



VAASAN AMMATTIKORKEAKOULU
VASA YRKESHÖGSKOLA
UNIVERSITY OF APPLIED SCIENCES

Richard Adeyeye

VAMK ANDROID APPLICATION

Technology and Communication

2014

VAASAN AMMATTIKORKEAKOULU
UNIVERSITY OF APPLIED SCIENCES
Degree Program in Information Technology

ABSTRACT

Author	Adeyeye Richard
Title	VAMK Android Application
Year	2014
Language	English
Pages	37 + 22 Appendices
Name of Supervisor	Ghodrat Mohgadampour

With the dynamic nature and fast growing nature of Information Technology in the recent decade, the Android operating system has taken over the tablets and mobile phones operating systems. As a result of this, VAMK android application needed to be developed so that the school Web information can be accessed easily by everyone on all mobile devices running the Android operating system.

The project was aimed at making some frequently needed information, such as the timetable, map location, list of registered courses, email and news on the school webpage available on mobile devices with the Android operating system. Other features of the application are possibility to send SMS, and taking photo by starting the device camera. The application has a server side part implemented in PHP, which takes of sending and retrieving data from a MySQL database used by the application.

The application was implemented successfully and met all the requirements set for it. There are still some parts, such as the timetable page which can be further developed in such a way that the user will be able to view the pages based on the current school calendar period. Also, the map page can be further developed so that user will be able to search for places.

ACKNOWLEDGEMENT

I really appreciate friends and family for their contributions in one way or other towards the completion of this project. Special appreciations to the supervisor of this project work, Dr. Ghodrat Moghadampour, for his time and patience. I also appreciate all the teachers who have been guiding me all through my study time. Above all, I say thank you God Almighty for keeping me and giving me the grace to complete this study.

ADEYEYE Richard Adeyinka.

CONTENTS

ABSTRACT

1	INTRODUCTION	5
2	RELEVANT TECHNOLOGIES	6
	2.1 Android Application	6
	2.2 Application Fundamentals	6
3	APPLICATION DESCRIPTION	8
	3.1 Installation of Eclipse and Android SDK	8
	3.2 Description, Requirements, Objectives and Constraints of the Project	8
	3.3 Quality Function Deployment.....	8
	3.4 Application Main Building Blocks	10
	3.5 Application Functions' Detailed Descriptions.....	11
	3.6 Application Building Blocks.....	13
	3.7 Application Working Diagram	13
4	DATABASE AND GUI DESIGN	15
	4.1 Design of the Database	15
	4.2 Design of Different Parts of GUI.....	17
5	IMPLEMENTATION	19
	5.1 Implementation of Different Parts	19
6	TESTING	26
	6.1 Test Cases	26
	6.2 Description of the Results of Implementation of Each Test Case	26
	6.3 Possible Improvements Made after Testing.....	28
7	CONCLUSIONS	29
	7.1 Future Work.....	29
	REFERENCES.....	31
	APPENDICES	

LIST OF FIGURES AND TABLES

Figure 1	Application Components	p. 12
Figure 2	Use-case Diagram Showing all Activities	p. 14
Figure 3	Application Main Building Block diagram	p.15
Figure 4	Application Log in Functions Diagram	p. 15
Figure 5	Application Register Functions Diagram	p. 16
Figure 6	Application Functions Diagram	p. 16
Figure 7	Application Building Block Diagram	p.17
Figure 8	Application Working Diagram	p. 17
Figure 9	Database diagram	p.18
Figure 10	Deployment diagram	p.21
Figure 11	Register Page	p.20
Figure 12	Login Page	p.21
Figure 13	Menu List Page	p.21
Figure 14	Register Page	p.24
Figure 15	Students Login Page	p. 25
Figure 16	User's Page	p.27
Figure 17	News Page	p.27
Figure 18	Maps Page	p. 28
Figure 19	Course Page	p.28
Figure 20	Timetable Page	p.29
Figure 21	Contacts Information	p. 29

Figure 22 Application Main Menu

p.30

LIST OF APPENDICES

APPENDIX 1.1. School.java

APPENDIX 1.2. MainActivity.java

APPENDIX 1.3. Register.java

APPENDIX 1.4. Menu.java

APPENDIX 1.5. Map.java

APPENDIX 1.6 Map Layout

APPENDIX 1.7 AndroidManifest.xml

1 INTRODUCTION

Google's Android mobile phone software platform may be the next big opportunity for application software developers. Google announced the Open Handset Alliance and the Android platform in November of 2007, releasing the first beta version of the Android Software Development Kit (SDK) at the same time. There are already several competing mobile phone software stacks in the market, so why is there such a great interest in Android? Resources are also available for developers to get to know of how to develop applications on it as well as earning a living by selling them in the Android Market, now called Google Play. Since the concept is the same, various user interfaces and functionalities can be implemented. Android has the potential for removing the barriers to success in the development and sale of a new generation of mobile phone application software. /3/

Android uses SQLite to manage its own private database. SQLite is based on SQL (Structured Query Language) syntax to manage data in mobile phones and tablets. As the name suggests, it is very light. SQLite helps in the creation of a database, insertion of records, updating of records and deleting of records. /2/

These have been the motivating factors and the reasons why VAMK Android application is developed so that some functions that are readily available on Android platform are implemented and used for the benefit of the students and staff of VAMK.

2 RELEVANT TECHNOLOGIES

2.1 Android Application

An Android application is a program which is developed to run on any device with an Android operating system. It is an open source mobile platform with no charging fees, emerged as a new mobile development option that offers many benefits over competing platforms. The native Android applications are written in Java./7/

The Android applications can be developed on a variety of operating systems including the Windows XP (32-bits), Vista (32 or 64-bits), Windows 7 (32 and 64-bits), Mac OS X 10.5.8 and Linux (tested on Linux Ubuntu 8.04 LTS Hardy Heron). /2/

2.2 Application Fundamentals

When developing an Android application, the source code is compiled with the Android SDK tools into a single installable package in apk format. During the packaging, data and resource files are also included. The apk file is then installed on any device with an Android operating system.

An application code runs in isolation from other applications because by default every application runs its own Linux process. Each parts of the Android package play very important roles in the completion of any working project.



Figure 1. Application Components /6/

Application components are the essential building blocks of an Android application.

1. Activities are similar to the standalone utilities on the desktop systems. They can interact with the user and request data or services from other activities or services via queries or Intents. /3/
2. Services are executable pieces of codes that run in the background without being exposed to the user interfaces.
3. Components are activated through intents. Activity intents facilitate the creation of new applications from existing applications (mobile mashup). /3/
4. A manifest file is created by ADT when a new Android project is created. It is an XML file. It represents an android deployment descriptor which lists any activity, services, content provider, or receiver contained in the application, along with the appropriate intent filters supported by the application.
5. We also have an application resources folder where drawable resources, layout resources and values folders are kept. These folders are generated automatically and they can be edited.

3 APPLICATION DESCRIPTION

3.1 Installation of Eclipse and Android SDK

To start the project, we have to carry it out in the eclipse environment using Android Software Development Kit (SDK). It is first checked if the eclipse is already installed on the computer but if not, check the system requirement is checked and the highlighted steps below are followed;

Double click on eclipse installable file, eclipse.exe. This will start the IDE and prompt to creating a workspace in the chosen directory. Thereafter, we obtain and install the Android SDK.

1. Eclipse from <http://www.eclipse.org/downloads/>
2. Android SDK from <http://developer.android.com/sdk/installing/index.html> to extract the required google API library to fix some bugs.
3. Install the Eclipse ADT(Android Development Tools) plug-in by following the steps in the following link
<http://developer.android.com/sdk/installing/installing-adt.html>

3.2 Description, Requirements, Objectives and Constraints of the Project

The purpose of the project was to develop an Android application that will contain some essential information from the school webpage and adding one or two functions to the application. To be able to use the application, the user must be able to register their information on the register page and login to access all the important features embedded in the app. All the user registered information is stored in a database for easy access.

3.3 Quality Function Deployment

This category is divided into three (3) parts, which are;

Normal Requirements (must have)

Users must be able to;

- Register: allows new user to register and have access to the application,
- Login: allows the user to login with the registered ID,
- Send sms: allows the user to send sms through the application
- Send email: allows user to send email through the application
- Make calls: allows the user to make calls to contacts through the application
- Take pictures through the in-built camera and re-used
- Get current location on the map automatically,
- View timetable page,
- View contact page, and
- View course(s) page,

Expected Requirements (should have)

The application will be user friendly. Users will be able to register easily and play around with all the important features on the application and help in getting certain information from the school. Therefore the application must have a login page for the username and password, so also a register page where the new user (s) will be able to register and enter their data.

Optional Requirements (nice to have)

- WinhaWille: to check the user's academic records
- Exams and Tests planner or reminders.

From the diagram below, we can see how the application works. The user and the administrator have access to all the features on the application. The database access is limited only to the administrator. This is always so for security reasons. The dependency of each activity on one another can also be seen. The user must register before he or she can login and have access to the features inside the menu list. The administrator also has access to all the information of the registered users. All the user data are stored in the database on the remote server. The server side was built using PHP which is used to populate the database remotely. On the client side, the user, Java is used for the design. There is login servlet which is responsible for the login of the user and authorizes the user to have access to all the features on the application.

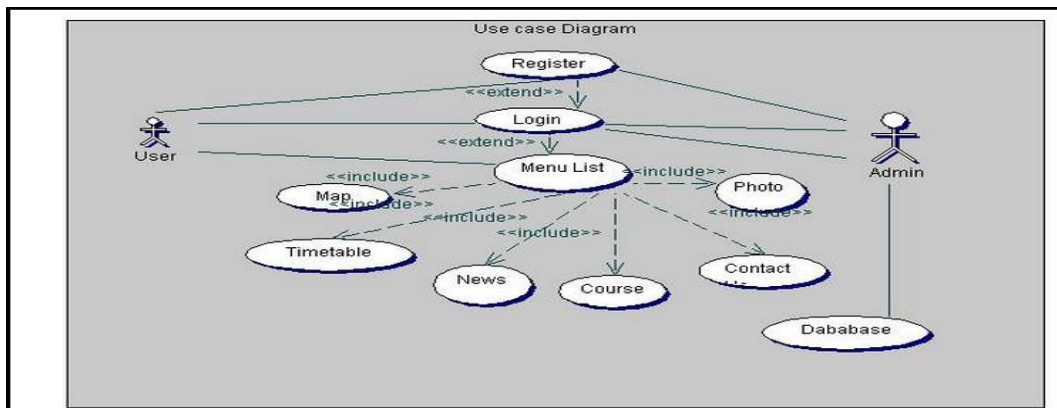


Figure 2. Use case diagram showing all the activities

3.4 Application Main Building Blocks

This diagram shows the declarations of each parameter in the different classes. It shows the connections, and the connectivity between the classes. It shows how each class is programmed. It shows that once the user is registered, the data are saved in the database which will help in the start-up and prompting the login.

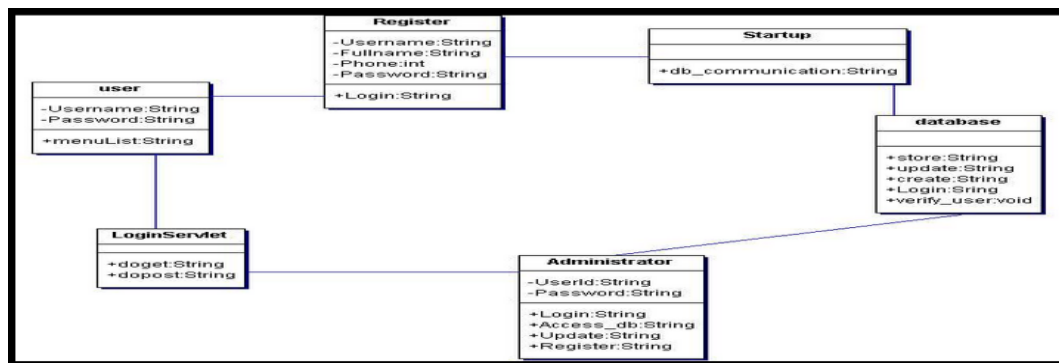


Figure 3.2. Application main building blocks diagram

3.5 Application Functions' Detailed Descriptions

The phase shows what happened in each stage and after each class. The new user has to first create an account by registering. The information provided in the registered page is saved inside the database with the help of query written in PHP. The user will be able to login into the application with the help of the loginPageServlet. The loginPageServlet checks the login information which will query the database to check if the user has already registered. When this is confirmed and logged in, the main page will be displayed. The main page contains the application menu list which holds all the important features on the application.

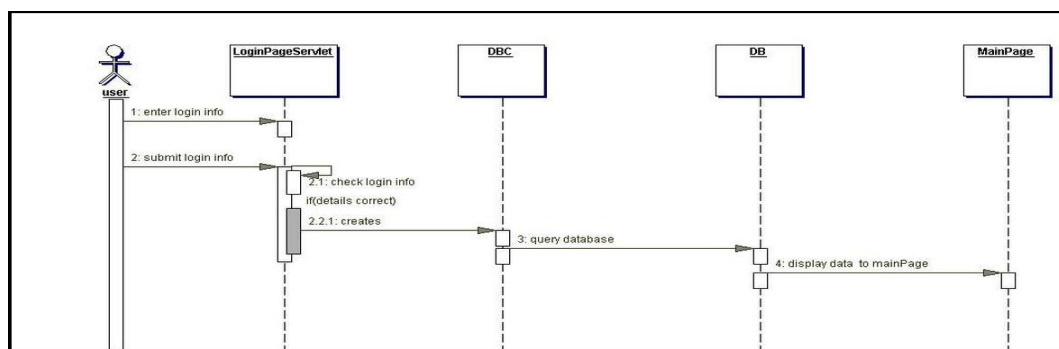


Figure 3. Application Login function diagram

The diagram above illustrates the sequence of login into the application. When the user enters the username and password, submit button is pressed. The login servlet checks the entered data and queries the database. When the parameters are accepted, the application main page will be displayed.

In figure 4 below, the registration sequence is explained diagrammatically. The user fills in all the required fields on the form and submits. The parameters will be saved in the database but if any of the fields is omitted, the application gives an error message and returns to the register page where the user will have to complete the form. This is possible with the help of the select method inside the register.php which enters the parameters into the database.

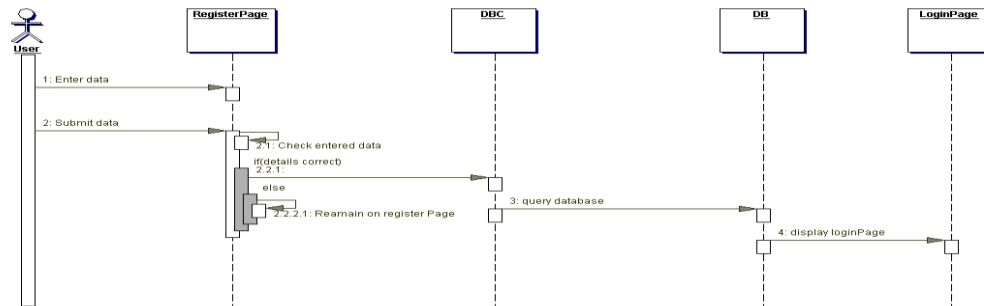


Figure 4. Application register function diagram

When all the required fields have been filled and accepted or saved in the database, the login page will open which will prompt the user to enter the registered username and password.

Figure 5 below shows the overall function sequence of the application. Here, the user is registered in to the application and saved in the database. When this has been done, the login page opens where the user should enter the registered username and password.

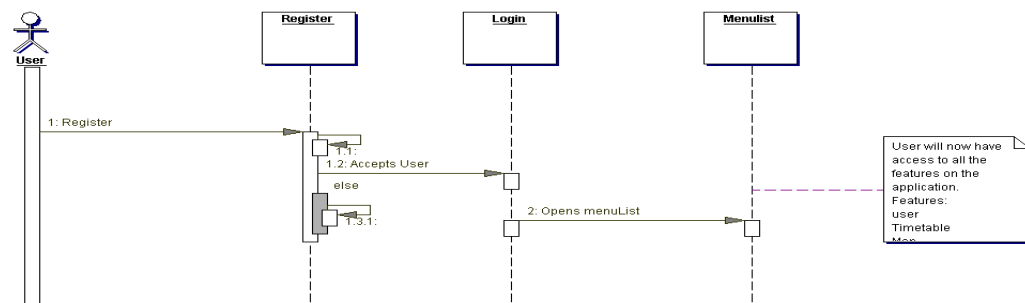


Figure 5. Application function diagram

When all the steps have been taken, the menu list page will open. Then the user has access to all the embedded features on the application.

3.6 Application Building Blocks

The different phases of the application are presented next. The application is divided into three different phases which are the controller, View and the database phase. The controller holds the registration; the user is the view (client) and the administration is in the database. These classes are the most important parts of the application and are the foundation of the application. After the user has registered, he or she will be able to login into the application with the information provided in the register page after the database has confirmed the user from the users table. This can be seen from the picture below.

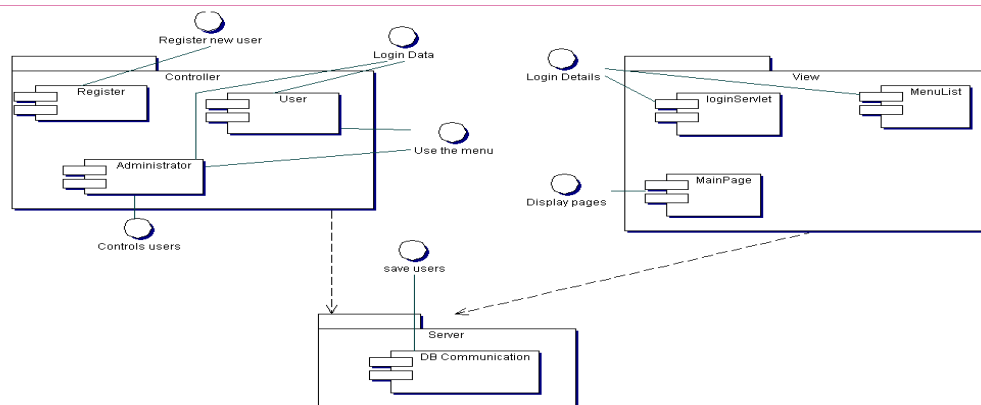


Figure 6. Application building blocks diagram

3.7 Application Working Diagram

The application working diagram shows what happens in each stage of the application and how the application works. The user needs to register and all the data will be saved on the database where the user can still retrieve and use it. After the registration, the user is logged in with the username and password created. After the login, the menu list opens where the user will be able to use all the important features on the application, such as Map, Timetable, and Courses etc

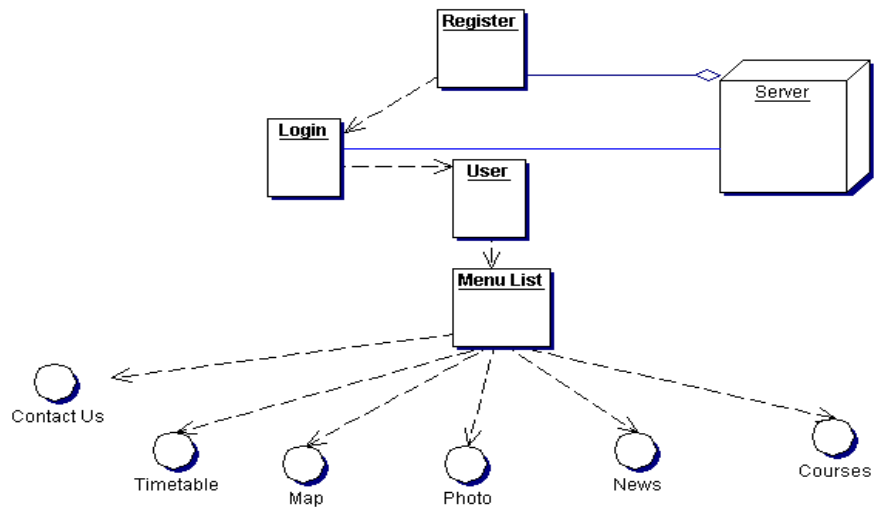


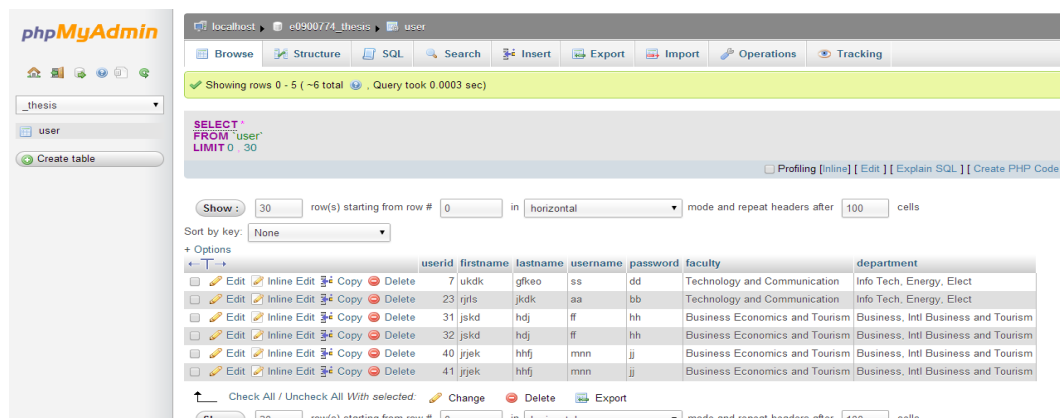
Figure 7. Application working diagram

4 DATABASE AND GUI DESIGN

4.1 Design of the Database

The database is created using MySQL. The database is used to populate, store and access link to all the information entered in the register page which the user can later recall for future use. Data are saved inside the specific tables created to hold specific data on the remote school server (www.mysql.cc.puv.fi). The server is written using PHP. The PHP will be used to parse the data to JSON format in order to be used for the server operations.

In order to create a communication between the database and the application, a URL handler is created which is a method that will help to link the Java operation with the PHP and deposit them on the server.



The screenshot shows the phpMyAdmin interface for a database named 'e0900774_thesis'. The 'user' table is selected, and a SQL query is executed: `SELECT * FROM 'user' LIMIT 0, 30`. The table structure is as follows:

userid	firstname	lastname	username	password	faculty	department
7	ukdk	gkeo	ss	dd	Technology and Communication	Info Tech, Energy, Elect
23	rjrls	jdkk	aa	bb	Technology and Communication	Info Tech, Energy, Elect
31	jskd	hdj	ff	hh	Business Economics and Tourism	Business, Intl Business and Tourism
32	jskd	hdj	ff	hh	Business Economics and Tourism	Business, Intl Business and Tourism
40	jrjk	hhfj	mnn	jj	Business Economics and Tourism	Business, Intl Business and Tourism
41	jrjk	hhfj	mnn	jj	Business Economics and Tourism	Business, Intl Business and Tourism

Figure 8. Database diagram

To design the database for the application, a table called; User, is created. This User will be holding all the information filled by the new user from the register page. These information stored in the User will be accessed the user anytime the user login to the application either on an old or new device. This means, there is no need to re-register in other to use the application when the device is changed.

Inside the user table, we have different columns holding the username, password, first_name, last_name, email, phone number, faculty and the department. PHP codes are also created to help populate the table and control activities inside the

database and to link the User table to the school server. The PHP folders are saved to the public html folder inside the U: drive in the student account. This is where it can be accessed by the school server. The db.php creates the table and always depends on the function.php.

The connect.php controls the connections between the database and the user. It holds the database and server username and password. The function.php handles the functions in the database and the application by storing the users' information. Login.php holds the username and password which will be used to log into the application. It saves the two parameters and the user can call for them from the login.php. Index.php holds all the parameters in the register page and serves as an engine room which controls all the parameters in the database by storing them. The test.php tests the workability of the database.

On the Java side from the database.java, different methods will be used to communicate with the database. These methods perform all the operations on the database from the Java side. It makes the login request and the http request in order to login to the application with the information saved in the database which is on the server. The serverTask java controls the server and the database by performing some background work which is so vital to the database-server operation using the AsyncTask method.

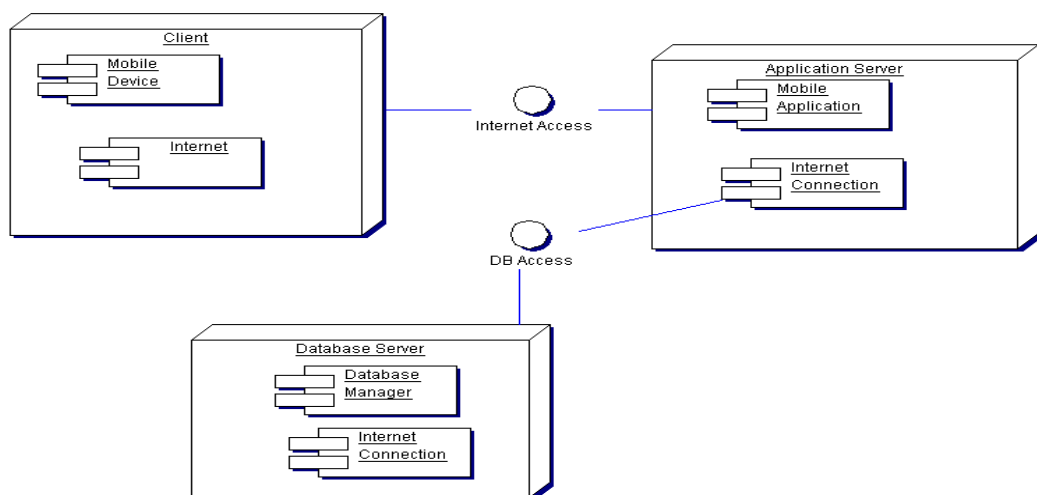
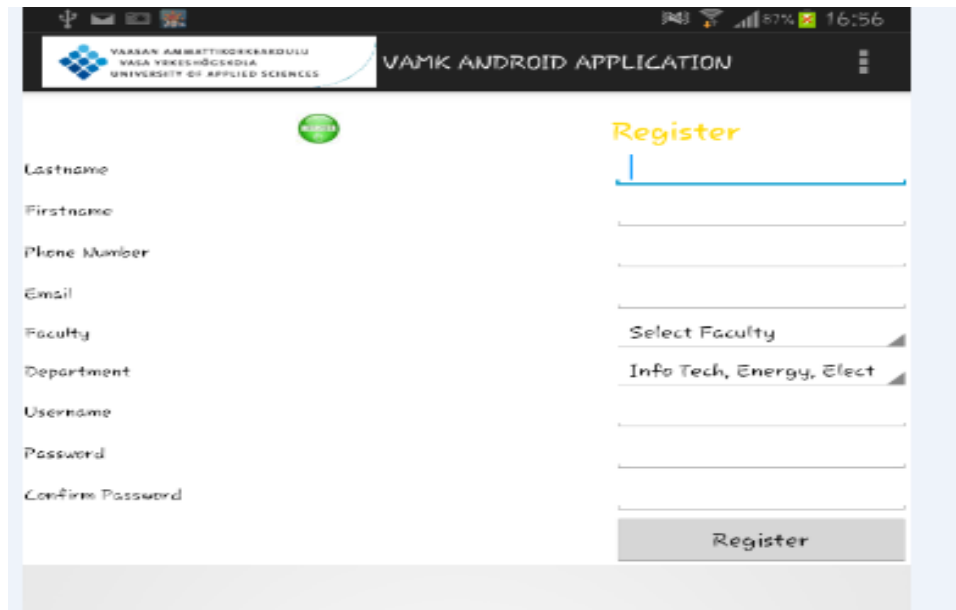


Figure 9. Deployment diagram

4.2 Design of Different Parts of GUI

Register Page

The register page was designed using xml. Each parameter has individual characteristics. The faculty and department were designed using a spinner. The method enables the user to select the faculty or department of his/her choice.



The screenshot shows the 'Register' page of the VAMK Android Application. The page is titled 'Register' in yellow text. On the left side, there is a list of input fields: Lastname, Firstname, Phone Number, Email, Faculty, Department, Username, Password, and Confirm Password. On the right side, there is a form with corresponding input fields. The Faculty and Department fields are spinners. The Faculty spinner is set to 'Select Faculty' and the Department spinner is set to 'Info Tech, Energy, Elect'. At the bottom of the form, there is a 'Register' button.

Figure 10. Register page

The register page was linked to the server using the register.php which will automatically populate the user database.

Login Page

The login page was designed using xml. It allows the user to enter their registered username and password. The page is linked to the server using the login.php which is also saved on the user database.

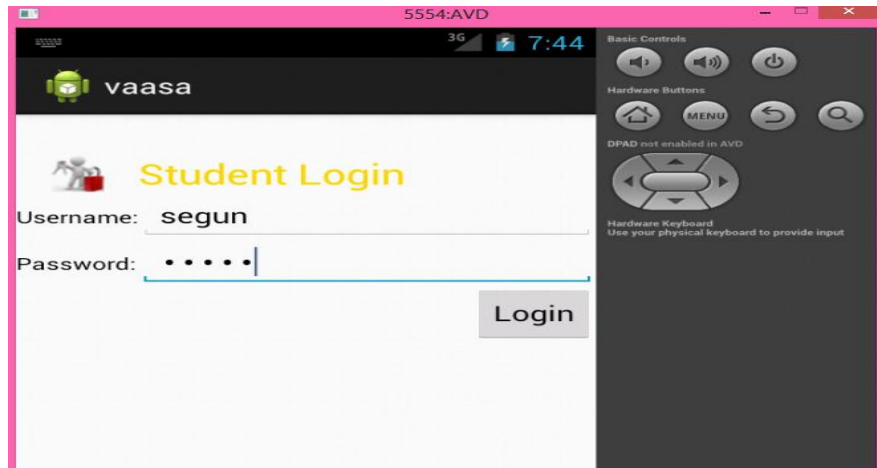


Figure 11. Login Page

The page was designed in such a way that it does not accept null variables or wrong Ids. All the parameters entered here are first checked on the database before it's accepted or proceeding to the next page.

Menu Lists Page

Menu list page contains all the icons to the important features embedded into the application. This was designed using xml. All the icon pictures are saved inside the drawable folder of the application workspace.

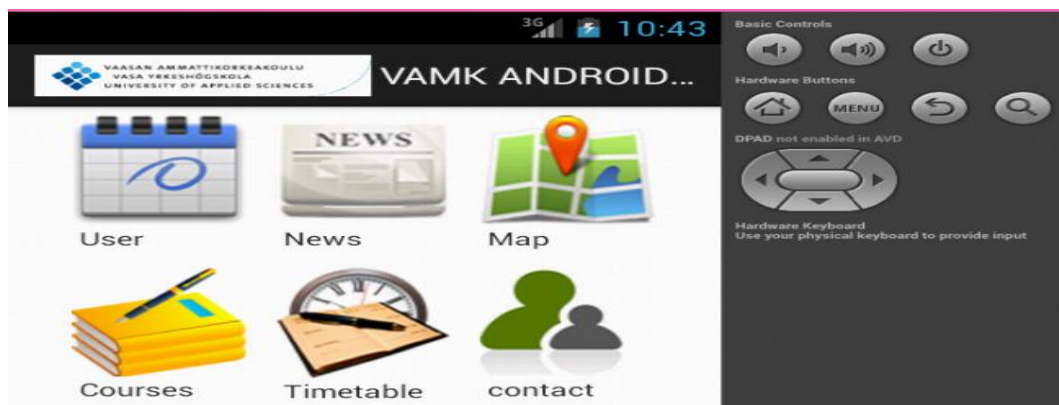


Figure 12. Menu lists page

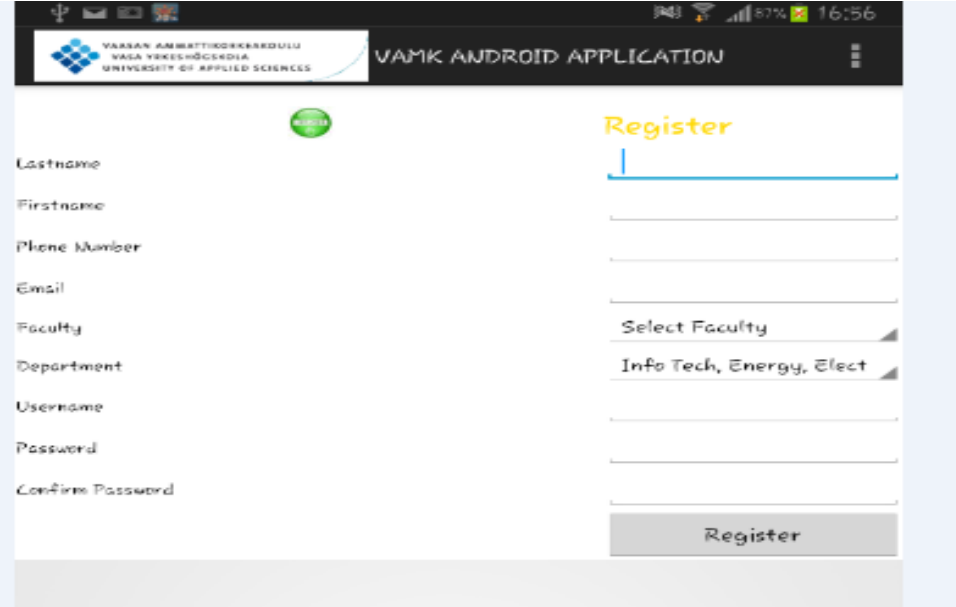
The page is the communication link between the user and application. The user will be able to access the features by clicking on the icons on the menu lists page.

5 IMPLEMENTATION

The application development is divided into two main parts which are the Java side and the server side in PHP. The Java side is used for the implementation of the features on the menu list of the application. The menu list of the application contains features like the user, timetable, courses, map, contact and the news page. The functions, methods of implementation and some details of each features will be enumerated fully soon. The server-database part of the implementation, controls the register and login side of the application. The server-database part were created to save register information and the login details so that the application can remember the user and be able to retrieve data anytime, and anywhere even with the change in the device.

5.1 Implementation of Different Parts

Users are allowed to register into the application in order to have access to the information on the application. The register page is as shown below.



The screenshot shows the 'Register' page of the VAMK Android Application. The page features a header with the VAMK logo and the text 'VAMK ANDROID APPLICATION'. Below the header, there is a green circular icon with the text 'REGISTER'. The main content area contains a registration form with the following fields: Lastname, Firstname, Phone Number, Email, Faculty (with a dropdown menu showing 'Select Faculty'), Department (with a dropdown menu showing 'Info Tech, Energy, Elect'), Username, Password, and Confirm Password. A 'Register' button is positioned at the bottom right of the form.

Figure 13. The Register page

The register page was implemented by creating an XML page for the register page and a Java method was used to control the page. Thereafter, the register page was linked the database with the register.php file through the register.java file.

Here, all the data are created and linked to the user table created in the database where they are saved and could be retrieved for future use.

Each user will be able to login with own created username and password. Thereafter, he or she will be prompted to the next page.

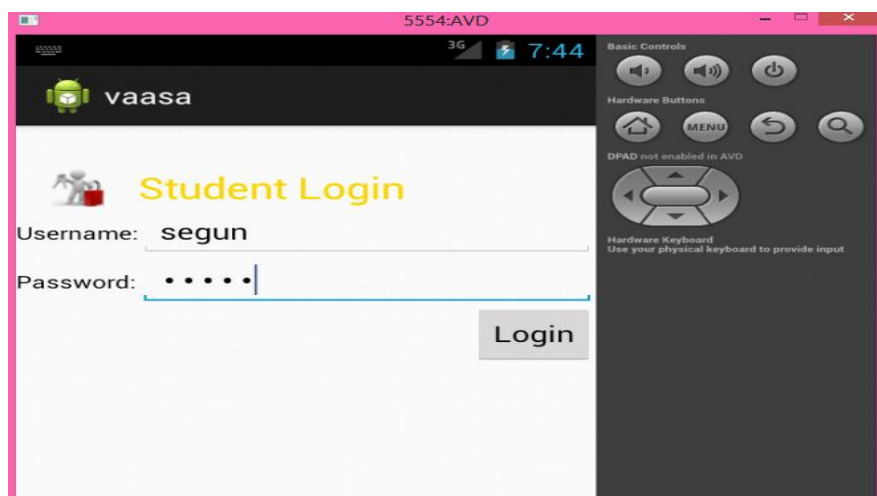


Figure 14. Student login page

The method of implementation is stated below. The username and password saved on the database will be called through the login.java and received from the database with the help of the login.php

Immediately after the user click, OK, he or she will be able to have access to all the features in the application menu and use the application. The page is designed in Java but the icons are defined in xml file for the menu list.

- User icon contains some tasks like call, sms, email, etc and the user has the privilege of choosing which one after the other. The page is designed using the onActivityResult method. The method will be used to start each activity on the user using an IF statement. The user can make a call through the default dial intent on the phone;


```
protected void onActivityResult(int requestCode, int resultCode, Intent data) {
    if (requestCode == task) {/*activity codes here*/}
```

The sms page is created with an sms manager method.

```
try {
    SmsManager smsManager = SmsManager.getDefault();
    smsManager.sendTextMessage(number, null, sms, null, null);
```

This will open the user defined page in the xml page to send sms.

The email page is also created using the Gmail sender method below, which is a user defined method.

```
try {
    GmailSender sender = new GmailSender(emailId, spass);
    sender.sendMail(sub,body,emailId,to);
```

The photo will open the camera intent which prompts the user to take picture and save it on the phone memory.



Figure 15. User's page

- News page displays the up-to-date events/activities going on in the school. There is a method for displaying the url inside a web view client. The web view client will display the page on the device and not on the web-browser.

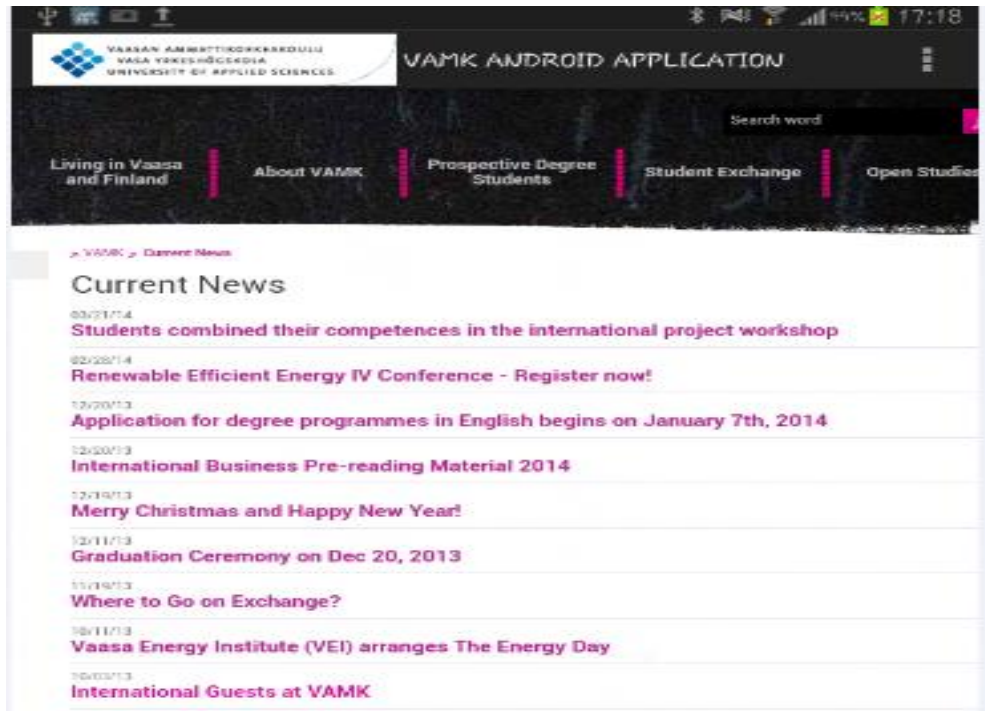


Figure 16. News page

- Map gives the location of places and from the application; we can determine the present location of the user. To execute the map, we can use location manager;

```

LocationManager mLocationManager = (LocationManager)
getSystemService(LOCATION_SERVICE);

        mLocationManager.requestLocationUpdates(LocationManager.GPS_PROVIDER,
LOCATION_REFRESH_TIME,
        LOCATION_REFRESH_DISTANCE, mLocationListener);

```

This will always prompt the GPS which will be activated to determine the present location of the user.



Figure 17. Map page

- Courses page will display all the courses the user registered for and offering as displayed on the user portal page.

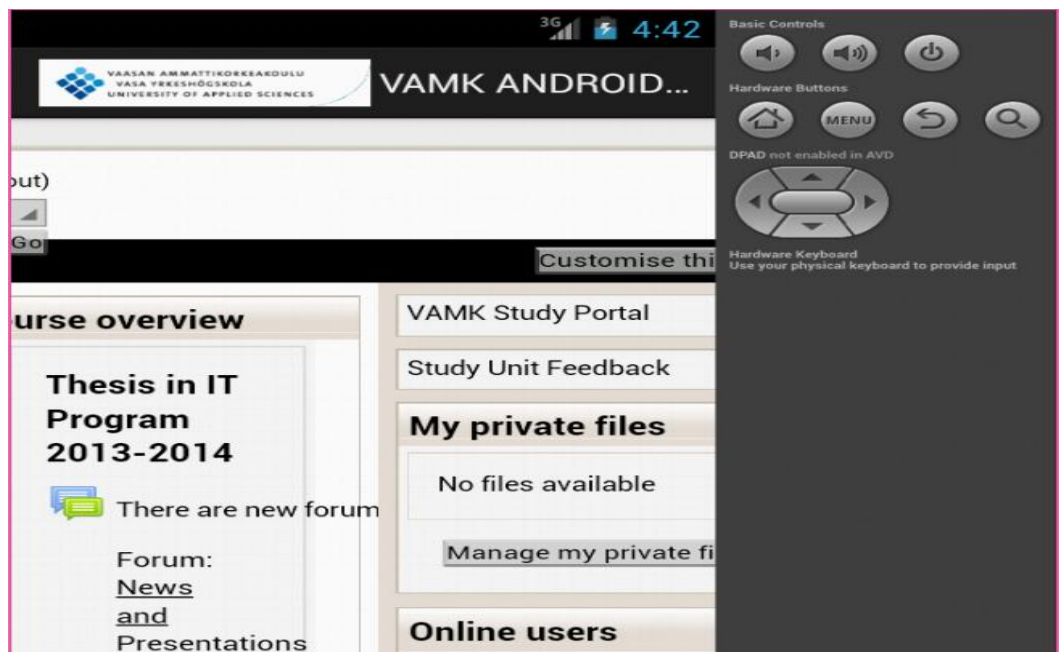


Figure 18. Courses page

- Timetable page will display the timetable according to the department registered with. This is as shown in figure 19 below.

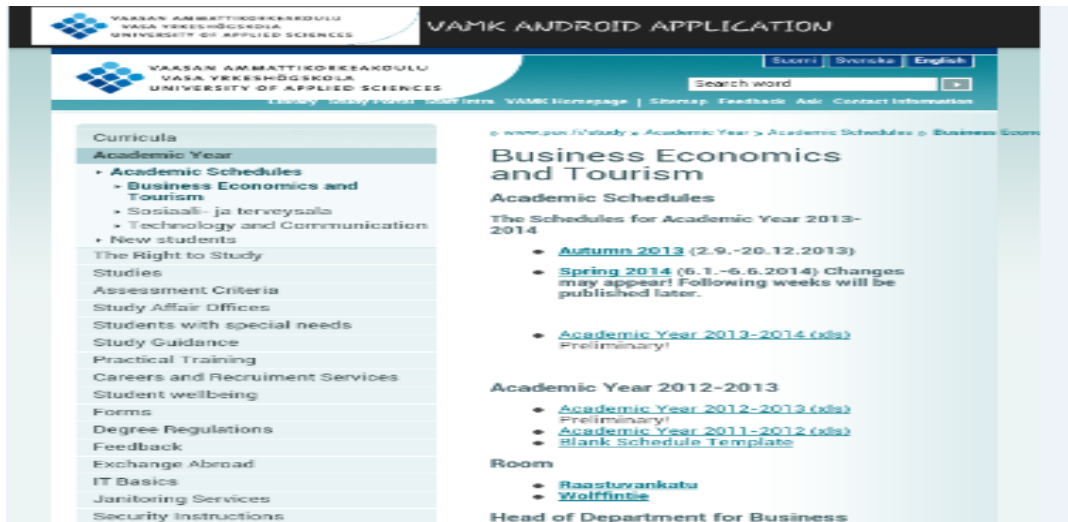


Figure 19. Timetable page

- Contact icon shows the basic information about the institution location and how to contact the school, such as phone number and address

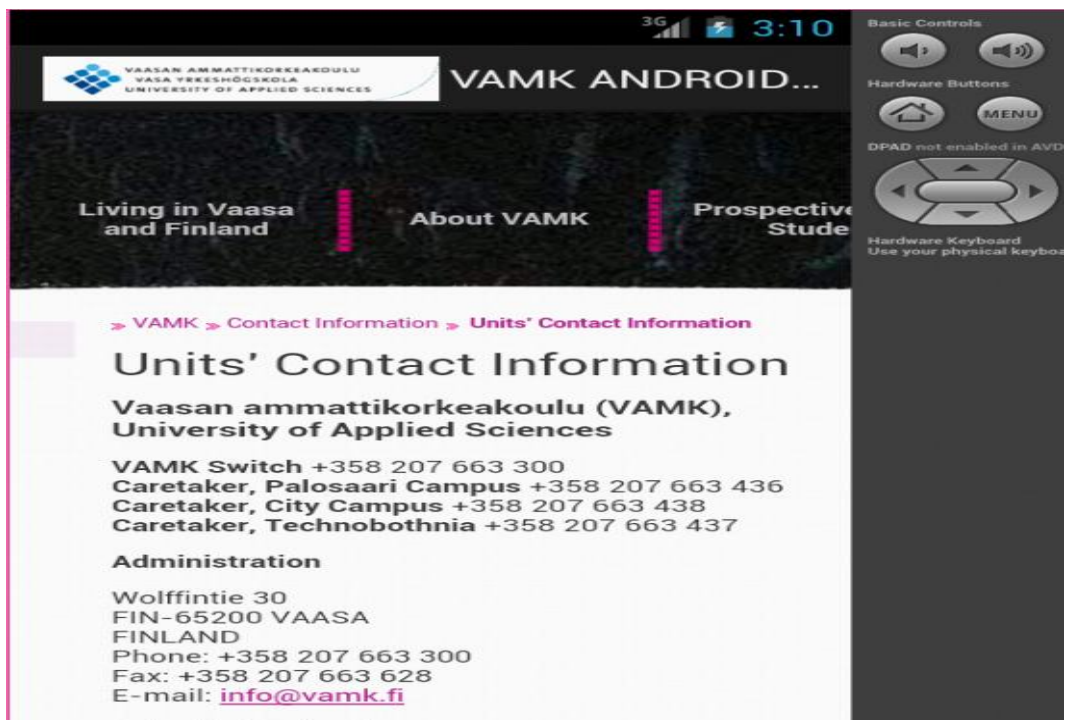


Figure 20. Contact information

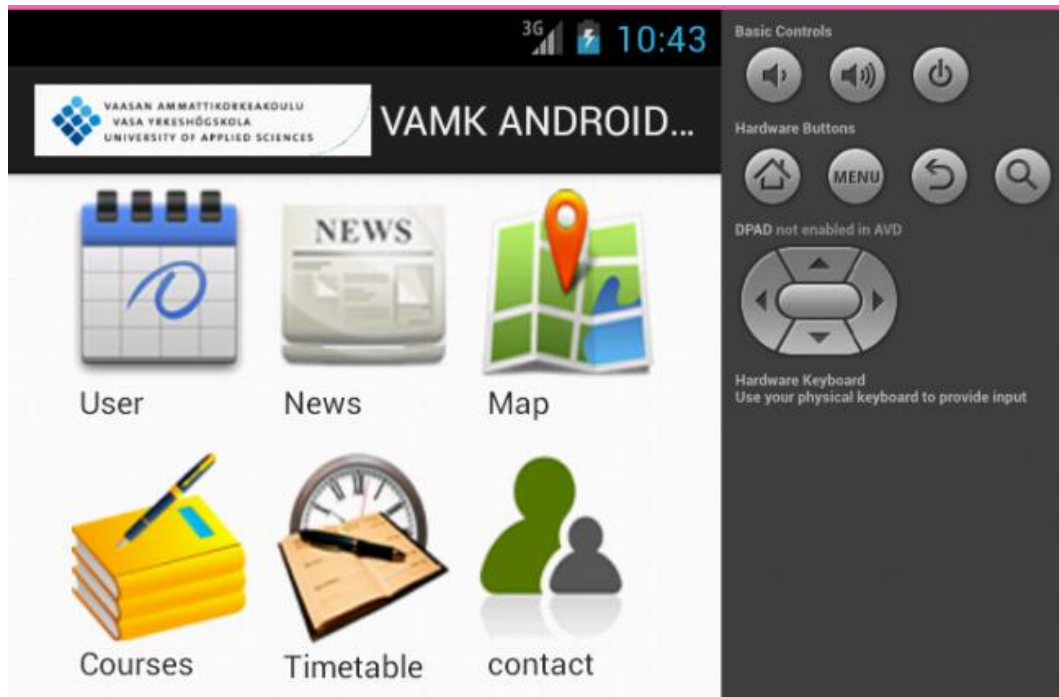


Figure 21. The application main menu

6 TESTING

6.1 Test Cases

After the source codes were written, bugs fixed and necessary errors corrected. It was time to test the application.

For this project and for quick debugging, a device with an Android operating system preferably android version 4.1.2 (Jelly bean) was used.

Different parts of the application, such as the register page, login page, menu page, and each components of the menu were tested.

6.2 Description of the Results of Implementation of Each Test Case

The Register Page

The register page was working perfectly well as expected as shown in figure 13 above. The new user can easily register into the application without any problem and all the user's information were saved in the database. The register page and the created database were communicating perfectly. After each successful registration, there is always a toast message displayed.

The Welcome Page

This is the next pop-up page after the user has registered. It is the confirmation that the information provided in the register page has been accepted and you can log in to continue using the application.



Figure 22. Welcome Page

This page confirms that the user has been accepted. This will be displayed for few seconds.

The Login Page

The login page was correctly displayed. All the required fields were shown. After the new user has registered, the login page came correctly for the new user to enter his/her registered username and password.

Menu Page

This was able to display all the needed features on the application. When each icon is clicked, it opens the linked or imbibed page. The icons responded very well when clicked.

User Icon

The page gives extra features to the application. All the in-built features, such as the sms, email and call buttons worked according to the design.

News Icon

The news page was well displayed when the news icon was clicked. The user can read the news details on the Android application in the same way as reading on the computer.

Map Icon

The map page was well displayed. The user's location can be viewed clearly and correctly. The GPS tracker worked so well and perfectly.

Courses Icon

This page displayed all the courses the user registered for in the school. This is a new invention on the school webpage and incorporated into this application. The user will still need to re-login with his/her school ID before he/she can view this page. The course page worked as expected.

Timetable Icon

This page will be displayed according to the faculty registered to by the user. There are three different faculties on the application and any one selected by the user will determine which timetable page he or she will view. All the functions created to make this page work, worked perfectly.

Contact Icon

This shows the contact information of the school. The page was displayed as designed and for easy access.

6.3 Possible Improvements Made after Testing

After the testing, the application was first built on shared preferences method where all the user information were stored on the device but the method was later changed to web server where we can save data on database. This will make the application accessible anytime, and anywhere even with different device. The application will be able to read and write data directly from the database. So there will not be any need to be registering the same username and password when the device is changed.

7 CONCLUSIONS

The application was implemented in the Android language and was targeted for mobile devices running on the Android operating system. It allows users to register, login and easily access the important features created on it. The registered data are saved on the database created on the school server. The saved data will be retrieved and used to log into the application. It is an application that incorporates some important information, features and applications on the school web page on itself like the Timetables, News, Courses, and Contact information for easy access for the user. It saves time and headache to login into the web browser all the time in order to get some information from the school webpage. Once the user is logged in, he/she can make calls, send sms, email without logging into the browser. It is a very easy and user friendly application and has the current version of the Android operating system.

The application was developed using the Java language and MySQL. The working application is converted to an apk format and then installed on the Android enabled devices. A welcome page appears and then a login page. One can then sign up with the username and password which will be saved into the database created. The user can then have access to the function in menu lists on the application. Every activity on the application is saved on the remote server. The server administrator has authority to access to all the information on the database and monitor traffics on the application for security reasons. The application met all the required details and specifications. The application works perfectly well.

7.1 Future Work

This application was designed purposely to transfer some important information on the school portal into an Android application which works on all Android devices. After the configuration and design, the application menu functions can still be improved on. The whole design can also be upgraded to suit the immediate need of the school in the nearest future.

The application can be deployed to the institution server where it can be monitored by the school administrator. The login information can also be modified in such a way that students will be able to log in with their individual student identification number so that there will be no need for creating a new username or password.

The timetable can be developed to work in such a way that it will be displaying the requested page according to the current academic period. Also the map page should be able to work in a way that the user will be able to search for places.

REFERENCES

- /1/ Android Developers Guide. Accessed 22.3.2014
<http://developer.android.com/reference/android/database/sqlite/package-summary.html>
- /2/ Ahmed, Misbah (2011). Introducing Mobile Application Development for Android. Accessed 22.3.2014
<http://www.egjug.org/files/Introducing%20Mobile%20Application%20Development%20for%20Android.pdf>
- /3/ Ableson , Frank (2008). Develop Android application with Eclipse (Get started with Google's Android Development Tools Eclipse plug-in). IBM Cooperation . Accessed 22.3.2014
<https://www6.software.ibm.com/developerworks/education/os-eclipse-android/os-eclipse-android-a4.pdf>
- /4/ Android Developers Guide. Accessed 22.3.2014
<http://developer.android.com/guide/topics/data/data-storage.html>
- /5/ Android Developers Guide. Accessed 22.3.2014
<http://developer.android.com/tools/help/avd-manager.html>
- /6/ Xamarin Developer Centre. Accessed 3.5.2014
http://docs.xamarin.com/guides/android/getting_started/hello,_multi-screen_applications/
- /7/ Conder, Shane, Lauren Darcey & Keith Vance (2010). Android Mobile (Application Development from A to Z). Internet.com. Developer eBook
- /8/ Rogers, Rick, John Lombardo, Zigurd Medinieks & Blake Meike (2009). Android Application Development. 1st Ed. O'Reily Media, Inc, USA.

APPENDICES

Appendix 1.0

VAMK Android Application

Appendix 1.1

School.java

```
package school.com;

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

public class school extends Activity {
    private int secondsDelayed = 2;

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

        try {
            new Handler().postDelayed(new Runnable() {
                @Override
                public void run() {
                    startActivity(new Intent(school.this, menu.class));
                    finish();
                }
            }, secondsDelayed * 1000);
        } catch (Exception e) {
        }
    }
}
```

Appendix 1.2**MainActivity.java**

```
package school.com;

import school.com.R.layout;
import android.os.Bundle;
import android.app.Activity;
import android.view.Menu;

public class MainActivity extends Activity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(layout.activity_main);
    }

    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        // Inflate the menu; this adds items to the action bar if it is present.
        getMenuInflater().inflate(R.menu.main, menu);
        return true;
    }
}
```

Appendix 1.3

Register.java

```
package school.com;

import android.app.Activity;
import android.app.AlertDialog;
import android.content.Intent;
import android.content.SharedPreferences;
import android.os.Bundle;
import android.util.Log;
import android.view.Menu;
import android.view.MenuInflater;
import android.view.MenuItem;
import android.view.View;
import android.widget.AdapterView;
import android.widget.AdapterView.OnItemClickListener;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Spinner;

public class register extends Activity implements View.OnClickListener {

    Button register;

    private EditText username;

    private EditText password;

    private EditText lastname;

    private EditText firstname;

    private EditText phonenumber;

    private EditText email;

    private EditText confirmPassword;

    public static Integer userId;
```

```

    public static String name;

    private Spinner department;

    private Spinner faculty;

    int pos;

    /** Called when the activity is first created. */

    @Override

    public void onCreate(Bundle savedInstanceState) {

        super.onCreate(savedInstanceState);

        setContentView(R.layout.register);

        register=(Button)findViewById(R.id.registerButton);

        username=(EditText)findViewById(R.id.username);

        password=(EditText)findViewById(R.id.password);

        lastname=(EditText)findViewById(R.id.lastname);

        firstname=(EditText)findViewById(R.id.firstname);

        email=(EditText)findViewById(R.id.email);

        phonenumber=(EditText)findViewById(R.id.phonenumber);

        confirmPassword=(EditText)findViewById(R.id.confirmpassword);

        department=(Spinner)findViewById(R.id.department);

        //department.setOnItemClickListener((OnItemSelectedListener) this);

        faculty=(Spinner)findViewById(R.id.faculty);

        //faculty.setOnItemClickListener((OnItemSelectedListener) this);

        register.setOnClickListener(this);

        faculty.setOnItemClickListener(new OnItemSelectedListener() {

            @Override

            public void onItemClick(AdapterView<?> arg0, View arg1,

```

```

        int arg2, long arg3) {
// TODO Auto-generated method stub
pos=arg2;
department.setVisibility(View.VISIBLE);
Log.v("pos", ""+pos);

if (pos==1){
    department.setSelection(pos-1);

}else if (pos==2){

    department.setSelection(pos-1);

}else if (pos==3){

    department.setSelection(pos-1);

}
}

@Override
public void onNothingSelected(AdapterView<?> arg0) {
// TODO Auto-generated method stub

}

});
}
@Override

```



```
public boolean onCreateOptionsMenu(Menu menu) {  
    MenuInflater inflater = getMenuInflater();  
    inflater.inflate(R.layout.menu_button, menu);  
    return true;  
}  
  
@Override  
public boolean onOptionsItemSelected(MenuItem item) {  
    int itemId = item.getItemId();  
    if (itemId == R.id.list) {  
        startActivity(new Intent(register.this, menu.class));  
        finish();  
        return true;  
    } else {  
        return super.onOptionsItemSelected(item);  
    }  
}  
  
public void onItemClick(AdapterView<?> parent,  
    View v, int position, long id) {  
  
    }  
    public void onNothingSelected(AdapterView<?> parent) {  
    }  
  
@Override  
    @SuppressWarnings({ "unused" })  
    public void onClick(View view) {  
        if (view == register)  
        {
```

```

String strUsername = username.getEditableText().toString();
String strPassword = password.getEditableText().toString();
String strLastname = lastname.getEditableText().toString();
String strFirstname= firstname.getEditableText().toString();
String strPhone = phonenumber.getEditableText().toString();
String strEmail = email.getEditableText().toString();

String strConfirmPassword =
confirmPassword.getEditableText().toString();

String fac = faculty.getSelectedItem().toString();
String dept = department.getSelectedItem().toString();

menu.name = username.getEditableText().toString();

if( strUsername.equals("") || strPassword.equals("") ||
strLastname.equals("") || strFirstname.equals("") || strConfirmPassword.equals("") ){

    //non field should be empty

    alertOkOnly("Registration Error","non field should be
empty");

}

else

{

    if (strPassword.equals(strConfirmPassword)){

        SharedPreferences.Editor ed =
this.getSharedPreferences("Timetable",MODE_PRIVATE).edit();

        ed.putString(Constants.KEY_FACULTY, fac);
        ed.putString(Constants.KEY_DEPARTMENT, dept);
        ed.putString(Constants.KEY_USERNAME, strUsername);
        ed.putString(Constants.KEY_PASSWORD, strPassword);
        ed.commit();
    }
}

```


Appendix 1.4

Menu.java

```
package school.com;

import android.app.Activity;
import android.content.Intent;
import android.content.SharedPreferences;
import android.content.SharedPreferences.Editor;
import android.os.Bundle;
import android.os.StrictMode;
import android.preference.PreferenceManager;
import android.view.Menu;
import android.view.MenuInflater;
import android.view.MenuItem;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;

public class menu extends Activity implements View.OnClickListener {

    //setting IP Contact_Us http://www.schooltwo.com/vaasa/vaasa/
    public static String IpAddress = "http://www.schooltwo.com/vaasa/vaasa/";
    //public static String IpAddress = "http://192.168.9.1/vaasa/";

    Button login;

    private EditText username;
    private EditText password;
    public static Integer userId;
    public static String name;
    static final String KEY_USERNAME = "username";
```

```

        static final String KEY_PASSWORD = "password";

        /** Called when the activity is first created. */

        @Override

        public void onCreate(Bundle savedInstanceState) {

            super.onCreate(savedInstanceState);

            setContentView(R.layout.login);

            StrictMode.ThreadPolicy policy = new
            StrictMode.ThreadPolicy.Builder().permitAll().build();

            StrictMode.setThreadPolicy(policy);

            login=(Button)findViewById(R.id.login);

            username=(EditText)findViewById(R.id.username);

            password=(EditText)findViewById(R.id.password);

            login.setOnClickListener(this);

            String strUsername = username.getEditableText().toString();

            String strPassword = password.getEditableText().toString();

            SharedPreferences prefs = PreferenceManager.getDefaultSharedPreferences(this);

            Editor ed = prefs.edit();

            ed.putString(KEY_USERNAME, strUsername);

            ed.putString(KEY_PASSWORD, strPassword);

            ed.commit();

        }

        @Override

        public boolean onCreateOptionsMenu(Menu menu) {

            MenuInflater inflater = getMenuInflater();

            inflater.inflate(R.layout.menu_button1, menu);

            return true;

```

```
}  
  
@Override  
public boolean onOptionsItemSelected(MenuItem item) {  
    int itemId = item.getItemId();  
    if (itemId == R.id.exit) {  
        finish();  
        return true;  
    } else if (itemId == R.id.signup) {  
        startActivity(new Intent(menu.this, register.class));  
        finish();  
        return true;  
    } else {  
        return super.onOptionsItemSelected(item);  
    }  
}
```

```
@Override  
public void onClick(View view) {  
    if (view == login)  
    {  
  
        name = username.getEditableText().toString();  
  
        startActivity(new Intent(menu.this, menuList.class));  
        finish();  
    }  
    }  
}
```

Appendix 1.5**Map.java**

```
package school.com;

import java.io.IOException;

import java.util.List;

import java.util.Locale;

import com.google.android.gms.maps.CameraUpdateFactory;

import com.google.android.gms.maps.GoogleMap;

import com.google.android.gms.maps.MapFragment;

import com.google.android.gms.maps.MapView;

import com.google.android.gms.maps.model.LatLng;

import com.google.android.maps.GeoPoint;

import android.annotation.SuppressLint;

import android.app.Activity;

import android.app.AlertDialog;

import android.content.Intent;

import android.location.Address;

import android.location.Geocoder;

import android.os.Bundle;

import android.view.Menu;

import android.view.MenuInflater;

import android.view.MenuItem;

import android.view.View;

import android.widget.EditText;

import android.widget.TextView;

public class map extends Activity {
```

```

public class Overlay {

    TextView error,pt;

    EditText cm;

    String i,returnStringLancar,stat;

    Geocoder geoCoder;

    GeoPoint p;

    MapController controller;

    MapView map;

    private GoogleMap mMap;

    private static final LatLng VassaLatLng = new LatLng(30, 65200);

    private static final Overlay MapOverlay = null;

    @SuppressWarnings("CutPasteId")

    @Override

    protected void onCreate(Bundle savedInstanceState) {

        super.onCreate(savedInstanceState);

        setContentView(R.layout.map);

        setUpMapIfNeeded();

        pt=(TextView)findViewById(R.id.map);

        cm=(EditText)findViewById(R.id.map);

    }

    private void setUpMapIfNeeded() {

        // Do a null check to confirm that we have not already instantiated the map.

        if (mMap == null) {

            mMap = ((MapFragment)
getFragmentManager().findFragmentById(R.id.map))

```



```

        .getMap();

        // Check if we were successful in obtaining the map.
        if (mMap != null) {

            // The Map is verified. It is now safe to manipulate the map.
            mMap.setMapType(GoogleMap.MAP_TYPE_HYBRID);

mMap.moveCamera(CameraUpdateFactory.newLatLngZoom(VassaLatLng,
15));

            mMap.animateCamera(CameraUpdateFactory.zoomIn());
        }
    }
}

public void clickHandler(View view){
int id = view.getId();

    if (id == R.id.launchmap) {

        geoCoder = new Geocoder(getApplicationContext(),
        Locale.getDefault());

        List<Address> addresses;

        try {

            addresses = geoCoder.getFromLocationName(cm.getText().toString(),5);
if(addresses.size() > 0)
{
            p = new GeoPoint( (int) (addresses.get(0).getLatitude() * 1E6),
                (int) (addresses.get(0).getLongitude() * 1E6));

            controller.animateTo(p);

```

```

        controller.setZoom(12);

        List<Overlay> listOfOverlays = Mapview.getOverlay();
        listOfOverlays.clear();
        listOfOverlays.add(MapOverlay);

        map.invalidate();

        cm.setText("");
    }
    else
    {
        AlertDialog.Builder adb = new AlertDialog.Builder(map.this);
        adb.setTitle("Google Map");
        adb.setMessage("Please Provide the Proper Place");
        adb.setPositiveButton("Close",null);
        adb.show();
    }
} catch (IOException e) {
    // TODO Auto-generated catch block
    e.printStackTrace();
}
}

protected boolean isRouteDisplayed() {
    // TODO Auto-generated method stub
    return false;
}

```

@Override

```
public boolean onCreateOptionsMenu(Menu menu) {  
    MenuInflater inflater = getMenuInflater();  
    inflater.inflate(R.layout.menu_button, menu);  
    return true;  
}
```

@Override

```
public boolean onOptionsItemSelected(MenuItem item) {  
    int itemId = item.getItemId();  
    if (itemId == R.id.exit) {  
        finish();  
        return true;  
    } else if (itemId == R.id.list) {  
        startActivity(new Intent(map.this, menuList.class));  
        finish();  
        return true;  
    } else {  
        return super.onOptionsItemSelected(item);  
    }  
}
```

```
public void alertOkOnly(String title, String message)  
{  
    AlertDialog.Builder builder=  
        new AlertDialog.Builder(map.this);  
    builder  
        .setTitle(title)
```

```
.setMessage(message)
.setPositiveButton("OK", null)
.show();
} }
```

Appendix 1.6 Map Layout

```
<?xml version="1.0" encoding="utf-8"?>  
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"  
    android:id="@+id/mapview">  
  
    <fragment  
        android:id="@+id/map"  
        android:name="com.google.android.gms.maps.MapFragment"  
        android:layout_width="match_parent"  
        android:layout_height="match_parent"  
        android:clickable="true"  
        android:orientation="vertical"  
    />  
</LinearLayout>
```

Appendix 1.7

AndroidManifest.xml

```
<?xml version="1.0" encoding="utf-8"?>

<manifest xmlns:android="http://schemas.android.com/apk/res/android"

    package="school.com"

    android:versionCode="1"

    android:versionName="1.0" >

    <uses-sdk

        android:minSdkVersion="12"

        android:targetSdkVersion="17" />

    <uses-feature

        android:glEsVersion="0x00020000"

        android:required="true"/>

    <uses-library android:name="com.google.android.maps" />

    <permission

        android:name="school.com.permission.MAPS_RECEIVE"

        android:protectionLevel="signature"/>

    <uses-permission android:name="school.com.permission.MAPS_RECEIVE"/>

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

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

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

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

    <uses-permission

        android:name="com.google.android.providers.gsf.permission.READ_GSERVICES"/>

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

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

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

    <application
```

```
android:allowBackup="true"
android:icon="@drawable/logo"
android:label="@string/app_name"
android:theme="@style/AppTheme" >
<activity android:name="school.com.school"
    android:label="@string/app_name"
    >
    <intent-filter>
        <action android:name="android.intent.action.MAIN" />
        <category android:name="android.intent.category.LAUNCHER" />
    </intent-filter>
</activity>
<activity android:name="school.com.menu"
    android:label="@string/app_name">
    <intent-filter>
        <action android:name="android.intent.action.MENU" />
        <category android:name="android.intent.category.DEFAULT" />
    </intent-filter>
</activity>
<activity android:name="school.com.menuList"
    android:label="@string/app_name">
    <intent-filter>
        <action android:name="android.intent.action.MENULIST" />
        <category android:name="android.intent.category.DEFAULT" />
    </intent-filter>
</activity>
<activity android:name="school.com.user"
    android:label="@string/app_name">
```

```
<intent-filter>
    <action android:name="android.intent.action.CONTACT" />
    <category android:name="android.intent.category.DEFAULT" />
</intent-filter>
</activity>
<activity android:name="school.com.news"
    android:label="@string/app_name">
    <intent-filter>
        <action android:name="android.intent.action.NEWS" />
        <category android:name="android.intent.category.DEFAULT" />
    </intent-filter>
</activity>
<activity android:name="school.com.map"
    android:label="@string/app_name">
    <intent-filter>
        <action android:name="android.intent.action.MAP" />
        <category android:name="android.intent.category.DEFAULT" />
    </intent-filter>
</activity>
<activity android:name="school.com.courses"
    android:label="@string/app_name">
    <intent-filter>
        <action android:name="android.intent.action.COURSES" />
        <category android:name="android.intent.category.DEFAULT" />
    </intent-filter>
</activity>
<activity android:name="school.com.contact"
    android:label="@string/app_name">
    <intent-filter>
```



```
<action android:name="android.intent.action.CONTACT" />
<category android:name="android.intent.category.DEFAULT" />
</intent-filter>
</activity>
<activity android:name="school.com.timetable"
    android:label="@string/app_name">
    <intent-filter>
        <action android:name="android.intent.action.TIMETABLE" />
        <category android:name="android.intent.category.DEFAULT" />
    </intent-filter>
</activity>
<activity android:name="school.com.register"
    android:label="@string/app_name">
    <intent-filter>
        <action android:name="android.intent.action.REGISTER" />
        <category android:name="android.intent.category.DEFAULT" />
    </intent-filter>
</activity>
<activity android:name="school.com.file"
    android:label="@string/app_name">
    <intent-filter>
        <action android:name="android.intent.action.FILE" />
        <category android:name="android.intent.category.DEFAULT" />
    </intent-filter>
</activity>
<meta-data
    android:name="com.google.android.gms.version"
```

```
android:value="@integer/google_play_services_version" />
    <meta-data
android:name="com.google.android.maps.v2.API_KEY"
android:value="AIzaSyDsKBeKvPF7-VW3B_bVNNVJEzgQN3kIT2g"/>
</application>
</manifest>
```