VAASAN AMMATTIKORKEAKOULU
VASA YRKESHÖGSKOLA
UNIVERSITY OF APPLIED SCIENCES

Oriyomi Oladele

STUDY TIME - AN ANDROID BASED MOBILE LEARNING APPLICATION

Technology and Communication

2014

ACKNOWLEDGEMENTS

To my parents who have always been a source of inspiration to me ever since childhood. To my wife who has been a sister, a friend and a good listener during every difficult time. To my close friends who have been true friends all through the years and to those who have been encouraging all along. I am grateful to everyone of them.

To my supervisor, Dr. Ghodrat Moghadampour, a teacher of immense knowledge. I am grateful for your guidance. To all inspiring lecturers and teachers that I have had the privilege of studying with: Jarmo Makelä, Seppo Mäkinen, Santiago Chavez, just to mention a few. It will always be a pleasure studying with you.

VAASAN AMMATTIKORKEAKOULU
UNIVERSITY OF APPLIED SCIENCES
Degree Programme in Information Technology

ABSTRACT

Author	Oriyomi Oladele
Title	Study Time - An Android Based Mobile Learning Application
Year	2014
Language	English
Pages	64
Name of Supervisor	Ghodrat Moghadampour

The main objective of this thesis work was to develop a mobile learning application that would make it possible for the users to study or learn on their own by attempting examination, such as questions from different subject areas. This will allow an individual user to refresh his or her knowledge in these subject areas. Timing is included so that a user can monitor how quick they are while answering those questions. The result of the examination is displayed to the user after the completion of the exam and can be viewed under history for future reference. The application can also be used in schools and other learning institutions. Teachers can give instructional materials and set homework or test questions for students to attempt and submit with the use of their mobile phones.

The application consists of client side and server side. Client side is made up of graphical user interfaces (GUI) which the user interacts with. The server side of the application consists of PHP script to query database for different requests. The data are stored on MySQL database. The application is currently developed for Android mobile devices running minimum of Android 2.2, API level 8.

Keywords Android, Mobile Learning, MySQL, PHP, Client, Server

CONTENTS

ABBREVIATIONS

API	Application Programming Interface
JSON	JavaScript Object Notation
XML	Extensible Markup Language
URL	Uniform Resource Locator
GUI	Graphical User Interface
SQL	Structured Query Language
UNESCO	The United Nations Educational, Scientific and Cultural Organization
WAEC	West African Examination Council
CSV	Comma-separated values
SDK	Software Development Kit
HTML	Hypertext Markup Language
IDE	Integrated Development Environment
HTTP	Hypertext Transfer Protocol
PDF	Portable Document Format
PHP	PHP: Hypertext Preprocessor
UI	User Interface
AVD	Android Virtual Device

1 INTRODUCTION

The advent of mobile learning in education has greatly influenced the manner in which teaching is disseminated in schools. Up till now, most of the applications developed for teaching in schools are restricted to computers and better part of these are contained on desktop computers.

As mobile devices become more and more important to our daily lives so has the need to develop applications that can reinforce teaching and learning on mobile platforms. Since education and process of learning are not confined to only class-rooms, it becomes so important to take advantage of the boom in the ownership of mobile devices to develop applications that can be incorporated into education administration in such a way that learning can be improved.

With this kind of application, students can learn at their own convenient time. They can as well use the application when preparing for examinations. The timing functionality included in the application will help them measure how fast they are while answering questions from each subject area of choice. The application will be particularly useful to the students preparing for the West African Examinations as questions and solutions from past WAEC examinations are used in the applica-tion.

1.1 Background

For few years now, there has been great emphasis on how mobile technologies can enable the achievement of Education For All./1/ This concept has started to be implemented by UNESCO in some of the developing countries of the world. In central Nigeria, a mobile application to help English language teachers improve the language literacy skills among the primary school students has been devel-oped./9/ In Senegal, UNESCO has also launched a mobile learning application to support student learning in mathematics. There have been similar projects by

UNESCO in Pakistan and Mexico. Mobile learning is therefore turning out to be one of the solutions to the challenges faced by education worldwide./1/

Apart from UNESCO, there have been lots of other organizations and individuals making efforts on how to improve learning and teaching through the use of mobile technologies./10/ In this regard, the topic of this project came at the right time and it will be of great importance not only to students in Nigeria but all over the world.

1.2 Motivations

This project topic came to my mind during my visit to Nigeria in early 2013. I observed that most students nowadays have mobile phones or mobile devices. This observation came as no surprise as Nigeria is rated the largest mobile market in Africa with more than 125 million subscribers. The market penetration for mobile devices was estimated to be around 75% as of early 2014./2/ However, most of these students basically use their mobile devices for playing games or chatting on social networks during their free time. As a way of encouraging teenagers spend better part of their free time studying, I decided to implement a mobile application which they can download on their mobile phones and subsequently use to study and to practice for examinations.

1.3 Objectives

The main idea of the topic is to implement an application with the help of which users can learn at their own pace on their mobile devices, anywhere and at anytime. This application will serve the purpose of making it possible for its users to prepare for examinations by attempting exam-like questions from different subject areas of choice.

The application was developed particularly for secondary school students. However, the use of the application is not limited to students alone as other users can

as well make use of the application for instance to refresh their memories about some subject areas.

1.4 Description

The project is a mobile learning application. It is developed on android platform for android mobile devices. The questions are set based on subject areas. The subject areas are grouped into categories. A user who wants to take an examination from a particular subject area will have to choose a category after which he/she can choose a subject. For each examination, there is a set number of questions. These questions are retrieved from an external database and are selected at random so that if two users start examination from the same subject at the same time, it is very unlikely that they will have the same order of the questions.

The time allowed for each examination is set on the database. A hint on answering each of the questions is provided. This hint is shown to the user when he/she clicks on the hint button. When a user starts the examination, the duration for the examination starts decreasing. This time is shown on the top right corner of the question view page. It is possible to go to the next question without answering the current question. The user can navigate back to the unanswered questions with the help of a previous button. If the time duration is up before the user finishes with the examination, the application submits itself and displays the result to the user.

2 RELEVANT TOOLS AND TECHNOLOGIES

There are several technologies and tools used in the development of this project work. These tools and technologies include: Android, Java, PHP, MySQL, JSON, phpMyAdmin and Android mobile phone.

2.1 Android

Android is an open-source mobile operating system that is based on Linux kernel. It is designed primarily for touchscreen mobile devices which include smart phones and tablet computers. Apart from its use in mobile devices, the technology has also been implemented in digital cameras, smart televisions and other electronics appliances. The Android operating system is made up of four main layers which are divided roughly into five sections. These sections are Linux kernel, Libraries, Android runtime, Application framework and Applications./8/

2.2 JSON

JSON is an acronym for JavaScript Object Notation. It is human-readable and super-lightweight data interchange technology. Though JSON is based on JavaScript, it also uses conventions which the programmers of other languages, such as C, C++, C#; Perl, Python, and many others are familiar with. JSON is in a text format which makes it completely easy for humans to read and write. Parsing and generating JSON is equally easy for the machines./3/

JSON consists of two data structures which are object and array. A JSON object is an unordered set of name/value pairs which normally begins with left brace ({) and ends with right brace (}). Each name is separated by a colon (:) while the name/value pairs are separated by comma (,). On the other hand, a JSON array is an ordered collection of values. It starts with left bracket ([) and ends with right bracket (]). The values are separated by comma (,)./3/

2.3 PHP: Hypertext Preprocessor

PHP is a server-side scripting language that was invented by Rasmus Lerdorf. The source code was released to the public under the GNU public license in June 1995. The software is completely free and it is designed primarily for the production of dynamic web pages. Presently, PHP is being developed by the PHP group. The acronym PHP which originally stood for Personal Home Page now stands for Hypertext Preprocessor./4/

For this thesis, PHP has been used on the server side of the application. It opens connection to MySQL database and takes care of execution of various queries to retrieve data from and save data to the database.

2.4 MySQL

MySQL is a cross-platform open-source relational database management system (RDBMS) and was initially released in May 1995 under the GNU General Public License. It is one of the world's most widely used open-source relational database management system. It was created by Michael Widenius and was partly named after his daughter, My. The phrase SQL is an acronym for Structured Query Language. MySQL was initially developed and owned by a Swedish company, MySQL Ab. The company is now owned by Oracle Corporation./7/

2.5 phpMyAdmin

phpMyAdmin is a free and open source MySQL administration tool written in PHP and was first released in 1998 under the GNU General Public License. It has cross-platform support for the major operating systems and supports administration of multiple servers./5/

It supports most of MySQL features and has an intuitive web interface. It also has supports for creating PDF graphics of database layout, importing data from CSV

and SQL formats as well as exporting data to various formats, such as SQL, XML, PDF, CSV, among others./5/

3 APPLICATION DESCRIPTION

3.1 Project Description

The project is a mobile learning application. It was developed to help teenagers utilize their free time studying with the aid of their mobile phones or mobile devices. Teachers can also make use of the application by setting sample examination questions for the students to study in preparation for an examination. Assignments can also be given out to students through the application for them to submit at a later time. At the moment, the application works so that a student or a user can register by creating an account, log in with his or her account details and attempt to take an examination from a chosen subject area.

The subjects are grouped into categories. A user will have to choose a category to be able to view the list of subject areas under that category. After a subject has been chosen, the user can proceed to the examination page which displays the set of questions for the subject area one after the other with the help of next and previous button. Hints on how to solve some problems are provided in order to help the student learn. When the user has got to the end of the questions, the application asks the user if he or she wants to proceed with the submission of the examination. If the user decides to submit, the next page that comes up has the details of the examination result.

All the questions from the subject areas are saved on an external database. When the user has completed an examination and submitted it, the result of the examination is saved automatically to a database table so that the user can have access to his past examination attempts record for reference purpose. The records are saved against the user email address.

Since there are lots of data exchange between the client side of the application and the server side, the application definitely requires an internet connection. For instance, an internet connection is needed to get the list of available categories, the

list of subject areas, the set of questions from a particular subject area and to save and retrieve user examination record from the database server.

3.2 Requirements Analysis

3.2.1 Quality Function Deployment - QFD

Table 1 below shows the requirements of the application and the priority given to each of them.

Table 1. Application requirements.

Task Number	Task Name	Priority
1	Registration	Good to have
2	Login	Good to have
3	Display categories	Should have
4	Display subjects	Should have
5	Display questions	Must have
6	Display duration	Must have
7	Display result	Must have
8	Save records	Nice to have
9	Display saved records	Nice to have
10	Display hint	Nice to have
11	Feedback	Nice to have

12	Complaints	Nice to have
13	Display instruction	Must have

3.2.2 Functional Definition

In order for the application to fulfill its requirements, it has to be able to perform some functions. These functionalities include:

- Communicating with an external database
- Allowing user registration
- Allowing user login
- Giving the user opportunity to take examinations
- Displaying the result of the examination
- Saving the result to a database
- Allowing the user to view the saved records

3.2.3 Use-case Diagram

The use-case diagram below shows the basic requirements the application should perform in order to fulfill its purpose. As shown in the diagram, the application should allow the user to register by simply creating an account. An already registered user should be able to login into the application using his or her registration details. A user who has successfully logged in should then be able to view the welcome page from where he or she can choose to take an exam, view records, read instructions, send feedback or go through practice questions.

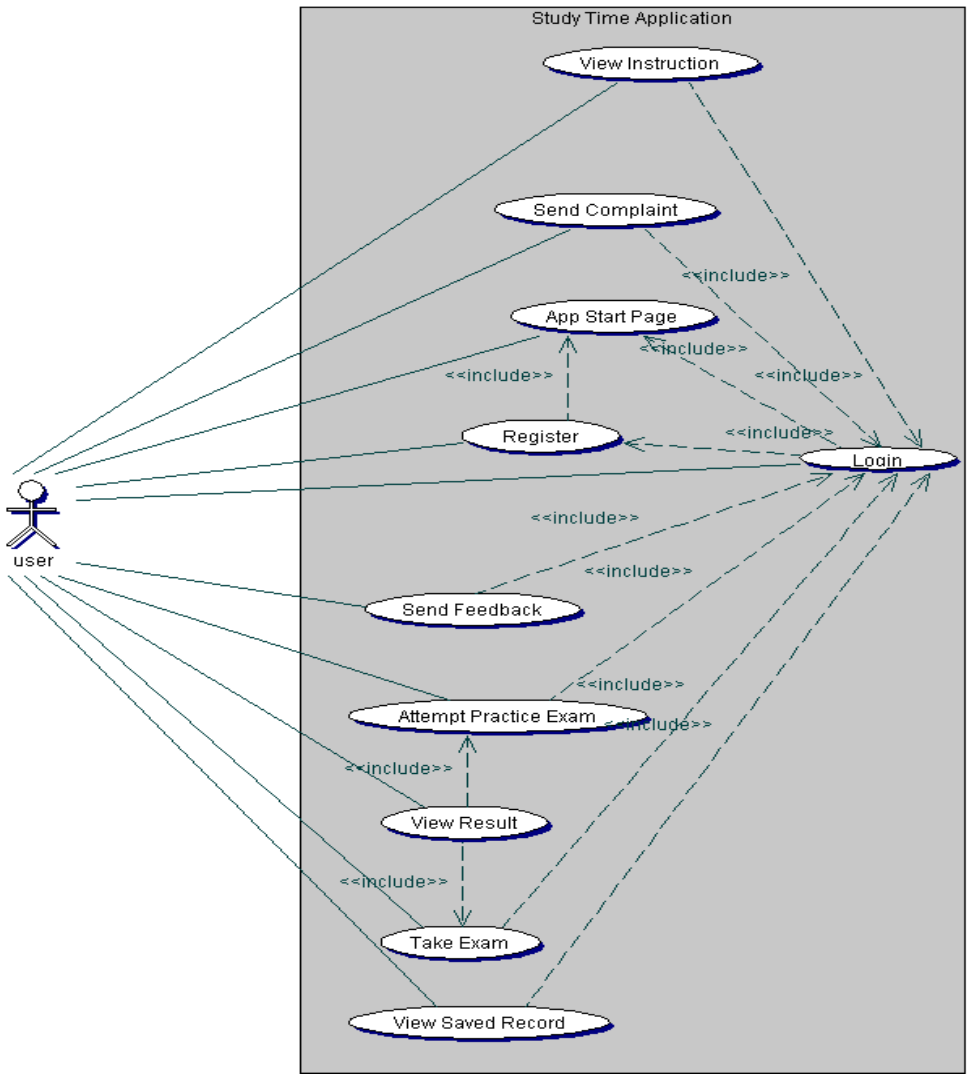

Figure 1. Use Case diagram.

3.2.4 Class Diagram

The class diagram for the application is as shown in Figure 2 below. It shows all the Java classes created for the application and the relationships between these classes.

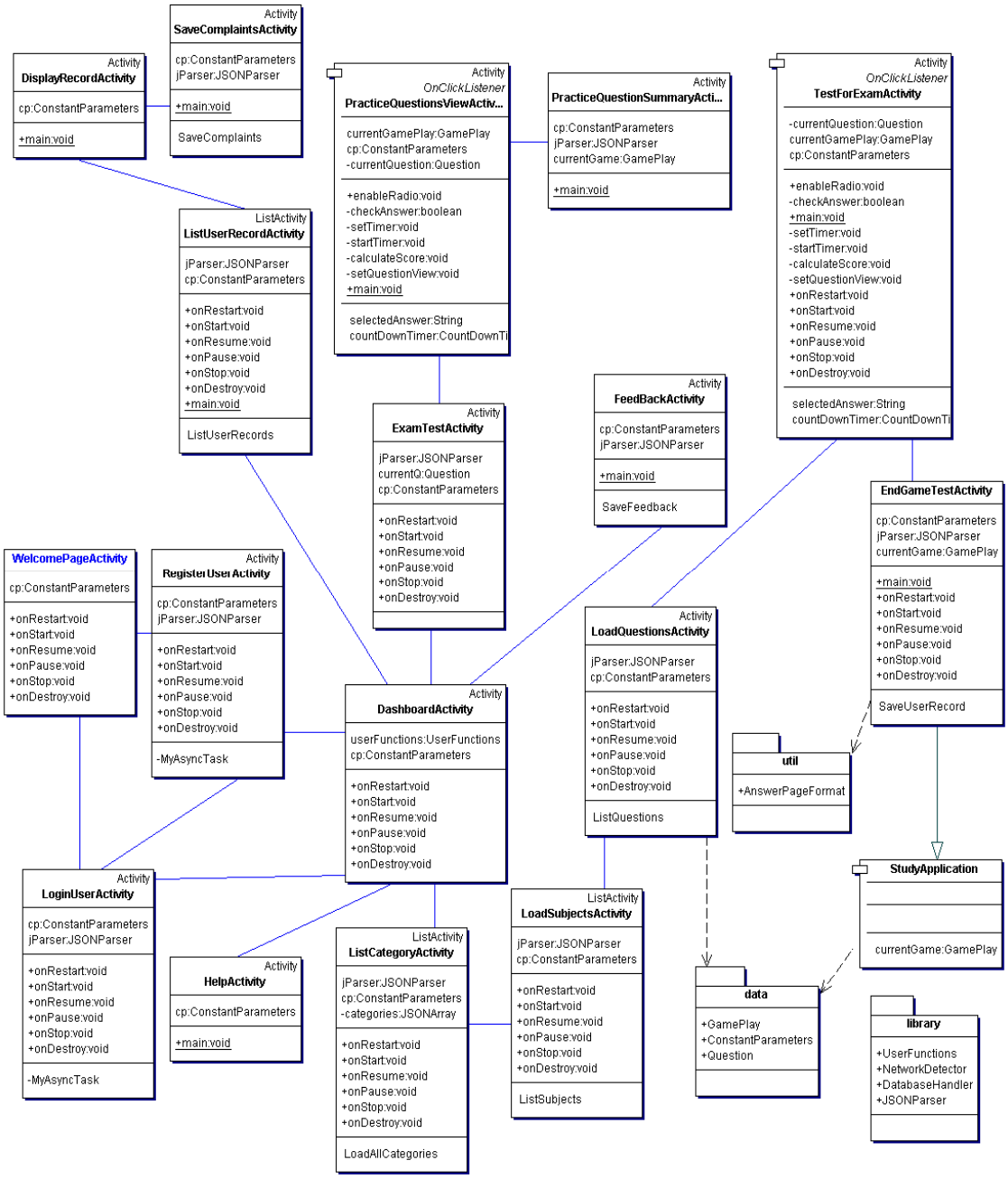

Figure 2. Class diagram.

3.2.5 Registration Sequence Diagram

The user needs to register in order to be able to use the application. To register, a user needs to supply some information which include firstname, surname, email

address and password. A field is added for password confirmation. When the user clicks on the submit button, the application checks for an empty or empty fields. If any of the required fields is empty, the application informs the user to enter the information for the fields that are empty. The application also checks for password match. However, if all the required information are supplied and the password entered matches with confirm password, the application then sends an HTTP request to the server side of the application. The PHP script that handles the registration checks the format of the email address entered to see if it is a valid email address and also checks from MySQL database if the email address entered by the user exists already or not. If the email exists, an error message is sent back to the user informing him that the email address supplied exists already in the database. If the email address supplied is not already in the database, the user will be directed to the welcome page indicating that the registration is successful.

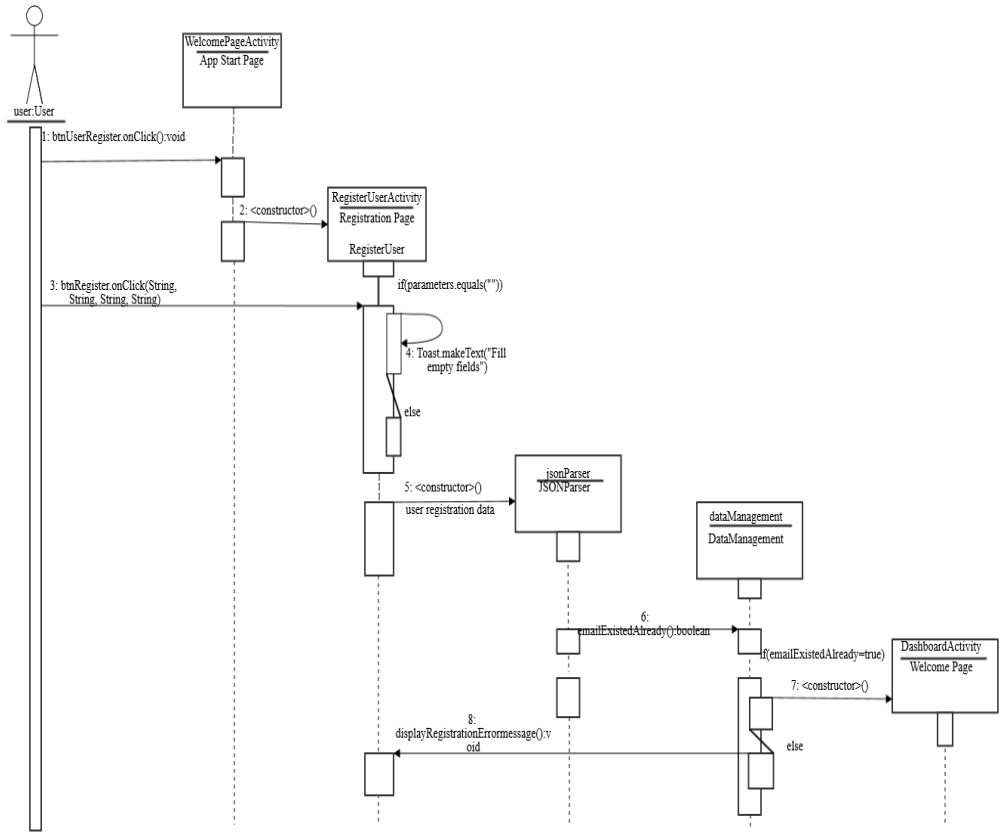

Figure 3. Registration Sequence diagram.

3.2.6 Login Sequence Diagram

The user who has already created an account can log in by supplying the email address and the password used in registration. If all fields are filled, the application sends and HTTP request to the server and the PHP script handles the process of checking the email address against the password supply from the database. If this user information exists in the database, the login process will be successful.

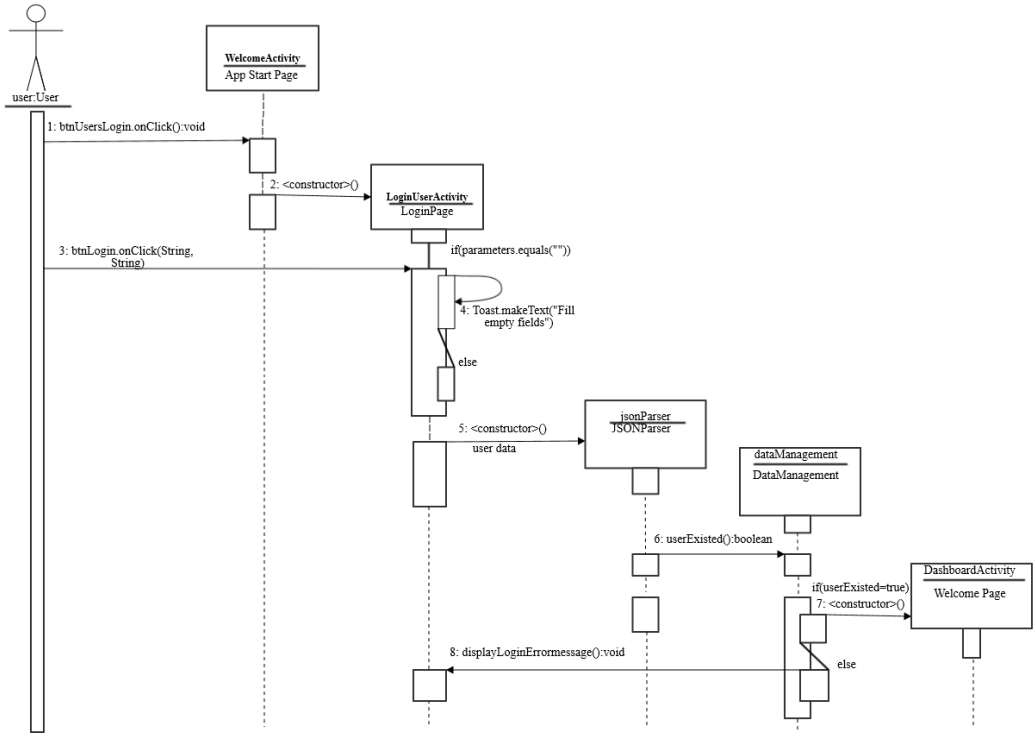

Figure 4. Login Sequence diagram.

3.2.7 Subject Selection Sequence Diagram

The user who has successfully logged in will be able to choose subject areas. Since the subjects are grouped into categories, the user must first choose a category and then click on the chosen category to view the list of subjects under it. If a chosen category, however, contains no subject yet, there will be an error message informing the user of this situation.

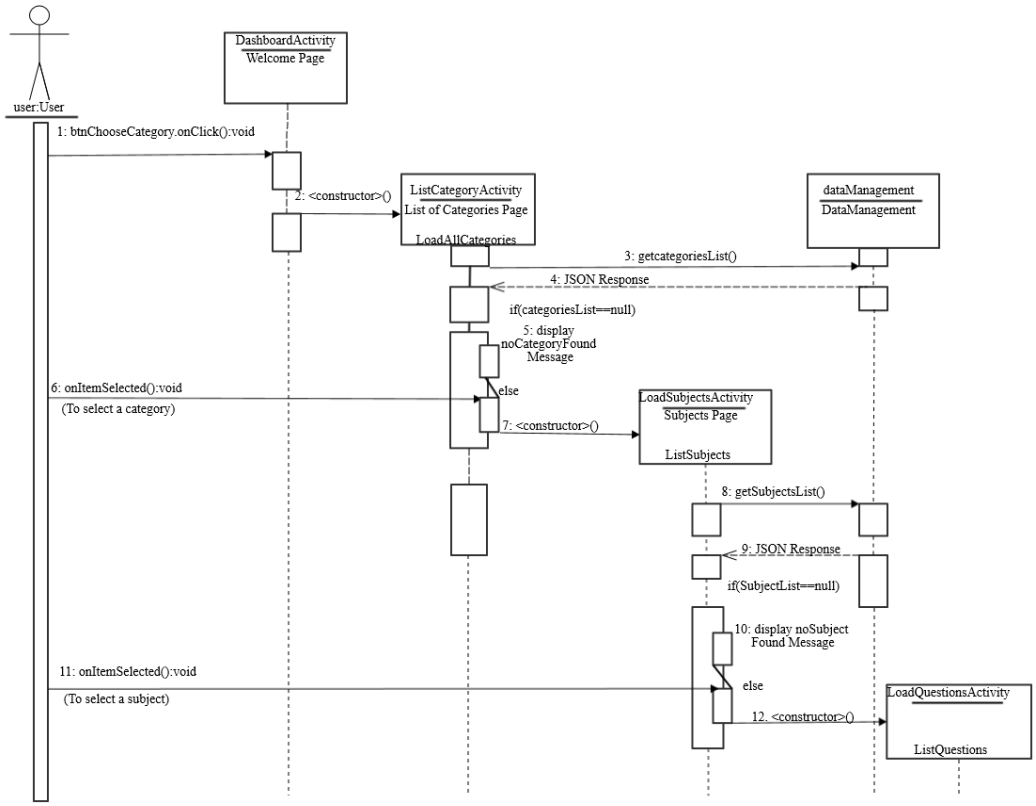

Figure 5. Select Subject Sequence diagram.

3.2.8 Take Examination Sequence Diagram

When the user clicks on a particular subject area, the questions from that subject area are retrieved from the database and saved in the application. The questions and the set of options for each will be displayed to the user one after the other after he has clicked the button to start the examination until the last question is reached. At this point the user can decide to submit the examination or go back to the previous questions with the help of the previous button. If the user decides to submit or the time is up, the result of the examination is calculated and the result page is shown to the user.

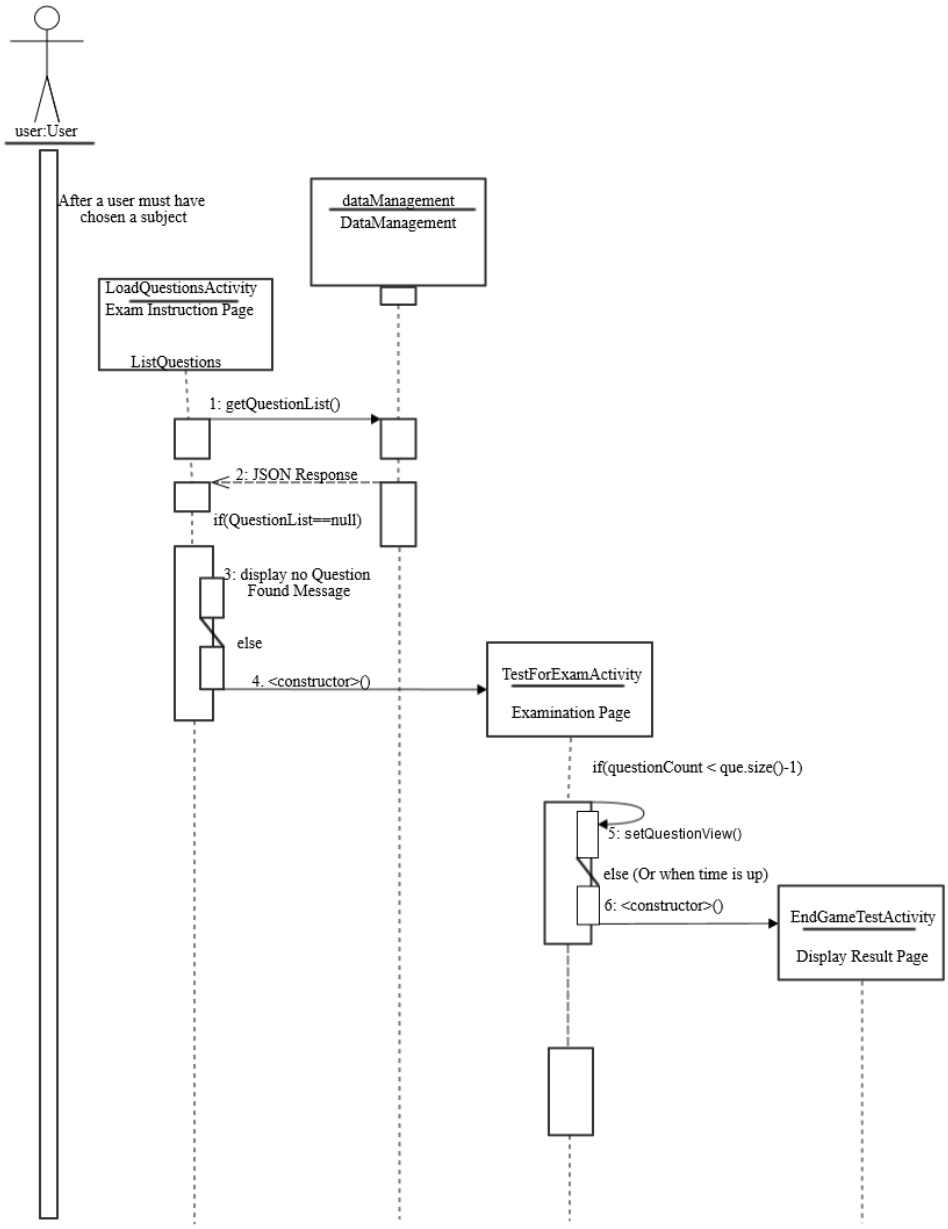

Figure 6. Take Examination Sequence diagram.

3.2.9 Examination Record Sequence Diagram

The user examination result for each examination attempt is saved automatically to MySQL database. It is therefore always possible for the user to view these saved records for future reference.

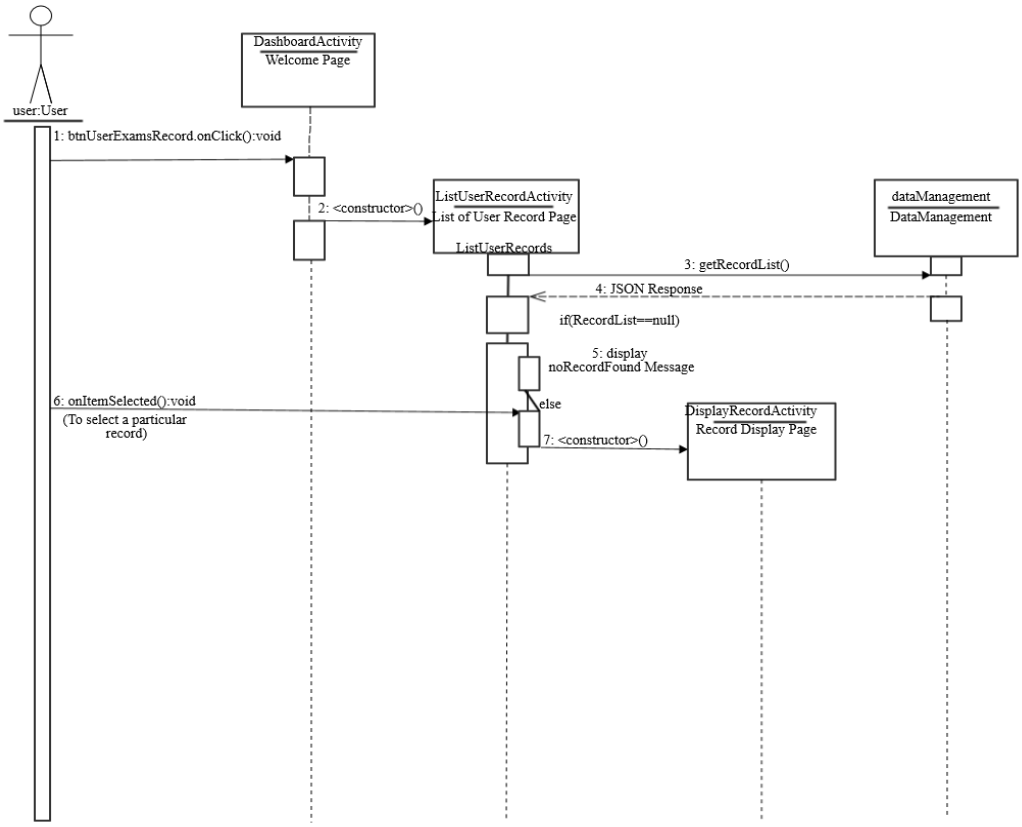

Figure 7. User Exam Record Sequence diagram.

3.2.10 Deployment Diagram

The deployment diagram for the application is as shown in Figure 8 below. It shows both the client and the server side of the application. Usually the client side communicates with the server side by sending an HTTP request. On the server side of the application, there are PHP scripts to process every request. These PHP scripts communicate with the database. After processing every request, a JSON response is sent back to the client side from the server side.

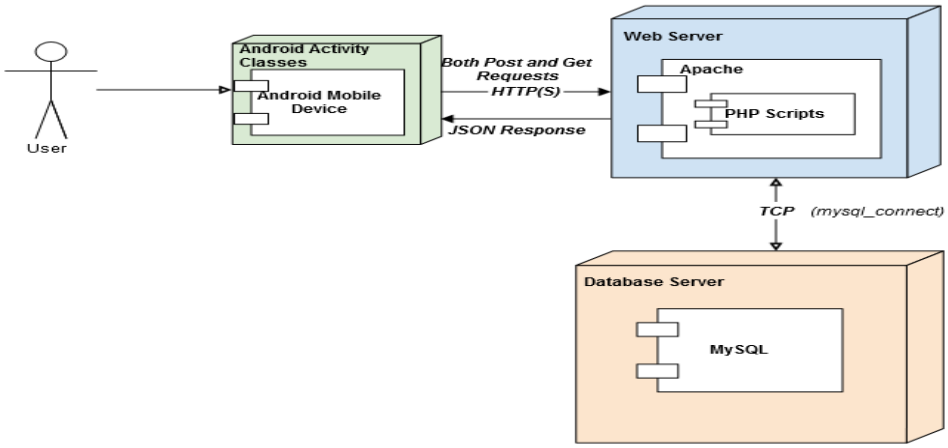

Figure 8. Deployment diagram.

4 DATABASE AND GUI DESIGN

MySQL database was chosen for this project. The database was designed in accordance with the requirements of the application as stated in Chapter 3 above. This chapter explains the design of the Database and the General User Interface for the application.

4.1 Database Design

For this application, a database named thesis_db was created. The database consists of 8 tables. These are users, categories, subjects, question, answer, user_exams_record, question_complaints and feedback tables. The relationships between the tables are as shown in the ER diagram below.

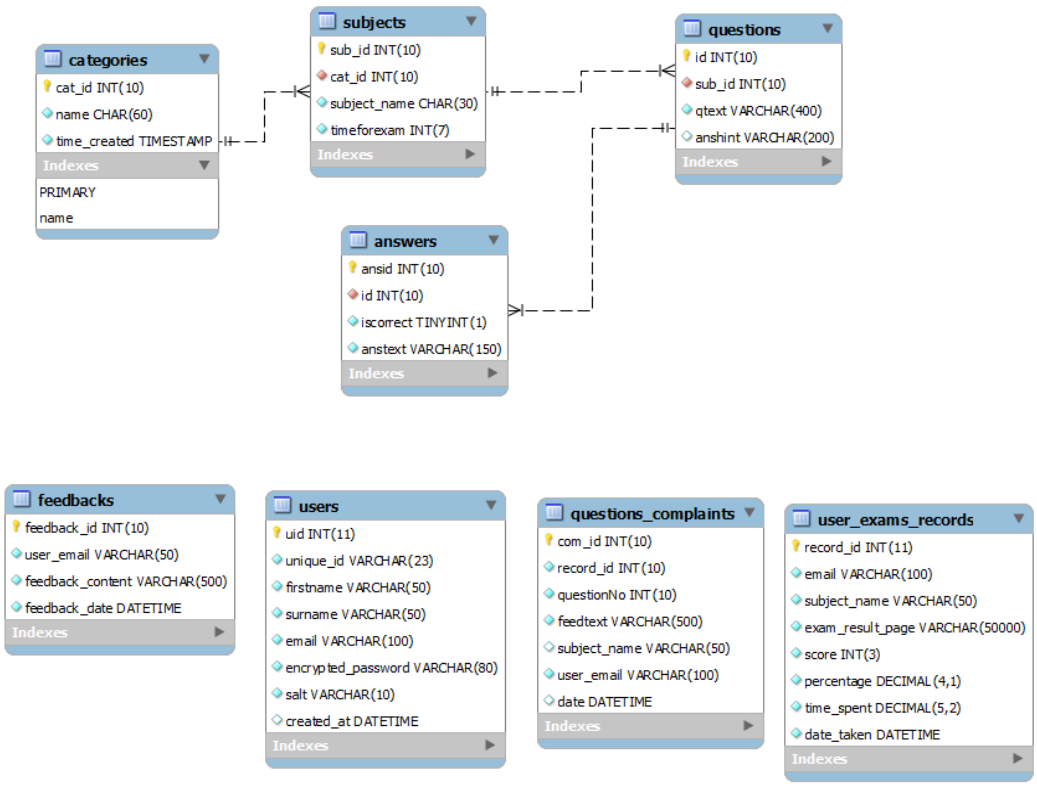

Figure 9. Entity Relationship diagram.

4.1.1 The Users Table

The structure of the table is as shown below.

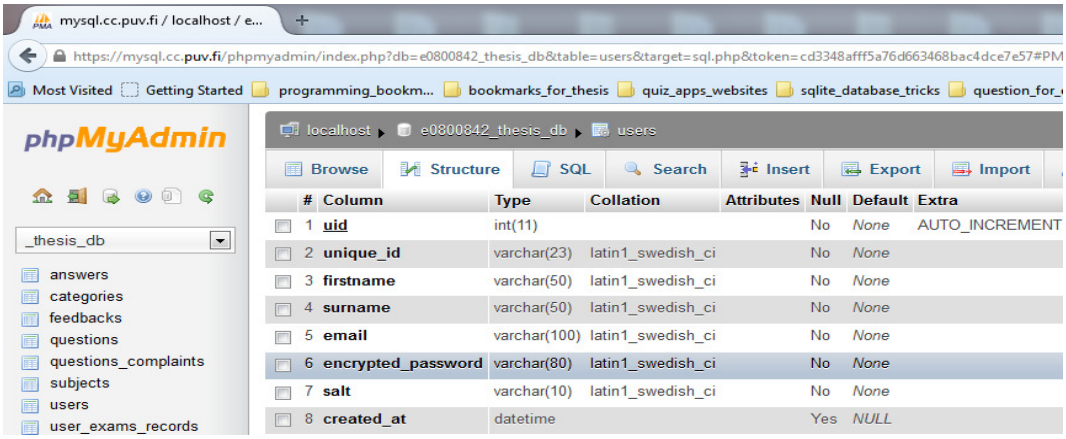

Figure 10. Structure of users table.

The users table is used to manage the user information. The table is made up of 8 columns to store uid, unique_id, firstname, surname, email, encrypted_password, salt and created_at. The primary key for the table is the user id which is represented in the table by uid. It is an integer and its value is set to auto increment.

The unique user id is generated by using the PHP function uniqid('', true). The sample uniqid generated is 53682b0ba49121.72753406.

The firstname and the surname columns of the table stores the user first and last names respectively. The email column stores the email address of the user and this should be unique for every user.

The encrypted_password column is needed for security purpose to encrypt user password in the database so that it is not saved in plain text. It uses the base64_encoding method. The salt column is used to store the value with which the password was encrypted.

The column created_at is included to be able to track when the user account was created. The column is saved in datetime format and uses NOW() function while saving user information into the database table.

4.1.2 The Categories Table

The table is used to save category information. It is created with the following sql statement:

```
CREATE TABLE IF NOT EXISTS `categories` (
 `cat_id` int(10) unsigned NOT NULL AUTO_INCREMENT,
 `name` varchar(60) NOT NULL,
 `time_created` timestamp NOT NULL DEFAULT CURRENT_TIMESTAMP,
 PRIMARY KEY (`cat_id`),
 UNIQUE KEY `name` (`name`)
) ENGINE=InnoDB
```

The primary key is the cat_id. It represents the category id. It has an integer value and this is set to auto increment. It has 3 columns altogether. The other two columns are category name represented by name and the time created which is represented in the table by time_created. The former has a varchar value while the latter has a timestamp value.

4.1.3 The Subjects Table

The table was created using the sql statement written below:

```
CREATE TABLE IF NOT EXISTS `subjects` (
 `sub_id` int(10) unsigned NOT NULL AUTO_INCREMENT,
```

```
 `cat_id` int(10) unsigned NOT NULL,
 `subject_name` varchar(30) NOT NULL,
 `timeforexam` int(7) NOT NULL,
 PRIMARY KEY (`sub_id`),
 UNIQUE KEY `name` (`subject_name`),
 KEY `cat_id` (`cat_id`)
) ENGINE=InnoDB
```

The table has 4 columns which are sub_id (Subject id), cat_id (Category id), subject_name and the timeforexam (The allowed duration of the exam under this subject). The sub_id is an integer and it is set to auto increment. cat_id is a foreign key referencing categories cat_id. There is one-many relationship between the categories and the subjects tables. That is, a category can have one or more subjects under it. However, a subject can only belong to one category.

The column timeforexam is an integer and this integer value represents the time for the exam in minutes. For example, a duration of 1 hour 30 minutes for an exam will be entered into this column as 90 which represents 90 minutes.

4.1.4 The Questions Table

The sql statement for creating this table is written below:

```
CREATE TABLE IF NOT EXISTS `questions` (
 `id` int(10) unsigned NOT NULL AUTO_INCREMENT,
 `sub_id` int(10) unsigned NOT NULL,
 `qtext` varchar(400) NOT NULL,
```

```
 `anshint` varchar(200) DEFAULT NULL,
 PRIMARY KEY (`id`),
 KEY `fk_questions_subjects1_idx` (`sub_id`)
) ENGINE=InnoDB
```

This table is used to store questions from each subject area. The table has 4 columns: id (Question id), sub_id (Subject id referencing subjects sub_id), qtext (Question text) and anshint (Correct answer hint). There is one-many relationship between subjects and questions table. One subject consists of many questions. The anshint column is used to save an hint for answering the question. The default value for this column is null.

4.1.5 The Answers Table

This table is used to save the set of options for each of the questions. Since each of the questions has a set of 5 options from which only one is the correct option, the table is designed to fulfil this condition. The sql statement for creating the table is shown below:

```
CREATE TABLE IF NOT EXISTS `answers` (
 `ansid` int(10) unsigned NOT NULL AUTO_INCREMENT,
 `id` int(10) unsigned NOT NULL,
 `iscorrect` tinyint(1) NOT NULL,
 `anstext` varchar(150) NOT NULL,
 PRIMARY KEY (`ansid`),
 KEY `fk_answers_questions1_idx` (`id`)
```

) ENGINE=InnoDB

The column ansid is the primary key and it represents the id of each of the options. The column named id is a foreign key which references questions id. The column named iscorrect is a binary with a value 1 for the correct option to a question and value 0 for the remaining 4 options which are all incorrect options to the same question. The anstext column stores the text value for each of the options.

4.1.6 The Table user_exams_records

This table stores the user exams record. It comprises of 8 columns which are record_id, email, subject_name, exam_result_page, score, percentage, time_spent and date_taken.

The email column stores the email of the user, exam_result_page column stores the result page, percentage column stores the user percentage score, time_spent column stores the time the exam takes the user to submit for grading while the date_taken stores the date and the time the user attempted the exam. The sql statement for creating the table is as shown below:

```
CREATE TABLE IF NOT EXISTS `user_exams_records` (

 `record_id` int(11) NOT NULL AUTO_INCREMENT,

 `email` varchar(100) NOT NULL DEFAULT '',

 `subject_name` varchar(50) NOT NULL DEFAULT '',

 `exam_result_page` varchar(50000) NOT NULL,

 `score` int(3) NOT NULL,

 `percentage` decimal(4,1) NOT NULL,

 `time_spent` decimal(5,2) NOT NULL,
```

```
 `date_taken` datetime NOT NULL,

 PRIMARY KEY (`record_id`)

) ENGINE=InnoDB
```

4.1.7 The Feedback Table

In order to be able to receive users' feedback about the application, I have included this table. The feedback will be needed in case users want addition of some other features the application is not currently having. The table was created with the statement shown below.

```
CREATE TABLE IF NOT EXISTS `feedbacks` (

 `feedback_id` int(10) NOT NULL AUTO_INCREMENT,

 `user_email` varchar(50) NOT NULL,

 `feedback_content` varchar(500) NOT NULL,

 `feedback_date` datetime NOT NULL,

 PRIMARY KEY (`feedback_id`)

) ENGINE=InnoDB
```

4.1.8 The Table questions_complaints

This table stores the complaints from users about questions or options. In case any of the questions contains incorrect information or some errors due to typing, users can complain about this question so that the error can be rectified. The table consists of 7 columns as shown in the sql statement below.

```
CREATE TABLE IF NOT EXISTS `questions_complaints` (

 `com_id` int(10) unsigned NOT NULL AUTO_INCREMENT,
```

```
  `record_id` int(10) unsigned NOT NULL,
  `questionNo` int(10) NOT NULL,
  `feedtext` varchar(500) NOT NULL,
  `subject_name` varchar(50) DEFAULT NULL,
  `user_email` varchar(100) NOT NULL,
  `date` datetime DEFAULT NULL,
  PRIMARY KEY (`com_id`),
  KEY `record_id` (`record_id`)
) ENGINE=InnoDB
```

4.2 GUI Design

The Graphical User Interface (GUI) was designed to fulfil all the requirements of the project as stated in Chapter 3 above. The client side interface is made up of 17 user interfaces which are interconnected to one another. The interfaces are as shown below.

4.2.1 Start Page

This is the first page that comes up when the application is started. It consists of two buttons asking the user to login or register depending on whether he/she already created an account or not.

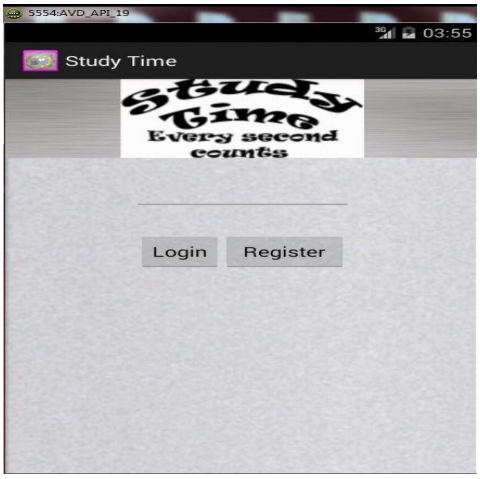

Figure 11. Start page.

4.2.2 Registration Page

This is the page used by the user for the registration purpose. It has edit text fields for first name, surname, email, password and password confirmation. These are the parameters needed for successful registration.

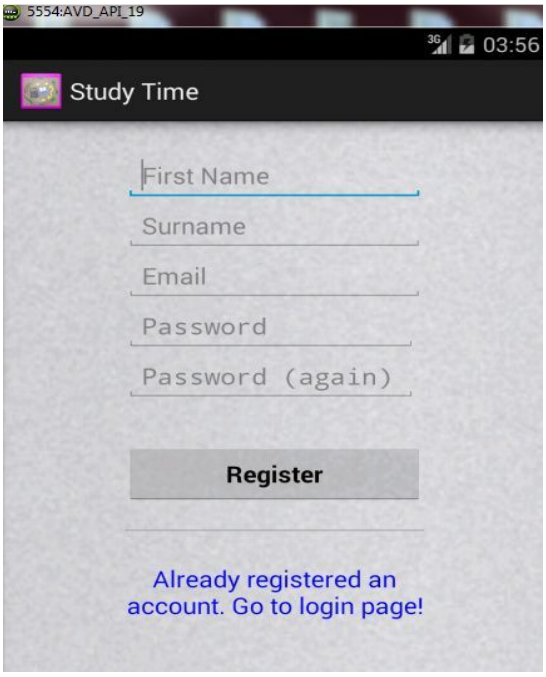

Figure 12. Registration page.

4.2.3 Login Page

This is the page used by the user for the login purpose. It has edit text fields for email and password which are the two parameters needed for login.

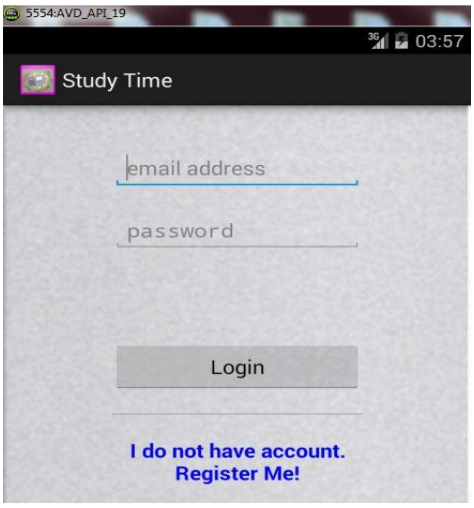

Figure 13. Login page.

4.2.4 Welcome Page

This is the page that comes up after the user successfully logged in or registered. It consists of buttons as shown in the figure below.

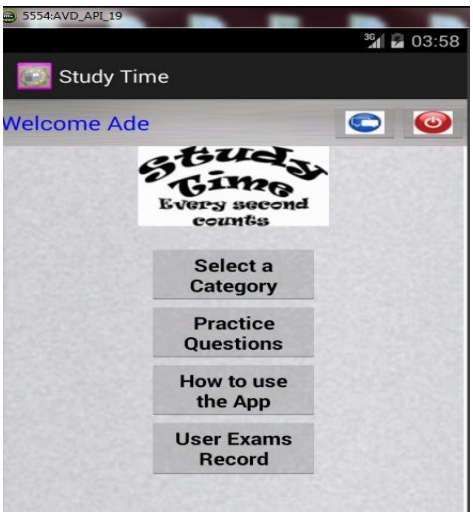

Figure 14. Welcome page.

4.2.5 Categories Page

When a user clicks on the button 'Select a Category' from the welcome page, he/she will be taken to this page listing the names of available categories.

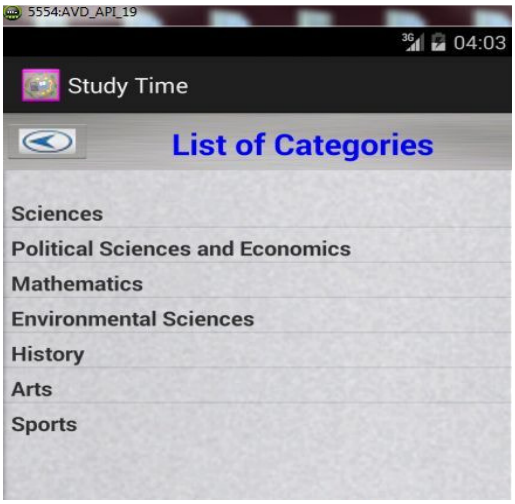

Figure 15. Categories page.

4.2.6 Subjects Page

When a user chooses a category from the categories page, a new page comes up on which subjects under that category are listed. For instance, if a user chooses Sciences from above, this page shown below is obtained.

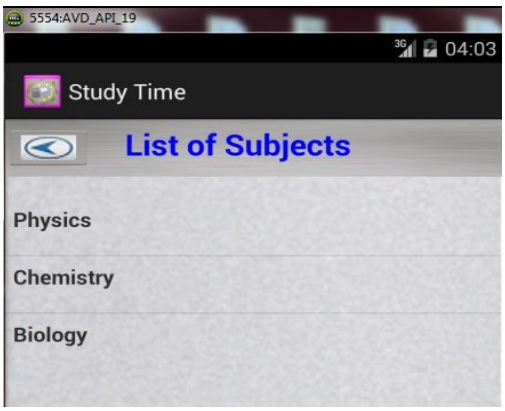

Figure 16. Subjects page.

4.2.7 Start Exam Page

This is the page that is shown to the user just before the start of an examination. It tells the user about the total number of questions and what is required from the user to perform well in the examination.

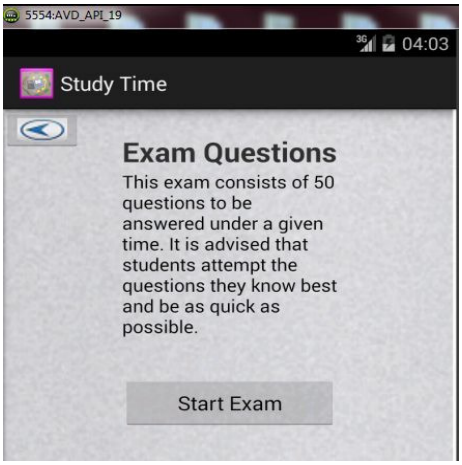

Figure 17. Start Exam page.

4.2.8 Exam Page

This is the page where questions for the examination are displayed one after the other with the help of next and previous buttons.

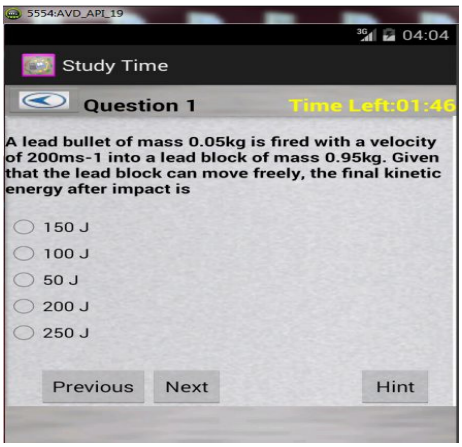

Figure 18. Exam page.

4.2.9 Result Page

This page displays the user examination score and the review of user response to each of the questions.

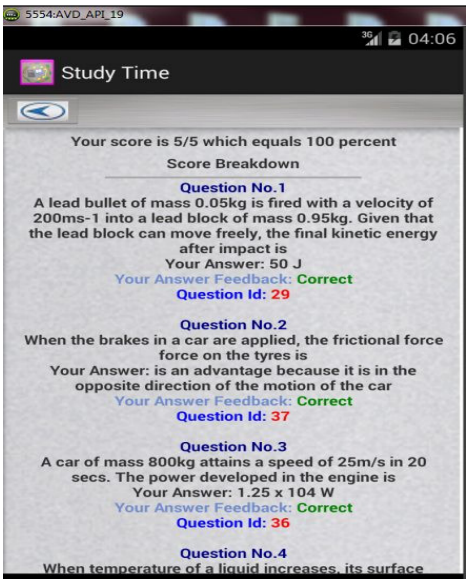

Figure 19. Result page.

4.2.10 Instruction Page

This page takes the user through the use of the application. It consists only of some texts telling the user how to navigate through it.

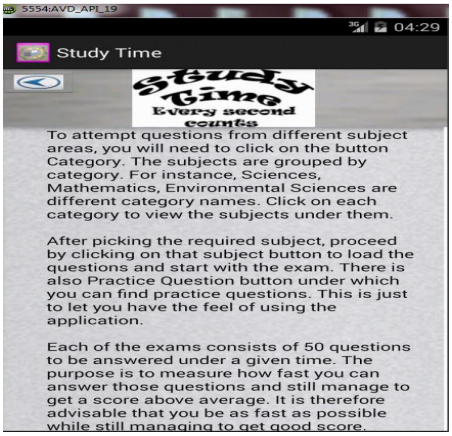

Figure 20. Instruction page.

4.2.11 Exams Record Page

This page lists the user examination record. It consists of the subject name, the score and the date the exam was taken.

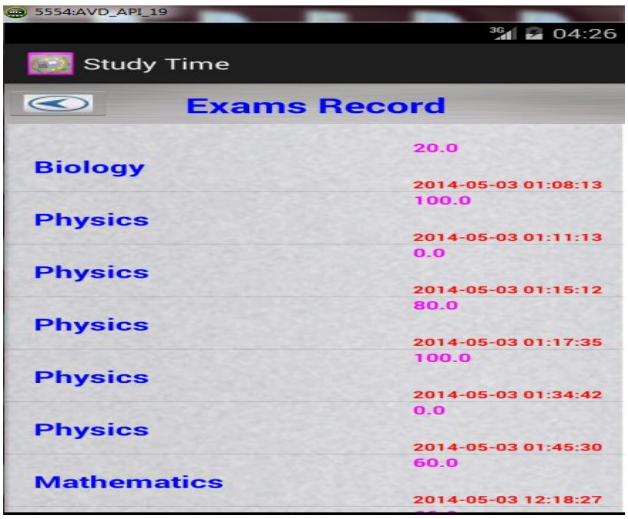

Figure 21. Exams Record page.

4.2.12 Single Record Page

This page displays the information about each of the exam records.

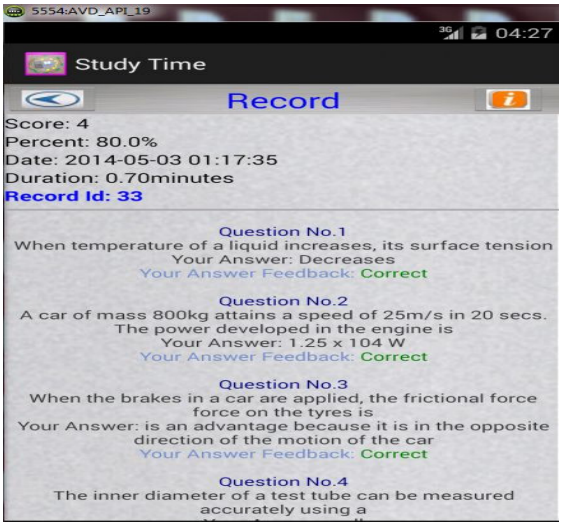

Figure 22. Single Record page.

4.2.13 Complaints Page

This page is included to allow the users report an incorrect information about a particular question or questions.

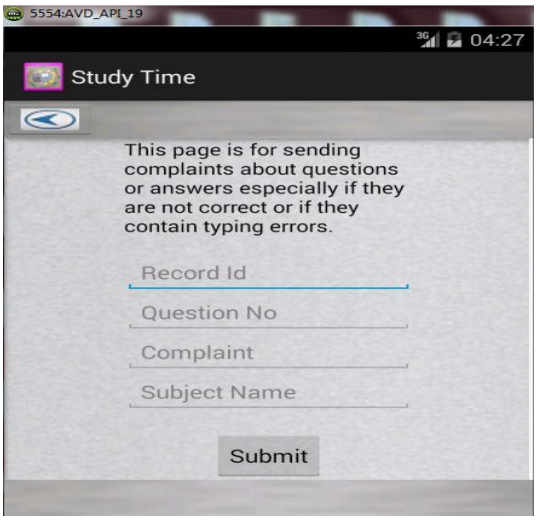

Figure 23. Complaints page.

4.2.14 Feedback Page

This page is meant for the users to give their feedback about the application. They can give suggestions on how the application can be improved.

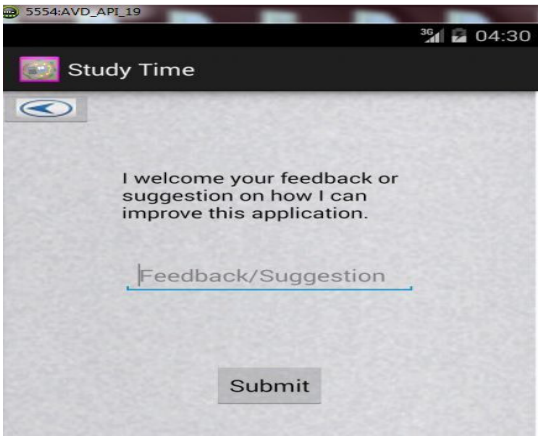

Figure 24. Feedback page.

5 IMPLEMENTATION

5.1 General Description of Implementation

The implementation of the project adhered strictly to the application requirements as stated in section 3.2.1 above. In the development of this project, an agile software development strategy was used because there was a constant need to add new functionality all through the implementation stage of the project. Some of these functionalities were suggested by my supervisor while the rest came from discussions with some teenagers who are the target users of the application.

The project was developed in the Eclipse integrated development environment (IDE) using the Android software development kit (Android SDK Tools). The application is targeted at any Android mobile device running at least Android 2.2, which corresponds to API level 8 up to the latest Android SDK Version. As at the time of writing this report, Android 4.4.2 (Android KitKat) API level 19 is the latest Android SDK Version.

5.2 Implementation of Different Parts

The implementation of this project consists of two parts. These are client interface and the web service part. The client interface is the part visible to the user and the one s/he interacts with. The web service part of the application is not visible to the user. Only the administrator can interact with server side of the application.

The client side of the application is made up of several java classes among which are the activity classes which are responsible for displaying the graphical user interface GUI and also for manipulating data for this application. In other words, an activity class normally takes care of the creation of a window in which user interface (UI) can be placed./6/

The overview of the project structure in Eclipse is as shown in the diagram below.

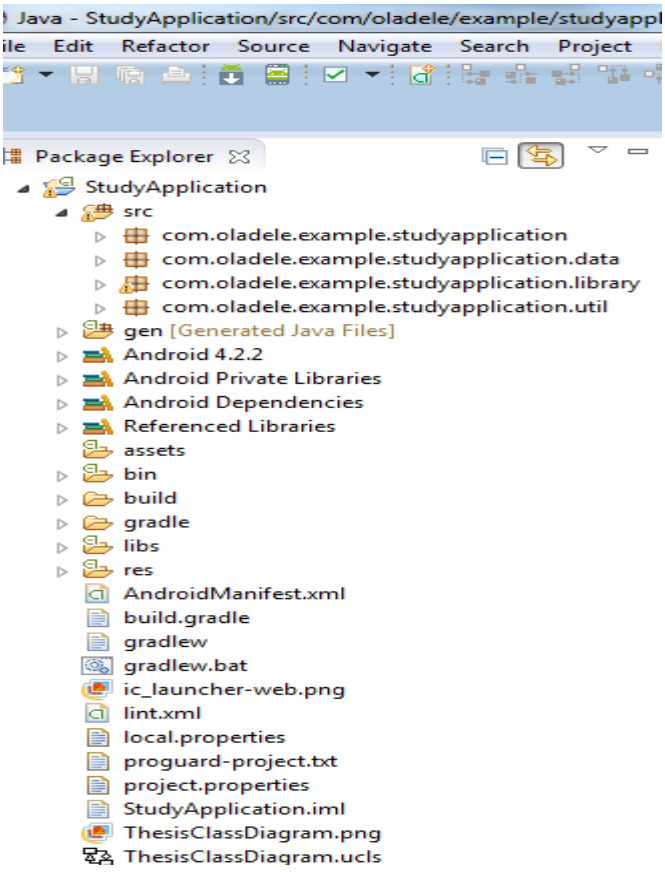

Figure 25. Project Structure in Eclipse.

As shown, the project consists of four java packages, namely:

- com.oladele.example.studyapplication
- com.oladele.example.studyapplication.data
- com.oladele.example.studyapplication.library
- com.oladele.example.studyapplication.util

The java classes contained in the studyapplication package are responsible for displaying different UI in the application and for data manipulation. The data package contains all data management java classes. The library package contains library classes while util package contains a java class to display the user result in html format.

The contents of the packages are as shown in the diagram below.

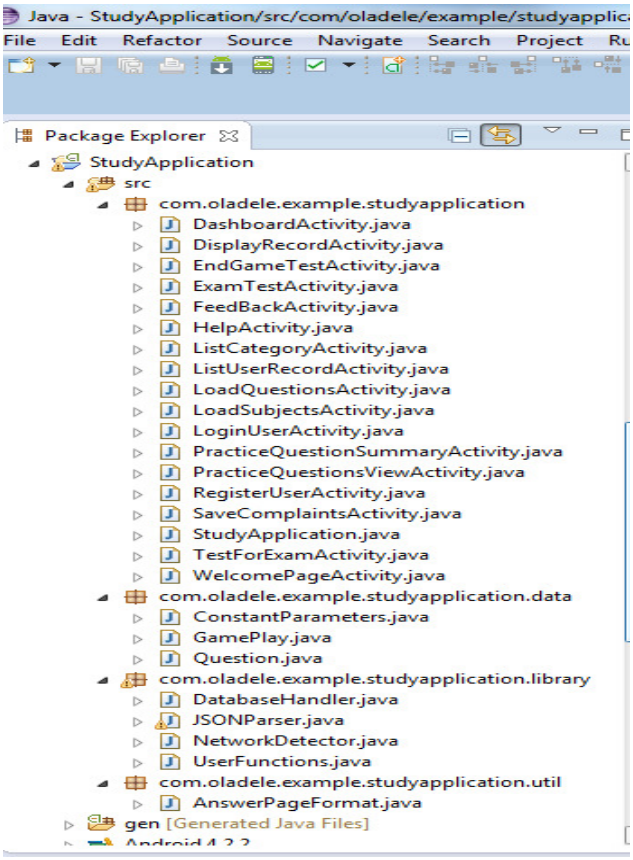

Figure 26. Application Java Classes.

5.2.1 WelcomePageActivity Class

The class is responsible for displaying the application start page. It has two buttons to take the user to either the login or registration page. If the login button is clicked by the user, the code snippet that gets executed is shown below:

```
Intent i = new Intent(getApplicationContext(), LoginUserActivity.class);
startActivity(i);
```

Snippet 1. Starting LoginUserActivity.

However, if the user clicks on the registration button, the code snippet that gets executed is similar to the one above and it is as written below:

```
Intent i = new Intent(getApplicationContext(), RegisterUserActiv-
ity.class);
```

```
startActivity(i);
```

Snippet 2. Starting RegisterUserActivity.

5.2.2 LoginUserActivity Class

This class is responsible for displaying the application login page. It checks for empty fields when the user clicks on login button. The code snippet for doing that is shown below:

```
btnLogin.setOnClickListener(new View.OnClickListener() {

	public void onClick(View view) {
		cp.email = inputEmail.getText().toString();
		cp.password = inputPassword.getText().toString();
		if (cp.email.trim().equals("") ||
			cp.password.trim().equals("")) {

		Toast.makeText(LoginUserActivity.this, "Please Fill the
			Empty Fields", Toast.LENGTH_LONG).show();
		return;
	}
	new LoginUser().execute(cp.email, cp.password);
			}
		});
```

Snippet 3. Implementation of login button.

As shown above, if all fields are filled, class LoginUser() is called with arguments email and password. This class is responsible for the background task of logging in the user. It sends the user email address and password to the server side of the application and returns JSON object. The content of LoginUser() is shown below:

This method onPreExecute() is responsible for showing the progress dialog before starting background thread.

```
@Override
protected void onPreExecute() {
	super.onPreExecute();
	cp.pDialog = new ProgressDialog(LoginUserActivity.this);
	cp.pDialog.setMessage("Logging in....wait a moment!");
	cp.pDialog.setIndeterminate(false);
	cp.pDialog.setCancelable(true);
```

```
        cp.pDialog.show();
              }
```

Snippet 4. Progress dialog.

The code snippet shown below is the doInBackground method. It takes care of the background thread of sending an http request to the web service side of the application.

```
protected JSONObject doInBackground(String... params) {

        if (params.length != 2)
              return null;

List<NameValuePair> param = new ArrayList<NameValuePair>();
              param.add(new BasicNameValuePair("tag", get
                    String(R.string.LOGIN)));
              param.add(new BasicNameValuePair("email",
                    cp.email));
              param.add(new BasicNameValuePair("password",
                    cp.password));
JSONObject json = jParser.getJSONFromUrl(getString(R.string.loginURL),
                    "POST", param);
              Log.e("JSON", json.toString());
              return json;

              }
```

Snippet 5. Background thread for login.

The following code snippet adds user details to an SQLite database on the mobile device. It also passes the user first name and email address to DashboardActivity class.

```
db.addUser(
        json_user.getString(getString(R.string.Firstname)),
        json_user.getString(getString(R.string.SURNAME)),
        json_user.getString(getString(R.string.EMAIL)),
        json.getString(getString(R.string.UID)),
        json_user.getString(getString(R.string.CREATEDAT)));
String firstname = json_user.getString(getString(R.string.Firstname));
String email = json_user.getString(getString(R.string.EMAIL));

Intent intent = new Intent(getApplicationContext(),
                    DashboardActivity.class);

intent.addFlags(Intent.FLAG_ACTIVITY_CLEAR_TOP);
```

```
intent.putExtra("Firstname", firstname);
intent.putExtra("Email", email);
startActivity(intent);
```

Snippet 6. Adding user information to a local database.

5.2.3 RegisterUserActivity Class

The class is similar in functionality to the LoginUserActivity class explained above. It displays the registration form and checks for empty fields when register button is clicked.

```
setContentView(R.layout.register);
```

Snippet 7. Setting the registration UI.

Shown above is the code snippet to display the registration form. The checking of empty fields done in this class is also very similar to the one in LoginUserActivity class, therefore this code snippet will not be repeated. The code snippets shown below are used to confirm password match and to set the minimum number of characters required for a password.

```
else if (!confirmPassword.equals(cp.password)) {

	Toast.makeText(RegisterUserActivity.this,
	"Password does not match", Toast.LENGTH_SHORT).show();
		return;
} else if (cp.password.length() < 2) {

	Toast.makeText(RegisterUserActivity.this,
	"Minimum of 2 characters are needed for password",
	Toast.LENGTH_SHORT).show();
		return;
```

Snippet 8. Checking password and its number of characters.

The next code snippet which is called when all the required fields has been filled is similar to that in LoginUserActivity class. The name of the class is Register-User and it extends AsyncTask. It sends all the user registration information in the form of NameValuePair to the web service side of the application in its doIn-

Background method and also gets back the response in its onPostExecute() method.

5.2.4 DashboardActivity Class

The DashboardActivity class is called after a successful user login. It displays the welcome page of the application. It consists of six buttons to select a category, to go to practice questions, to view the instruction, to view user exams record, to send feedback and to logout from the application. When a user clicks on each of these buttons, another activity class that handles that operation is called.

```
btnLogout.setOnClickListener(new View.OnClickListener() {

   public void onClick(View arg0) {
      // TODO Auto-generated method stub
      AlertDialog.Builder builder = new AlertDialog.Builder(
             DashboardActivity.this);
      final AlertDialog alertDialog = builder.create();

      alertDialog.setMessage("Do you really want to log out?");
      alertDialog.setButton(DialogInterface.BUTTON_POSITIVE,
             "Yes", new DialogInterface.OnClickListener() {

                @Override
                public void onClick(DialogInterface dialog,
                       int which) {
                   // TODO Auto-generated method stub
                   alertDialog.dismiss();

                   userFunctions
                        .logoutUser(getApplicationContext());

                   Intent intent = new Intent(
                          getApplicationContext(),
                          WelcomePageActivity.class);
                   intent.addFlags(Intent.FLAG_ACTIVITY_CLEAR_TOP);
                   startActivity(intent);
                   // Closing dashboard screen
                   finish();
             }
       });
```

Snippet 9. Logout button implementation.

The above code snippet shows the logout operation. If a user chooses to log out from the application, this code snippet is executed. The logoutUser method in UserFunctions class is called. The method is shown below.

```
public boolean logoutUser(Context context) {
		DatabaseHandler db = new DatabaseHandler(context);
		db.resetTables();
		return true;
	}
```

Snippet 10. Resetting login table after logout.

This method calls another method named resetTables(). The resetTables method is defined in DatabaseHandler class. Its content is as shown below.

```
public void resetTables() {
		SQLiteDatabase db = this.getWritableDatabase();
		db.delete(TABLE_LOGIN, null, null);
		db.close();
	}
```

Snippet 11. Content of resetTables method.

It simply resets the table by deleting the user information. This way, the table is made to be empty.

5.2.5 ListCategoryActivity Class

This class is responsible for sending a request to the web service side of the application to get the list of available subject categories. The class is called when a user clicks on the button to select a category from the welcome page. If categories exist, it displays the list of these categories, otherwise, it displays a feedback saying no category is available.

```
class LoadAllCategories extends AsyncTask<String, String, JSONObject> {

	@Override
	protected void onPreExecute() {
		super.onPreExecute();
		cp.pDialog = new ProgressDialog(ListCategoryActivity.this);
```

```
        cp.pDialog.setMessage("Loading categories. Please
              wait...");
        cp.pDialog.setIndeterminate(false);
        cp.pDialog.setCancelable(false);
        cp.pDialog.show();
}

protected JSONObject doInBackground(String... args) {

        List<NameValuePair> params = new ArrayList
              <NameValuePair>();
        params.add(new BasicNameValuePair("tag", getString
              (R.string.CATEGORIES)));
        JSONObject jObj = jParser.getJSONFromUrl
              (getString(R.string.categoryURL), "GET", params);
        Log.d("All Categories: ", jObj.toString());
        return jObj;
}

protected void onPostExecute(JSONObject jObj) {
        cp.pDialog.dismiss();

        try {

          if (jObj != null && jObj.getString(getString
              (R.string.SUCCESS)) != null) {

              cp.categoryerrorMessage.setText("");

              String res = jObj.getString(getString
                     (R.string.SUCCESS));
              categories = jObj.getJSONArray(getString
                     (R.string.CATEGORIES));

              if (Integer.parseInt(res) == 1) {

                for (int i = 0; i < categories.length(); i++) {
                   JSONObject c = categories.getJSONObject(i);

                    String cat_id = c.getString(getString
                           (R.string.CID));
                    String name = c.getString(getString
                           (R.string.CNAME));

                   HashMap<String, String> map = new
                           HashMap<String, String>();

                    map.put(getString(R.string.CID), cat_id);
                    map.put(getString(R.string.CNAME), name);

                    categoriesList.add(map);
                    cp.pDialog.dismiss();
              }
        } else {
```

```
                cp.categoryerrorMessage.setText("No category found");
                cp.pDialog.dismiss();
             }
        } else {

             cp.pDialog.dismiss();
             Toast.makeText(ListCategoryActivity.this,
                   "Unknown Command", Toast.LENGTH_LONG).show();
        }
      } catch (JSONException e) {
             e.printStackTrace();
      }

      runOnUiThread(new Runnable() {
         public void run() {

            ListAdapter adapter = new SimpleAdapter(
              ListCategoryActivity.this, categoriesList,
              R.layout.list_item, new String[] { getString
                   (R.string.CID), getString(R.string.CNAME) }, new
                   int[] { R.id.cat_id, R.id.name });
            setListAdapter(adapter);

          }
      });
    }
}
```

Snippet 12. Background thread to load all categories in a listview.

The class uses ListView to display the available category names. In order to achieve this, a ListAdapter is created to wrap the categories information. When a user selects a category, the code snippet shown below is used to get the position and the id of that category. The string cat_id of the chosen category is received and passed on to another activity to load the available subjects under the chosen category. Also the first name and the email of the user are passed along to the next activity class.

```
ListView lv = getListView();

lv.setOnItemClickListener(new OnItemClickListener() {

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

            Intent in = new Intent(getApplicationContext(),
                        LoadSubjectsActivity.class);
```

```
                String cat_id = ((TextView) view.findViewById(R.id.cat_id))
                            .getText().toString();

                in.putExtra("Cat_id", cat_id);
                in.putExtra("Firstname", cp.firstname);
                in.putExtra("Email", cp.email);

                startActivity(in);
        }
});
```

Snippet 13. Selecting a category.

5.2.6 LoadSubjectsActivity Class

This class is responsible for displaying the available subjects under each of the categories. It works in the same way as the ListCategoryActivity class. The only difference is that it requires cat_id which represents the category id from the previous activity class. The code snippet shown below indicates how to retrieve the cat_id.

```
Intent i = getIntent();
cp.cat_id = i.getStringExtra("Cat_id");
```

Snippet 14. Retrieving category id from intent.

The code snippet below shows how the cat_id is used in sending a GET request to the server side of the application to get the list of subjects belonging to that category. url_subject contains the path to the PHP file to get the list of subjects on the server side of the application.

```
List<NameValuePair> params = new ArrayList<NameValuePair>();
params.add(new BasicNameValuePair("tag",
       getString(R.string.SUBJECTS)));
params.add(new BasicNameValuePair(getString(R.string.CID),
       cp.cat_id));

String url_subject = getString(R.string.subjectURL);

JSONObject json = jParser.getJSONFromUrl(url_subject, "GET", params);
```

```
Log.d("All Subjects: ", json.toString());
       return json;
       }
```

Snippet 15. Getting a list of subjects belonging to a category.

After getting and displaying the list of available subjects, the user can click on any of the subject areas to get the list of questions. In order to get the list of questions, the subject id (sub_id) is needed. The sub_id, sub_name (Subject name), exam_time (Exam duration), email and the first name are passed on to the next activity class as shown below.

```
Intent in = new Intent(getApplicationContext(),
       LoadQuestionsActivity.class);

String sub_id = ((TextView) view.findViewById(R.id.sub_id))
              .getText().toString();

String sub_name = ((TextView) view
       .findViewById(R.id.subject_name)).getText().toString();

String exam_time = ((TextView) view
              .findViewById(R.id.timeforexam)).getText().toString();

       in.putExtra("Sub_id", sub_id);
       in.putExtra("SubjectName", sub_name);
       in.putExtra("ExamTime", exam_time);
       in.putExtra("Email", cp.email);
       in.putExtra("Firstname", cp.firstname);

       startActivity(in);
```

Snippet 16. Passing string values from one activity to another.

5.2.7 LoadQuestionsActivity Class

This class is called immediately after the user clicks on a subject area. The purpose of this class is to take the sub_id and make use of that in a request that is sent to the server side of the application to get the list of questions belonging to that subject area.

```
cp.questions = json.getJSONArray(getString(R.string.QUESTIONS));
quesList = new ArrayList<Question>();
int n = cp.questions.length();

for (int i = 0; i < n; i++) {

	Question que = new Question();

	JSONObject c = cp.questions.getJSONObject(i);

	que.setId(c.getString(getString(R.string.QID)));
	que.setQuestions(c.getString(getString(R.string.QTEXT)));
	que.setAnsHint(c.getString(getString(R.string.ANSHINT)));
	que.setOption1(c.getString(getString(R.string.OID1)));
	que.setOption2(c.getString(getString(R.string.OID2)));
	que.setOption3(c.getString(getString(R.string.OID3)));
	que.setOption4(c.getString(getString(R.string.OID4)));
	que.setOption5(c.getString(getString(R.string.OID5)));
	que.setAnswer(c.getString(getString(R.string.ANSW)));
	quesList.add(que);

}

for (Question d1 : quesList) {
		System.out.println(d1.getId() + d1.getQuestions()
			+ d1.getQuestionOptions() + d1.getAnsHint());

}

} catch (JSONException e) {
	e.printStackTrace();
}
```

Snippet 17. Retrieving list of questions and putting them in an array list.

As shown in the code snippet above, an instance of Question class is created and it is used to set the values of question object such as Id, Questions, AnsHint and so on. These question objects are then saved in quesList which is an ArrayList.

5.2.8 TestForExamActivity Class

The purpose of this class is to display each of the questions together with its set of options to the user one after the other. The class implements 'next' and 'previous' buttons to allow the user navigate to the next question and vice versa. The class also displays the duration for the exam.

```
que = getIntent().getParcelableArrayListExtra("key");
currentGamePlay = new GamePlay();
currentGamePlay.setQuestions(que);
((StudyApplication) getApplication()).setCurrentGame(currentGamePlay);

Log.d("Class", "This is TestForExamActivity");

for (Question d1 : que) {
      System.out.println(d1.getId() + d1.getQuestions()
                        + d1.getQuestionOptions() + d1.getAnswer()
                        + d1.getChosenOption() + d1.getAnsHint());
            }

currentGamePlay = ((StudyApplication) getApplication())
                        .getCurrentGame();

currentQuestion = que.get(cp.count);
```

Snippet 18. Getting current question.

As shown above, the array of questions is retrieved from the intent and saved in the variable que. Then a new instance of GamePlay class is created with a name currentGamePlay and the value of variable Questions in GamePlay class is set to que. The value of variable CurrentGame is set to currentGamePlay. The variable currentQuestion is used to iterate through the ArrayList que to get the current question. The variable count has an initial value of zero.

```
private void setQuestionView() {

      cp.hint = currentQuestion.getAnsHint();
      cp.textQuestion.setText(currentQuestion.getQuestions());
      List<String> answers = currentQuestion.getQuestionOptions();
      cp.option1.setText(answers.get(0));
      cp.option2.setText(answers.get(1));
      cp.option3.setText(answers.get(2));
      cp.option4.setText(answers.get(3));
      cp.option5.setText(answers.get(4));

      String chosenOption = currentQuestion.getChosenOption();
      if (chosenOption != null) {

            if (chosenOption.equalsIgnoreCase(cp.option1.getText()
                  .toString())) {
                  cp.option1.setChecked(true);
            } else if (chosenOption.equalsIgnoreCase
                  (cp.option2.getText().toString())) {
                  cp.option2.setChecked(true);
            } else if (chosenOption.equalsIgnoreCase
```

```
                (cp.option3.getText().toString())) {
                cp.option3.setChecked(true);
          } else if (chosenOption.equalsIgnoreCase
                (cp.option4.getText().toString())) {
                cp.option4.setChecked(true);
          } else if (chosenOption.equalsIgnoreCase
                (cp.option5.getText().toString())) {
                cp.option5.setChecked(true);
                }
          }

    }
```

Snippet 19. Setting a view for current question and its set of options.

This code snippet is used to set the current values of question and its set of options each time the next and the previous buttons are clicked. Also it checks whether currentQuestion's chosenOption parameter is null or not. If it is not null, the appropriate radiobutton containing the option is checked. However, if no option has been chosen previously for the current question, all the radiobuttons are left unchecked. The hint for answering the current question is also retrieved and shown only when the user clicks on the hint button.

5.2.9 EndGameTestActivity Class

The purpose of this class is to display the result of every examination as well as sending a request to the server side so that the result can be saved automatically into the database.

```
cp.examScore = Integer.toString(cp.score);
cp.exampercentScore = Integer.toString(cp.percentScore);

int numberofquestions = b.getInt("number");

String str = Integer.toString(cp.score);
String str1 = Integer.toString(cp.percentScore);
examScore.setText("Your score is" + " " + str + "/" + numberofquestions
                + " " + "which equals" + " " + str1 + " " + "percent");
examSummary.setText(R.string.summarytext);

currentGame = ((StudyApplication) getApplicationContext())
                                .getCurrentGame();

cp.answers = AnswerPageFormat.getAnswersToQuestions(currentGame
```

```
                                    .getQuestions());
scoreSummary.setText(Html.fromHtml(cp.answers));

new SaveUserRecord().execute(cp.email, cp.sub_name, cp.exam_result_page,
                                    cp.examScore, cp.exampercentScore,
                                    cp.time_taken);
```

Snippet 20. Displaying the examination result.

The above code snippet shows a part of implementation of this class. The score and the percentage score from the examination are both converted to string values so that each of them can be displayed in a textview. The variable answers holds the result of the examination as shown. Also, the value of variable scoreSummary which is a textview is set to the value of answers and this is displayed in HTML format.

Finally, the class SaveUserRecord is called with arguments email, sub_name (subject name), exam_result_page, examScore, exampercentScore and the time_taken. These parameters are saved to the database table on the server side of the application.

6 TESTING

6.1 Client and Server Sides Testing

All the testing done in the Client side of the application were carried out on a real Android mobile device - Samsung Galaxy Trend. The screen of the mobile phone was printed and saved to a file with the help of DroidAtScreen software program. The software basically displays the screen of any Android mobile device connected to a computer on the computer screen and whatever application running on the phone will be displayed simultaneously on the computer.

6.1.1 Registration Page Testing

The following screenshots show the testing done on the registration page.

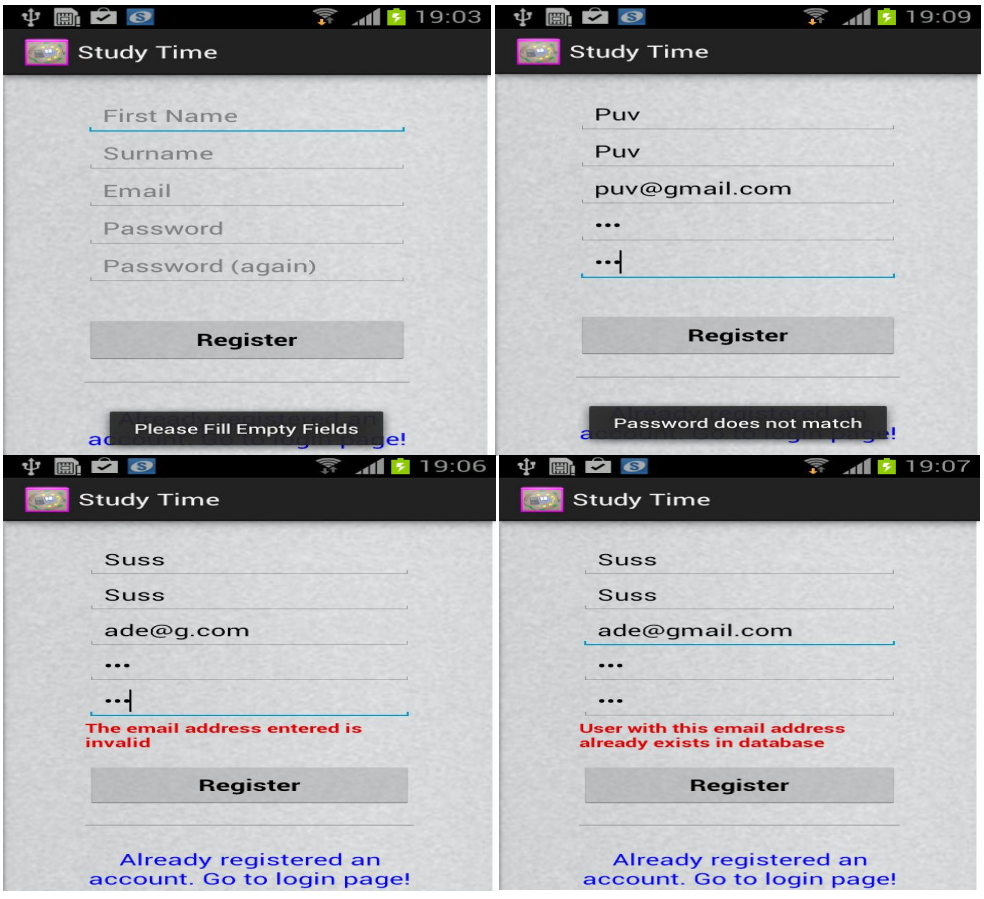

Figure 27. Registration sequence testing.

The first screenshot in the Figure 27 above shows the response obtained when the register button is clicked with all the fields empty. This same response is obtained as long as any of the fields is empty when the register button is pressed. The second screenshot shows the response obtained when the password and password confirmation entered are not equal. The third screenshot shows the response got from the server side of the application when an invalid email address is entered for the registration purpose. The last screenshot shows the response obtained from the server side of the application when email address supplied for the registration is already on the database.

6.1.2 Welcome, Records, Category and Subject Pages Testing

The following screenshots show the result of testing each of the pages shown.

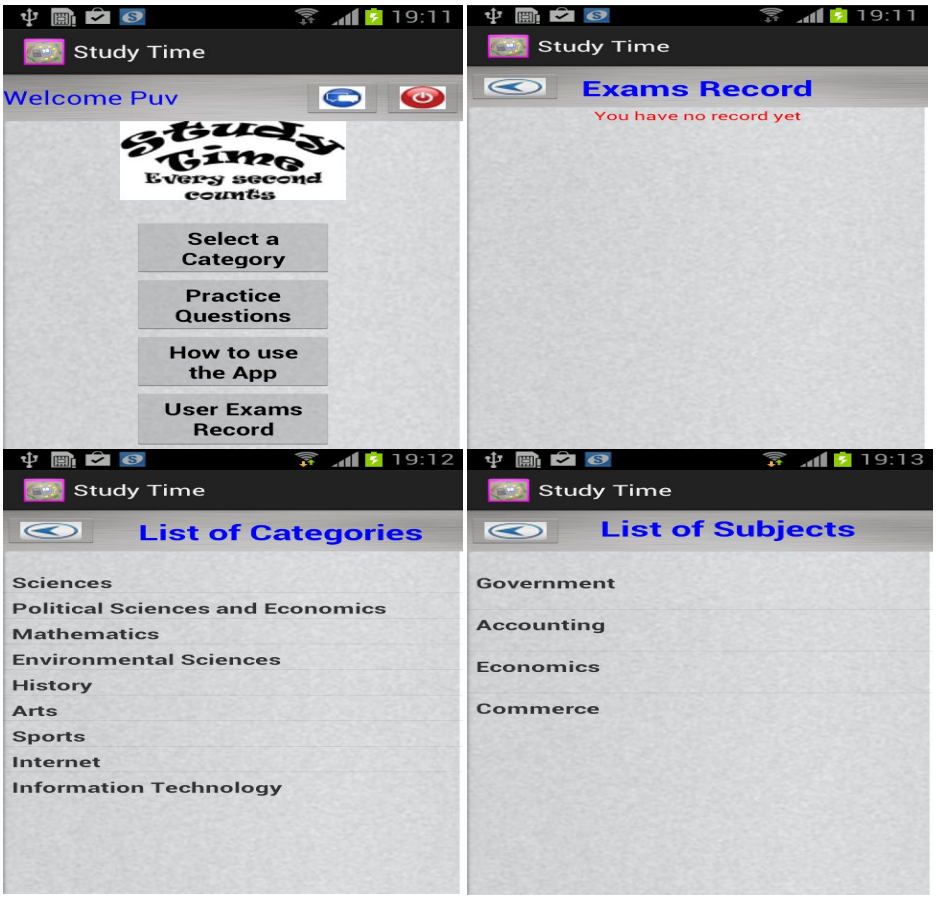

Figure 28. Testing Welcome, Record, Category and Subject pages.

The first screenshot in figure 28 above represents the welcome page after a successful login or successful account creation. The name of the user as shown in the screenshot is Puv. The second screenshot shows the response obtained when the user clicked on the User Exam Record button. The response shows that the user has no examination record yet. The third screenshot shows the list of available categories after clicking on Select a Category button. The last screenshot displays the subject areas available under Political Sciences and Economics. The figure 29 below shows the JSON response obtained when the list of subjects belonging to category with id 2 is searched from the web browser.

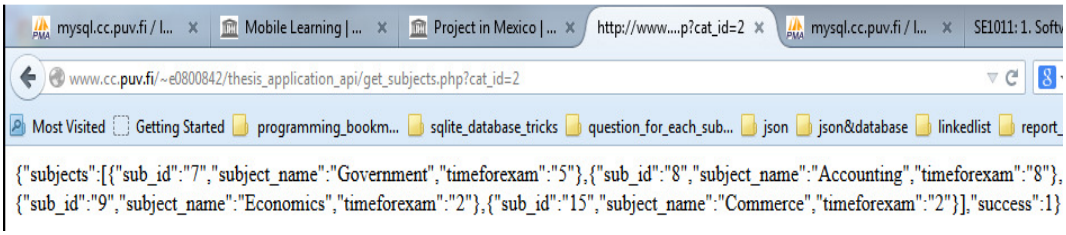

Figure 29. Testing the server side for subjects list.

The table below is the subjects table. The table can be used to verify the JSON response shown above in the web browser. As can be seen from the database table, the subjects with cat_id equals to 2 are Government, Accounting, Economics and Commerce.

sub_id	cat_id	subject_name	timeforexam
1	1	Physics	2
2	4	Agricultural Science	7
3	1	Chemistry	10
4	3	Further Mathematics	15
5	4	Geography	7
6	1	Biology	7
7	2	Government	5
8	2	Accounting	8
9	2	Economics	2
10	3	Mathematics	15
11	6	Fine Arts	5
12	6	English Language	10
13	5	English Literature	7
14	5	History	7
15	2	Commerce	2

Figure 30. Showing the list of subjects in subjects table.

The table below shows the addition of user Puv to the users database table after a successful registration of the user.

uid	unique_id	firstname	surname	email	encrypted_password	salt	created_at
31	53682b0ba49121.72753406	Dewaleiladddelle	Dfddfggfff	dgffff@gmail.com	PyYaffllOPqlDsk4m0KNezG0X5dhODAxMTg0YWM4	a801184ac8	2014-05-06 03:21:31
32	536aa1d43166e5.41600059	Ramat	Ramat	ramat@gmail.com	Wk0fMKwl5X4ssxf+EEF0MsEpKw02NDAzZTFkNzE2	6403e1d716	2014-05-08 00:12:52
33	536ab5e6f05de9.86748179	Car	Car	car@gmail.com	XtvYRcVcMEe/F27iKOrbeD0YY5BjZjI3ZmJkMWMz	cf27fbd1c3	2014-05-08 01:38:30
34	536ab828a91578.33513370	Bus	Bus	bus@gmail.com	hW6fai31sGZZ8axkYw1sG7/X4HwwZTBjZTkzMTUx	0e0ce93151	2014-05-08 01:48:08
35	536bff74414789.16769308	Cup	Cup	cup@gmail.com	SyP487BYTTCdETDdd1vAm3NO84E3MTM2YzI0YmM0	7136c24bc4	2014-05-09 01:04:36
36	536c042a4d12e8.98678167	Book	Book	book@gmail.com	lJ9AY2xSQBEGJsvo3LGNHYaDc10yM2JlOGUyY2Nm	23be8e2ccf	2014-05-09 01:24:42
37	536c0c27d3e467.51282062	Mouse	Mouse	mouse@gmail.com	88Lo63/0xg70drMlWV7gKSQ11oc1ODZjZWU0ODg4	586cee4888	2014-05-09 01:58:47
38	536c8155ad0837.39294930	Bulb	Bulb	bulb@gmail.com	U0S7NHQsKh85t8N9apUmkPuJvHdiOTBiZmYwOTBh	b90bff090a	2014-05-09 10:18:45
39	53767ac53760a3.85082869	Tv	Tv	tv@gmail.com	o3rT8W6L+l2ykvyF3K4cOP3q1XNmMmlyYjNmYTYx	f2b2b3fa61	2014-05-16 23:53:25
40	537c3b1d3dfdf5.30920513	Bed	Bed	bed@gmail.com	ngSy8Jq5dC+O4/EYedOOYRZUZeEzOGNjYWM1YjMz	38ccac5b33	2014-05-21 08:35:25
41	537c5637c2d340.92257345	Vamk	Vamk	vamk@puv.fi	76cmYwFBH4Ou0gkT6rScxX/e7CU4M2EzYTM5NjBh	83a3a3960a	2014-05-21 10:31:03
42	5389ff2ebe26f5.52740733	Puv	Puv	puv@gmail.com	CRBd4n+tdAD7w0cNvQN2wLT92l9hNzdkYjI1MTFk	a77db2511d	2014-05-31 19:11:26

Figure 31. Showing the users table with the addition of new user, Puv.

7 SUMMARY

An Android mobile learning application consisting of the client and the server side was developed. The application was developed for all Android mobile devices running at least Android 2.2. The server side of the application consists of PHP scripts to query the database. MySQL database was used in order to keep the tables associated with the application.

The application was designed to help students prepare for examinations and study on their own during their free time with the use of their mobile phones. The application works so that a new user has to register to be able to have access to the subject areas and the questions available for each subject. The questions are saved on MySQL database.

With the application, the user can attempt examination questions from different subject areas of choice. Each of the examinations comes with a time duration under which a user is expected to finish and submit. However if the user is unable to submit the examination himself before time is up, the application submits the examination and the result is displayed to the user. This examination record is also saved automatically to an external database located on the server side of the application.

Since the application was designed to model real life examination, it is possible for the user to navigate from one question to the other even when he or she has not answered the current question. The questions are displayed one after the other with the help of next and previous buttons. The application was designed so that each of the questions consists of five options only one of which is the correct answer to the question. The questions are selected at random from the database so that if two friends decide to take examination from the same subject at the same time, they will not have the same order of the questions. The positions of the options are also rearranged every time next or previous button is clicked.

A hint on answering each question is also provided and can be viewed by the user when he or she clicks on hint button. The user can have access to his or her saved examination records for reference.

8 CONCLUSIONS

The project was designed primarily to provide assistance to students while preparing for examination. Since the application runs on the user's mobile phone, he or she can have access to the materials anywhere and at anytime. The hint on answering each question included in the application is also an advantage as students can have the opportunity to learn the tricks associated with answering the questions correctly.

The application can also be used in schools to conduct real examinations. Since examination results are saved to an external database, teachers and school authority can have access to these results. With this in mind the project was developed so that all the data used in the application including the questions, the answers and the set of incorrect options are saved on an external database rather than the local database (SQLite) to prevent any possibility of user having access to them. Even though this situation will be a difficult one, it is never totally impossible.

The timing functionality included in the application is another interesting part as a user can monitor the time he or she spends each time answering different examination questions and how accurate the responses are.

Most of the goals set at the beginning of the project design stage for this application have been achieved with its implementation. There are quite a few parts of the application which I found challenging. The most difficult part of the implementation was finding a way to hold the set of questions retrieved from the database and displaying them in a view with their set of options one after the other. For this problem, I created a class to hold each question object and added them to an array. To display each of the questions, I only needed to iterate through the array using the position of each of the elements in that array.

To develop this application further, there are some additional features that could be added. One of these features is the functionality that will make it possible for a

user to challenge a friend to a quiz. It could be implemented in such a way that the question is displayed to both of them at the same time but only one person whose turn it is to answer will be enable to answer the question under a given time. However, if the person fails to answer within the allocated time, the opponent will then have the opportunity to answer the same question. Points are awarded for a right answer to every question.

9 REFERENCES

/1/ ICT in Education. UNESCO Official website. Accessed 30.05.2014. http://www.unesco.org/new/en/unesco/themes/icts/m4ed/

/2/ Nigeria - Mobile Market - Overview, Statistics and Forecasts. Accessed 28.05.2014.

http://www.budde.com.au/Research/Nigeria-Mobile-Market-Overview-Statistics-and-Forecasts.html

/3/ Introducing JSON. JSON Official website. Accessed 31.05.2014. http://www.json.org/

/4/ History of PHP. PHP Official website. Accessed 30.05.2014. http://www.php.net/manual/en/history.php.php

/5/ About phpMyAdmin. phpMyAdmin Official website. Accessed 30.05.2014. http://www.phpmyadmin.net/home_page/index.php

/6/ Android Activity. Official Android Developer Reference website. Accessed 30.05.2014.

http://developer.android.com/reference/android/app/Activity.html

/7/ MySQL. Wikipedia Foundation. Accessed 30.05.2014. http://en.wikipedia.org/wiki/MySQL

/8/ Android (operating system). Wikipedia Foundation. Accessed 30.05.2014. http://en.wikipedia.org/wiki/Android_%28operating_system%29

/9/ Project in Nigeria. Official UNESCO website. Accessed 30.05.2014.

http://www.unesco.org/new/en/unesco/themes/icts/m4ed/teacher-support-and-development/teacher-development-with-mobile-technologies-projects-in-mexico-nigeria-pakistan-and-senegal/project-in-nigeria/

/10/ Mobile learning is gaining momentum. Training Magazine Official website. Accessed 30.05.2014.

http://www.trainingmag.com/content/mobile-learning-gaining-momentum