize="20px"
android:layout_centerHorizontal="true" android:layout_marginTop="200px" android:text="Search Title:"
/>
<EditText android:id="@+id/search_title_item" android:layout_width="400px"
android:layout_height="wrap_content" android:layout_marginLeft="40px"
android:layout_marginTop="225px" />
<Button android:id="@+id/search_library_btn" android:layout_width="200px"
android:layout_height="wrap_content" android:onClick="search_library_button"
android:layout_marginLeft="140px" android:layout_marginTop="325px" android:text="Search" />
<Button android:id="@+id/main_menu_btn_16" android:layout_width="match_parent"
android:layout_height="wrap_content" android:layout_alignParentBottom="true"
android:layout_alignParentLeft="true" android:onClick="main_menu_16"
android:background="@drawable/emu_button" />
</RelativeLayout>
```

Library_Search

```
package emu.library;
import main.menu.EMUActivity; import main.menu.R; import android.app.Activity; import
android.content.Intent; import android.os.Bundle; import android.view.View; import
android.widget.EditText;
public class Library_Search extends Activity {
	private EditText search_title;
	@Override
	public void onCreate(Bundle savedInstanceState) {
		super.onCreate(savedInstanceState);
		setContentView(R.layout.library_search);
		search_title = (EditText) findViewById(R.id.search_title_item);
	}
	public void search_library_button(View view) {
		String data_sent = search_title.getText().toString();
```

```
                Intent btn = new Intent(this, Library_Catalog.class);
                btn.putExtra("DATA", data_sent);
                startActivity(btn);
        }
        @Override
        public void onBackPressed() {
                finish();
         }
        public void main_menu_16(View view) {
                Intent btn = new Intent(this, EMUActivity.class);
                btn.setFlags(Intent.FLAG_ACTIVITY_CLEAR_TOP);
                startActivity(btn);
                finish();
        }
}
```

library_catalog.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
android:orientation="vertical" android:layout_width="fill_parent"  android:layout_height="fill_parent"
android:background="@drawable/background" android:padding="6dip" >
<TextView android:id="@+id/library_results_title" android:layout_width="wrap_content"
android:layout_height="wrap_content" android:layout_centerHorizontal="true"
android:layout_marginTop="40px" android:text="Library Catalog" android:textColor="#448275"
android:textSize="25px" />
<ListView android:id="@+id/library_search_items" android:layout_width="match_parent"
android:layout_height="wrap_content" android:layout_above="@+id/main_menu_btn_17"
android:layout_marginTop="70px" android:layout_alignLeft="@+id/main_menu_btn_17"
android:cacheColorHint="#00000000" android:background="@drawable/background"
android:layout_below="@+id/library_results_title" ></ListView>
<Button android:id="@+id/main_menu_btn_17" android:layout_width="match_parent"
android:layout_height="wrap_content" android:layout_alignParentBottom="true"
android:layout_alignParentLeft="true" android:onClick="main_menu_17"
android:background="@drawable/emu_button" />
</RelativeLayout>
```

library_item.xml

```
<TextView xmlns:android="http://schemas.android.com/apk/res/android"
android:id="@+id/library_item_line" android:layout_width="fill_parent"  android:layout_height="180px"
android:textColor="#FFFFFF" android:background="@drawable/background" android:padding="5px"
android:singleLine="false" />
```

Library_Catalog

```
package emu.library;
import java.io.BufferedReader; import java.io.InputStream; import java.io.InputStreamReader; import
java.util.ArrayList; import main.menu.EMUActivity; import main.menu.R; import
org.apache.http.HttpResponse; import org.apache.http.client.methods.HttpGet; import
org.apache.http.impl.client.DefaultHttpClient; import org.jsoup.Jsoup; import
org.jsoup.nodes.Document; import org.jsoup.nodes.Element; import org.jsoup.select.Elements; import
android.app.Activity; import android.app.ProgressDialog;
import android.content.Context; import android.content.Intent; import android.os.AsyncTask; import
android.os.Bundle; import android.view.View; import android.widget.ArrayAdapter; import
android.widget.ListView;
public class Library_Catalog extends Activity {
	private ListView library_results;
	private ArrayAdapter<String> adapter;
	private Context myContext;
	public void onCreate(Bundle savedInstanceState) {
		super.onCreate(savedInstanceState);
		setContentView(R.layout.library_catalog);
		Intent btn = getIntent();
		String data_received = btn.getStringExtra("DATA");
		myContext = this;
		library_results = (ListView) findViewById(R.id.library_search_items);
		String web =
	"http://portal.emich.edu/vwebv/search?searchCode=GKEY%5E&limitTo=TYPE%3Dam&recCo
	unt=50&searchType=1&page.search.search.button=Search";
		web += "&searchArg=" + data_received;
		Page_Download task = new Page_Download();
		task.execute(new String[] { web });
	}
	private class Page_Download extends AsyncTask<String, Void, String> {
	private ProgressDialog progressDialog;
	private ArrayList<String> search_results_items = new ArrayList<String>();
	protected void onPreExecute() {
		progressDialog = ProgressDialog.show(Library_Catalog.this, "", "Loading. Please
	wait...", true);
	}
	@Override
	protected String doInBackground(String... urls) {
		String response = "";
		for (String url : urls) {
			DefaultHttpClient client = new DefaultHttpClient();
			HttpGet httpget = new HttpGet(url);
			try {
				HttpResponse execute = client.execute(httpget);
				InputStream content = execute.getEntity().getContent();
```

```
                                BufferedReader buffer = new BufferedReader(new
InputStreamReader(content));
                                String s = "";
                                while ((s = buffer.readLine()) != null) {
                                    response += s;
                                }
                                Document doc = Jsoup.parseBodyFragment(response);
                                Elements results_content = doc.select("div.resultListTextCell");
                                response = "";
                                for (Element book : results_content) {
                                    String line_1 = book.select("div.line1Link").first().text();
                                    String line_2 = book.select("div.line2Link").first().text();
                                    String line_3 = book.select("div.line4Link").first().text();
                                    String line_4 = book.select("div.line5Link").first().text();
                                    response = line_1 + "\n" + line_2 + "\n" + line_3 + "\n" + line_4 + "\n";
                                    search_results_items.add(response);
                                    }
                            }
                            catch (Exception e) {
                                    e.printStackTrace();
                            }
                    }
                    return response;
            }
            @Override
            protected void onPostExecute(String result) {
                    progressDialog.dismiss();
                    adapter = new ArrayAdapter<String>(myContext, R.layout.library_item,
            search_results_items);
                    library_results.setAdapter(adapter);
            }
    }
    @Override
     public void onBackPressed() {
            finish();
    }
    public void main_menu_17(View view) {
            Intent btn = new Intent(this, EMUActivity.class);
            btn.setFlags(Intent.FLAG_ACTIVITY_CLEAR_TOP);
            startActivity(btn);
            finish();
} }
```

Appendix F: Tuition & Fees Calculator code

tuition_calculator.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
android:layout_width="fill_parent" android:layout_height="wrap_content"
android:background="@drawable/background" android:orientation="vertical" >
<TextView android:id="@+id/residency_choice" android:layout_width="wrap_content"
android:layout_height="wrap_content" android:layout_centerHorizontal="true"
android:layout_marginBottom="2px" android:layout_marginTop="20px" android:text="Residency:"
android:textColor="#448275" android:textSize="20px" />
<Spinner android:id="@+id/residency" android:layout_width="440px"
android:layout_height="wrap_content" android:layout_centerHorizontal="true"
android:layout_marginTop="50px" />
<TextView android:id="@+id/credits" android:layout_width="wrap_content"
android:layout_height="wrap_content" android:layout_marginLeft="20px"
android:layout_marginTop="120px" android:text="Credits:"  android:textColor="#448275"
android:textSize="20px" />
<TextView android:id="@+id/level" android:layout_width="wrap_content"
android:layout_height="wrap_content" android:layout_marginLeft="140px"
android:layout_marginTop="120px" android:text="Level:"  android:textColor="#448275"
android:textSize="20px" />
<TextView android:id="@+id/grouping" android:layout_width="wrap_content"
android:layout_height="wrap_content" android:layout_marginLeft="270px"
android:layout_marginTop="120px" android:text="Course Grouping:"  android:textColor="#448275"
android:textSize="20px" />
<EditText android:id="@+id/credits_0" android:layout_width="40px"
android:layout_height="wrap_content" android:layout_alignBaseline="@+id/level0"
android:layout_alignBottom="@+id/level0" android:layout_alignLeft="@+id/credits"
android:layout_alignRight="@+id/credits" />
<EditText android:id="@+id/credits_1" android:layout_width="40px"
android:layout_height="wrap_content" android:layout_alignBottom="@+id/level1"
android:layout_alignLeft="@+id/credits_2" android:layout_alignRight="@+id/credits_2" />
<EditText android:id="@+id/credits_2" android:layout_width="40px"
android:layout_height="wrap_content" android:layout_alignParentLeft="true"
android:layout_alignRight="@+id/credits" android:layout_marginLeft="20px"
android:layout_marginTop="310px" />
<EditText android:id="@+id/credits_3" android:layout_width="40px"
android:layout_height="wrap_content" android:layout_alignParentLeft="true"
android:layout_alignRight="@+id/credits" android:layout_marginLeft="20px"
android:layout_marginTop="390px" />
<EditText android:id="@+id/credits_4" android:layout_width="40px"
android:layout_height="wrap_content" android:layout_alignParentLeft="true"
```

```
android:layout_alignRight="@+id/credits" android:layout_marginLeft="20px"
android:layout_marginTop="470px" />
<Spinner android:id="@+id/level0" android:layout_width="130px"
android:layout_height="wrap_content" android:layout_marginLeft="100px"
android:layout_marginTop="150px" />
<Spinner android:id="@+id/level1" android:layout_width="130px"
android:layout_height="wrap_content" android:layout_marginLeft="100px"
android:layout_marginTop="230px" />
<Spinner android:id="@+id/level2" android:layout_width="130px"
android:layout_height="wrap_content" android:layout_marginLeft="100px"
android:layout_marginTop="310px" />
<Spinner android:id="@+id/level3" android:layout_width="130px"
android:layout_height="wrap_content" android:layout_marginLeft="100px"
android:layout_marginTop="390px" />
<Spinner android:id="@+id/level4" android:layout_width="130px"
android:layout_height="wrap_content" android:layout_marginLeft="100px"
android:layout_marginTop="470px" />
<Spinner android:id="@+id/grouping0" android:layout_width="220px"
android:layout_height="wrap_content" android:layout_alignBaseline="@+id/level0"
android:layout_alignBottom="@+id/level0" android:layout_alignRight="@+id/residency" />
<Spinner android:id="@+id/grouping1" android:layout_width="220px"
android:layout_height="wrap_content" android:layout_alignBaseline="@+id/credits_1"
android:layout_alignBottom="@+id/credits_1" android:layout_alignLeft="@+id/grouping0" />
<Spinner android:id="@+id/grouping2" android:layout_width="220px"
android:layout_height="wrap_content" android:layout_alignBottom="@+id/credits_2"
android:layout_alignLeft="@+id/grouping1" />
<Spinner android:id="@+id/grouping3" android:layout_width="220px"
android:layout_height="wrap_content" android:layout_alignBottom="@+id/credits_3"
android:layout_alignLeft="@+id/grouping2" />
<Spinner android:id="@+id/grouping4" android:layout_width="220px"
android:layout_height="wrap_content" android:layout_alignBaseline="@+id/level4"
android:layout_alignBottom="@+id/level4" android:layout_alignRight="@+id/tuition_calculator_btn" />
<Button android:id="@+id/tuition_calculator_btn" android:layout_width="440px"
android:layout_height="wrap_content" android:layout_centerHorizontal="true"
android:layout_marginTop="550px" android:onClick="tuition_calculator_button" android:text="Calculate
Tuition" />
<Button android:id="@+id/main_menu_btn_10" android:layout_width="match_parent"
android:layout_height="wrap_content" android:layout_alignParentBottom="true"
android:layout_alignParentLeft="true" android:background="@drawable/emu_button"
android:onClick="main_menu_10" />
</RelativeLayout>
```

Tuition_Calculator

```
package emu.tuition;
```

```
import java.io.BufferedReader; import java.io.InputStream; import java.io.InputStreamReader; import
java.util.ArrayList; import java.util.List;
import main.menu.EMUActivity; import main.menu.R; import org.apache.http.HttpResponse; import
org.apache.http.NameValuePair; import org.apache.http.client.entity.UrlEncodedFormEntity; import
org.apache.http.client.methods.HttpPost; import org.apache.http.impl.client.DefaultHttpClient; import
org.apache.http.message.BasicNameValuePair; import org.jsoup.Jsoup; import
org.jsoup.nodes.Document; import org.jsoup.nodes.Element; import org.jsoup.select.Elements; import
android.app.Activity; import android.app.ProgressDialog;
import android.content.Intent; import android.os.AsyncTask; import android.os.Bundle;
import android.view.View; import android.widget.AdapterView; import android.widget.ArrayAdapter;
import android.widget.EditText; import android.widget.Spinner;
public class Tuition_Calculator extends Activity implements AdapterView.OnItemSelectedListener {
        private static final String[] residency_values = {"Choose", "MI/OH", "Non MI/OH"};
        private static final String[] level_values = {"Choose", "100-299", "300-499", "500-699", "700+"};
        private static final String[] grouping_values = {"Choose", "Business", "Education", "Fine Arts",
"Foreign Languages", "Health & Human Services (except Nursing)", "Leadership and Counseling
(Doctoral)", "Nursing", "Science (Bio,Chem,Comp. Sc.,Math,Phy,Astronomy)", "Technology (except
Military Science)", "All other courses"};
        public Spinner residency;
        public String residency_value;
        public int[] spinner_ids;
        public int spinner_check;
        public int spinner_index;
        public EditText credits_0, credits_1, credits_2, credits_3, credits_4;
        public Spinner level_0, level_1, level_2, level_3, level_4;
        public String level0, level1, level2, level3, level4;
        public Spinner grouping_0, grouping_1, grouping_2, grouping_3, grouping_4;
        public String grouping0, grouping1, grouping2, grouping3, grouping4;
   @Override
   public void onCreate(Bundle savedInstanceState) {
      super.onCreate(savedInstanceState);
      setContentView(R.layout.tuition_calculator);
      spinner_ids = new int[11];
      spinner_check = 0;
      spinner_index = 0;
      residency_value = "-" + 1;
      level0 = "-" + 1; level1 = "-" + 1; level2 = "-" + 1; level3 = "-" + 1; level4 = "-" + 1;
      grouping0 = "-" + 1; grouping1 = "-" + 1; grouping2 = "-" + 1; grouping3 = "-" + 1; grouping4 = "-" +
1;
      residency = (Spinner) findViewById(R.id.residency);
      residency.setOnItemSelectedListener(this);
      credits_0 = (EditText) findViewById(R.id.credits_0);
      credits_1 = (EditText) findViewById(R.id.credits_1);
      credits_2 = (EditText) findViewById(R.id.credits_2);
      credits_3 = (EditText) findViewById(R.id.credits_3);
      credits_4 = (EditText) findViewById(R.id.credits_4);
      level_0 = (Spinner) findViewById(R.id.level0);
```

```
        level_1 = (Spinner) findViewById(R.id.level1);
        level_2 = (Spinner) findViewById(R.id.level2);
        level_3 = (Spinner) findViewById(R.id.level3);
        level_4 = (Spinner) findViewById(R.id.level4);
        level_0.setOnItemSelectedListener(this);
        level_1.setOnItemSelectedListener(this);
        level_2.setOnItemSelectedListener(this);
        level_3.setOnItemSelectedListener(this);
        level_4.setOnItemSelectedListener(this);
        grouping_0 = (Spinner) findViewById(R.id.grouping0);
        grouping_1 = (Spinner) findViewById(R.id.grouping1);
        grouping_2 = (Spinner) findViewById(R.id.grouping2);
        grouping_3 = (Spinner) findViewById(R.id.grouping3);
        grouping_4 = (Spinner) findViewById(R.id.grouping4);
        grouping_0.setOnItemSelectedListener(this);
        grouping_1.setOnItemSelectedListener(this);
        grouping_2.setOnItemSelectedListener(this);
        grouping_3.setOnItemSelectedListener(this);
        grouping_4.setOnItemSelectedListener(this);
        ArrayAdapter<String> res = new ArrayAdapter<String>(this,
android.R.layout.simple_spinner_item, residency_values);
        ArrayAdapter<String> lev = new ArrayAdapter<String>(this,
android.R.layout.simple_spinner_item, level_values);
        ArrayAdapter<String> courses = new ArrayAdapter<String>(this,
android.R.layout.simple_spinner_item, grouping_values);
        res.setDropDownViewResource(android.R.layout.simple_spinner_dropdown_item);
        lev.setDropDownViewResource(android.R.layout.simple_spinner_dropdown_item);
courses.setDropDownViewResource(android.R.layout.simple_spinner_dropdown_item);
        residency.setAdapter(res);
        level_0.setAdapter(lev); level_1.setAdapter(lev); level_2.setAdapter(lev); level_3.setAdapter(lev);
level_4.setAdapter(lev);
        grouping_0.setAdapter(courses); grouping_1.setAdapter(courses);
grouping_2.setAdapter(courses); grouping_3.setAdapter(courses); grouping_4.setAdapter(courses);
    }
    public void onResume() {
            super.onResume();
            spinner_ids = new int[11];
            spinner_check = 0;
            spinner_index = 0;
    }
    public void onRestart() {
            super.onRestart();
            spinner_ids = new int[11];
            spinner_check = 0;
            spinner_index = 0;
    }
    public void onItemSelected(AdapterView<?> parent, View v, int position, long id) {
```

```
        int b = parent.getId();
        if (spinner_index < 11) {
                spinner_ids[spinner_index] = b;
                spinner_index++;
        }
        int pos = position - 1;
        setSpinner(b, pos);
    }

    public void onNothingSelected(AdapterView<?> parent) {
    }
    public void setSpinner(int spin_id, int value)
    {
        if (spin_id == spinner_ids[0])
                residency_value = "" + value;
        else if (spin_id == spinner_ids[1])
                level0 = "" + value;
        else if (spin_id == spinner_ids[2])
                level1 = "" + value;
        else if (spin_id == spinner_ids[3])
                level2 = "" + value;
        else if (spin_id == spinner_ids[4])
                level3 = "" + value;
        else if (spin_id == spinner_ids[5])
                level4 = "" + value;
        else if (spin_id == spinner_ids[6])
                grouping0 = "" + value;
        else if (spin_id == spinner_ids[7])
                grouping1 = "" + value;
        else if (spin_id == spinner_ids[8])
                grouping2 = "" + value;
        else if (spin_id == spinner_ids[9])
                grouping3 = "" + value;
        else
                grouping4 = "" + value;
    }
    private class Page_Download extends AsyncTask<String, Void, String> {
                private ProgressDialog progressDialog;
                protected void onPreExecute() {
                        progressDialog = ProgressDialog.show(Tuition_Calculator.this, "", "Loading.
Please wait...", true);
                }
                @Override
                protected String doInBackground(String... urls) {
                        String response = "";
                        for (String url : urls) {
                                DefaultHttpClient client = new DefaultHttpClient();
```

```
                HttpPost httppost = new HttpPost(url);
                try {
                        String n1 = credits_0.getText().toString();
                        String n2 = credits_1.getText().toString();
                        String n3 = credits_2.getText().toString();
                        String n4 = credits_3.getText().toString();
                        String n5 = credits_4.getText().toString();

                        List<NameValuePair> nameValuePairs = new
ArrayList<NameValuePair>(16);
                nameValuePairs.add(new BasicNameValuePair("residency",
residency_value));

                nameValuePairs.add(new BasicNameValuePair("credits0", n1));
                nameValuePairs.add(new BasicNameValuePair("credits1", n2));
                nameValuePairs.add(new BasicNameValuePair("credits2", n3));
                nameValuePairs.add(new BasicNameValuePair("credits3", n4));
                nameValuePairs.add(new BasicNameValuePair("credits4", n5));

                nameValuePairs.add(new BasicNameValuePair("level0", level0));
                nameValuePairs.add(new BasicNameValuePair("level1", level1));
                nameValuePairs.add(new BasicNameValuePair("level2", level2));
                nameValuePairs.add(new BasicNameValuePair("level3", level3));
                nameValuePairs.add(new BasicNameValuePair("level4", level4));

                nameValuePairs.add(new BasicNameValuePair("grouping0",
grouping0));
                nameValuePairs.add(new BasicNameValuePair("grouping1",
grouping1));
                nameValuePairs.add(new BasicNameValuePair("grouping2",
grouping2));
                nameValuePairs.add(new BasicNameValuePair("grouping3",
grouping3));
                nameValuePairs.add(new BasicNameValuePair("grouping4",
grouping4));

                httppost.setEntity(new UrlEncodedFormEntity(nameValuePairs));

                HttpResponse execute = client.execute(httppost);
                    InputStream content = execute.getEntity().getContent();

                    BufferedReader buffer = new BufferedReader(new
InputStreamReader(content));
                    String s = "";
                    while ((s = buffer.readLine()) != null) {
                            response += s;
                    }
```

```
                                        Document doc = Jsoup.parseBodyFragment(response);
                                        response = "";
                                        Element text_content = doc.select("div.mobile").first();
                                        if (text_content.text().substring(0,4).equals("Your")) {
                                                Elements result_rows = text_content.select("tr");
                                                for (Element row : result_rows) {
                                                        response += row.text() + "\n";
                                                }
                                        }
                                        else
                                                response = "There was an error in your request, try again
later.";
                                }
                                catch (Exception e) {
                                        response = "Error processing your request, please try again later.";
                                        e.printStackTrace();
                                }
                        }
                        return response;
                }

                @Override
                protected void onPostExecute(String result) {
                        progressDialog.dismiss();
                        tuition_display_view(result);
                }
        }
        @Override
        public void onBackPressed() {
                finish();
        }
        public void tuition_calculator_button(View view)
        {
                String web = "http://www.emich.edu/sbs/tuitionfeesresults.php";
                Page_Download task = new Page_Download();
                task.execute(new String[] { web });
        }
        public void tuition_display_view(String res)
        {
                Intent results_display = new Intent(this, Tuition_Display.class);
                results_display.putExtra("res_value", res);
                startActivity(results_display);
                finish();
        }
        public void main_menu_10(View view) {
                Intent btn = new Intent(this, EMUActivity.class);
                btn.setFlags(Intent.FLAG_ACTIVITY_CLEAR_TOP);
```

```
            startActivity(btn);
            finish();
    }
}
```

tuition_display.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
android:layout_width="fill_parent" android:layout_height="wrap_content" android:orientation="vertical"
android:background="@drawable/background" >
<TextView android:id="@+id/tuition_display_title" android:layout_width="wrap_content"
android:layout_height="wrap_content" android:layout_marginBottom="2px"
android:layout_marginTop="50px" android:layout_centerHorizontal="true" android:text="Tuition Costs"
android:textColor="#448275" android:textSize="30px" />
<TextView android:id="@+id/tuition_display" android:layout_width="450px"
android:layout_height="wrap_content" android:layout_marginBottom="2px"
android:layout_marginTop="150px" android:layout_centerHorizontal="true"
android:textColor="#448275" android:textSize="25px" />
<Button android:id="@+id/main_menu_btn_11" android:layout_width="match_parent"
android:layout_height="wrap_content" android:layout_alignParentBottom="true"
android:layout_alignParentLeft="true" android:onClick="main_menu_11"
android:background="@drawable/emu_button" />
</RelativeLayout>
```

Tuition_Display

```
package emu.tuition;
import main.menu.EMUActivity; import main.menu.R; import android.app.Activity; import
android.content.Intent; import android.os.Bundle; import android.view.View; import
android.widget.TextView;
public class Tuition_Display extends Activity {
            private TextView show_results;
            private String res;
            public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.tuition_display);
        show_results = (TextView) findViewById(R.id.tuition_display);
        Intent results_display = getIntent();
        res = results_display.getStringExtra("res_value");
        show_results.setText(res);
            }
            @Override
    public void onBackPressed() {
```

```
        Intent btn = new Intent(this, Tuition_Calculator.class);
                startActivity(btn);
                finish();
    }
        public void main_menu_11(View view) {
                Intent btn = new Intent(this, EMUActivity.class);
                btn.setFlags(Intent.FLAG_ACTIVITY_CLEAR_TOP);
                startActivity(btn);
                finish();
        }
}
```

Appendix G: Online Directory code

directory_search.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
android:layout_width="fill_parent" android:layout_height="fill_parent" android:orientation="vertical"
android:background="@drawable/background" >
<TextView android:id="@+id/first_name" android:layout_width="400px"
android:layout_height="wrap_content" android:textColor="#448275" android:textSize="20px"
android:layout_marginLeft="40px" android:layout_marginTop="100px" android:text="First Name" />
<EditText android:id="@+id/search_first_name" android:layout_width="400px"
android:layout_height="wrap_content" android:layout_marginLeft="40px"
android:layout_marginTop="125px" ></EditText>
<TextView android:id="@+id/last_name" android:layout_width="400px"
android:layout_height="wrap_content" android:textColor="#448275" android:textSize="20px"
android:layout_marginLeft="40px" android:layout_marginTop="200px" android:text="Last Name:" />
<EditText android:id="@+id/search_last_name" android:layout_width="400px"
android:layout_height="wrap_content" android:layout_marginLeft="40px"
android:layout_marginTop="225px" />
<Button android:id="@+id/search_directory_btn" android:layout_width="200px"
android:layout_height="wrap_content" android:onClick="search_directory_button"
android:layout_marginLeft="140px" android:layout_marginTop="325px" android:text="Search" />
<Button android:id="@+id/main_menu_btn_4" android:layout_width="match_parent"
android:layout_height="wrap_content" android:layout_alignParentBottom="true"
android:layout_alignParentLeft="true" android:onClick="main_menu_4"
android:background="@drawable/emu_button" />
<TextView android:id="@+id/more_results" android:layout_width="wrap_content"
android:layout_height="wrap_content" android:layout_below="@+id/search_directory_btn"
android:layout_centerHorizontal="true" android:layout_marginTop="28dp" android:textColor="#448275"
android:textSize="20px" android:text="" />
</RelativeLayout>
```

Directory_Search

```
package emu.directory;
import java.io.BufferedReader; import java.io.InputStream; import java.io.InputStreamReader; import
java.util.ArrayList; import java.util.List;
import main.menu.EMUActivity; import main.menu.R;
import org.apache.http.HttpResponse; import org.apache.http.NameValuePair; import
org.apache.http.client.entity.UrlEncodedFormEntity; import org.apache.http.client.methods.HttpPost;
import org.apache.http.impl.client.DefaultHttpClient; import
org.apache.http.message.BasicNameValuePair; import org.jsoup.Jsoup; import
org.jsoup.nodes.Document; import org.jsoup.nodes.Element;
```

```
import android.app.Activity; import android.app.ProgressDialog; import android.content.Intent; import
android.os.AsyncTask; import android.os.Bundle; import android.view.View; import
android.widget.EditText; import android.widget.TextView;
public class Directory_Search extends Activity {
	private EditText fname, lname;
	private String first_name, last_name;
	private String[] searchNames;
	private String[] searchLinks;
	private TextView no_results;
	@Override
	public void onCreate(Bundle savedInstanceState) {
		super.onCreate(savedInstanceState);
		setContentView(R.layout.directory_search);
		fname = (EditText) findViewById(R.id.search_first_name);
		lname = (EditText) findViewById(R.id.search_last_name);
		no_results = (TextView) findViewById(R.id.more_results);
	}
	@Override
   public void onBackPressed() {
	finish();
   }
	@Override
	public void onResume() {
		super.onResume();
		no_results.setText("");
	}
	private class Page_Download extends AsyncTask<String, Void, String> {
		private ProgressDialog progressDialog;
		private int num = 0;
		private int counter;
		protected void onPreExecute() {
			progressDialog = ProgressDialog.show(Directory_Search.this, "", "Loading.
Please wait...", true);
		}
		@Override
		protected String doInBackground(String... urls) {
			String response = "";
			for (String url : urls) {
				DefaultHttpClient client = new DefaultHttpClient();
				HttpPost httppost = new HttpPost(url);
				try {
					first_name = fname.getText().toString();
					last_name = lname.getText().toString();
					List<NameValuePair> nameValuePairs = new
ArrayList<NameValuePair>(3);
				nameValuePairs.add(new BasicNameValuePair("First_name",
first_name));
```

```
                        nameValuePairs.add(new BasicNameValuePair("Last_name",
last_name));
                        nameValuePairs.add(new BasicNameValuePair("Type", "ALL"));
                        httppost.setEntity(new UrlEncodedFormEntity(nameValuePairs));
                        HttpResponse execute = client.execute(httppost);
                            InputStream content = execute.getEntity().getContent();
                            BufferedReader buffer = new BufferedReader(new
InputStreamReader(content));
                            String s = "";
                            while ((s = buffer.readLine()) != null) {
                                    response += s;
                            }
                            Document doc = Jsoup.parseBodyFragment(response);
                            Element body = doc.body();
                            Element table = body.select("table.text").get(1);
                            Element paragraph = table.select("tbody tr td").first();
                            Element table2 = paragraph.select("table.text tbody").first();
                            Element numberOfResults = paragraph.select("b").first();
                            String hundred_results = numberOfResults.text();
                            String initial_id = "";
                            if (hundred_results.equals("100 matches")) {
                                    response = "There are more than 20 results. Please be more
specific and try again.";
                            }
                            else {
                                    num = Integer.parseInt(numberOfResults.text());
                                    if ((num <= 20) && (num != 0)) {
                                            counter = num;
                                            searchNames = new String[counter];
                                            searchLinks = new String[counter];
                                            response = "";
                                            for (int i=1; i<=10; i++) {
                                                    Element row = table2.select("tr").get(i);
                                                    Element column = row.select("td").first();
                                                    Element link = column.select("a").first();
                                                    searchNames[i-1] = link.text();
                                                    searchLinks[i-1] = link.attr("href");
                                                    if (i==1)
                                                            initial_id = searchLinks[i-
1].substring(92);
                                            }
                                            for (int i=10; i<=num; i+=10)
                                            {
                                                    String rep = "" + i;
                                                    List<NameValuePair> nameValuePairs_2 =
new ArrayList<NameValuePair>(5);
```

```
                                                    nameValuePairs_2.add(new
BasicNameValuePair("First_name", first_name));
                                                    nameValuePairs_2.add(new
BasicNameValuePair("Last_name", last_name));
                                                    nameValuePairs_2.add(new
BasicNameValuePair("Type", "ALL"));
                                                    nameValuePairs_2.add(new
BasicNameValuePair("ID", initial_id));
                                                    nameValuePairs_2.add(new
BasicNameValuePair("StartRow", rep));
                                                    httppost.setEntity(new
UrlEncodedFormEntity(nameValuePairs_2));
                                                    HttpResponse execute_2 =
client.execute(httppost);
                                                    InputStream content_2 =
execute_2.getEntity().getContent();
                                                    BufferedReader buffer_2 = new
BufferedReader(new InputStreamReader(content_2));
                                                    String t = "";
                                                    while ((t = buffer_2.readLine()) != null) {
                                                            response += t;
                                                    }
                                                    Document doc_2 =
Jsoup.parseBodyFragment(response);
                                                    Element body_2 = doc_2.body();
                                                    Element table_2 =
body_2.select("table.text").get(1);
                                                    Element paragraph_2 = table_2.select("tbody
tr td").first();
                                                    for (int j=1; j<=10; j++) {
                                                            Element table3 =
paragraph_2.select("table.text tbody").first();
                                                            Element row_2 =
table3.select("tr").get(j);
                                                            Element column_2 =
row_2.select("td").first();
                                                            Element link_2 =
column_2.select("a").first();
                                                            searchNames[i-1+j] = link_2.text();
                                                            searchLinks[i-1+j] =
link_2.attr("href");
                                                    }
                                            }
                                    }
                                    else if (num==0)
                                            response = "There are no results for your search.";
                                    else if ((num>20) || (hundred_results.equals("100 matches")))
```

```
                                                response = "There are more than 20 results. Please
be more specific and try again.";
                                                }
                                        }
                                        catch (Exception e) {
                                                e.printStackTrace();
                                        }
                                }
                                return response;
                        }
                        @Override
                        protected void onPostExecute(String result) {
                                progressDialog.dismiss();
                                startIntent(searchNames, searchLinks, num, result);
                        }
                }
                public void search_directory_button(View view) {
                        String web =
"http://it.emich.edu/service/online/directory/index.cfm?fuseaction=search_results";
                        Page_Download task = new Page_Download();
                        task.execute(new String[] { web });
                }
                public void startIntent(String[] array1, String[] array2, int num, String result) {
                        if (result.equals("100 matches") || (num > 20) || (num == 0)) {
                                no_results.setText(result);
                        }
                        else {
                                Intent btn = new Intent(this, Directory_Results.class);
                                btn.putExtra("resNames", array1);
                                btn.putExtra("resLinks", array2);
                                startActivity(btn);
                        }
                }
                public void main_menu_4(View view) {
                        Intent btn = new Intent(this, EMUActivity.class);
                        btn.setFlags(Intent.FLAG_ACTIVITY_CLEAR_TOP);
                        startActivity(btn);
                        finish();
                }
}
```

directory_results.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
```

```
android:orientation="vertical" android:layout_width="match_parent"
android:layout_height="match_parent" android:background="@drawable/background" >
<TextView android:id="@+id/search_results_title" android:layout_width="wrap_content"
android:layout_height="wrap_content" android:layout_centerHorizontal="true"
android:layout_marginTop="30px" android:layout_marginBottom="30px" android:textColor="#448275"
android:textSize="25px" android:text="Search Results:" />
<Button android:id="@+id/main_menu_btn_5" android:layout_width="match_parent"
android:layout_height="wrap_content" android:layout_alignParentBottom="true"
android:layout_alignParentLeft="true" android:background="@drawable/emu_button"
android:onClick="main_menu_button_5" />
<ListView android:id="@+id/search_directory_list" android:layout_width="match_parent"
android:layout_height="wrap_content" android:layout_above="@+id/main_menu_btn_5"
android:layout_alignParentRight="true" android:layout_below="@+id/search_results_title"
android:cacheColorHint="#00000000"android:background="@drawable/background" >
</ListView>
</RelativeLayout>
```

Directory_Results

```
package emu.directory;
import main.menu.EMUActivity; import main.menu.R; import android.app.Activity; import
android.content.Intent; import android.os.Bundle; import android.view.View; import
android.widget.AdapterView; import android.widget.AdapterView.OnItemClickListener; import
android.widget.ArrayAdapter; import android.widget.ListView;
public class Directory_Results extends Activity {
	private String[] names = new String[100];
	private String[] links = new String[100];
	private ListView directory_search_listview;
	@Override
	public void onCreate(Bundle savedInstanceState) {
		super.onCreate(savedInstanceState);
		setContentView(R.layout.directory_results);
	directory_search_listview = (ListView) findViewById(R.id.search_directory_list);
		Intent btn = getIntent();
		names = btn.getStringArrayExtra("resNames");
		links = btn.getStringArrayExtra("resLinks");
		directory_search_listview.setAdapter(new
ArrayAdapter<String>(this,android.R.layout.simple_list_item_1, names));
		directory_search_listview.setOnItemClickListener(new OnItemClickListener() {
			public void onItemClick(AdapterView<?> parent, View view, int position, long
id) {
				goIntent(links[position]);
			}
		});
	}
```

```
        @Override
    public void onBackPressed() {
        finish();
    }
        public void goIntent(String item) {
                Intent btn = new Intent(this, Directory_Person.class);
                btn.putExtra("ITEM_LINK", item);
                startActivity(btn);
        }
        public void main_menu_button_5(View view) {
                Intent btn = new Intent(this, EMUActivity.class);
                btn.setFlags(Intent.FLAG_ACTIVITY_CLEAR_TOP);
                startActivity(btn);
                finish();
        }
}
```

directory_person.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
android:layout_width="match_parent" android:layout_height="match_parent"
android:background="@drawable/background" android:orientation="vertical" >
<TextView android:id="@+id/person_title" android:layout_width="wrap_content"
android:layout_height="wrap_content" android:layout_centerHorizontal="true"
android:layout_marginTop="60px" android:text="EMU Online Directory"  android:textColor="#448275"
android:textSize="25px" />
<TextView android:id="@+id/person_name" android:layout_width="wrap_content"
android:layout_height="wrap_content" android:layout_marginLeft="110px"
android:layout_marginTop="140px" android:text="Person:"  android:textColor="#448275"
android:textSize="20px" />
<TextView android:id="@+id/person_name_text" android:layout_width="wrap_content"
android:layout_height="wrap_content" android:layout_marginLeft="200px"
android:layout_marginTop="140px" android:text="Person" android:textColor="#448275"
android:textSize="20px" />
<TextView android:id="@+id/person_email" android:layout_width="wrap_content"
android:layout_height="wrap_content" android:layout_above="@+id/person_address_text"
android:layout_alignRight="@+id/person_name" android:text="Email:"  android:textColor="#448275"
android:textSize="20px" />
<TextView android:id="@+id/person_email_text" android:layout_width="wrap_content"
android:layout_height="wrap_content" android:layout_alignLeft="@+id/person_name_text"
android:layout_below="@+id/person_name_text" android:layout_marginTop="30dp"
android:clickable="true" android:onClick="send_email" android:text="Email"
android:textColor="#448275" android:textSize="20px" />
```

```
<TextView android:id="@+id/person_address" android:layout_width="wrap_content"
android:layout_height="wrap_content" android:layout_alignBaseline="@+id/person_address_text"
android:layout_alignBottom="@+id/person_address_text"
android:layout_alignRight="@+id/person_email" android:text="Address:"  android:textColor="#448275"
android:textSize="20px" />
<TextView android:id="@+id/person_address_text" android:layout_width="wrap_content"
android:layout_height="wrap_content" android:layout_alignLeft="@+id/person_email_text"
android:layout_below="@+id/person_email_text" android:layout_marginTop="30dp"
android:text="Address" android:textColor="#448275" android:textSize="20px" />
<Button android:id="@+id/main_menu_btn_6" android:layout_width="match_parent"
android:layout_height="wrap_content" android:layout_alignParentBottom="true"
android:layout_alignParentLeft="true" android:background="@drawable/emu_button"
android:onClick="main_menu_button_6" />
</RelativeLayout>
```

Directory_Person

```
package emu.directory;
import java.io.BufferedReader; import java.io.InputStream; import java.io.InputStreamReader; import
org.apache.http.HttpResponse; import org.apache.http.client.methods.HttpPost; import
org.apache.http.impl.client.DefaultHttpClient; import org.jsoup.Jsoup; import
org.jsoup.nodes.Document; import org.jsoup.nodes.Element; import main.menu.EMUActivity; import
main.menu.R; import android.app.Activity; import android.app.ProgressDialog; import
android.content.Intent; import android.os.AsyncTask; import android.os.Bundle; import
android.view.View; import android.widget.TextView;
public class Directory_Person extends Activity {
	private TextView name;
	private TextView address;
	private TextView email;
	private String personName;
	private String personAddress;
	private String personEmail;
	@Override
	public void onCreate(Bundle savedInstanceState) {
		super.onCreate(savedInstanceState);
		setContentView(R.layout.directory_person);
		name = (TextView) findViewById(R.id.person_name_text);
		address = (TextView) findViewById(R.id.person_address_text);
		email = (TextView) findViewById(R.id.person_email_text);
		personName = "";
		personAddress = "";
		personEmail = "";

		Intent btn = getIntent();
		String selected_url = btn.getStringExtra("ITEM_LINK");
```

```
                Page_Download task = new Page_Download();
                task.execute(new String[] { selected_url });
        }
        @Override
    public void onBackPressed() {
        finish();
    }
        private class Page_Download extends AsyncTask<String, Void, String[]> {
                private ProgressDialog progressDialog;
                protected void onPreExecute() {
                        progressDialog = ProgressDialog.show(Directory_Person.this, "", "Loading.
Please wait...", true);
                }
                @Override
                protected String[] doInBackground(String... urls) {
                        String[] parameters = new String[] { personName, personAddress,
personEmail };
                        String response = "";
                        for (String url : urls) {
                                DefaultHttpClient client = new DefaultHttpClient();
                                HttpPost httppost = new HttpPost(url);
                                try {
                                        HttpResponse execute = client.execute(httppost);
                                        InputStream content = execute.getEntity().getContent();
BufferedReader buffer = new BufferedReader(new InputStreamReader(content));
                                        String s = "";
                                        while ((s = buffer.readLine()) != null) {
                                                response += s;
                                        }
                                        Document doc = Jsoup.parseBodyFragment(response);
                                        Element body = doc.body();
                                        Element table = body.select("table").get(2);
                                        Element tr = table.select("tbody tr").get(2);
                                        Element td = tr.select("td.text").first();
                                        Element table2 = td.select("table.text tbody").first();
                                        Element nameTR = table2.select("tr").get(1);
                                        Element addressTR = table2.select("tr").get(3);
                                        Element emailTR = table2.select("tr").get(5);
                                        Element name = nameTR.select("td b").first();
                                        Element address = addressTR.select("td").first();
                                        Element email = emailTR.select("td a").first();
                                        parameters[0] = name.text();
                                        parameters[1] = address.text();
                                        parameters[2] = email.text();
                                }
                                catch (Exception e) {
                                        e.printStackTrace();
```

```
                        }
                    }
                    return parameters;
                }
                @Override
                protected void onPostExecute(String[] result) {
                    progressDialog.dismiss();
                    name.setText(result[0]);
                    address.setText(result[1]);
                    email.setText(result[2]);
                }
            }
            public void send_email(View view) {
                Intent email_btn = new Intent(Intent.ACTION_SEND);
                email_btn.setType("plain/text");
                email_btn.putExtra(Intent.EXTRA_EMAIL, new String[]{email.getText().toString()});
                startActivity(email_btn);
            }
            public void main_menu_button_6(View view) {
                Intent btn = new Intent(this, EMUActivity.class);
                btn.setFlags(Intent.FLAG_ACTIVITY_CLEAR_TOP);
                startActivity(btn);
                finish();
            }
}
```

Appendix H: Weekly Schedule code

schedule_login_layout.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android=”http://schemas.android.com/apk/res/android”
android:layout_width="fill_parent" android:layout_height="fill_parent" android:orientation="vertical"
android:background="@drawable/background" >
<TextView android:id="@+id/schedule_login_title" android:layout_width="wrap_content"
android:layout_height="wrap_content" android:layout_centerHorizontal="true"
android:textColor="#448275" android:textSize="25px" android:layout_marginLeft="40px"
android:layout_marginTop="50px" android:text="Weekly Class Schedule" />
<TextView android:id="@+id/user_name" android:layout_width="400px"
android:layout_height="wrap_content" android:textColor="#448275" android:textSize="20px"
android:layout_marginLeft="40px" android:layout_marginTop="100px" android:text="EID:" />
<EditText android:id="@+id/schedule_user" android:layout_width="400px"
android:layout_height="wrap_content" android:layout_marginLeft="40px"
android:layout_marginTop="125px" ></EditText>
<TextView android:id="@+id/pin_number" android:layout_width="400px"
android:layout_height="wrap_content" android:textColor="#448275" android:textSize="20px"
android:layout_marginLeft="40px" android:layout_marginTop="200px" android:text="PIN:" />
<EditText android:id="@+id/schedule_pin" android:layout_width="400px"
android:layout_height="wrap_content" android:layout_marginLeft="40px"
android:layout_marginTop="225px" android:inputType="textPassword" />
<TextView android:id="@+id/error_message_view" android:layout_width="wrap_content"
android:layout_height="wrap_content" android:layout_centerHorizontal="true"
android:textColor="#FFFFFF" android:textSize="20px" android:layout_marginTop="430px"
android:text="" />
<Button android:id="@+id/schedule_login_btn" android:layout_width="200px"
android:layout_height="wrap_content" android:onClick="schedule_login_button"
android:layout_marginLeft="140px" android:layout_marginTop="325px" android:text="Login" />
<Button android:id="@+id/main_menu_btn_1" android:layout_width="match_parent"
android:layout_height="wrap_content" android:layout_alignParentBottom="true"
android:layout_alignParentLeft="true" android:onClick="main_menu_1"
android:background="@drawable/emu_button" />
</RelativeLayout>
```

Schedule_Login

```
package emu.schedule;
import java.io.BufferedReader; import java.io.DataOutputStream; import java.io.IOException; import
java.io.InputStreamReader; import java.net.MalformedURLException; import java.net.URL; import
java.net.URLEncoder; import java.util.ArrayList; import javax.net.ssl.HttpsURLConnection; import
javax.net.ssl.SSLContext; import javax.net.ssl.TrustManager; import javax.net.ssl.X509TrustManager;
```

```
import org.jsoup.Jsoup; import org.jsoup.nodes.Document; import org.jsoup.nodes.Element; import
org.jsoup.select.Elements; import main.menu.EMUActivity; import main.menu.R; import
android.app.Activity; import android.app.ProgressDialog; import android.content.Intent; import
android.os.AsyncTask; import android.os.Bundle; import android.view.View; import
android.widget.EditText; import android.widget.TextView;
public class Schedule_Login extends Activity {
        private EditText schedule_user, schedule_pin;
        private static final String VALDN_ADDR =
"https://bannerweb.emich.edu/pls/berp/twbkwbis.P_ValLogin";
        private static final String SCHED_ADDR =
"https://bannerweb.emich.edu/pls/berp/bwskfshd.P_CrseSchd";
        private static final String LOGIN_COOK = "BW_COOKIE=R2158265430; TESTID=set";
        private URL url;
        private HttpsURLConnection connection;
        private String cookie;
        private ArrayList<String> different_classes;
        private TextView error_message;
   @Override
   public void onCreate(Bundle savedInstanceState) {
      super.onCreate(savedInstanceState);
      setContentView(R.layout.schedule_login_layout);
      schedule_user = (EditText) findViewById(R.id.schedule_user);
      schedule_pin = (EditText) findViewById(R.id.schedule_pin);
      different_classes = new ArrayList<String>();
      error_message = (TextView) findViewById(R.id.error_message_view);
   }
   public void onResume() {
        super.onResume();
        error_message.setText("");
   }
   public void onRestart() {
        super.onRestart();
        error_message.setText("");
   }
        private class Page_Download extends AsyncTask<String, Void, String> {
                private ProgressDialog progressDialog;
                private int login_check;
                protected void onPreExecute() {
                        login_check = 0;
                        progressDialog = ProgressDialog.show(Schedule_Login.this, "", "Loading.
Please wait...", true);
                }
                @Override
                protected String doInBackground(String... urls) {
                        String response = "";
                        String user = schedule_user.getText().toString();
                        String pin = schedule_pin.getText().toString();
```

```
                        for (String url_1 : urls) {
                                try {
                                        response = getSchedule(user, pin, url_1);
                                }
                                catch (Exception e) {
                                        login_check = 1;
                                        e.printStackTrace();
                                }
                        }
                        return response;
                }
                @Override
                protected void onPostExecute(String result) {
                        // Stop the progressDialog
                        progressDialog.dismiss();
                        schedule_view_display(result, login_check);
                }
        }
    private void parseCookie()
    {
            cookie = connection.getHeaderField("Set-Cookie");
    }
   private void login(String userId, String password, String pass_url)  throws MalformedURLException,
IOException {
            url = new URL(pass_url);
            TrustManager[] trustAllCerts = new TrustManager[] { new X509TrustManager() {
           public java.security.cert.X509Certificate[] getAcceptedIssuers() {
              return null;
           }
           public void checkClientTrusted(
                 java.security.cert.X509Certificate[] certs, String authType) {
           }
           public void checkServerTrusted(
                 java.security.cert.X509Certificate[] certs, String authType) {
           } } };
            try {
                        SSLContext sc = SSLContext.getInstance("SSL");
                        sc.init(null, trustAllCerts, new java.security.SecureRandom());
                        HttpsURLConnection.setDefaultSSLSocketFactory(sc.getSocketFactory());
            }
            catch (Exception e) {
            e.printStackTrace();
            }
            connection = (HttpsURLConnection) url.openConnection();
            connection.setRequestMethod("POST");
            connection.setDoInput(true);
            connection.setDoOutput(true);
```

```
		connection.setUseCaches(true);
		connection.setRequestProperty("Cookie", LOGIN_COOK);
		DataOutputStream out = new DataOutputStream(connection.getOutputStream());
		String content = "sid=" + URLEncoder.encode(userId, "UTF-8") + "&pin=" +
URLEncoder.encode(password, "UTF-8");
		out.writeBytes(content);
		out.flush();
		out.close();
		parseCookie();

		BufferedReader input = new BufferedReader(new InputStreamReader(
connection.getInputStream()));
		  input.close();
}
public String getSchedule(String username, String password, String pass_url) throws IOException,
MalformedURLException {
		login(username, password, pass_url);
		url = new URL(SCHED_ADDR);
		connection = (HttpsURLConnection) url.openConnection();
		connection.setRequestMethod("POST");
		connection.setDoInput(true);
		connection.setDoOutput(true);
		connection.setUseCaches(true);
		connection.setRequestProperty("Cookie", cookie);
		connection.setRequestProperty("Content-Type", "text/html");
		DataOutputStream out = new DataOutputStream(connection.getOutputStream());
		out.flush();
		out.close();
		BufferedReader input = new BufferedReader(new InputStreamReader(
connection.getInputStream()));
		String alllines = ""; String line = "";
		while ((line = input.readLine()) != null)
		{
			alllines += line + "\n";
		}
		input.close();
		Document doc = Jsoup.parseBodyFragment(alllines);
		Element body = doc.body();
		Element labels = body.select("table.datadisplaytable").first();
		Elements content = labels.select("td.ddlabel");
		for (Element some_data : content) {
			int check = 0;
			String s = some_data.text();
			if (!s.equals("1am") || !s.equals("2am") || !s.equals("3am") || !s.equals("4am")
					|| !s.equals("5am") || !s.equals("6am") || !s.equals("7am") ||
!s.equals("8am")
```

```
                                || !s.equals("9am") || !s.equals("10am") || !s.equals("11am") ||
!s.equals("12pm")
                                || !s.equals("1pm") || !s.equals("2pm") || !s.equals("3pm") ||
!s.equals("4pm")
                           || !s.equals("5pm") || !s.equals("6pm") || !s.equals("7pm") || !s.equals("8pm")
                           || !s.equals("9pm") || !s.equals("10pm") || !s.equals("11pm") ||
!s.equals("12am")) {
                        for (String test : different_classes) {
                                if (test.equals(s))
                                        check = 1;
                                }
                                if (check == 0) {
                                        different_classes.add(s);
                                }
                        }
                }
                alllines = labels.toString();
                return alllines;
        }
   @Override
    public void onBackPressed() {
        finish();
    }
    public void main_menu_1(View view) {
        Intent btn = new Intent(this, EMUActivity.class);
        btn.setFlags(Intent.FLAG_ACTIVITY_CLEAR_TOP);
        startActivity(btn);
    }
    public void schedule_login_button(View view) {
                Page_Download task = new Page_Download();
                task.execute(new String[] { VALDN_ADDR });
    }
    public void schedule_view_display(String a, int check) {
        if (check == 0) {
                Intent btn = new Intent(this, Schedule_View.class);
                btn.putExtra("HTML_CODE", a);
                btn.putStringArrayListExtra("INDIVIDUAL_CLASSES", different_classes);
                startActivity(btn);
        }
        else {
                error_message.setText("Error processing your request.\n" + "Verify that:\n" + "- you
are connected to the Internet,\n" + "- you are a student,\n" + "- your credentials are correct.");
        }
    }
}
```

schedule_view_success.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android=”http://schemas.android.com/apk/res/android”
android:background="@drawable/background" android:layout_width="fill_parent"
android:layout_height="fill_parent" android:orientation="vertical" >
<TextView android:layout_width="wrap_content" android:layout_height="wrap_content"
android:textColor="#448275" android:textSize="25px" android:layout_centerHorizontal="true"
android:layout_marginTop="25px" android:text="Weekly Class Schedule" />
<WebView android:id="@+id/schedule_calendar_view" android:layout_width="match_parent"
android:layout_height="500px" android:layout_alignParentLeft="true"
android:layout_above="@+id/schedule_locations_btn" />
<Button android:id="@+id/schedule_locations_btn" android:layout_width="match_parent"
android:layout_height="wrap_content" android:layout_centerHorizontal="true"
android:onClick="schedule_locations_button" android:layout_above="@+id/main_menu_btn_2"
android:text="Get Directions" />
<Button android:id="@+id/main_menu_btn_2" android:layout_width="match_parent"
android:layout_height="wrap_content" android:layout_alignParentBottom="true"
android:layout_alignParentLeft="true" android:onClick="main_menu_2"
android:background="@drawable/emu_button" />
</RelativeLayout>
```

Schedule_View

```
package emu.schedule;
import java.util.ArrayList; import main.menu.EMUActivity; import main.menu.R;
import android.app.Activity; import android.content.Intent; import android.os.Bundle;
import android.view.View; import android.webkit.WebView; import android.webkit.WebViewClient;
public class Schedule_View extends Activity {
        private WebView Schedule_Calendar_WebView;
        private String schedule_code;
        private ArrayList<String> names;
        private ArrayList<String> names_for_addresses;
        @Override
         public void onCreate(Bundle savedInstanceState) {
                 super.onCreate(savedInstanceState);
                setContentView(R.layout.schedule_view_success);
                 Intent btn = getIntent();
                schedule_code = btn.getStringExtra("HTML_CODE");
                names = btn.getStringArrayListExtra("INDIVIDUAL_CLASSES");
                names_for_addresses = new ArrayList<String>();
                for (String name : names) {
                        String s = ""; int index = 0; int char_at = 0;
                        for (int i=0; i<name.length(); i++) {
```

```
                        if (name.charAt(i) == ' ')
                                char_at++;
                        if (char_at == 6)
                                index = i+2;
                }
                s = name.substring(index);
                names_for_addresses.add(s);
        }
        Schedule_Calendar_WebView = (WebView)
findViewById(R.id.schedule_calendar_view);
        Schedule_Calendar_WebView.getSettings().setJavaScriptEnabled(true);
    Schedule_Calendar_WebView.getSettings().setJavaScriptCanOpenWindowsAutomatically(fals
e);
    Schedule_Calendar_WebView.loadDataWithBaseURL("",schedule_code,"text/html","utf-8","");
        Schedule_Calendar_WebView.setWebViewClient(new WebViewClient() {
    @Override public boolean shouldOverrideUrlLoading(WebView view, String url) {
        return true;
    }
  });
  }
  @Override
  public void onBackPressed() {
        finish();
  }
  public void main_menu_2(View view) {
        Intent btn = new Intent(this, EMUActivity.class);
        btn.setFlags(Intent.FLAG_ACTIVITY_CLEAR_TOP);
        startActivity(btn);
        finish();
  }
  public void schedule_locations_button(View view) {
        Intent btn = new Intent(this, Schedule_Locations.class);
        btn.putStringArrayListExtra("ADDRESSES", names_for_addresses);
        startActivity(btn);
  }
}
```

schedule_locations.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android=”http://schemas.android.com/apk/res/android”
android:orientation="vertical" android:layout_width="fill_parent"  android:layout_height="fill_parent"
android:background="@drawable/background"  android:padding="6dip" >
<TextView android:id="@+id/schedule_locations_title" android:layout_width="wrap_content"
android:layout_height="wrap_content" android:layout_centerHorizontal="true"
```

```
android:layout_marginTop="40px" android:text="Directions to Classes" android:textColor="#448275"
android:textSize="25px" />
<ListView android:id="@+id/schedule_locations_items" android:layout_width="match_parent"
android:layout_height="wrap_content" android:layout_above="@+id/main_menu_btn_18"
android:layout_marginTop="70px" android:layout_alignLeft="@+id/main_menu_btn_18"
android:cacheColorHint="#00000000" android:background="@drawable/background"
android:layout_below="@+id/schedule_locations_title" ></ListView>
<Button android:id="@+id/main_menu_btn_18" android:layout_width="match_parent"
android:layout_height="wrap_content" android:layout_alignParentBottom="true"
android:layout_alignParentLeft="true" android:onClick="main_menu_18"
android:background="@drawable/emu_button" />
</RelativeLayout>
```

schedule_locations_item.xml

```
<TextView xmlns:android=”http://schemas.android.com/apk/res/android”
android:id="@+id/schedule_item_line" android:layout_width="fill_parent"
android:layout_height="80px" android:textSize="25px" android:textColor="#FFFFFF"
android:background="@drawable/background" android:gravity="center_vertical"
android:singleLine="false" />
```

Schedule_Locations

```
package emu.schedule;
import java.util.ArrayList; import main.menu.EMUActivity; import main.menu.R; import
android.app.Activity; import android.content.Intent; import android.os.Bundle; import android.view.View;
import android.widget.AdapterView; import android.widget.ArrayAdapter; import
android.widget.ListView; import android.widget.AdapterView.OnItemClickListener;
public class Schedule_Locations extends Activity {
        private final String[] buildings =
{"ALEXAN","BEST","BOONE","BOWEN","BRIGGS","CTRLST","DC-1", "FLETCH",
"FORD","HALLE","KING","MARKJ","MARSHL","MCKENN","OWEN","PORTER", "PRAY-H",
"PSYCLN","QUIRK","RACKHM","RECIM","ROOSEV","SCHLHS",
"SCIENC","SCULPT","SHERZ","SILL","STRONG","WALTON","WARNER"};
        private final String[] latitude =
{"42.249582","42.250615","42.245993","42.249662","42.246858","42.255546","42.250829","42.254030
","42.246358","42.248693","42.247406","42.247271","42.248034","42.246151","42.241394","42.248899
","42.249551","42.245294","42.249463","42.248510","42.250226","42.247668","42.248226","42.247271
","42.254824","42.247073","42.248685","42.247875","42.252052","42.249606"};
        private final String[] longitude = {"-83.620226","-83.621428","-83.622801","-83.625827","-
83.625741","-83.631492","-83.622254","-83.634753","-83.623756","-83.627522","-83.623885","-
83.626299","-83.623928","-83.625677","-83.616675","-83.623745","-83.622608","-83.621793","-
83.621503","-83.625226","-83.625076","-83.622619","-83.626826","-83.626299","-83.631567","-
83.624818","-83.620462","-83.626170","-83.622254","-83.625119",};
```

```
	private ListView individual_locations;
	private ArrayList<String> addresses;
	private String[] list_for_addresses;
	@Override
	public void onCreate(Bundle savedInstanceState) {
		super.onCreate(savedInstanceState);
		setContentView(R.layout.schedule_locations);
		Intent btn = getIntent();
		 addresses = btn.getStringArrayListExtra("ADDRESSES");
		int size = addresses.size();
		list_for_addresses = new String[size];
		int i=0;
		try {
			for (String a : addresses) {
				list_for_addresses[i] = a;
				i++;
			}
		}
		catch (Exception e) {
			e.printStackTrace();
		 }
		individual_locations = (ListView) findViewById(R.id.schedule_locations_items);
		ArrayAdapter<String> adapter = new ArrayAdapter<String>(this,
R.layout.schedule_location_item, addresses);
		individual_locations.setAdapter(adapter);
		individual_locations.setOnItemClickListener(new OnItemClickListener() {
		public void onItemClick(AdapterView<?> parent, View view, int position, long id) {
			String test = list_for_addresses[position];
			int index = 0;
			for (int j=0; j<test.length(); j++){
				if (test.charAt(j) == ' ') {
					index = j;
					break;
				}
			}
			String building_code = test.substring(0,index);
			int check = 0;
			int index_check = 0;
			for (int t=0; t<30; t++){
				if (building_code.equals(buildings[t])) {
					check = 1;
					index_check = t;
					break;
				}
			}
				goIntent(check, index_check);
			}
```

```
                });
        }
        @Override
         public void onBackPressed() {
                finish();
         }
        public void goIntent(int ind_1, int ind_2) {
                Intent btn = new Intent(this, Schedule_Directions.class);
                if (ind_1 == 1) {
                        btn.putExtra("VALID", ind_1);
                        btn.putExtra("BLDG", buildings[ind_2]);
                        btn.putExtra("LAT", latitude[ind_2]);
                        btn.putExtra("LONG", longitude[ind_2]);
                }
                else
                        btn.putExtra("VALID", ind_1);
                startActivity(btn);
        }
        public void main_menu_18(View view) {
                Intent btn = new Intent(this, EMUActivity.class);
                btn.setFlags(Intent.FLAG_ACTIVITY_CLEAR_TOP);
                startActivity(btn);
        }
}
```

schedule_directions.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android=”http://schemas.android.com/apk/res/android”
android:layout_width="match_parent" android:layout_height="match_parent"
android:background="@drawable/background" android:orientation="vertical" >
<TextView android:id="@+id/schedule_directions_top" android:layout_width="wrap_content"
android:layout_height="wrap_content" android:textColor="#448275" android:layout_marginTop="40px"
android:layout_centerHorizontal="true" android:textSize="25px" android:text="Directions to" />
<WebView android:id="@+id/schedule_directions_view" android:layout_width="match_parent"
android:layout_height="550px" android:layout_above="@+id/main_menu_btn_19"
android:layout_alignParentLeft="true" />
<Button android:id="@+id/main_menu_btn_19" android:layout_width="match_parent"
android:layout_height="wrap_content" android:layout_alignParentBottom="true"
android:layout_alignParentLeft="true" android:background="@drawable/emu_button"
android:onClick="main_menu_button_19" />
</RelativeLayout>
```

Schedule_Directions

```
package emu.schedule;
import main.menu.EMUActivity; import main.menu.R; import android.app.Activity; import
android.content.Context; import android.content.Intent; import android.location.Criteria;
import android.location.Location; import android.location.LocationListener; import
android.location.LocationManager; import android.os.Bundle; import android.view.View; import
android.webkit.WebView; import android.webkit.WebViewClient; import android.widget.TextView;
public class Schedule_Directions extends Activity implements LocationListener {
	private TextView text_test;
	private WebView schedule_directions_map;
	private LocationManager locationManager;
	private String from, provider;
	@Override
	public void onCreate(Bundle savedInstanceState) {
		super.onCreate(savedInstanceState);
		setContentView(R.layout.schedule_directions);
		locationManager = (LocationManager)
getSystemService(Context.LOCATION_SERVICE);
		Criteria criteria = new Criteria();
		provider = locationManager.getBestProvider(criteria, false);
		Location location = locationManager.getLastKnownLocation(provider);
		 if (location != null) {
		double lat = (double) (location.getLatitude());
		double lng = (double) (location.getLongitude());
		from = ""+lat+","+lng;
		} else {
			from = "0.0,0.0";
		}
		text_test = (TextView) findViewById(R.id.schedule_directions_top);
		 Intent btn = getIntent();
		 int check = btn.getIntExtra("VALID", 0);
		 if (check == 0) {
			text_test.setText("Wrong query.");
		}
		else {
			String url = "http://maps.google.com/maps?saddr=" + from + "&daddr=" +
btn.getStringExtra("LAT") + "," + btn.getStringExtra("LONG");
			text_test.setText("Directions to: " + btn.getStringExtra("BLDG"));
			schedule_directions_map = (WebView)
findViewById(R.id.schedule_directions_view);
			schedule_directions_map.getSettings().setJavaScriptEnabled(true);
 schedule_directions_map.getSettings().setJavaScriptCanOpenWindowsAutomatically(false);
			schedule_directions_map.loadUrl(url);
			schedule_directions_map.setWebViewClient(new WebViewClient() {
```

```
                        @Override public boolean shouldOverrideUrlLoading(WebView view, String
url) {
                                // returns true to handle the click yourself
                                return true;
                        } });
        }
}
@Override
public void onBackPressed() {
            finish();
}
@Override
protected void onResume() {
        locationManager.requestLocationUpdates(provider, 500, 1, this);
        super.onResume();
}
/* Remove the locationlistener updates when Activity is paused */
@Override
protected void onPause() {
        super.onPause();
        locationManager.removeUpdates(this);
}
public void onLocationChanged(Location location) {
        double lat = (double) (location.getLatitude());
        double lng = (double) (location.getLongitude());
        from =  ""+lat+","+lng;
}
public void onProviderDisabled(String provider) { }
public void onProviderEnabled(String provider) { }

public void onStatusChanged(String provider, int status, Bundle extras) { }
public void main_menu_button_19(View view) {
            Intent btn = new Intent(this, EMUActivity.class);
            btn.setFlags(Intent.FLAG_ACTIVITY_CLEAR_TOP);
            startActivity(btn);
            finish();
} }
```

Appendix I: Course Lookup code

classes_login.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
android:layout_width="fill_parent" android:layout_height="fill_parent" android:orientation="vertical"
android:background="@drawable/background" >
<TextView android:id="@+id/classes_login_title" android:layout_width="wrap_content"
android:layout_height="wrap_content" android:layout_centerHorizontal="true"
android:textColor="#448275" android:textSize="25px" android:layout_marginLeft="40px"
android:layout_marginTop="50px" android:text="Login" />
<TextView android:id="@+id/user_name_2" android:layout_width="400px"
android:layout_height="wrap_content" android:textColor="#448275" android:textSize="20px"
android:layout_marginLeft="40px" android:layout_marginTop="100px" android:text="EID:" />
<EditText android:id="@+id/classes_user" android:layout_width="400px"
android:layout_height="wrap_content" android:layout_marginLeft="40px"
android:layout_marginTop="125px" ></EditText>
<TextView android:id="@+id/pin_number_2" android:layout_width="400px"
android:layout_height="wrap_content" android:textColor="#448275" android:textSize="20px"
android:layout_marginLeft="40px" android:layout_marginTop="200px" android:text="PIN:" />
<EditText android:id="@+id/classes_pin" android:layout_width="400px"
android:layout_height="wrap_content" android:layout_marginLeft="40px"
android:layout_marginTop="225px" android:inputType="textPassword" />
<TextView android:id="@+id/error_message_view_4" android:layout_width="wrap_content"
android:layout_height="wrap_content" android:layout_centerHorizontal="true"
android:textColor="#FFFFFF" android:textSize="20px" android:layout_marginTop="430px"
android:text="" />
<Button android:id="@+id/classes_login_btn" android:layout_width="200px"
android:layout_height="wrap_content" android:onClick="classes_login_button"
android:layout_marginLeft="140px" android:layout_marginTop="325px" android:text="Login" />
<Button android:id="@+id/main_menu_btn_23" android:layout_width="match_parent"
android:layout_height="wrap_content" android:layout_alignParentBottom="true"
android:layout_alignParentLeft="true" android:onClick="main_menu_23"
android:background="@drawable/emu_button" />
</RelativeLayout>
```

Classes_Login

```
package emu.classes;
import java.io.BufferedReader; import java.io.DataOutputStream; import java.io.IOException; import
java.io.InputStreamReader; import java.net.MalformedURLException; import java.net.URL; import
java.net.URLEncoder; import javax.net.ssl.HttpsURLConnection; import javax.net.ssl.SSLContext;
import javax.net.ssl.TrustManager; import javax.net.ssl.X509TrustManager; import org.jsoup.Jsoup;
```

```
import org.jsoup.nodes.Document; import org.jsoup.nodes.Element; import main.menu.EMUActivity;
import main.menu.R; import android.app.Activity; import android.app.ProgressDialog; import
android.content.Intent; import android.os.AsyncTask; import android.os.Bundle; import
android.view.View; import android.widget.EditText; import android.widget.TextView;
public class Classes_Login extends Activity {
	private EditText us, pn;
	private String user, pin;
	private static final String VALDN_ADDR =
"https://bannerweb.emich.edu/pls/berp/twbkwbis.P_ValLogin";
	private static final String LOGIN_COOK = "BW_COOKIE=R2158265430; TESTID=set";
	private URL url;
	private HttpsURLConnection connection;
	private String cookie;
	private TextView error_message;
	private int login_check, activity_check;
	@Override
	public void onCreate(Bundle savedInstanceState) {
		super.onCreate(savedInstanceState);
		setContentView(R.layout.classes_login);
		Intent btn = getIntent();
		activity_check = btn.getIntExtra("ACTIVITY_CHECK", 1);

		us = (EditText) findViewById(R.id.classes_user);
		pn = (EditText) findViewById(R.id.classes_pin);
		error_message = (TextView) findViewById(R.id.error_message_view_4);
		login_check = 0;
	}
public void onResume() {
	super.onResume();
	error_message.setText("");
	login_check = 0;
   }
   public void onRestart() {
	super.onRestart();
	error_message.setText("");
	login_check = 0;
   }
	private class Page_Download extends AsyncTask<String, Void, String> {
		private ProgressDialog progressDialog;
		protected void onPreExecute() {
			login_check = 0;
			progressDialog = ProgressDialog.show(Classes_Login.this, "", "Loading.
Please wait...", true);
		}
		@Override
		protected String doInBackground(String... urls) {
			String response = "";
```

```
                        String user = us.getText().toString();
                        String pin = pn.getText().toString();
                        for (String url : urls) {
                                try {
                                        login(user, pin, url);
                                }
                                catch (Exception e) {
                                        login_check = 1;
                                        e.printStackTrace();
                                }
                        }
                        return response;
                }
                @Override
                protected void onPostExecute(String result) {
                        progressDialog.dismiss();
                        startIntent(login_check);
                }
        }
        private void parseCookie()
        {
                cookie = connection.getHeaderField("Set-Cookie");
        }
        private void login(String userId, String password, String url_page)  throws
MalformedURLException, IOException {
                url = new URL(url_page);
                TrustManager[] trustAllCerts = new TrustManager[] { new X509TrustManager() {
      public java.security.cert.X509Certificate[] getAcceptedIssuers() {
         return null;
      }
      public void checkClientTrusted(
            java.security.cert.X509Certificate[] certs, String authType) {
      }
      public void checkServerTrusted(
            java.security.cert.X509Certificate[] certs, String authType) {
      }
   } };
                try {
                        SSLContext sc = SSLContext.getInstance("SSL");
                        sc.init(null, trustAllCerts, new java.security.SecureRandom());
       HttpsURLConnection.setDefaultSSLSocketFactory(sc.getSocketFactory());
                }
                catch (Exception e) {
      e.printStackTrace();
   }
                connection = (HttpsURLConnection) url.openConnection();
                connection.setRequestMethod("POST");
```

```
                connection.setDoInput(true);
                connection.setDoOutput(true);
                connection.setUseCaches(true);
                connection.setRequestProperty("Cookie", LOGIN_COOK);
                DataOutputStream out = new DataOutputStream(connection.getOutputStream());
                String content = "sid=" + URLEncoder.encode(userId, "UTF-8") + "&pin=" +
URLEncoder.encode(password, "UTF-8");
                out.writeBytes(content);
                out.flush();
                out.close();
                parseCookie();
                BufferedReader input = new BufferedReader(new InputStreamReader(
connection.getInputStream()));
                String alllines = "";
                String line = "";
                while ((line = input.readLine()) != null)
                {
                        alllines += line + "\n";
                }
                input.close();
                try {
                        Document doc = Jsoup.parseBodyFragment(alllines);
                        Element body = doc.body();
                        Element label = body.select("table.plaintable").get(2);
                        if (label.text().equals("Authorization Failure - Invalid User ID or PIN.")) {
                                login_check = 1;
                        }
                        else
                                login_check = 0;
                }
                catch (Exception e){
                        login_check = 0;
                }
        }
        @Override
   public void onBackPressed() {
        finish();
   }
        public void classes_login_button(View view) {
                String web = VALDN_ADDR;
                Page_Download task = new Page_Download();
                task.execute(new String[] { web });
        }
        public void startIntent(int check) {
                if (check == 0)
                {
                        Intent search_classes;
```

```
                        if (activity_check == 1) {
                                search_classes = new Intent(this, Classes_Search.class);
                        }
                        else {
                                search_classes = new Intent(this, Classes_Terms.class);
                        }
                        search_classes.putExtra("Cookie_Value", cookie);
                        search_classes.putExtra("USER", user);
                        search_classes.putExtra("PIN", pin);
                        startActivity(search_classes);
                }
                else
                        error_message.setText("Invalid credentials, please try again.");
        }
        public void main_menu_23(View view) {
        Intent btn = new Intent(this, EMUActivity.class);
        btn.setFlags(Intent.FLAG_ACTIVITY_CLEAR_TOP);
        startActivity(btn);
        finish();
   }
}
```

classes_search.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
android:layout_width="fill_parent" android:layout_height="fill_parent" android:orientation="vertical"
android:background="@drawable/background" >
<TextView android:id="@+id/classes_search_title" android:layout_width="wrap_content"
android:layout_height="wrap_content" android:layout_centerHorizontal="true"
android:textColor="#448275" android:textSize="25px" android:layout_marginLeft="40px"
android:layout_marginTop="50px" android:text="Search Available Classes" />
<TextView android:id="@+id/classes_search_item_term" android:layout_width="400px"
android:layout_height="wrap_content" android:textColor="#448275" android:textSize="20px"
android:layout_marginLeft="40px" android:layout_marginTop="100px" android:text="Term" />
<Spinner android:id="@+id/classes_search_term" android:layout_width="400px"
android:layout_height="wrap_content" android:layout_marginLeft="40px"
android:layout_marginTop="130px" />
<TextView android:id="@+id/classes_search_item_group" android:layout_width="400px"
android:layout_height="wrap_content" android:textColor="#448275" android:textSize="20px"
android:layout_marginLeft="40px" android:layout_marginTop="200px" android:text="Class Subject" />
<Spinner android:id="@+id/classes_search_subject" android:layout_width="400px"
android:layout_height="wrap_content" android:layout_marginLeft="40px"
android:layout_marginTop="230px" />
```

```
<TextView android:id="@+id/classes_search_item_title" android:layout_width="400px"
android:layout_height="wrap_content" android:textColor="#448275" android:textSize="20px"
android:layout_marginLeft="40px" android:layout_marginTop="300px" android:text="Class Title" />
<EditText android:id="@+id/classes_search_item" android:layout_width="400px"
android:layout_height="wrap_content" android:layout_marginLeft="40px"
android:layout_marginTop="330px" />
<TextView android:id="@+id/error_message_view_3" android:layout_width="wrap_content"
android:layout_height="wrap_content" android:layout_centerHorizontal="true"
android:textColor="#FFFFFF" android:textSize="20px" android:layout_marginTop="530px"
android:text="" />
<Button android:id="@+id/classes_search_btn" android:layout_width="200px"
android:layout_height="wrap_content" android:onClick="classes_search_button"
android:layout_marginLeft="140px" android:layout_marginTop="425px" android:text="Search" />
<Button android:id="@+id/main_menu_btn_24" android:layout_width="match_parent"
android:layout_height="wrap_content" android:layout_alignParentBottom="true"
android:layout_alignParentLeft="true" android:onClick="main_menu_24"
android:background="@drawable/emu_button" />
</RelativeLayout>
```

Classes_Search

```
package emu.classes;
import java.io.BufferedReader; import java.io.DataOutputStream; import java.io.IOException; import
java.io.InputStreamReader; import java.net.MalformedURLException; import java.net.URL; import
java.util.Calendar; import java.util.GregorianCalendar; import javax.net.ssl.HttpsURLConnection; import
main.menu.EMUActivity; import main.menu.R; import org.jsoup.Jsoup; import
org.jsoup.nodes.Document; import org.jsoup.nodes.Element; import android.app.Activity; import
android.app.ProgressDialog; import android.content.Intent; import android.os.AsyncTask; import
android.os.Bundle; import android.view.View; import android.widget.AdapterView; import
android.widget.ArrayAdapter; import android.widget.EditText; import android.widget.Spinner;
public class Classes_Search extends Activity implements AdapterView.OnItemSelectedListener {
        private String user, pin, cookie, search_title, search_term, search_subject;
        private URL url_address;
        private HttpsURLConnection conn;
        private EditText class_title;
        private Spinner class_term, class_subject;
        private String[] term_values;
        private String[] term_strings;
        public int[] spinner_ids;
        public int spinner_check;
        public int spinner_index;
        private final String[] class_subjects =
        {"ACC","AAS","ANTH","ATM","AMUS","ARTE","ARTH","ARTS","AHPR","ASTR","ATTR","ATHL
","SPAI","AVT","BIO","BIOT","BMMT",
        "BEDU","CTAA","CTAC","CTAO","CTAT","CTAR","CTWE","CHEM","CHL","CHNE","CLSC","C
```

```
LRA","CASI","COB","COT","CMT",
        "CADM","CAE","CET","COSC","CNST","COUN","CSIE","CRTW","CRM","CURR","DANC","DS"
,"DTC","ECE","ESSC","ECON","EDLD",
        "EDMT","EDPS","EDST","ELEC","EM","ET","ENGL","ESLN","ENVI","FETE","FIN","FLAN","FR
NH","GEOG","GERN","GERT","GREK",
        "IHHS","HLAD","HLED","HPHP","GHPR","HIST","HRM","ID","IA","IS","IMC","IDE","IB","JPNE","
JRNL","LNGE","LATN","LAW","LEGL","LING",
        "LITR","MGMT","MKTG","MATH","MET","MSL","MUSC","NURS","OCTH","OM","ORPR","PHIL"
,"PHED","PEGN","PSCI","PHY","PLSC",
        "PC","PRCT","PDD","PSY","QUAL","RDNG","RECR","SET","STS","SAG","SOFD","SWRK","S
OCL","SPNH","SPCI","SPHI","SPEI","SPGN",
        "SPLI","SPPI","SPSI","SPVI","SMGT","SPMD","EDUC","SABR","SCM","TSLN","TEDU","TM","
TS","THRC","UNIV","GPLN","URED","WGST"};
        @Override
        public void onCreate(Bundle savedInstanceState) {
                super.onCreate(savedInstanceState);
                setContentView(R.layout.classes_search);
                Intent search_classes = getIntent();
                user = search_classes.getStringExtra("USER");
                pin = search_classes.getStringExtra("PIN");
                cookie = search_classes.getStringExtra("Cookie_Value");
                set_term_values();
                spinner_ids = new int[2];
                spinner_check = 0;
                spinner_index = 0;
                class_title = (EditText) findViewById(R.id.classes_search_item);
                class_subject = (Spinner) findViewById(R.id.classes_search_subject);
                class_subject.setOnItemSelectedListener(this);
                class_term = (Spinner) findViewById(R.id.classes_search_term);
                class_term.setOnItemSelectedListener(this);
                ArrayAdapter<String> srch_term = new ArrayAdapter<String>(this,
android.R.layout.simple_spinner_item, term_strings);
                ArrayAdapter<String> srch_subj = new ArrayAdapter<String>(this,
android.R.layout.simple_spinner_item, class_subjects);
srch_term.setDropDownViewResource(android.R.layout.simple_spinner_dropdown_item);
srch_subj.setDropDownViewResource(android.R.layout.simple_spinner_dropdown_item);
      class_term.setAdapter(srch_term);
      class_subject.setAdapter(srch_subj);
          }
          public void onResume() {
          super.onResume();
          spinner_ids = new int[2];
          spinner_check = 0;
          spinner_index = 0;
   }
   public void onRestart() {
          super.onRestart();
```

```
		spinner_ids = new int[2];
		spinner_check = 0;
		spinner_index = 0;
	}
	@Override
	public void onBackPressed() {
		finish();
	}
	public void onItemSelected(AdapterView<?> parent, View v, int position, long id) {
		int b = parent.getId();
		if (spinner_index < 2) {
			spinner_ids[spinner_index] = b;
			spinner_index++;
		}
		setSpinner(b, position);
	}
	public void onNothingSelected(AdapterView<?> parent) {
	}
	public void setSpinner(int spin_id, int value)
	{
		if (spin_id == spinner_ids[0])
			search_term = term_values[value];
		else if (spin_id == spinner_ids[1])
			search_subject = class_subjects[value];
	}
		private class Page_Download extends AsyncTask<String, Void, String> {
			private ProgressDialog progressDialog;
			protected void onPreExecute() {
				progressDialog = ProgressDialog.show(Classes_Search.this, "", "Loading.
Please wait...", true);
			}
			@Override
			protected String doInBackground(String... urls) {
				String response = "";
				for (String url : urls) {
					try {
						search_title = class_title.getText().toString();
						url += "&term_in=" + search_term + "&sel_title=" +
search_title + "&sel_subj=" + search_subject;
						response = getClassSearchResults(user, pin, url);
					}
					catch (Exception e) {
						e.printStackTrace();
					}
				}
				return response;
			}
```

```
		@Override
		protected void onPostExecute(String result) {
			progressDialog.dismiss();
			startIntent(result);
		}
	}
	public String getClassSearchResults(String username, String password, String address)
throws IOException, MalformedURLException {
		url_address = new URL(address);
		conn = (HttpsURLConnection) url_address.openConnection();

		conn.setRequestMethod("POST");
		conn.setDoInput(true);
		conn.setDoOutput(true);
		conn.setUseCaches(true);
		conn.setRequestProperty("Cookie", cookie);
		conn.setRequestProperty("Content-Type", "text/html");
		DataOutputStream out = new DataOutputStream(conn.getOutputStream());
	out.flush();
		out.close();
		BufferedReader input = new BufferedReader(new
InputStreamReader(conn.getInputStream()));
		String alllines = "";
		String line = "";
		while ((line = input.readLine()) != null)
		{
			alllines += line + "\n";
		}
		input.close();
		Document doc = Jsoup.parseBodyFragment(alllines);
		Element body = doc.body();
		Element labels =  body.select("table.datadisplaytable").first();
		alllines = labels.toString();
		return alllines;
	}
	public void classes_search_button(View view) {
		String web = "https://bannerweb.emich.edu/pls/berp/bwskfcls.P_GetCrse?" +
			"sel_subj=dummy&sel_day=dummy&sel_schd=dummy&" + ""
"sel_schd=%25&sel_insm=dummy&sel_camp=dummy&sel_camp=%25&sel_levl=dummy&" + "" +

"sel_sess=dummy&sel_instr=dummy&sel_instr=%25&sel_ptrm=dummy&sel_ptrm=%25&" + "" +
				"sel_attr=dummy&sel_attr=%25&sel_crse=&sel_from_cred=&" + "" +

"sel_to_cred=&begin_hh=0&begin_mi=0&begin_ap=a&end_hh=0&end_mi=0&end_ap=a";
		Page_Download task = new Page_Download();
		task.execute(new String[] { web });
	}
```

```
public void startIntent(String res) {
	Intent data_sent = new Intent(this, Classes_View.class);
	data_sent.putExtra("data", res);
	startActivity(data_sent);
}
public void set_term_values() {
	GregorianCalendar today = new GregorianCalendar();
	int month = today.get(Calendar.MONTH) + 1;
	int year = today.get(Calendar.YEAR);
	if ((month == 1) || (month == 2) || (month == 11) || (month == 12)) {
		term_values = new String[2];
		term_strings = new String[2];
		if (month == 1 || month == 2) {
			String FALL = year + "10";
			term_values[0] = FALL;
			FALL = "Fall " + (year-1);
			term_strings[0] = FALL;
			String WINTER = year + "20";
			term_values[1] = WINTER;
			WINTER = "Winter" + year;
			term_strings[1] = WINTER;
		}
		else {
			String FALL = year + "10";
			term_values[1] = FALL;
			FALL = "Fall " + year;
			term_strings[1] = FALL;
			String WINTER = year + "20";
			term_values[0] = WINTER;
			WINTER = "Winter " + (year + 1);
			term_strings[0] = WINTER;
		}
	}
	else {
		term_values = new String[4];
		String WINTER = year + "20";
		term_values[3] = WINTER;
		WINTER = "Winter " + year;
		term_strings[3] = WINTER;
		String SPRING = year + "30";
		term_values[2] = SPRING;
		SPRING = "Spring " + year;
		term_strings[2] = SPRING;
		String SUMMER = year + "40";
		term_values[1] = SUMMER;
		SUMMER = "Summer " + year;
		term_strings[1] = SUMMER;
```

```
                        String FALL = (year + 1) + "10";
                        term_values[0] = FALL;
                        FALL = "Fall " + year;
                        term_strings[0] = FALL;
                }
        }
        public void main_menu_24(View view) {
        Intent btn = new Intent(this, EMUActivity.class);
        btn.setFlags(Intent.FLAG_ACTIVITY_CLEAR_TOP);
        startActivity(btn);
        finish();
   }
}
```

classes_view.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
android:orientation="vertical" android:layout_width="fill_parent" android:layout_height="fill_parent"
android:background="@drawable/background"  android:padding="6dip" >
<TextView android:id="@+id/classes_view_title" android:layout_width="wrap_content"
android:layout_height="wrap_content" android:layout_centerHorizontal="true"
android:layout_marginTop="40px" android:text="Available Classes" android:textColor="#448275"
android:textSize="25px" />
<ListView android:id="@+id/classes_view_items" android:layout_width="match_parent"
android:layout_height="wrap_content" android:layout_above="@+id/main_menu_btn_25"
android:layout_marginTop="70px" android:layout_alignLeft="@+id/main_menu_btn_25"
android:cacheColorHint="#00000000" android:background="@drawable/background"
android:layout_below="@+id/classes_view_title" ></ListView>
<Button android:id="@+id/main_menu_btn_25" android:layout_width="match_parent"
android:layout_height="wrap_content" android:layout_alignParentBottom="true"
android:layout_alignParentLeft="true" android:onClick="main_menu_25"
android:background="@drawable/emu_button" />
</RelativeLayout>
```

classes_item.xml

```
<TextView xmlns:android="http://schemas.android.com/apk/res/android"
android:id="@+id/classes_item_line" android:layout_width="fill_parent"  android:layout_height="180px"
android:textColor="#FFFFFF"
android:background="@drawable/background" android:padding="5px"
android:singleLine="false" />
```

Classes_View

```
package emu.classes;
import java.util.ArrayList; import main.menu.EMUActivity; import main.menu.R; import org.jsoup.Jsoup;
import org.jsoup.nodes.Document; import org.jsoup.nodes.Element; import org.jsoup.select.Elements;
import android.app.Activity; import android.content.Context; import android.content.Intent; import
android.os.Bundle; import android.view.View; import android.widget.ArrayAdapter; import
android.widget.ListView;
public class Classes_View extends Activity {
	private ListView classes_results;
	private ArrayAdapter<String> adapter;
	private ArrayList<String> class_search_results_items = new ArrayList<String>();
	private Context myContext;
	public void onCreate(Bundle savedInstanceState) {
		super.onCreate(savedInstanceState);
		setContentView(R.layout.classes_view);
		Intent btn = getIntent();
		String data_received = btn.getStringExtra("data");
		myContext = this;
		classes_results = (ListView) findViewById(R.id.classes_view_items);
		Document doc = Jsoup.parseBodyFragment(data_received);
		Elements items = doc.select("tr");
		int i = 0;
		 for (Element item : items) {
			if (i == 0 || i==1) {
			i++;
			}
			else {
				String CRN = "";
				String subj = "";
				String crse = "";
				String cred = "";
				String title = "";
				String days = "";
				String time = "";
				String avail = "";
				String instr = "";
				String date = "";
				String loc = "";

				Elements class_data = item.select("td");
				for (int j=0; j<23; j++) {
					if (j==1)
						CRN = class_data.get(j).text();
					else if (j==2)
						subj = class_data.get(j).text();
```

```
                                        else if (j==3)
                                                crse = class_data.get(j).text();
                                        else if (j==6)
                                                cred = class_data.get(j).text();
                                        else if (j==7)
                                                title = class_data.get(j).text();
                                        else if (j==8)
                                                days = class_data.get(j).text();
                                        else if (j==9)
                                                time = class_data.get(j).text();
                                        else if (j==12)
                                                avail = class_data.get(j).text();
                                        else if (j==19)
                                                instr = class_data.get(j).text();
                                        else if (j==20)
                                                date = class_data.get(j).text();
                                        else if (j==21)
                                                loc = class_data.get(j).text();

                                        i++;
                                }
                                String c = "CRN: " + CRN + "   " + subj + "-" + crse + "    Credits: " +
cred + "\n"
                                  + title + "\n"
                                  + "Days: " + days + "    Time: " + time + "    Available: " + avail + "\n"
                                  + "Instructor: " + instr + "\n"
                                  + "Date: " + date + "    Location: " + loc;
                        class_search_results_items.add(c);
                }
            }
            adapter = new ArrayAdapter<String>(myContext, R.layout.classes_item,
class_search_results_items);
                        classes_results.setAdapter(adapter);
                }
                @Override
    public void onBackPressed() {
                finish();
    }
                public void main_menu_25(View view) {
                        Intent btn = new Intent(this, EMUActivity.class);
                        btn.setFlags(Intent.FLAG_ACTIVITY_CLEAR_TOP);
                        startActivity(btn);
                        finish();
                }
}
```

Appendix J: Final Grades code

classes_terms.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
android:layout_width="fill_parent" android:layout_height="fill_parent" android:orientation="vertical"
android:background="@drawable/background" >
<TextView android:id="@+id/classes_grades_title" android:layout_width="wrap_content"
android:layout_height="wrap_content" android:layout_centerHorizontal="true"
android:textColor="#448275" android:textSize="25px" android:layout_marginLeft="40px"
android:layout_marginTop="50px" android:text="View Final Grades" />
<TextView android:id="@+id/classes_grades_item_term" android:layout_width="400px"
android:layout_height="wrap_content" android:textColor="#448275" android:textSize="20px"
android:layout_marginLeft="40px" android:layout_marginTop="150px" android:text="Term" />
<Spinner android:id="@+id/classes_grades_term" android:layout_width="400px"
android:layout_height="wrap_content" android:layout_marginLeft="40px"
android:layout_marginTop="180px" />
<TextView android:id="@+id/error_message_view_6" android:layout_width="wrap_content"
android:layout_height="wrap_content" android:layout_centerHorizontal="true"
android:textColor="#FFFFFF" android:textSize="20px" android:layout_marginTop="400px"
android:text="" />
<Button android:id="@+id/classes_grades_btn" android:layout_width="200px"
android:layout_height="wrap_content" android:onClick="classes_grades_button"
android:layout_marginLeft="140px" android:layout_marginTop="300px" android:text="View Grades" />
<Button android:id="@+id/main_menu_btn_26" android:layout_width="match_parent"
android:layout_height="wrap_content" android:layout_alignParentBottom="true"
android:layout_alignParentLeft="true" android:onClick="main_menu_26"
android:background="@drawable/emu_button" />
</RelativeLayout>
```

Classes_Terms

```
package emu.classes;
import java.io.BufferedReader; import java.io.DataOutputStream; import java.io.IOException; import
java.io.InputStreamReader; import java.net.MalformedURLException; import java.net.URL; import
javax.net.ssl.HttpsURLConnection; import main.menu.EMUActivity; import main.menu.R; import
org.jsoup.Jsoup; import org.jsoup.nodes.Document; import org.jsoup.nodes.Element; import
org.jsoup.select.Elements; import android.app.Activity; import android.app.ProgressDialog; import
android.content.Intent; import android.os.AsyncTask; import android.os.Bundle; import
android.view.View; import android.widget.AdapterView; import android.widget.ArrayAdapter; import
android.widget.Spinner;
public class Classes_Terms extends Activity implements AdapterView.OnItemSelectedListener {
        private String user, pin, cookie;
```

```
	private URL url_address;
	private HttpsURLConnection conn;
	private Spinner class_term;
	private String[] term_values;
	private String[] term_strings;
	private String term_val, term_text;
	@Override
	public void onCreate(Bundle savedInstanceState) {
		super.onCreate(savedInstanceState);
		setContentView(R.layout.classes_terms);
		Intent search_classes = getIntent();
		user = search_classes.getStringExtra("USER");
		pin = search_classes.getStringExtra("PIN");
		cookie = search_classes.getStringExtra("Cookie_Value");
		String web = "https://bannerweb.emich.edu/pls/berp/bwskogrd.P_ViewTermGrde";
		Page_Download task = new Page_Download();
		task.execute(new String[] { web });
	}
	public void onResume() {
	super.onResume();
   }
   public void onRestart() {
	super.onRestart();
   }
   @Override
   public void onBackPressed() {
	finish();
   }
   public void set_term_values() {
		class_term = (Spinner) findViewById(R.id.classes_grades_term);
		class_term.setOnItemSelectedListener(this);
		ArrayAdapter<String> classes_term = new ArrayAdapter<String>(this,
android.R.layout.simple_spinner_item, term_strings);
classes_term.setDropDownViewResource(android.R.layout.simple_spinner_dropdown_item);
	class_term.setAdapter(classes_term);
	}
   public void onItemSelected(AdapterView<?> parent, View v, int position, long id) {
	term_val = term_values[position];
	term_text = term_strings[position];
   }
   public void onNothingSelected(AdapterView<?> parent) {
   }
	private class Page_Download extends AsyncTask<String, Void, String> {
		private ProgressDialog progressDialog;
		protected void onPreExecute() {
			progressDialog = ProgressDialog.show(Classes_Terms.this, "", "Loading.
Please wait...", true);
```

```
		}
		@Override
		protected String doInBackground(String... urls) {
			String response = "";
			for (String url : urls) {
				try {
					getTermValues(user, pin, url);
				}
				catch (Exception e) {
					e.printStackTrace();
				}
			}
			return response;
		}
		@Override
		protected void onPostExecute(String result) {
			progressDialog.dismiss();
			set_term_values();
		}
	}
	public void getTermValues(String username, String password, String address) throws
IOException, MalformedURLException {
		url_address = new URL(address);
		conn = (HttpsURLConnection) url_address.openConnection();
		conn.setRequestMethod("POST");
		conn.setDoInput(true);
		conn.setDoOutput(true);
		conn.setUseCaches(true);
		conn.setRequestProperty("Cookie", cookie);
		conn.setRequestProperty("Content-Type", "text/html");
		DataOutputStream out = new DataOutputStream(conn.getOutputStream());
		out.flush();
		out.close();
		BufferedReader input = new BufferedReader(new
InputStreamReader(conn.getInputStream()));
		String alllines = "";
		String line = "";
		while ((line = input.readLine()) != null)
		{
			alllines += line + "\n";
		}
		input.close();
		Document doc = Jsoup.parseBodyFragment(alllines);
		Element body = doc.body();
		Element labels =  body.select("table.dataentrytable").first();
		Elements term_data = labels.select("option");
		int list_size = term_data.size();
```

```
                term_values = new String[list_size];
                term_strings = new String[list_size];
                int index = 0;
                for (Element item : term_data) {
                        term_values[index] = item.val();
                        term_strings[index] = item.text();
                        index++;
                }
        }
        public void classes_grades_button(View view) {
                Intent btn = new Intent(this, Classes_Grades.class);
                btn.putExtra("Cookie_Value", cookie);
                btn.putExtra("USER", user);
                btn.putExtra("PIN", pin);
                btn.putExtra("TERM_VALUE", term_val);
                btn.putExtra("TERM_TEXT", term_text);
        startActivity(btn);
        finish();
        }
        public void main_menu_26(View view) {
        Intent btn = new Intent(this, EMUActivity.class);
        btn.setFlags(Intent.FLAG_ACTIVITY_CLEAR_TOP);
        startActivity(btn);
        finish();
   }
}
```

classes_grades.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
android:layout_width="fill_parent" android:layout_height="fill_parent" android:orientation="vertical"
android:background="@drawable/background" >
<TextView android:id="@+id/final_grades_title" android:layout_width="wrap_content"
android:layout_height="wrap_content" android:layout_centerHorizontal="true"
android:textColor="#448275" android:textSize="25px" android:layout_marginLeft="40px"
android:layout_marginTop="50px" android:text="Final Grades" />
<TextView android:id="@+id/final_grades_title_2" android:layout_width="wrap_content"
android:layout_height="wrap_content" android:layout_centerHorizontal="true"
android:textColor="#448275" android:textSize="25px" android:layout_marginLeft="40px"
android:layout_marginTop="90px" android:text="" />
<ScrollView android:layout_width="450px" android:layout_height="wrap_content"
android:layout_centerHorizontal="true"android:layout_marginTop="150px" >
   <TextView android:id="@+id/final_grades" android:layout_width="450px"
      android:layout_height="wrap_content" android:textColor="#448275"
```

```
        android:textSize="20px" android:layout_centerHorizontal="true"
        android:layout_marginTop="150px" android:text="" />
</ScrollView>
<Button android:id="@+id/main_menu_btn_27" android:layout_width="match_parent"
android:layout_height="wrap_content" android:layout_alignParentBottom="true"
android:layout_alignParentLeft="true" android:onClick="main_menu_27"
android:background="@drawable/emu_button" />
</RelativeLayout>
```

Classes_Grades

```
package emu.classes;
import java.io.BufferedReader; import java.io.DataOutputStream; import java.io.IOException; import
java.io.InputStreamReader; import java.net.MalformedURLException; import java.net.URL; import
javax.net.ssl.HttpsURLConnection; import main.menu.EMUActivity; import main.menu.R; import
org.jsoup.Jsoup; import org.jsoup.nodes.Document; import org.jsoup.nodes.Element; import
org.jsoup.select.Elements; import android.app.Activity; import android.app.ProgressDialog; import
android.content.Intent; import android.os.AsyncTask; import android.os.Bundle; import
android.view.View; import android.widget.TextView;
public class Classes_Grades extends Activity {
            private String user, pin, cookie;
            private URL url_address;
            private HttpsURLConnection conn;
            private TextView display_term, display;
            private String term_val, term_text;
            @Override
            public void onCreate(Bundle savedInstanceState) {
                        super.onCreate(savedInstanceState);
                        setContentView(R.layout.classes_grades);
                        Intent btn = getIntent();
                        user = btn.getStringExtra("USER");
                        pin = btn.getStringExtra("PIN");
                        cookie = btn.getStringExtra("Cookie_Value");
                        term_val = btn.getStringExtra("TERM_VALUE");
                        term_text = btn.getStringExtra("TERM_TEXT");
                        display_term = (TextView) findViewById(R.id.final_grades_title_2);
                        display = (TextView) findViewById(R.id.final_grades);
                        String web = "https://bannerweb.emich.edu/pls/berp/bwskogrd.P_ViewGrde?term_in="
+ term_val;
                        Page_Download task = new Page_Download();
                        task.execute(new String[] { web });
            }
            public void onResume() {
            super.onResume();
   }
```

```
    public void onRestart() {
        super.onRestart();
    }
    @Override
    public void onBackPressed() {
        finish();
    }
        private class Page_Download extends AsyncTask<String, Void, String> {
                private ProgressDialog progressDialog;
                protected void onPreExecute() {
                        progressDialog = ProgressDialog.show(Classes_Grades.this, "", "Loading.
Please wait...", true);
                }
                @Override
                protected String doInBackground(String... urls) {
                        String response = "";
                        for (String url : urls) {
                                try {
                                        response = getTermValues(user, pin, url);
                                }
                                catch (Exception e) {
                                        e.printStackTrace();
                                }
                        }
                        return response;
                }
                @Override
                protected void onPostExecute(String result) {
                        progressDialog.dismiss();
                        display_term.setText(term_text);
                        display.setText(result);
                }
        }
        public String getTermValues(String username, String password, String address) throws
IOException, MalformedURLException {
                url_address = new URL(address);
                conn = (HttpsURLConnection) url_address.openConnection();
                conn.setRequestMethod("POST");
                conn.setDoInput(true);
                conn.setDoOutput(true);
                conn.setUseCaches(true);
                conn.setRequestProperty("Cookie", cookie);
                conn.setRequestProperty("Content-Type", "text/html");
                DataOutputStream out = new DataOutputStream(conn.getOutputStream());
                out.flush();
                out.close();
```

```
                BufferedReader input = new BufferedReader(new
InputStreamReader(conn.getInputStream()));
                String alllines = "";
                String line = "";
                while ((line = input.readLine()) != null)
                {
                        alllines += line + "\n";
                }
                input.close();
                Document doc = Jsoup.parseBodyFragment(alllines);
                Element body = doc.select("div.pagebodydiv").first();
                Element table_1 =  body.select("table.datadisplaytable").get(1);
                Element table_2 =  body.select("table.datadisplaytable").get(2);
                alllines = "";
                Elements table_1_trs = table_1.select("tr");
                int table_1_check = 0;
                for (Element item_1 : table_1_trs) {
                        if (table_1_check != 0) {
                                alllines += item_1.select("td").get(1).text() + "-" +
item_1.select("td").get(2).text() + " - ";
                                alllines += item_1.select("td").get(4).text() + ": ";
                                alllines += item_1.select("td").get(6).text() + "\n";
                        }
                        else
                                table_1_check = 1;
                }
                alllines += "\nGPA: \n";
                Elements table_2_trs = table_2.select("tr");
                int table_2_check = 0;
                for (Element item_2 : table_2_trs) {
                        if (table_2_check != 0) {
                                alllines += item_2.select("th").first().text() +
item_2.select("td").get(4).text() + "\n";
                        }
                        else
                                table_2_check = 1;
                }
                return alllines;
        }
        public void main_menu_27(View view) {
        Intent btn = new Intent(this, EMUActivity.class);
        btn.setFlags(Intent.FLAG_ACTIVITY_CLEAR_TOP);
        startActivity(btn);
        finish();
   }
}
```

Appendix K: Twitter Updates code

twitter_options.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
android:layout_width="match_parent" android:layout_height="match_parent"
android:background="@drawable/background" android:orientation="vertical" >
<TextView android:id="@+id/twitter_updates_menu" android:layout_width="wrap_content"
android:layout_height="wrap_content" android:layout_centerHorizontal="true"
android:layout_marginTop="60px" android:text="Twitter Updates" android:textColor="#448275"
android:textSize="25px" />
<Button android:id="@+id/twitter_menu_btn1" android:layout_width="match_parent"
android:layout_height="wrap_content" android:layout_marginTop="150px"
android:layout_centerHorizontal="true" android:onClick="twitter_choice_1" android:text="News" />
<Button android:id="@+id/twitter_menu_btn2" android:layout_width="match_parent"
android:layout_height="wrap_content" android:layout_alignParentLeft="true"
android:layout_marginTop="220px" android:onClick="twitter_choice_2" android:text="Events" />
<Button  android:id="@+id/twitter_menu_btn3" android:layout_width="match_parent"
android:layout_height="wrap_content" android:layout_marginTop="290px"
android:layout_alignParentLeft="true" android:onClick="twitter_choice_3" android:text="Athletics" />
<Button android:id="@+id/twitter_menu_btn4" android:layout_width="match_parent"
android:layout_height="wrap_content" android:layout_marginTop="360px"
android:layout_alignParentLeft="true" android:onClick="twitter_choice_4" android:text="Buildings and
Offices" />
<Button android:id="@+id/twitter_menu_btn5" android:layout_width="match_parent"
android:layout_height="wrap_content" android:layout_marginTop="430px"
android:layout_alignParentLeft="true" android:onClick="twitter_choice_5" android:text="Greek Life" />
<Button android:id="@+id/twitter_menu_btn6" android:layout_width="match_parent"
android:layout_height="wrap_content" android:layout_marginTop="500px"
android:layout_alignParentLeft="true" android:onClick="twitter_choice_6" android:text="Jobs and
Rewards" />
<Button android:id="@+id/twitter_menu_btn7" android:layout_width="match_parent"
android:layout_height="wrap_content" android:layout_marginTop="570px"
android:layout_alignParentLeft="true" android:onClick="twitter_choice_7" android:text="Housing and
Dining" />
<Button android:id="@+id/main_menu_btn_15" android:layout_width="match_parent"
android:layout_height="wrap_content" android:layout_alignParentBottom="true"
android:layout_alignParentLeft="true" android:background="@drawable/emu_button"
android:onClick="main_menu_button_15" />
</RelativeLayout>
```

Twitter_Options

```
package emu.twitter;
import main.menu.EMUActivity; import main.menu.R; import android.app.Activity; import
android.content.Intent; import android.os.Bundle; import android.view.View;
public class Twitter_Options extends Activity {
   @Override
   public void onCreate(Bundle savedInstanceState) {
      super.onCreate(savedInstanceState);
      setContentView(R.layout.twitter_options);
   }
   @Override
   public void onBackPressed() {
         finish(); }
public void twitter_choice_1(View view) {
         String url =
"http://search.twitter.com/search.json?q=from%3aEMU_Swoop+OR+from%3aemunews+OR+from%3a
TheEasternEcho+OR+from3%aemusg&rpp=50&page=1";
         Intent btn = new Intent(this, Twitter_Feed.class);
         btn.putExtra("URL_QUERY", url);
         startActivity(btn);
   }
   public void twitter_choice_2(View view) {
         String url =
"http://search.twitter.com/search.json?q=from%3aEMUTheatre+OR+from%3aEMUcampuslife&rpp=50
&page=1";
         Intent btn = new Intent(this, Twitter_Feed.class);
         btn.putExtra("URL_QUERY", url);
         startActivity(btn);
   }
   public void twitter_choice_3(View view) {
         String url =
"http://search.twitter.com/search.json?q=from%3aron_english+OR+from%3aMHart2032+OR+from%3a
EMUHoops+OR+from%3aEMUwrestling+OR+from%3aEMU_EagleNation+OR+from%3aemuathletics+
OR+from%3aemueaglesradio+OR+from%3aEMUWBB+OR+from%3aEagleRadioEMU+OR+from%3a
EMU_Baseball+OR+from%3aEMUFB+OR+from%3arobmurphyEMU+OR+from%3aAnnMarieGilbert&r
pp=50&page=1";
         Intent btn = new Intent(this, Twitter_Feed.class);
         btn.putExtra("URL_QUERY", url);
         startActivity(btn);
   }
   public void twitter_choice_4(View view) {
         String url =
"http://search.twitter.com/search.json?q=from%3aEMUOIS+OR+from%3aAdvising_Career+OR+from%
3aEMUAlumni+OR+from%3aVISIONEMU+OR+from%3aEMU_Recim+OR+from%3aemulibrary+OR+fr
om%3algbtrc_emu&rpp=50&page=1";
```

```
        Intent btn = new Intent(this, Twitter_Feed.class);
        btn.putExtra("URL_QUERY", url);
        startActivity(btn);
    }
    public void twitter_choice_5(View view) {
        String url =
"http://search.twitter.com/search.json?q=from%3aemuDSP+OR+from%3aDeltaZetaEMU+OR+from%3
aTKE_Fraternity+OR+from%3aASTAlphasEMU+OR+from%3aEMUAlphaGams+OR+from%3aSigmaN
uPhi+OR+from%3aemuPHISIGS+OR+from%3aAXiDatEMU+OR+from%3aemusigmanu+OR+from%3
aEMUDelts&rpp=50&page=1";
        Intent btn = new Intent(this, Twitter_Feed.class);
        btn.putExtra("URL_QUERY", url);
        startActivity(btn);
    }
    public void twitter_choice_6(View view) {
        String url =
"http://search.twitter.com/search.json?q=from%3aEMUJobs+OR+from%3aeaglerewards&rpp=50&pag
e=1";
        Intent btn = new Intent(this, Twitter_Feed.class);
        btn.putExtra("URL_QUERY", url);
        startActivity(btn);
    }
    public void twitter_choice_7(View view) {
        String url =
"http://search.twitter.com/search.json?q=from%3aEMUvillage+OR+from%3aBuellHall+OR+from%3aDo
wningHall+OR+from%3aEMUDining&rpp=50&page=1";
        Intent btn = new Intent(this, Twitter_Feed.class);
        btn.putExtra("URL_QUERY", url);
        startActivity(btn);
    }
    public void main_menu_button_15(View view) {
        Intent btn = new Intent(this, EMUActivity.class);
        btn.setFlags(Intent.FLAG_ACTIVITY_CLEAR_TOP);
        startActivity(btn);
        finish();
    }
}
```

twitter_feed.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
android:layout_width="fill_parent" android:layout_height="fill_parent" android:orientation="vertical"
android:background="@drawable/background" >
```

```
<ListView android:id="@+id/twitter_list_view" android:layout_width="match_parent"
android:layout_height="wrap_content" android:layout_above="@+id/main_menu_btn_14"
android:layout_alignParentLeft="true" android:layout_alignParentTop="true"
android:cacheColorHint="#00000000" />
<Button android:id="@+id/main_menu_btn_14" android:layout_width="match_parent"
android:layout_height="wrap_content" android:layout_alignParentBottom="true"
android:layout_alignParentLeft="true" android:onClick="main_menu_14"
android:background="@drawable/emu_button" />
</RelativeLayout>
```

twitter_tweet.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
android:orientation="vertical" android:layout_width="fill_parent"  android:layout_height="150px"
android:padding="6dip" >
<ImageView android:id="@+id/avatar" android:layout_width="80px" android:layout_height="80px" />
<TextView android:id="@+id/toptext" android:layout_width="250px"
android:layout_height="wrap_content" android:textColor="#448275" android:layout_marginLeft="90px"
android:singleLine="true" />
<TextView android:id="@+id/time_sent" android:layout_width="150px"
android:layout_height="wrap_content" android:textColor="#448275"
android:layout_marginLeft="350px" android:singleLine="true" />
<TextView android:id="@+id/bottomtext" android:layout_width="wrap_content"
android:layout_height="wrap_content" android:layout_alignLeft="@+id/toptext"
android:layout_alignRight="@+id/time_sent" android:layout_below="@+id/toptext"
android:singleLine="false" android:textColor="#FFFFFF" />
</RelativeLayout>
```

Twitter_Feed

```
package emu.twitter;
import java.net.URI; import java.util.ArrayList; import main.menu.EMUActivity; import main.menu.R;
import org.apache.http.HttpResponse; import org.apache.http.HttpStatus; import
org.apache.http.client.HttpClient; import org.apache.http.client.methods.HttpGet; import
org.apache.http.impl.client.DefaultHttpClient; import org.apache.http.util.EntityUtils; import
org.json.JSONArray; import org.json.JSONObject; import android.app.Activity; import
android.app.ProgressDialog; import android.content.Intent; import android.graphics.Bitmap; import
android.os.AsyncTask; import android.os.Bundle; import android.view.View; import
android.widget.ListView;
public class Twitter_Feed extends Activity {
        private ListView list_tweets;
        public Bitmap placeholder;
        private String url_query;
```

```
        private ArrayList<Tweet> tweets;
        private Activity myActivity;
    @Override
    public void onCreate(Bundle savedInstanceState) {
       super.onCreate(savedInstanceState);
       setContentView(R.layout.twitter_feed);
       list_tweets = (ListView) findViewById(R.id.twitter_list_view);
       myActivity = this;
       Intent btn = getIntent();
       url_query = btn.getStringExtra("URL_QUERY");
       Page_Download task = new Page_Download();
       task.execute(new String[] {url_query});
       list_tweets = (ListView) findViewById(R.id.twitter_list_view);
    }
    @Override
    public void onBackPressed() {
        finish();
    }
    public class Page_Download extends AsyncTask<String, Void, String> {
        private ProgressDialog progressDialog;
        protected void onPreExecute() {
            progressDialog = ProgressDialog.show(Twitter_Feed.this, "", "Loading. Please wait...",
true);
        }
        @Override
        protected String doInBackground(String... urls) {
        tweets = new ArrayList<Tweet>();
        String response = "";
            for (String url : urls) {
                try {
                    tweets = getTweets(url);
                }
                catch (Exception e) {
                    e.printStackTrace();
                }
            }
            return response;
        }
        @Override
        protected void onPostExecute(String result) {
            list_tweets.setAdapter(new Twitter_Adapter(myActivity, R.layout.twitter_tweet,
tweets));
            progressDialog.dismiss();
        }
    }
    public ArrayList<Tweet> getTweets(String url) {
            String searchUrl = url;
```

```
		ArrayList<Tweet> tweets = new ArrayList<Tweet>();
		try {
			HttpClient hc = new DefaultHttpClient();
		HttpGet get = new HttpGet();
		get.setURI(new URI(searchUrl));
		HttpResponse rp = hc.execute(get);
			 if(rp.getStatusLine().getStatusCode() == HttpStatus.SC_OK)
			{
				String result = EntityUtils.toString(rp.getEntity());
				JSONObject root = new JSONObject(result);
				JSONArray sessions = root.getJSONArray("results");
				for (int i = 0; i < sessions.length(); i++) {
					JSONObject session = sessions.getJSONObject(i);
	Tweet tweet = new Tweet(session.getString("from_user"), session.getString("text"),
	session.getString("profile_image_url"), session.getString("created_at"));
					 tweets.add(tweet);
				}
			}
		}
		catch (Exception e) {
			e.printStackTrace();
		}
		return tweets;
	}
	public void main_menu_14(View view) {
		Intent btn = new Intent(this, EMUActivity.class);
		btn.setFlags(Intent.FLAG_ACTIVITY_CLEAR_TOP);
		startActivity(btn);
		finish();
	}
}
```

Tweet

```
package emu.twitter;
import java.text.SimpleDateFormat; import java.util.Date; import java.util.Locale;
public class Tweet {
	String author;
	 String content;
	String image_url;
	Boolean usernameSet = false;
	Boolean messageSet = false;
	Boolean imageSet = false;
	String time_stamp;
	public Tweet(String author, String content, String image_url, String time_date) {
```

```
		this.author = author;
		this.content = content;
		this.image_url = image_url;
		this.time_stamp = twitterHumanFriendlyDate(time_date);
	}
	public String twitterHumanFriendlyDate(String date_time) {
		// parse Twitter date
		SimpleDateFormat dateFormat = new SimpleDateFormat("EEE, dd MMM yyyy
HH:mm:ss ZZZZZ", Locale.ENGLISH);
		dateFormat.setLenient(false);
		Date created = null;
		try {
			created = dateFormat.parse(date_time);
		} catch (Exception e) {
			return null;
		}
		// today
		Date today = new Date();
		// how much time since (ms)
		Long duration = today.getTime() - created.getTime();
		int second = 1000;
		int minute = second * 60;
		int hour = minute * 60;
		int day = hour * 24;
		if (duration < second * 7) {
			return "right now";
		}
		if (duration < minute) {
			int n = (int) Math.floor(duration / second);
			return n + " seconds ago";
		}
		if (duration < minute * 2) {
			return "about 1 minute ago";
		}
		if (duration < hour) {
			int n = (int) Math.floor(duration / minute);
			return n + " minutes ago";
		}
		if (duration < hour * 2) {
			return "about 1 hour ago";
		}
		if (duration < day) {
			int n = (int) Math.floor(duration / hour);
			return n + " hours ago";
		}
		if (duration > day && duration < day * 2) {
			return "yesterday";
```

```
                }
                if (duration < day * 365) {
                        int n = (int) Math.floor(duration / day);
                        return n + " days ago";
                } else {
                        return "over a year ago";
                }
        }
}
```

Twitter_Adapter

```
package emu.twitter;
import java.util.ArrayList; import main.menu.R; import android.app.Activity; import
android.content.Context; import android.view.LayoutInflater; import android.view.View;
import android.view.ViewGroup; import android.widget.ArrayAdapter; import
android.widget.ImageView; import android.widget.TextView;
public class Twitter_Adapter extends ArrayAdapter<Tweet> {
        private ArrayList<Tweet> tweets;
        private Activity activity;
        public Twitter_Image_Manager imageManager;
        public Twitter_Adapter(Activity a, int textViewResourceId, ArrayList<Tweet> tweets) {
                super(a, textViewResourceId, tweets);
                this.tweets = tweets;
                activity = a;
                imageManager = new Twitter_Image_Manager(activity.getApplicationContext());
        }
        public static class ViewHolder{
                public TextView username;
                public TextView message;
                public TextView time_date;
                public ImageView image;
        }
        @Override
        public View getView(int position, View convertView, ViewGroup parent) {
                View v = convertView;
                ViewHolder holder;
                if (v == null) {
                        LayoutInflater vi =
        (LayoutInflater)activity.getSystemService(Context.LAYOUT_INFLATER_SERVICE);
                        v = vi.inflate(R.layout.twitter_tweet, null);
                        holder = new ViewHolder();
                        holder.username = (TextView) v.findViewById(R.id.toptext);
                        holder.message = (TextView) v.findViewById(R.id.bottomtext);
                        holder.time_date = (TextView) v.findViewById(R.id.time_sent);
```

```
                holder.image = (ImageView) v.findViewById(R.id.avatar);
                v.setTag(holder);
            }
            else
                holder=(ViewHolder)v.getTag();
            final Tweet tweet = tweets.get(position);
            if (tweet != null) {
                holder.username.setText(tweet.author);
                holder.message.setText(tweet.content);
                holder.time_date.setText(tweet.time_stamp);
                holder.image.setTag(tweet.image_url);
                imageManager.displayImage(tweet.image_url, activity, holder.image);
            }
            return v;
        }
}
```

Twitter_Image_Manager

```
package emu.twitter;
import java.io.File; import java.io.FileOutputStream; import java.lang.ref.SoftReference; import
java.net.URL; import java.util.HashMap; import java.util.Stack; import main.menu.R;
import android.app.Activity; import android.content.Context; import android.graphics.Bitmap; import
android.graphics.BitmapFactory; import android.widget.ImageView;
public class Twitter_Image_Manager {
        private HashMap<String, SoftReference<Bitmap>> imageMap = new HashMap<String,
SoftReference<Bitmap>>();
        private File cacheDir;
        private ImageQueue imageQueue = new ImageQueue();
        private Thread imageLoaderThread = new Thread(new ImageQueueManager());
        public Twitter_Image_Manager(Context context) {
            // Make background thread low priority, to avoid affecting UI performance
            imageLoaderThread.setPriority(Thread.NORM_PRIORITY-1);
            // Find the dir to save cached images
            String sdState = android.os.Environment.getExternalStorageState();
            if (sdState.equals(android.os.Environment.MEDIA_MOUNTED)) {
                File sdDir = android.os.Environment.getExternalStorageDirectory();
                cacheDir = new File(sdDir,"data/imagedata");
            }
            else
                cacheDir = context.getCacheDir();
            if(!cacheDir.exists())
                cacheDir.mkdirs();
        }
        public void displayImage(String url, Activity activity, ImageView imageView) {
```

```
                if(imageMap.containsKey(url))
                        imageView.setImageBitmap(imageMap.get(url).get());
                else {
                        queueImage(url, activity, imageView);
                        imageView.setImageResource(R.drawable.ic_launcher);
                }
        }
        private void queueImage(String url, Activity activity, ImageView imageView) {
                // This ImageView might have been used for other images, so we clear
                // the queue of old tasks before starting.
                imageQueue.Clean(imageView);
                ImageRef p=new ImageRef(url, imageView);
                synchronized(imageQueue.imageRefs) {
                        imageQueue.imageRefs.push(p);
                        imageQueue.imageRefs.notifyAll();
                }
                // Start thread if it's not started yet
                if(imageLoaderThread.getState() == Thread.State.NEW)
                        imageLoaderThread.start();
        }
        private Bitmap getBitmap(String url) {
                String filename = String.valueOf(url.hashCode());
                File f = new File(cacheDir, filename);
                // Is the bitmap in our cache?
                Bitmap bitmap = BitmapFactory.decodeFile(f.getPath());
                if(bitmap != null) return bitmap;
                // Nope, have to download it
                try {
                        bitmap = BitmapFactory.decodeStream(new
URL(url).openConnection().getInputStream());
                        // save bitmap to cache for later
                        writeFile(bitmap, f);
                        return bitmap;
                } catch (Exception ex) {
                        ex.printStackTrace();
                        return null;
                }
        }
        private void writeFile(Bitmap bmp, File f) {
                FileOutputStream out = null;
                try {
                        out = new FileOutputStream(f);
                        bmp.compress(Bitmap.CompressFormat.PNG, 80, out);
                } catch (Exception e) {
                        e.printStackTrace();
                }
                finally {
```

```
                try { if (out != null ) out.close(); }
                catch(Exception ex) {}
            }
        }
        /** Classes **/
        private class ImageRef {
            public String url;
            public ImageView imageView;
            public ImageRef(String u, ImageView i) {
                url=u;
                imageView=i;
            }
        }
        // stores list of images to download
        private class ImageQueue {
            private Stack<ImageRef> imageRefs =
                new Stack<ImageRef>();
            // removes all instances of this ImageView
            public void Clean(ImageView view) {
                for(int i = 0 ;i < imageRefs.size();) {
                    if(imageRefs.get(i).imageView == view)
                        imageRefs.remove(i);
                    else ++i;
                }
            }
        }
        private class ImageQueueManager implements Runnable {
            public void run() {
                try {
                    while(true) {
                        // Thread waits until there are images in the
                        // queue to be retrieved
                        if(imageQueue.imageRefs.size() == 0) {
                            synchronized(imageQueue.imageRefs) {
                                imageQueue.imageRefs.wait();
                            }
                        }
                        // When we have images to be loaded
                        if(imageQueue.imageRefs.size() != 0) {
                            ImageRef imageToLoad;
                            synchronized(imageQueue.imageRefs) {
                                imageToLoad =
imageQueue.imageRefs.pop();
                            }
                            Bitmap bmp = getBitmap(imageToLoad.url);
                            imageMap.put(imageToLoad.url, new
SoftReference<Bitmap>(bmp));
```

```
                                        Object tag = imageToLoad.imageView.getTag();

                                        // Make sure we have the right view - thread
                                        // safety defender
                                        if(tag != null &&
((String)tag).equals(imageToLoad.url)) {
                                                BitmapDisplayer bmpDisplayer =
                                                        new BitmapDisplayer(bmp,
imageToLoad.imageView);
                                                Activity a =
(Activity)imageToLoad.imageView.getContext();
                                                a.runOnUiThread(bmpDisplayer);
                                        }
                                }
                                if(Thread.interrupted())
                                        break;
                        }
                } catch (InterruptedException e) {}
        }
    }
    //Used to display bitmap in the UI thread
    private class BitmapDisplayer implements Runnable {
        Bitmap bitmap;
        ImageView imageView;
        public BitmapDisplayer(Bitmap b, ImageView i) {
            bitmap=b;
            imageView=i;
        }
        public void run() {
            if(bitmap != null)
                imageView.setImageBitmap(bitmap);
            else
                imageView.setImageResource(R.drawable.ic_launcher);
        }
    }
}
```

Appendix L: Athletics News code

athletics.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
android:layout_width="match_parent" android:layout_height="match_parent"
android:background="@drawable/background" android:orientation="vertical" >
<Button android:id="@+id/main_menu_btn_7" android:layout_width="match_parent"
android:layout_height="wrap_content" android:layout_alignParentBottom="true"
android:layout_alignParentLeft="true" android:background="@drawable/emu_button"
android:onClick="main_menu_7" />
<WebView android:id="@+id/athletics_view" android:layout_width="match_parent"
android:layout_height="match_parent" android:layout_above="@+id/main_menu_btn_7"
android:layout_alignParentLeft="true" android:layout_alignParentTop="true" />
</RelativeLayout>
```

Athletics

```
package emu.athletics;
import main.menu.EMUActivity; import main.menu.R; import android.app.Activity; import
android.content.Intent; import android.os.Bundle; import android.view.View; import
android.webkit.WebView; import android.webkit.WebViewClient;
public class Athletics extends Activity {
        private WebView page;
   @Override
   public void onCreate(Bundle savedInstanceState) {
      super.onCreate(savedInstanceState);
      setContentView(R.layout.athletics);
      page = (WebView) findViewById(R.id.athletics_view);
      page.setWebViewClient(new WebViewClient());
      page.getSettings().setJavaScriptEnabled(true);
      page.loadUrl("http://emueagles.com/mobile/?");
   }
   @Override
   public void onBackPressed() {
        finish();
   }
   public void main_menu_7(View view) {
        Intent btn = new Intent(this, EMUActivity.class);
        btn.setFlags(Intent.FLAG_ACTIVITY_CLEAR_TOP);
        startActivity(btn);
        finish();
   }}
```

Appendix M: Dining Locations code

dining_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
android:layout_width="match_parent" android:layout_height="match_parent"
android:background="@drawable/background" android:orientation="vertical" >
<TextView android:id="@+id/dining_main_menu" android:layout_width="wrap_content"
android:layout_height="wrap_content" android:layout_centerHorizontal="true"
android:layout_marginTop="60px" android:text="Dining Options" android:textColor="#448275"
android:textSize="25px" />
<Button android:id="@+id/dining_locations_view" android:layout_width="300px"
android:layout_height="wrap_content" android:layout_marginTop="150px"
android:layout_centerHorizontal="true" android:onClick="choice_1" android:text="Dining Locations" />
<Button android:id="@+id/dining_menu_options" android:layout_width="300px"
android:layout_height="wrap_content" android:layout_marginTop="250px"
android:layout_centerHorizontal="true" android:onClick="choice_2" android:text="Dining Menus" />
<Button android:id="@+id/main_menu_btn_31" android:layout_width="match_parent"
android:layout_height="wrap_content" android:layout_alignParentBottom="true"
android:layout_alignParentLeft="true" android:background="@drawable/emu_button"
android:onClick="main_menu_button_31" />
</RelativeLayout>
```

Dining_Main

```
package emu.dining;
import main.menu.EMUActivity; import main.menu.R; import android.app.Activity; import
android.content.Intent; import android.os.Bundle; import android.view.View;
public class Dining_Main extends Activity {
   @Override
   public void onCreate(Bundle savedInstanceState) {
      super.onCreate(savedInstanceState);
      setContentView(R.layout.dining_main);
   }
   @Override
   public void onBackPressed() {
         finish();
   }
   public void choice_1(View view) {
         Intent pick = new Intent(this, Dining_View.class);
         pick.putExtra("URL",
"http://maps.google.com/maps/ms?hl=en&ie=UTF8&t=h&msa=0&msid=211900276239851867603.000
48653b984f58be26d7&ll=42.250186,-83.623939&spn=0.00953,0.016072&z=16&source=embed");
```

```
        startActivity(pick);
        finish();
    }
    public void choice_2(View view) {
        Intent pick = new Intent(this, Dining_Menus.class);
        startActivity(pick);
        finish();
    }
    public void main_menu_button_31(View view) {
        Intent btn = new Intent(this, EMUActivity.class);
        btn.setFlags(Intent.FLAG_ACTIVITY_CLEAR_TOP);
        startActivity(btn);
        finish();
    }
}
```

dining_menus.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
android:orientation="vertical" android:layout_width="fill_parent"  android:layout_height="fill_parent"
android:background="@drawable/background"  android:padding="6dip" >
<TextView android:id="@+id/dining_locations_title" android:layout_width="wrap_content"
android:layout_height="wrap_content" android:layout_centerHorizontal="true"
android:layout_marginTop="40px" android:text="Dining_Locations Menus"
android:textColor="#448275" android:textSize="25px" />
<ListView android:id="@+id/dining_items" android:layout_width="match_parent"
android:layout_height="wrap_content" android:layout_above="@+id/main_menu_btn_32"
android:layout_marginTop="70px" android:layout_alignLeft="@+id/main_menu_btn_32"
android:cacheColorHint="#00000000" android:background="@drawable/background"
android:layout_below="@+id/dining_locations_title" ></ListView>
<Button android:id="@+id/main_menu_btn_32" android:layout_width="match_parent"
android:layout_height="wrap_content" android:layout_alignParentBottom="true"
android:layout_alignParentLeft="true" android:onClick="main_menu_button_32"
android:background="@drawable/emu_button" />
</RelativeLayout>
```

dining_item.xml

```
<TextView xmlns:android="http://schemas.android.com/apk/res/android"
android:id="@+id/dining_item_line" android:layout_width="fill_parent"  android:layout_height="80px"
android:textColor="#FFFFFF" android:background="@drawable/background" android:padding="5px"
android:singleLine="false" />
```

Dining_Menus

```
package emu.dining;
import main.menu.EMUActivity; import main.menu.R; import android.app.Activity; import
android.content.Intent; import android.os.Bundle; import android.view.View; import
android.widget.AdapterView; import android.widget.ArrayAdapter; import android.widget.ListView;
import android.widget.AdapterView.OnItemClickListener;
public class Dining_Menus extends Activity {
	private final String[] menus = {"The Commons","E-Street Grill","Freshens","Jump Asian
Cuisine", "Sunset Strips","The Upper Crust","Wrap It Up","Quick Fixx","International Kitchen","Salsa
Grille","The Bistro","Starbucks","Einstein Bros. Bagels","MarketPlace","Lobby Shop","Eagle Cafe
@Mark Jefferson","Eagle Cafe @Alexander Music Building","Eagle Cafe @The COB","Eagle Cafe
@Mckenny Hall","Eagle Cafe @Halle Library","Eagle Cafe @Marshall Hall","Eagle Cafe @Sill Hall"};
	private ListView locations;
	String url;
	@Override
	public void onCreate(Bundle savedInstanceState) {
		super.onCreate(savedInstanceState);
		setContentView(R.layout.dining_menus);
		url = "";
		locations = (ListView) findViewById(R.id.dining_items);
		locations.setAdapter(new ArrayAdapter<String>(this,R.layout.dining_item, menus));
		locations.setOnItemClickListener(new OnItemClickListener() {
			public void onItemClick(AdapterView<?> parent, View view, int position, long
id) {
				if (position == 0)
					url =
"http://www.emich.edu/dining/services/menu/commons2.html";
				else if (position == 1)
					url =
"http://docs.google.com/gview?embedded=true&url=http://www.emich.edu/dining/services/menu/estree
tmenu.pdf";
				else if (position == 2)
					url =
"http://docs.google.com/gview?embedded=true&url=http://www.emich.edu/dining/services/menu/freshe
nsmenu.pdf";
				else if (position == 3)
					url =
"http://docs.google.com/gview?embedded=true&url=http://www.emich.edu/dining/services/menu/jumpm
enu.pdf";
				else if (position == 4)
					url =
"http://docs.google.com/gview?embedded=true&url=http://www.emich.edu/dining/services/menu/sunset
stripsmenu.pdf";
				else if (position == 5)
```

```
                url =
"http://docs.google.com/gview?embedded=true&url=http://www.emich.edu/dining/services/menu/upper
crustmenu.pdf";
            else if (position == 6)
                url =
"http://docs.google.com/gview?embedded=true&url=http://www.emich.edu/dining/services/menu/wrapit
upmenu.pdf";
            else if (position == 7)
                url =
"http://docs.google.com/gview?embedded=true&url=http://www.emich.edu/dining/services/menu/quickfi
xxmenu.pdf";
            else if (position == 8)
                url =
"http://docs.google.com/gview?embedded=true&url=http://www.emich.edu/dining/services/menu/ikmen
u.pdf";
            else if (position == 9)
                url =
"http://docs.google.com/gview?embedded=true&url=http://www.emich.edu/dining/services/menu/salsa
menu.pdf";
            else if (position == 10)
                url =
"http://docs.google.com/gview?embedded=true&url=http://www.emich.edu/dining/services/menu/vegeta
rianmenu.pdf";
            else if (position == 11)
                url =
"http://docs.google.com/gview?embedded=true&url=http://www.emich.edu/dining/services/menu/starbu
cksmenu.pdf";
            else if (position == 12)
                url =
"http://docs.google.com/gview?embedded=true&url=http://www.emich.edu/dining/services/menu/einstei
nbrosmenu.pdf";
            else if (position == 13)
                url =
"http://docs.google.com/gview?embedded=true&url=http://www.emich.edu/dining/services/menu/marke
tplacemenu.pdf";
            else if (position == 14)
                url =
"http://docs.google.com/gview?embedded=true&url=http://www.emich.edu/dining/services/menu/lobbys
hopmenu.pdf";
            else if (position == 15)
                url =
"http://docs.google.com/gview?embedded=true&url=http://www.emich.edu/dining/services/menu/ecmar
kjmenu.pdf";
            else if (position == 16)
                url =
"http://docs.google.com/gview?embedded=true&url=http://www.emich.edu/dining/services/menu/ecalex
andermenu.pdf";
```

```
                                        else if (position == 17)
                                                url =
"http://docs.google.com/gview?embedded=true&url=http://www.emich.edu/dining/services/menu/eccobmenu.pdf";
                                        else if (position == 18)
                                                url =
"http://docs.google.com/gview?embedded=true&url=http://www.emich.edu/dining/services/menu/ecmckennymenu.pdf";
                                        else if (position == 19)
                                                url =
"http://docs.google.com/gview?embedded=true&url=http://www.emich.edu/dining/services/menu/echallemenu.pdf";
                                        else if (position == 20)
                                                url =
"http://docs.google.com/gview?embedded=true&url=http://www.emich.edu/dining/services/menu/commongroundmenu.pdf";
                                        else
                                                url =
"http://docs.google.com/gview?embedded=true&url=http://www.emich.edu/dining/services/menu/foodforthoughtmenu.pdf";

                                        startIntent(url);
                                }
                        });

        }
        @Override
    public void onBackPressed() {
        finish();
    }
        public void startIntent(String url_to_send) {
                Intent btn = new Intent(this, Dining_View.class);
                btn.putExtra("URL", url_to_send);
                startActivity(btn);
                finish();
        }

        public void main_menu_button_32(View view) {
        Intent btn = new Intent(this, EMUActivity.class);
        btn.setFlags(Intent.FLAG_ACTIVITY_CLEAR_TOP);
        startActivity(btn);
        finish();
    }
}
```

dining_view.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
android:layout_width="match_parent" android:layout_height="match_parent"
android:background="@drawable/background" android:orientation="vertical" >
<WebView android:id="@+id/dining_locations_view" android:layout_width="match_parent"
android:layout_height="match_parent" android:layout_above="@+id/main_menu_btn_33"
android:layout_alignParentLeft="true" android:layout_alignParentTop="true" />
<Button android:id="@+id/main_menu_btn_33" android:layout_width="match_parent"
android:layout_height="wrap_content" android:layout_alignParentBottom="true"
android:layout_alignParentLeft="true" android:background="@drawable/emu_button"
android:onClick="main_menu_button_33" />
</RelativeLayout>
```

Dining_View

```
package emu.dining;
import main.menu.EMUActivity; import main.menu.R; import android.app.Activity; import
android.content.Intent; import android.os.Bundle; import android.view.View; import
android.webkit.WebView; import android.webkit.WebViewClient;
public class Dining_View extends Activity {
        private WebView map;
   @Override
   public void onCreate(Bundle savedInstanceState) {
      super.onCreate(savedInstanceState);
      setContentView(R.layout.dining_view);

      Intent btn = getIntent();
      String url = btn.getStringExtra("URL");
      map = (WebView) findViewById(R.id.dining_locations_view);
      map.setWebViewClient(new WebViewClient());
      map.getSettings().setJavaScriptEnabled(true);
      map.loadUrl(url);
   }
   @Override
   public void onBackPressed() {
        finish();
   }
   public void main_menu_button_33(View view) {
        Intent btn = new Intent(this, EMUActivity.class);
        btn.setFlags(Intent.FLAG_ACTIVITY_CLEAR_TOP);
        startActivity(btn);
        finish();
   }
}
```

Appendix N: Campus Map code

map.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
android:layout_width="match_parent" android:layout_height="match_parent"
android:background="@drawable/background" android:orientation="vertical" >
<WebView android:id="@+id/map_view" android:layout_width="match_parent"
android:layout_height="match_parent" android:layout_above="@+id/main_menu_btn_12"
android:layout_alignParentLeft="true" android:layout_alignParentTop="true" />
<Button android:id="@+id/main_menu_btn_12" android:layout_width="match_parent"
android:layout_height="wrap_content" android:layout_alignParentBottom="true"
android:layout_alignParentLeft="true" android:background="@drawable/emu_button"
android:onClick="main_menu_button_12" />
</RelativeLayout>
```

Map

```
package emu.map;
import main.menu.EMUActivity; import main.menu.R; import android.app.Activity; import
android.content.Intent; import android.os.Bundle; import android.view.View; import
android.webkit.WebView; import android.webkit.WebViewClient;
public class Map extends Activity {
        private WebView map;
   @Override
   public void onCreate(Bundle savedInstanceState) {
      super.onCreate(savedInstanceState);
      setContentView(R.layout.map);
      map = (WebView) findViewById(R.id.map_view);
      map.setWebViewClient(new WebViewClient());
      map.getSettings().setJavaScriptEnabled(true);
map.loadUrl("http://docs.google.com/gview?embedded=true&url=http://www.emich.edu/maps/pdf/map_
2011.pdf");
   }
   @Override
   public void onBackPressed() {
        finish();
   }
   public void main_menu_button_12(View view) {
        Intent btn = new Intent(this, EMUActivity.class);
        btn.setFlags(Intent.FLAG_ACTIVITY_CLEAR_TOP);
        startActivity(btn);
        finish();    }}
```

Appendix O: Bus Schedule code

bus_menu.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
android:layout_width="match_parent" android:layout_height="match_parent"
android:background="@drawable/background" android:orientation="vertical" >
<TextView android:id="@+id/bus_routes_menu" android:layout_width="wrap_content"
android:layout_height="wrap_content" android:layout_centerHorizontal="true"
android:layout_marginTop="60px" android:text="Bus Routes" android:textColor="#448275"
android:textSize="25px" />
<Button android:id="@+id/bus_menu_btn1" android:layout_width="300px"
android:layout_height="wrap_content" android:layout_marginTop="150px"
android:layout_centerHorizontal="true" android:onClick="choice_1" android:text="33 - COB Shuttle" />
<Button android:id="@+id/main_menu_btn_8" android:layout_width="match_parent"
android:layout_height="wrap_content" android:layout_alignParentBottom="true"
android:layout_alignParentLeft="true" android:background="@drawable/emu_button"
android:onClick="main_menu_button_8" />
<Button android:id="@+id/bus_menu_btn2" android:layout_width="300px"
android:layout_height="wrap_content" android:layout_alignLeft="@+id/bus_menu_btn1"
android:layout_below="@+id/bus_menu_btn1" android:layout_marginTop="20dp"
android:onClick="choice_2" android:text="33 - Schedule" />
<Button android:id="@+id/bus_menu_btn3" android:layout_width="300px"
android:layout_height="wrap_content" android:layout_alignLeft="@+id/bus_menu_btn2"
android:layout_below="@+id/bus_menu_btn2" android:layout_marginTop="20dp"
android:onClick="choice_3" android:text="34 - West Campus Shuttle" />
<Button android:id="@+id/bus_menu_btn4" android:layout_width="300px"
android:layout_height="wrap_content" android:layout_alignLeft="@+id/bus_menu_btn3"
android:layout_below="@+id/bus_menu_btn3" android:layout_marginTop="20dp"
android:onClick="choice_4" android:text="34 - Schedule" />
</RelativeLayout>
```

Bus_Menu

```
package emu.bus;
import main.menu.EMUActivity; import main.menu.R; import android.app.Activity; import
android.content.Intent; import android.os.Bundle; import android.view.View;
public class Bus_Menu extends Activity {
   @Override
   public void onCreate(Bundle savedInstanceState) {
      super.onCreate(savedInstanceState);
      setContentView(R.layout.bus_menu);
   }
```

```
    @Override
    public void onBackPressed() {
            finish();
    }
    public void choice_1(View view) {
            Intent pick = new Intent(this, Bus_Display.class);
            pick.putExtra("url", "http://mobile.aata.org/rideguide_m.asp?route=33");
            pick.putExtra("choice", 1);
            startActivity(pick);
    }
    public void choice_2(View view) {
            Intent pick = new Intent(this, Bus_Display.class);
            pick.putExtra("url",
"http://docs.google.com/gview?embedded=true&url=http://aata.org/rideguide/33inot.pdf");
            pick.putExtra("choice", 2);
            startActivity(pick);
    }
    public void choice_3(View view) {
            Intent pick = new Intent(this, Bus_Display.class);
            pick.putExtra("url", "http://mobile.aata.org/rideguide_m.asp?route=34");
            pick.putExtra("choice", 3);
            startActivity(pick);
    }
    public void choice_4(View view) {
            Intent pick = new Intent(this, Bus_Display.class);
            pick.putExtra("url",
"http://docs.google.com/gview?embedded=true&url=http://aata.org/rideguide/34inot.pdf");
            pick.putExtra("choice", 4);
            startActivity(pick);
    }
    public void main_menu_button_8(View view) {
            Intent btn = new Intent(this, EMUActivity.class);
            btn.setFlags(Intent.FLAG_ACTIVITY_CLEAR_TOP);
            startActivity(btn);
            finish();
    }
}
```

bus_display.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
android:layout_width="match_parent" android:layout_height="match_parent"
android:background="@drawable/background" android:orientation="vertical" >
```

```
<Button android:id="@+id/main_menu_btn_9" android:layout_width="match_parent"
android:layout_height="wrap_content" android:layout_alignParentBottom="true"
android:layout_alignParentLeft="true" android:background="@drawable/emu_button"
android:onClick="main_menu_button_9" />
<WebView android:id="@+id/bus_display_view" android:layout_width="match_parent"
android:layout_height="match_parent" android:layout_above="@+id/main_menu_btn_9"
android:layout_alignParentLeft="true" android:layout_alignParentTop="true" />
</RelativeLayout>
```

Bus_Display

```
package emu.bus;
import main.menu.EMUActivity; import main.menu.R; import android.app.Activity; import
android.content.Intent; import android.os.Bundle; import android.view.View; import
android.webkit.WebView; import android.webkit.WebViewClient;
public class Bus_Display extends Activity {
        private WebView myPick;
        private String my_url;
        private int received_choice;
   @Override
   public void onCreate(Bundle savedInstanceState) {
      super.onCreate(savedInstanceState);
      setContentView(R.layout.bus_display);
      Intent pick = getIntent();
      my_url = pick.getStringExtra("url");
      received_choice = pick.getIntExtra("choice", 1);
      myPick = (WebView) findViewById(R.id.bus_display_view);
      myPick.setWebViewClient(new WebViewClient());
      if ((received_choice == 2) || (received_choice == 4))
         myPick.getSettings().setJavaScriptEnabled(true);
      else
         myPick.getSettings().setJavaScriptEnabled(false);
      myPick.loadUrl(my_url);
   }
   @Override
   public void onBackPressed() {
         finish();
   }
   public void main_menu_button_9(View view) {
         Intent btn = new Intent(this, EMUActivity.class);
         btn.setFlags(Intent.FLAG_ACTIVITY_CLEAR_TOP);
         startActivity(btn);
         finish();
   }
}
```

Appendix P: Contact EMU code

contact_emu.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
android:layout_width="match_parent" android:layout_height="match_parent"
android:background="@drawable/background" android:orientation="vertical" >
<TextView android:id="@+id/contact_emu_title" android:layout_width="wrap_content"
android:layout_height="wrap_content" android:text="Contact EMU" android:textColor="#448275"
android:textSize="25px" android:layout_marginTop="30px" android:layout_marginBottom="30px"
android:layout_centerHorizontal="true" />
<ListView android:id="@+id/contact_emu_list" android:layout_width="match_parent"
android:layout_height="wrap_content" android:layout_above="@+id/main_menu_btn_22"
android:layout_alignParentRight="true" android:cacheColorHint="#00000000"
android:layout_below="@+id/contact_emu_title" />
<Button android:id="@+id/main_menu_btn_22" android:layout_width="match_parent"
android:layout_height="wrap_content" android:layout_alignParentBottom="true"
android:layout_alignParentLeft="true" android:background="@drawable/emu_button"
android:onClick="main_menu_button_22" />
</RelativeLayout>
```

Contact

```
package emu.contact;
import main.menu.EMUActivity; import main.menu.R; import android.app.Activity;
import android.content.Intent; import android.net.Uri; import android.os.Bundle; import
android.view.View; import android.widget.AdapterView; import android.widget.ArrayAdapter; import
android.widget.ListView;  import android.widget.AdapterView.OnItemClickListener;
public class Contact extends Activity {
        private final String[] titles = {"College of Arts and Sciences\n734.487.4344", "College of
Business\n734.487.4140","College of Education\n734.487.1416",
"College of Health and Human Services\n734.487.0077","College of
Technology\n734.487.0354","Graduate School\n734.487.0042","Honors
College\n734.487.0341","Extended Programs\n734.487.0407","Admissions\n1-800-GO-TO-
EMU","Alumni Relations\n734.487.0250","Eagles
Athletics\n734.487.1050","Commencement\n734.487.2300","I.T. Help Desk\n734.487.2120","Dining
Services\n734.487.0418","Financial Aid\n734.487.0455","EMU Foundation\n734.484.1322","EMU
Online\n734.484.1322","Housing\n734.487.1300","I.T. (Information
Technology)\n734.487.3141","Library\n734.487.0020","My.emich\n734.487.2120",
"Ombudsman\n734.487.0074","Office of the President\n734.487.2211","Parking\n734.487.3450","Pay
Online\n734.487.3335","Public Safety\n734.487.1222","Records and
Registration\n734.487.4111","Student Business Services\n734.487.3335","Student
```

```
Center\n734.487.1157","Tuition and Fees\n734.487.3335","Division of
Communications\n734.487.4400","Severe Weather Policy\n734.487.2460" };
        private final String[] phones =
{"7344874344","7344874140","7344871416","7344870077","7344870354","7344870042",
"7344870341","7344870407","18004686368","7344870250","7344871050","7344872300",
"7344872120","7344870418","7344870455","7344841322","7344841322","7344871300",
"7344873141","7344870020","7344872120","7344870074","7344872211","7344873450",
"7344873335","7344871222","7344874111","7344873335","7344871157","7344873335",
"7344874400","7344872460" };
        private ListView contact_emu_entries;
        @Override
        public void onCreate(Bundle savedInstanceState) {
                super.onCreate(savedInstanceState);
                setContentView(R.layout.contact_emu);
                contact_emu_entries = (ListView) findViewById(R.id.contact_emu_list);
                contact_emu_entries.setAdapter(new
ArrayAdapter<String>(this,android.R.layout.simple_list_item_1, titles));
                contact_emu_entries.setOnItemClickListener(new OnItemClickListener() {
                        public void onItemClick(AdapterView<?> parent, View view, int position, long
id) {
                                goIntent(phones[position]);
                        }
                });
        }
        @Override
    public void onBackPressed() {
        finish();
    }
        public void goIntent(String number) {
                String numberToDial = "tel:"+number;
       startActivity(new Intent(Intent.ACTION_DIAL, Uri.parse(numberToDial)));
        }
        public void main_menu_button_22(View view) {
                Intent btn = new Intent(this, EMUActivity.class);
                btn.setFlags(Intent.FLAG_ACTIVITY_CLEAR_TOP);
                startActivity(btn);
                finish();
        }
}
```

Appendix Q: GPA Calculator code

gpa_calculator.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
android:layout_width="match_parent"  android:layout_height="match_parent"
android:orientation="vertical" android:background="@drawable/background" >
<TextView android:id="@+id/gpa_title" android:layout_width="wrap_content"
android:layout_height="wrap_content" android:textColor="#448275" android:textSize="25px"
android:layout_centerHorizontal="true" android:layout_marginTop="30px" android:text="GPA
Calculator" />
<TextView android:id="@+id/gpa_credits_title_1" android:layout_width="wrap_content"
android:layout_height="wrap_content" android:textColor="#448275" android:textSize="20px"
android:layout_marginLeft="10px" android:layout_marginTop="120px" android:text="Credits:" />
<TextView android:id="@+id/gpa_grade_title_1" android:layout_width="wrap_content"
android:layout_height="wrap_content" android:textColor="#448275" android:textSize="20px"
android:layout_alignRight="@+id/gpa_credits_title_1" android:layout_marginTop="180px"
android:text="Grade:" />
<TextView android:id="@+id/gpa_credits_title_2" android:layout_width="wrap_content"
android:layout_height="wrap_content" android:textColor="#448275" android:textSize="20px"
android:layout_marginLeft="10px" android:layout_marginTop="280px" android:text="Credits:" />
<TextView android:id="@+id/gpa_grade_title_2" android:layout_width="wrap_content"
android:layout_height="wrap_content" android:textColor="#448275" android:textSize="20px"
android:layout_alignRight="@+id/gpa_credits_title_2" android:layout_marginTop="340px"
android:text="Grade:" />
<EditText android:id="@+id/gpa_credits_0" android:layout_width="60px" android:layout_height="60px"
android:layout_marginTop="105px" android:layout_marginLeft="125px" />
<EditText android:id="@+id/gpa_credits_1" android:layout_width="60px" android:layout_height="60px"
android:layout_marginLeft="250px" android:layout_alignBaseline="@+id/gpa_credits_0" />
<EditText android:id="@+id/gpa_credits_2" android:layout_width="60px" android:layout_height="60px"
android:layout_marginLeft="380px" android:layout_alignBaseline="@+id/gpa_credits_0" />
<EditText android:id="@+id/gpa_credits_3" android:layout_width="60px" android:layout_height="60px"
android:layout_marginTop="260px" android:layout_alignLeft="@+id/gpa_credits_0" />
<EditText android:id="@+id/gpa_credits_4"  android:layout_width="60px" android:layout_height="60px"
android:layout_marginTop="260px" android:layout_alignLeft="@+id/gpa_credits_1" />
<EditText android:id="@+id/gpa_credits_5" android:layout_width="60px" android:layout_height="60px"
android:layout_marginTop="260px" android:layout_alignLeft="@+id/gpa_credits_2" />
<Spinner android:id="@+id/gpa_grade0" android:layout_width="130px" android:layout_height="60px"
android:layout_marginLeft="90px" android:layout_marginTop="165px" />
<Spinner android:id="@+id/gpa_grade1" android:layout_width="130px" android:layout_height="60px"
android:layout_marginLeft="220px" android:layout_alignBaseline="@+id/gpa_grade0" />
<Spinner android:id="@+id/gpa_grade2" android:layout_width="130px" android:layout_height="60px"
android:layout_marginLeft="350px" android:layout_alignBaseline="@+id/gpa_grade0" />
```

```
<Spinner android:id="@+id/gpa_grade3" android:layout_width="130px" android:layout_height="60px"
android:layout_alignLeft="@+id/gpa_grade0" android:layout_marginTop="325px" />
<Spinner android:id="@+id/gpa_grade4" android:layout_width="130px" android:layout_height="60px"
android:layout_marginTop="350px" android:layout_alignLeft="@+id/gpa_grade1"
android:layout_alignBaseline="@+id/gpa_grade3" />
<Spinner android:id="@+id/gpa_grade5" android:layout_width="130px" android:layout_height="60px"
android:layout_alignLeft="@+id/gpa_grade2" android:layout_alignBaseline="@+id/gpa_grade3" />
<Button android:id="@+id/calculate"  android:onClick="calculate_gpa" android:text="Calculate GPA"
android:layout_marginTop="420px" android:layout_centerHorizontal="true"
android:layout_width="wrap_content" android:layout_height="60px" />
<TextView android:id="@+id/gpa_results_text" android:layout_centerHorizontal="true"
android:layout_width="wrap_content" android:layout_height="wrap_content"
android:textColor="#FFFFFF" android:text="" android:textSize="25px"
android:layout_marginTop="500px" />
<Button android:id="@+id/main_menu_btn_13" android:layout_width="match_parent"
android:layout_height="wrap_content" android:layout_alignParentBottom="true"
android:layout_alignParentLeft="true" android:background="@drawable/emu_button"
android:onClick="main_menu_button_13" />
</RelativeLayout>
```

GPA_Calculator

```
package emu.gpa;
import java.text.DecimalFormat; import main.menu.EMUActivity; import main.menu.R; import
android.app.Activity; import android.content.Intent; import android.os.Bundle; import android.view.View;
import android.widget.AdapterView; import android.widget.ArrayAdapter; import
android.widget.EditText; import android.widget.Spinner; import android.widget.TextView;
public class GPA_Calculator extends Activity implements AdapterView.OnItemSelectedListener {
        private final String[] grades = {"Select","A","A-","B+","B","B-","C+","C","C-","D+","D","D-","E"};
        private final double[] values = {0.0, 4.0, 3.7, 3.3, 3.0, 2.7, 2.3, 2.0, 1.7, 1.3, 1.0, 0.7, 0.0};
        private EditText gpa_credits0, gpa_credits1, gpa_credits2, gpa_credits3, gpa_credits4,
gpa_credits5;
        private int gpa_credits_0, gpa_credits_1, gpa_credits_2, gpa_credits_3, gpa_credits_4,
gpa_credits_5;
        private Spinner gpa_grade0, gpa_grade1, gpa_grade2, gpa_grade3, gpa_grade4,
gpa_grade5;
        private double gpa_grade_0, gpa_grade_1, gpa_grade_2, gpa_grade_3, gpa_grade_4,
gpa_grade_5;
        private TextView gpa;
        public int[] spinner_ids;
        public int spinner_check;
        public int spinner_index;
   @Override
   public void onCreate(Bundle savedInstanceState) {
      super.onCreate(savedInstanceState);
```

```
        setContentView(R.layout.gpa_calculator);
        spinner_ids = new int[6];
        spinner_check = 0;
        spinner_index = 0;
        gpa = (TextView) findViewById(R.id.gpa_results_text);
        gpa_credits0 = (EditText) findViewById(R.id.gpa_credits_0);
        gpa_credits1 = (EditText) findViewById(R.id.gpa_credits_1);
        gpa_credits2 = (EditText) findViewById(R.id.gpa_credits_2);
        gpa_credits3 = (EditText) findViewById(R.id.gpa_credits_3);
        gpa_credits4 = (EditText) findViewById(R.id.gpa_credits_4);
        gpa_credits5 = (EditText) findViewById(R.id.gpa_credits_5);
        gpa_grade0 = (Spinner) findViewById(R.id.gpa_grade0);
        gpa_grade1 = (Spinner) findViewById(R.id.gpa_grade1);
        gpa_grade2 = (Spinner) findViewById(R.id.gpa_grade2);
        gpa_grade3 = (Spinner) findViewById(R.id.gpa_grade3);
        gpa_grade4 = (Spinner) findViewById(R.id.gpa_grade4);
        gpa_grade5 = (Spinner) findViewById(R.id.gpa_grade5);
        gpa_grade0.setOnItemSelectedListener(this);
        gpa_grade1.setOnItemSelectedListener(this);
        gpa_grade2.setOnItemSelectedListener(this);
        gpa_grade3.setOnItemSelectedListener(this);
        gpa_grade4.setOnItemSelectedListener(this);
        gpa_grade5.setOnItemSelectedListener(this);
        ArrayAdapter<String> individual_grades = new ArrayAdapter<String>(this,
android.R.layout.simple_spinner_item, grades);

            individual_grades.setDropDownViewResource(android.R.layout.simple_spinner_dropdown_ite
m);
        gpa_grade0.setAdapter(individual_grades);
        gpa_grade1.setAdapter(individual_grades);
        gpa_grade2.setAdapter(individual_grades);
        gpa_grade3.setAdapter(individual_grades);
        gpa_grade4.setAdapter(individual_grades);
        gpa_grade5.setAdapter(individual_grades);
    }
    public void onRestart() {
            super.onRestart();
            gpa.setText("");
            spinner_ids = new int[6];
            spinner_check = 0;
            spinner_index = 0;
    }
    public void onItemSelected(AdapterView<?> parent, View v, int position, long id) {
            int b = parent.getId();
            if (spinner_index < 6) {
                    spinner_ids[spinner_index] = b;
                    spinner_index++;
```

```
        }
        setSpinner(b, position);
}
public void onNothingSelected(AdapterView<?> parent) {
}
public void setSpinner(int spin_id, int value) {
        if (spin_id == spinner_ids[0])
                gpa_grade_0 = values[value];
        else if (spin_id == spinner_ids[1])
                gpa_grade_1 = values[value];
        else if (spin_id == spinner_ids[2])
                gpa_grade_2 = values[value];
        else if (spin_id == spinner_ids[3])
                gpa_grade_3 = values[value];
        else if (spin_id == spinner_ids[4])
                gpa_grade_4 = values[value];
        else
                gpa_grade_5 = values[value];
}
public void calculate_gpa(View view) throws Exception {
        try {
                if (!gpa_credits0.getText().toString().equals("")) {
                        gpa_credits_0 = Integer.parseInt(gpa_credits0.getText().toString());
                }
                else {
                        gpa_credits_0 = 0;
                        gpa_grade_0 = 0.0;
                }
                if (!gpa_credits1.getText().toString().equals("")) {
                        gpa_credits_1 = Integer.parseInt(gpa_credits1.getText().toString());
                }
                else {
                        gpa_credits_1 = 0;
                        gpa_grade_1 = 0.0;
                }
                if (!gpa_credits2.getText().toString().equals("")) {
                        gpa_credits_2 = Integer.parseInt(gpa_credits2.getText().toString());
                }
                else {
                        gpa_credits_2 = 0;
                        gpa_grade_2 = 0.0;
                }
                if (!gpa_credits3.getText().toString().equals("")) {
                        gpa_credits_3 = Integer.parseInt(gpa_credits3.getText().toString());
                }
                else {
                        gpa_credits_3 = 0;
```

```
                gpa_grade_3 = 0.0;
            }
            if (!gpa_credits4.getText().toString().equals("")) {
                gpa_credits_4 = Integer.parseInt(gpa_credits4.getText().toString());
            }
            else {
                gpa_credits_4 = 0;
                gpa_grade_4 = 0.0;
            }
            if (!gpa_credits5.getText().toString().equals("")) {
                gpa_credits_5 = Integer.parseInt(gpa_credits5.getText().toString());
            }
            else {
                gpa_credits_5 = 0;
                gpa_grade_5 = 0.0;
            }
            int credits_attempted = gpa_credits_0 + gpa_credits_1 + gpa_credits_2 +
gpa_credits_3 + gpa_credits_4 + gpa_credits_5;
            double credit_pts_earned = gpa_credits_0*gpa_grade_0 +
gpa_credits_1*gpa_grade_1 + gpa_credits_2*gpa_grade_2 + gpa_credits_3*gpa_grade_3 +
gpa_credits_4*gpa_grade_4 + gpa_credits_5*gpa_grade_5;
            double calc_gpa = credit_pts_earned/credits_attempted;
            DecimalFormat gpa_format = new DecimalFormat("#.##");
            String final_gpa = gpa_format.format(calc_gpa);
            String res = "Your GPA is: " + final_gpa;
            gpa.setText(res);
        }
        catch (Exception e) {
            gpa.setText("Wrong input, please try again.");
        }
    }
    @Override
    public void onBackPressed() {
        finish();
    }
    public void main_menu_button_13(View view) {
        Intent btn = new Intent(this, EMUActivity.class);
        btn.setFlags(Intent.FLAG_ACTIVITY_CLEAR_TOP);
        startActivity(btn);
        finish();
    }
}
```