CLASSROOM QUIZ APP

by

HEMALA KONGANDA

B.E., Visvesvaraya Technological University, 2011

A REPORT

submitted in partial fulfillment of the requirements for the degree

MASTER OF SCIENCE

Department of Computing and Information Sciences
College Of Engineering

KANSAS STATE UNIVERSITY
Manhattan, Kansas

2014

Approved by:

Major Professor
Dr. David A. Gustafson

Copyright

Abstract

As a part of enhancing Teaching and Learning in class, this project implements a Student Response System (SRS) by providing interactive in class quizzes between Students and Instructors by the use of a smart device. The application of this project is limited to the scope of android devices. This application includes a way in which the Instructor can enter his questions and answer for the quiz and the Student receives these questions instantly, allowing the student to choose his answer to the best of his knowledge. The answers are then validated and visualized. Based on which the Instructor implements the concept of Talk to your Partner (TTYP). The Instructor would have the option of deciding if he wants to pair the users either randomly or based on their result of the previous question.

Table of Contents

List of Figures

List of Tables

Acknowledgements

This project would not have been possible without the support and help of my Major Professor Dr. David A Gustafson. I would like to extend my sincere thanks him. I am grateful to him for his guidance and constant supervision as well as for providing necessary information regarding the project

I would like to express my special gratitude and thanks to Dr. Jana Fallin without who's encouragement this project would not have been possible.

A special thanks to Dr. Mitchell L. Neilsen for being a part of my committee and providing me the resources to work on the application.

I would also like to thank my family and friends for their immense support in every possible way.

Chapter 1 - Introduction

1.1 Motivation

In the field of education, there exists the concept of Classroom Performance Systems (CPS) which is a technological way to assess students, more commonly known as Student Response System (SRS) or Audience Response Systems (ARS). [1] (Keng Siau, 2006)

Technology based classroom with activities are more interactive than the traditional classroom. The learning benefits of questions cannot be achieved if students does not participate in the processing of answering questions. In the existing systems, this student-instructor interaction is achieved by the use of "Clickers". [2][3] (Caldwell, 2007)

With smart phones being more of a necessity these days, the main idea of developing this application was to ease things by allowing the students to be able to use an application on their own device instead of using or carrying around a clicker device. The main aim of the concept of Classroom Performance Systems(CPS) and this application is to increase the efficiency of classroom learning and making the best use of technology to enhance teaching and learning in class. So another striking feature used in this application was a way to team or group students in pairs so that they could answer as teams. This concept was used as it has been proved by research that talking to your partner while deciding on an answer helps you to learn more and increases the interest and ability of a student to learn more in class. [4](Noreen M. Webb*, 2009)

1.2 Description

This application function has two kinds of users, the Instructor and the Student. So when the professor is ready for this activity he posts his question from his user interface which is a web page. On receiving this question the student responds with this answer from his android mobile device which should have the application running. On receiving all the student responses immediately, the instructor analyzes the graph and data of the students who answered wrong and the students who answered right, and decides if he wants to team them up or not for the next question. The student receives the instructor's response/decision to pair and their team based on their performance in the previous question answered individually.

Chapter 2 - Requirements

2.1 Requirement Analysis

This section describes the requirements for this application.

- *Instructor Interface and Student Interface*: The Instructor can type in his questions and answer for the quiz and the Student (android user) receives these questions and can choose the answer to the best of his knowledge.
- *Validation of Answers:* The answers are then validated based on the correct answer entered by the Instructor.
- *Graph of Student Answers:* The student answer data is visualized based on which the answers are analyzed by the Instructor.
- *Talk to your Partner:* Based on the student answers the professor implements the concept of Talk to Your Partner(TTYP). The professor would have the option of deciding if he wants to pair the students or not based on their performance in previous question.

2.1.1 Software Requirements

Operating System: Android 3.0 or higher

Language: Android, Java, PHP

Database: MySQL

Technologies: Java, XML, JSON

Tools: ADT, SDK, Linux server

2.1.2 Hardware Requirements

Samsung galaxy tab 10.1

Samsung galaxy young.

Chapter 3 - System Overview and Design

3.1 System Overview

The main components of this application include: Android, PHP based web services, JSON, MySQL Database. Let us see why we need all these components

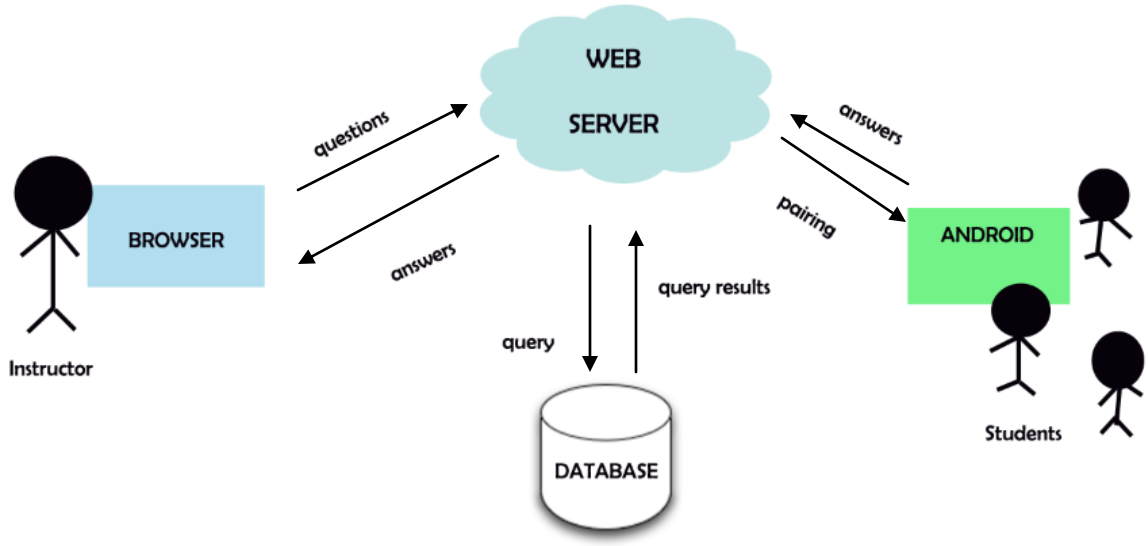

Figure 3.1 System Overview

3.1.1 MySQL

MySQL is the world's 2nd most popular, widely used open source relational database management system and is easily accessible.[5] (Wikipedia, MySQL, 2014) Apart from that it's just the fact that SQL is a pretty simple language to save, retrieve, update, and delete and store information from a database very efficiently, which can communicate with the web service too effectively.

3.1.2 PHP JSON

MySQL database and an Android app cannot communicate directly, so PHP and JSON had to be used to make this happen. PHP acts as an interpreter, which will do all the talking between android and MySQL.

3.1.3 PHP, MySQL, JSON and Android together

The android application will call a PHP script to perform basic CRUD (Create, Read, Update, Delete) operations. First your android app calls a PHP script in order to perform a data operation. The PHP script then connects to MySQL database to perform the operation. So the data flows from Android app to PHP script then finally is stored in your MySQL database.

For Example, to create a new Student User from our Android device, we will want the application to have some input fields, such as username, password, email, etc. When the user fills out this information and hits Register, it will pass the information to our PHP web service. The web service will connect to the MySQL Database, and find the "Users" table and insert a new row into the MySQL database with the information that was sent from the Android device. Once this row is created, the web service display some JSON data that will tell whether entered data successfully or not. Lastly, Android device will parse this JSON data and display a status.

3.2 System Design

Once the requirements were gathered the design diagrams are essential to implement this application which we can see are further used in the implementation of the project.

3.2.1 Use Case Diagram

Use case Diagram shows the Actors and their role in the System. The Actors in this System are Instructor and Student.

Below are the use case diagrams for Student and Instructor:

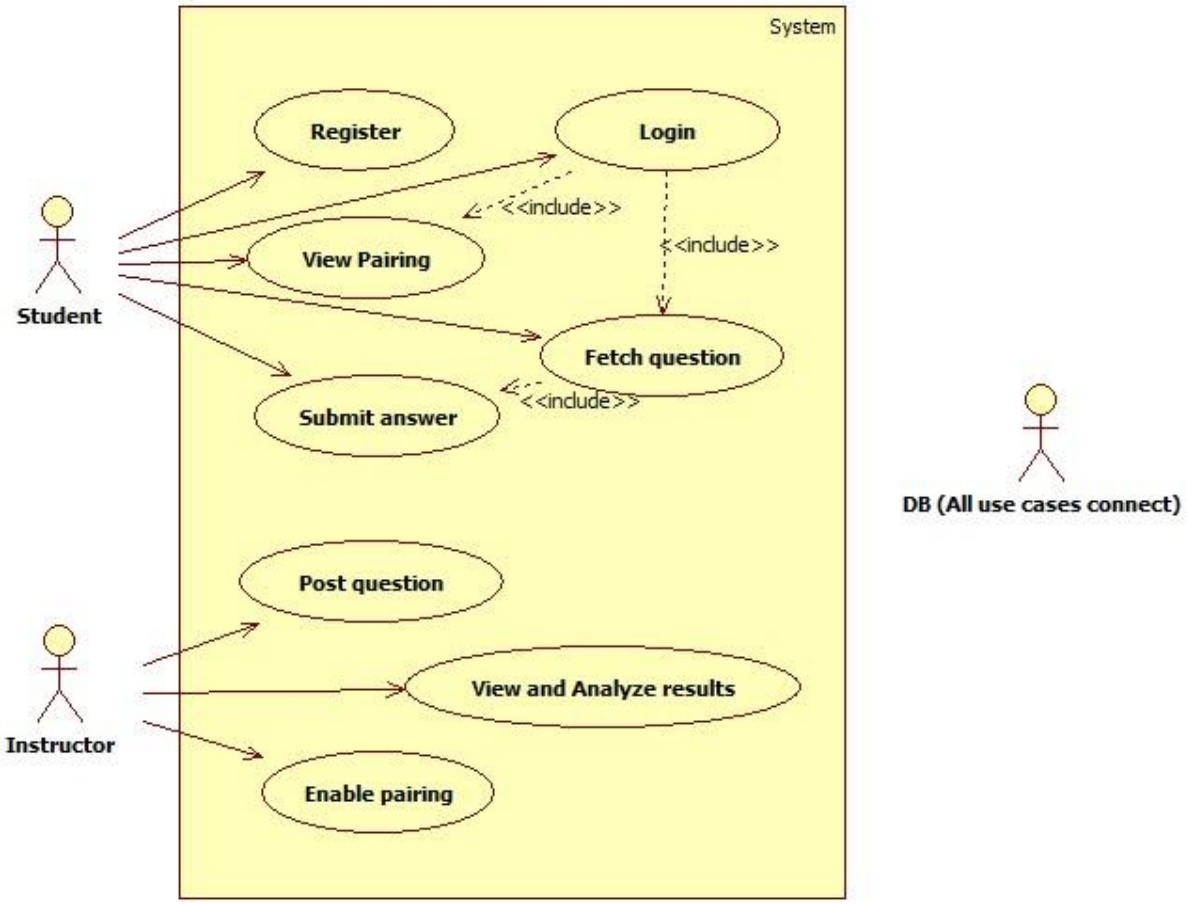

Figure 3.2 Use case diagram

3.2.2 Data Flow Diagram

The data flow diagram describes the flow of the application based on user action and all the data flow path between the client and server.

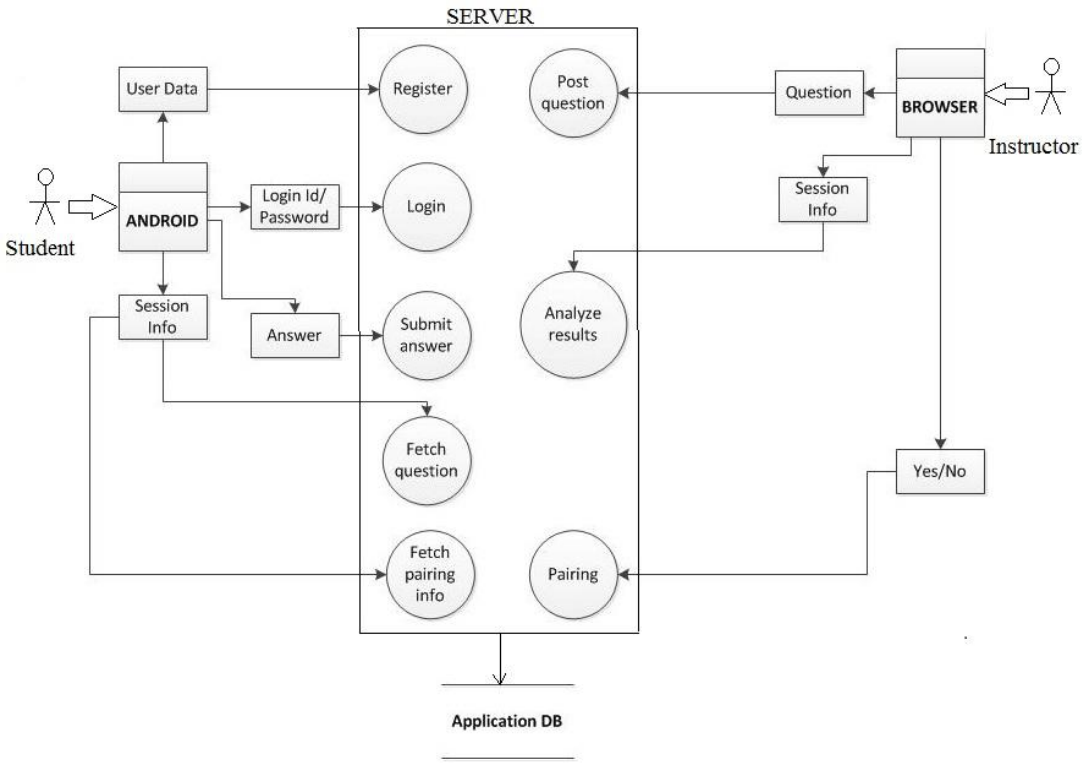

Figure 3.3 Data Flow Diagram

Chapter 4 - Implementation

This chapter briefly discusses android module and how it was implemented to interact with other component modules.

4.1 Android Module

This section discusses details of structural overview, layouts, activities and manifest implemented on the client-side.

4.1.1 Structural Overview [6](Brahler, 2010)

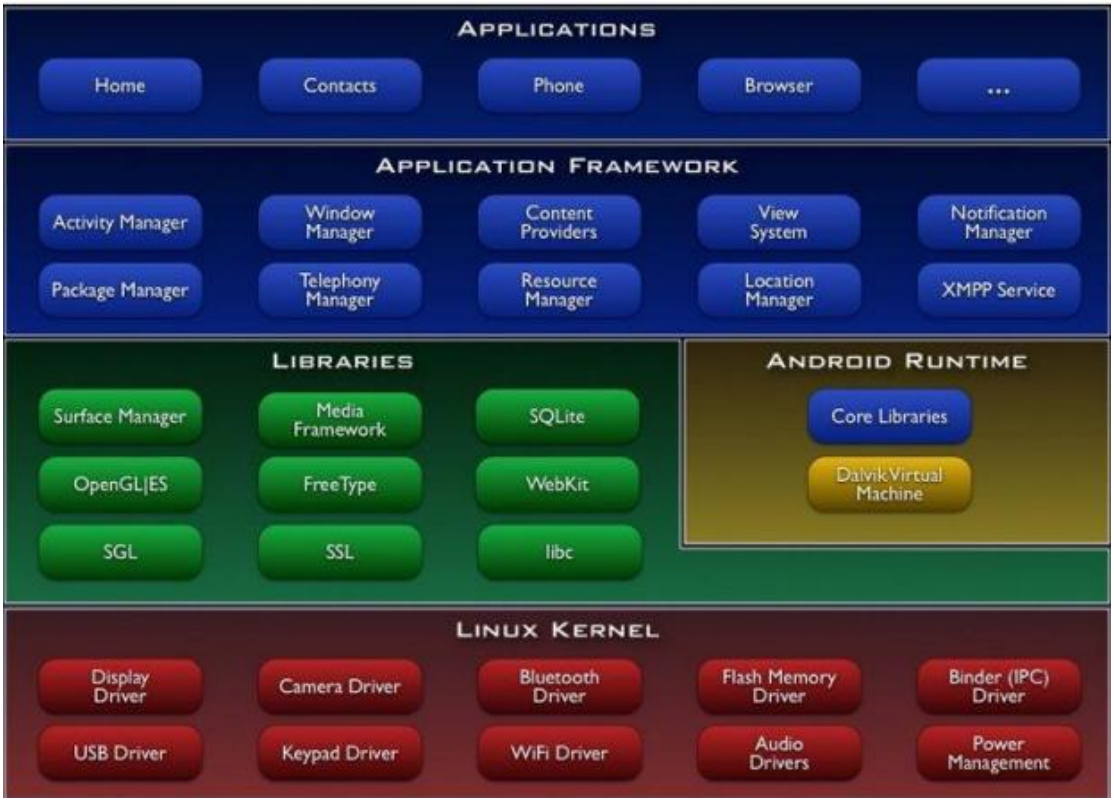

Figure 4.1 Android Architecture [6](Brahler, 2010)

Figure 4.1 depicts the different layers and component of the Android operating system. Each layer offers varied services to the layer above it. The different layers in the architecture are: The kernel and low level tools, native libraries, the Android Runtime, the framework layer and on top of all the applications. The kernel in use is a Linux 2.6 series kernel, modified for special needs in power management, memory management and the runtime environment. Right above the

kernel run some Linux typical daemons like bluez for Bluetooth support and wpa supplicant for WiFi encryption.

As Android is supposed to run on devices with little main memory and low powered CPUs, the libraries for CPU and GPU intensive tasks are compiled to device optimized native code. Basic libraries like the libc or libm were developed especially for low memory consumption and because of licensing issues on Android. In this layer the surface manager handles screen access for the window manager from the framework layer. Opposing to other frameworks, the media framework resides in this layer, as it includes audio and video codecs that have to be heavily optimized.

The Android Runtime consists of the Dalvik virtual machine and the Java core libraries. The Dalvik virtual machine is an interpreter for byte code that has been transformed from Java byte code to Dalvik byte code. Dalvik itself is compiled to native code whereas the the core libraries are written in Java, thus interpreted by Dalvik.

Frameworks in the Application Framework layer are written in Java and provide abstractions of the underlying native libraries and Dalvik capabilities to applications. Android applications run in their own sandboxed Dalvik VM and can consist of multiple components: Activities, services, broadcast receivers and content providers. Components can interact with other components of the same or a different application via intents.

4.1.2 XML Layouts

The user interface is defined using a hierarchy of View and ViewGroup objects. But it's not always necessary to use Views and View groups as in this application. Android provides several standard layouts where you just have to define the content. Android defines the layout with an XML file. This is a part of the application framework Figure 4.1.The XML layouts developed for this application include:

4.1.2.1 Login.xml, Register.xml

This login file corresponds to the login screen where the student login details are required to login to the application and similarly the register file where the layout for the student details are defined.

4.1.2.2 Main.xml

This is the layout file provides details for the student to proceed to the quiz.

4.1.2.3 GetQuestion.xml: Displays the question with options following ListView in linear layout format.

4.1.2.4 Answer.xml: Corresponds to the screen to submit the answer using TextView in linear layout.

4.1.2.5 AnswerValidation.xml: Again this screen corresponds to TextView in Linear layout for the validation of answers.

4.1.2.6 Pairing.xml: Corresponds to the screen using ListView in Linear Layout to display pairs.

4.1.3 Activity

Activities interact with the user, so the Activity class takes care of creating a window in which you can place your UI. [7](Google, 2014) All the java classes follow the same base functionality. This is a part of the application framework Figure 4.1.

4.1.3.1 onCreate()

To start with, the method that the subclass of an activity will implement is onCreate().This is used to initialize activity, declaring the user interface (defined in an XML layout file), defining member variables, and configuring some of the UI. Here the setContentView() is called with a layout resource defining the UI.

4.1.3.1 Intent

An intent is an abstract description of an operation to be performed. An Intent provides a facility for performing late runtime binding between the code in different applications. It is used in the launching of activities, where it can be thought of as the binding between activities. It is basically a passive data structure holding an abstract description of an action to be performed.

4.1.3.1 Async

In this application the activities are using AsyncTask because connecting php scripts may take some time, and also to avoid using same thread that runs the GUI the application. AsyncTask will create another thread to execute the tasks to be completed. PHP scripts are placed in the server-side. (explained in next section). It is defined by 3 steps called onPreExecute, doInBackground and onPostExecute. [8](Google, Developers Android, 2014) The AsyncTask of these java classes of this application implements:

- POST/GET that data to or from the URL that handles each functionality, that is, to POST/GET data to the corresponding script.

- Retrieve the JSON response from the server and interpret the response- POST/GET the information to/from the URL using HTTP request, and we parse the result using JSONParser.java class. What it mainly does is determine what type of method to use, either POST or GET, and then it makes httprequest and gets the response. In the JSONParser class will use a buffer reader to get the different elements of the JSON object.
- If it is successful continue. If not, display a status explaining why.

Figure 4.2 is a snippet of the AsynTask used in this application

```
class LoadAll extends AsyncTask<String, String, String> {
    /**Before starting background thread Show Progress Dialog* */
    @Override
    protected void onPreExecute() {
        super.onPreExecute();
        pDialog = new ProgressDialog(TeacherPaired.this);
        pDialog.setMessage("Does your Prof want pairing ???..");
        pDialog.setIndeterminate(false);
        pDialog.setCancelable(false);
        pDialog.show();
    }
    protected String doInBackground(String... args) {
        // Building Parameters
        List<NameValuePair> params = new ArrayList<NameValuePair>();
        // getting JSON string from URL
        JSONObject json = jParser.makeHttpRequest(url_correct, "GET",
                params);
        // Check your log cat for JSON reponse
        try {
            // Checking for SUCCESS TAG
            int success = json.getInt(TAG_SUCCESS);
            if (success == 1) {
                decision = json.getJSONArray(TAG_NODE);
                JSONObject c = decision.getJSONObject(0);
                yesOrno = c.getString(TAG_NAME);
            }
            else {
                message = json.getString(TAG_MESSAGE);
            }
        } catch (JSONException e) {
            e.printStackTrace();
        }
        return null;
    }

    /**
     * After completing background task Dismiss the progress dialog
     * **/
    protected void onPostExecute(String file_url) {
        pDialog.dismiss();
```

Figure 4.2 Code Snippet-1 (AsyncTask)

In this application, based on the above description the following summarizes the important java classes used:

Register.java

Registers the user details into the database

Login.java

Logs into the application and validates the user

Main.java

Gives the student user the option to start quiz or log out.

GetQuestion.java

Fetches the question data from the database via server

Answer.java

Student User submits the answer to the database

AnswerValidation.java

Validation of the answer and the correct answer.

InstuctorDecisionPair.java

Fetches the decision of the instructor either to pair or not

Pairing.java

The logic of pairing student users, that is, the student who answered wrong is paired with the student who answered correct. And if either is one more they are made into team of three else students who answered wrong are paired together and students who answered right are paired together.

UpdatePairing.java

This is used to update the database with the teams.

JSONParser.java: This is used to parse the JSON information. What it mainly does is determine what type of method to use, either POST or GET, and then it makes httprequest and gets the response. In the JSONParser class will use a buffer reader to get the different elements of the JSON object.

```
public class JSONParser {

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

    // constructor
    public JSONParser() {

    }

    // function get json from url
    // by making HTTP POST or GET mehtod
    public JSONObject makeHttpRequest(String url, String method,
            List<NameValuePair> params) {

        // Making HTTP request
        try {

            // check for request method
            if(method == "POST"){
                // request method is POST
                // defaultHttpClient
                DefaultHttpClient httpClient = new DefaultHttpClient();
                HttpPost httpPost = new HttpPost(url);
                httpPost.setEntity(new UrlEncodedFormEntity(params));

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

            }else if(method == "GET"){
                // request method is GET
                DefaultHttpClient httpClient = new DefaultHttpClient();
                String paramString = URLEncodedUtils.format(params, "utf-8");
                url += "?" + paramString;
                HttpGet httpGet = new HttpGet(url);

                HttpResponse httpResponse = httpClient.execute(httpGet);
                HttpEntity httpEntity = httpResponse.getEntity();
                is = httpEntity.getContent();
            }
```

Figure 4.3 Code Snippet-2 (JSON parser)

4.1.4 Android Manifest

To interact with the PHP scripts and our remote MySQL database, the Android application should have internet permission. All the activities in our application should also be mentioned in the Android Manifest.

```
  </application>
   <!-- Allow to connect with internet and to know the current network state-->
  <uses-permission android:name="android.permission.INTERNET" />
  <uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />
  <uses-permission android:name="ACCESS_WIFI_STATE" />
  <uses-permission android:name="android.permission.ACCESS_WIFI_STATE" />


/manifest>
```

Figure 4.4 Code Snippet-3 (Android Manifest)

4.2 Web Module

PHP scripts are being used to interact with the database. The information from the android which is the client-side is send to the PHP script to be processed, the script will interact with the database and information will be returned back to the application when required. The goal is to use Android app to post or get data to or from this URL and have it return a JSON for the android to parse.

The following PHP scripts are being used as follows:

4.2.1 dbConfig.php

Connects to the MySQL database.

4.2.2 Index.php

Instructor main page for posting question and answers

4.2.3 Insert.php

Updates database with the question and answer.

4.2.4 GetQuestions.php

GetQuestions.php- fetches MySQL data and converts it into JSON

4.2.5 StudentAnswer.php

Get student answer from client side and updates into database.

4.2.6 StudentAnswerInfo.php

Used to display and fetch the students with the right and wrong answer.

4.2.7 StudentRightAnswer.php

Converts the data of students who answered right into JSON.

4.2.8 StudentWrongAnswer.php

Converts the data of students who answered right into JSON.

4.2.9 WantPairing.php

Displays an option to the instructor if he wants to pair or not.

4.2.10 PairDecision.php

The option of the instructor is converted to JSON.

4.2.11 UpdatePairing.php

The teams are updated to the database from the client-side.

4.2.12 StudentAnswerInfoPair.php

Displays and fetched the results of answers after the teams were formed.

4.2.13 Graph.php

Google Chart API is used to display the graph of student results. [9](Google, Google Charts, 2014)

Chapter 5 - Testing

5.1 Compatibility Testing:

The application was tested with different phones and tablets and the configuration works well and ensures uniformity.

5.2 Unit Testing:

The following test cases were used for unit testing:

Test No	Required Result	Result Got	Result
1	User should be authenticated: 1)Invalid User 2)Invalid password	Invalid user cannot log into the application without registering	Pass
2	User cannot answer more than one question for a session	If user tried to enter answer for the second time for the same question, he is not allowed	Pass
3	Navigation of activities in order	The navigation of activities are in order without crashing	Pass
4	Pairing of Student Users with odd number of	One 3 person team resulted with odd number of	Pass

	students: There should be one 3 person team	students who answered.	
5	Pairing of Student Users with even number of students: No three person team	When even number of students answered only teams containing two members were formed.	Pass
6	Pairing should be with priority to - student who answered right with students who answered wrong	Pairing in this order was achieved and tested for all inputs.	Pass

Table 5.1 Unit Testing

5.3 Performance Testing Using JMeter:

The application was tested by pairing 50 users and the response time taken to retrieve the pairs. Performance is tested using JMeter. Below is the graph showing the response time with respect to the number of users who answered the questions and were paired up.

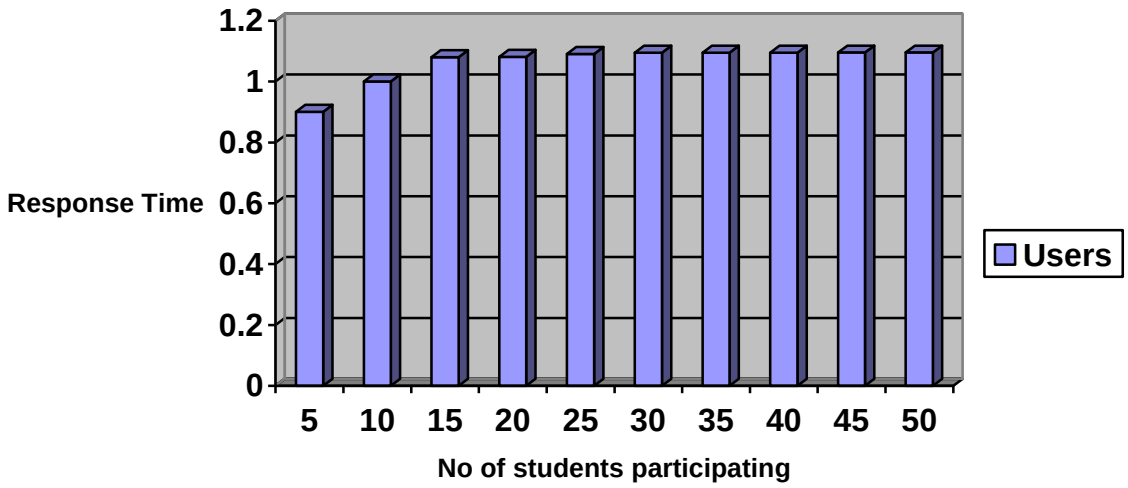

Figure 5.2 Performance Test Results

5.3.1 Performance Analysis

From the performance testing and Figure 5.1 it we can see that the response time of the activity increases minutely with the number of users, this is because of the time taken to retrieve the data and pairing of the students. But the overall performance is less than 1.2 sec, so it can be said that the overall performance of the application is good.

Chapter 6 - Future Work

With this application there would exist a fun way for Professors to analyze immediately the level in which each student has understood in class, and also students are alert and allowing them to instantly assess what they have learned in class, also implementing the concept of TTYP thereby enhancing the learning process by communicating with fellow students.

This system has the potential to extend and enhance the existing System improving the traditional Classroom Response System.

The Future work could include:

- The system to have a feature for the Professor to give immediate comment/review to students based on their answers.
- Categorizing the questions based on topic or courses. Giving the instructor an option to have previously asked questions based on courses.
- A timer system maintained by the professor to allot time based on the difficulty level of the question.
- Expanded to other Operating Systems

Chapter 7 - Conclusion

This application has a large scope of expanding where it can make a significant difference to the classroom teaching and learning compared to the traditional classroom response systems.

This project is limited to the scope of Android and with huge support for android communities online developing this application was easier. This application was not just about have a good user interface but also about having a robust backend system and understanding of the client server architecture between android and web services.

Lastly, this application provided the most important experience, for me as a developer to be a part of all the important phases of software development life cycle from requirement gathering to testing.

Chapter 8 - Bibliography

[1]Caldwell, J. E. (2007). *Clickers in the Large Classroom: Current Research and Best-Practice Tips.* CBE-Life Sciences Education.

[2]Keng Siau, H. S.-H. (2006). *Use of a Classroom Response System to Enhance Classroom Interactivity.* IEEE.

[3]Noreen M. Webb*, M. L. (2009). *Explain to your partner': teachers' instructional practices and students' dialogue in small groups.* Cambridge Journal of Education.

[4] Wikipedia. (2014, March 26). *Classroom Performance System*. Retrieved from Wiki: http://en.wikipedia.org/wiki/Classroom_Performance_Systems

[5] Wikipedia. (2014, March 5). *MySQL*. Retrieved from Wikipedia: http://en.wikipedia.org/wiki/MySQL

[6] Brahler, S. (2010). *Analysis of the Android.* www.kit.edu.

[7] Google. (2014, April 1). *Android Developers*. Retrieved from developer.android: http://developer.android.com/reference/android/app/Activity.html

[8] Google. (2014, April 5). *Developers Android*. Retrieved from developers.android: http://developer.android.com/reference/android/os/AsyncTask.html

[9] Google. (2014, April 23). *Google Charts*. Retrieved from developer.google: https://developers.google.com/chart/

Appendix A-User Manual

The web interface or the instructor interface is the part where the instructor posts the questions.

A.1 Instructor Interface

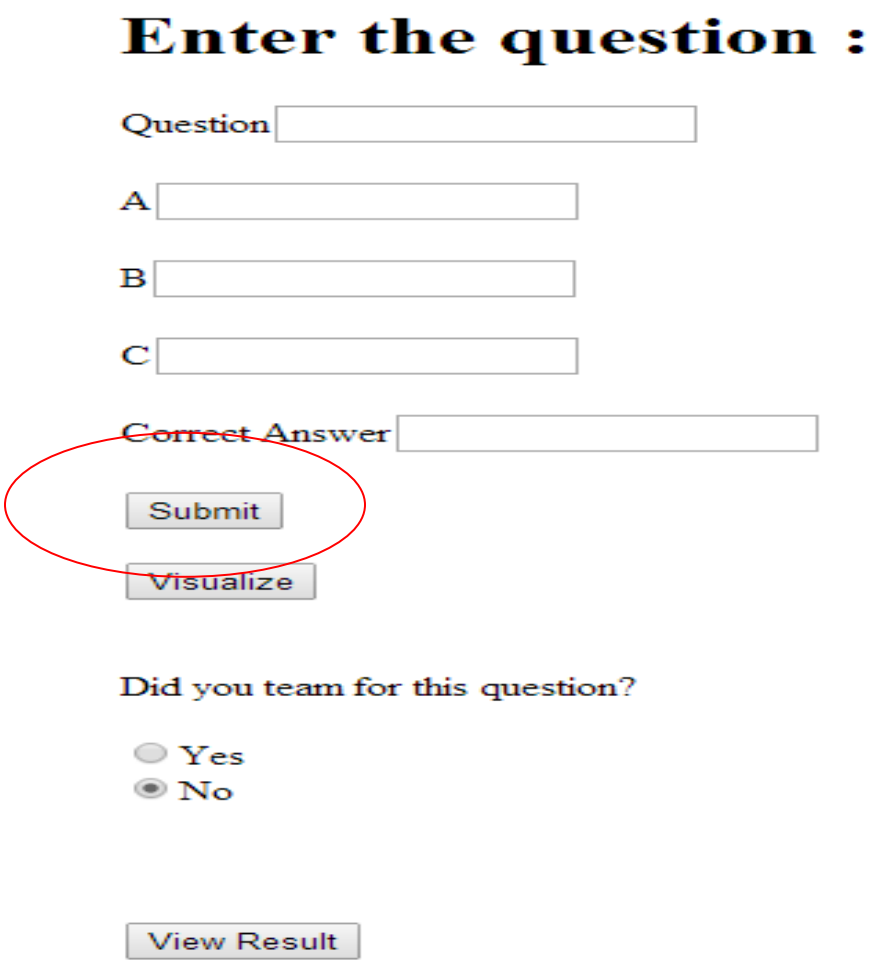

Figure A.1.1 Instructor question page

The instructor has to make sure that all the options are complete to proceed further, that is, before hitting submit

.The instructor has the ability to post, view answers and decide on teaming or not teaming the students for the next question.

From the Figure A.1.1 we can see that when the instructor hits the submit button. The questions are posted to the student interface that is the android device.

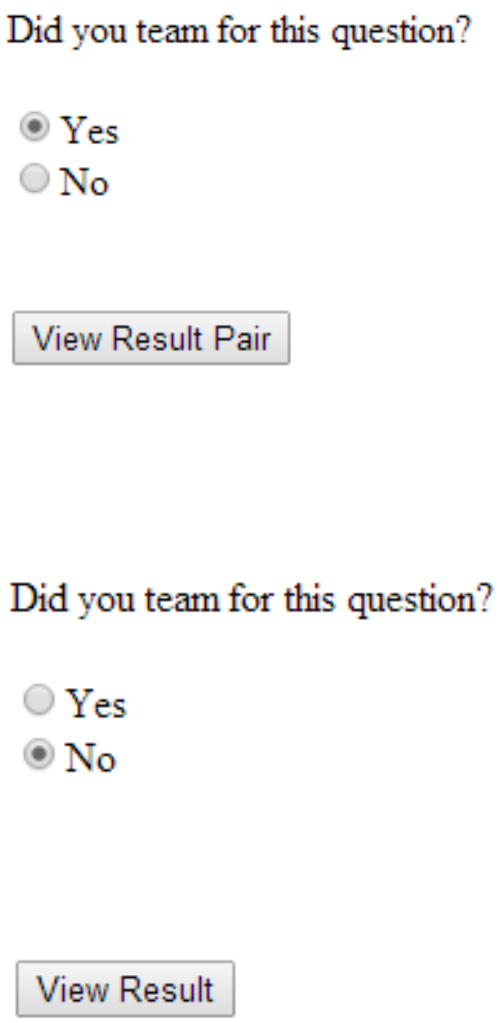

Figure A.1.2 Instructor student teaming details page

In order to view the results once the student submits their answer the instructor has to make a choice if he made a team or not for that question. The next section A.2 explain how the student logs into android application and eventually gets to view the questions posted by the professor.

The instructor also has the option to visualize the results on the click of visualize button, as shown below.

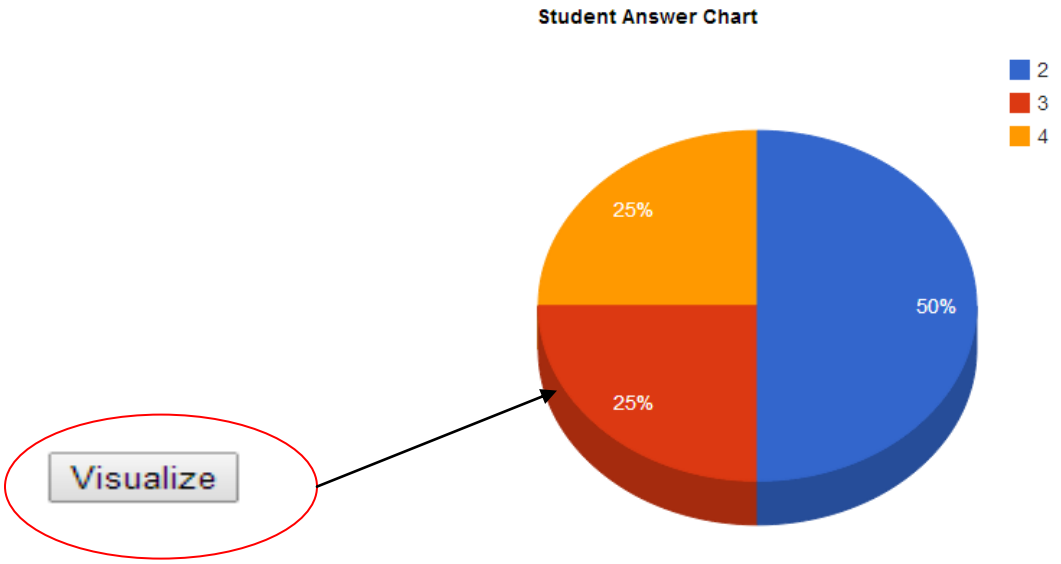

Figure A.1.3 Instructor student graph page

On hitting the "View Results" button the instructor can view the students who answered correct and the students who answered wrong also giving him an option if he wants to team the students or not for the next question. Eventually, allowing him to move on to the next question.

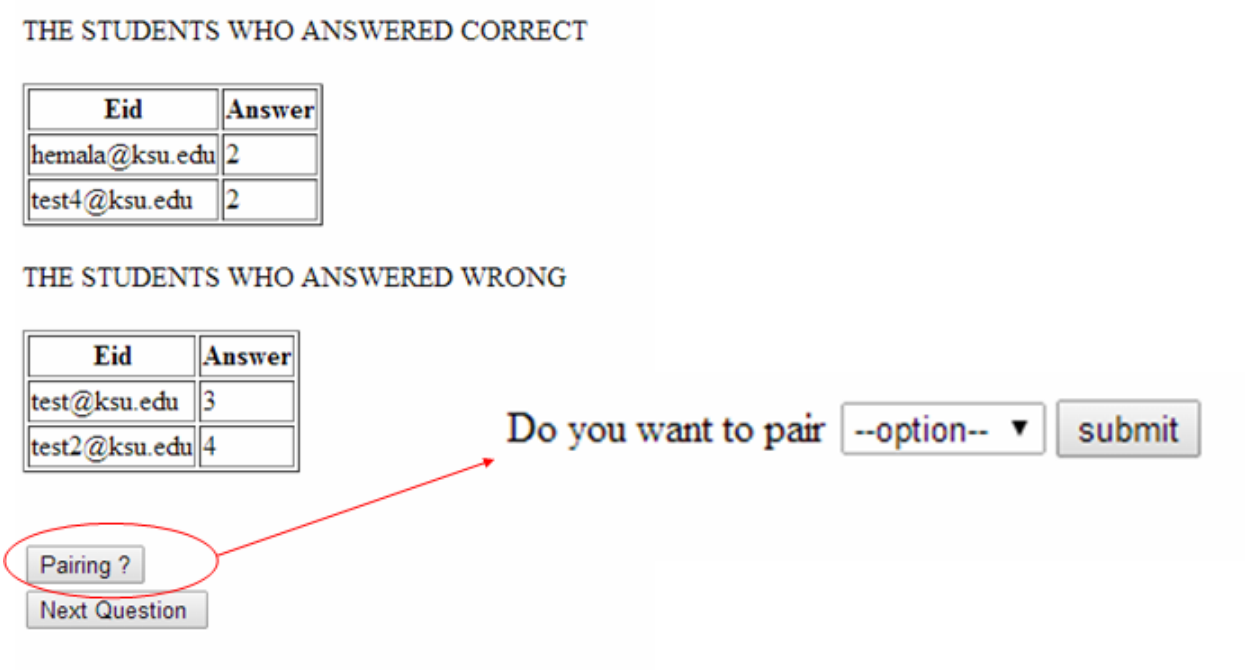

Figure A.1.4 Instructor teaming decision page

A.2 Student Interface

When the Student user clicks on the app icon as shown below

Figure A.2.1 Student Application icon screen

The Student user is then directed to the login screen:

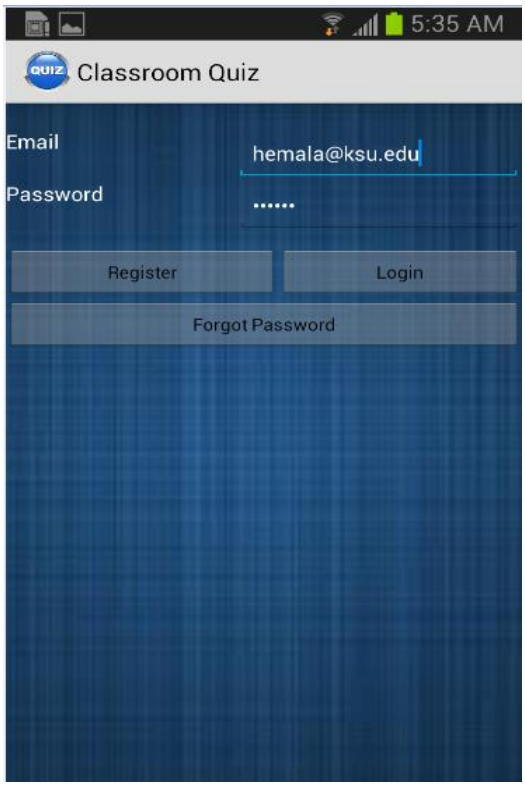

Figure A.2.2 Student Login screen

The Student logs in to the app through the login screen as shown in Figure A.2.2 where he enters his registered Email and password, before which it is mandatory for him to be a registered user.

Figure A.2.3 Student Register screen

To be validated user. The Student user has to enter his details like Firstname, Lastname, email, Username and Password as shown in Figure A.2.3, also he has to make sure that all the details are entered as it is mandatory to validate his credentials.

Figure A.2.4 Student reset password screen

If the user forgets his password he is redirected to forgot password screen as shown in Figure A.2.4. If the user enters his email address the reset password details and instructions to be followed to reset his password would be sent to his email.

Figure A.2.5 Student change password screen

Once the reset password instructions are sent the user has an option to change his password as desired.

Figure A.2.6 Student main start quiz screen

The Figure A.2.6 screen is the main welcome screen where the user hits on the quiz button he is redirected to the question with multiple choice answers.

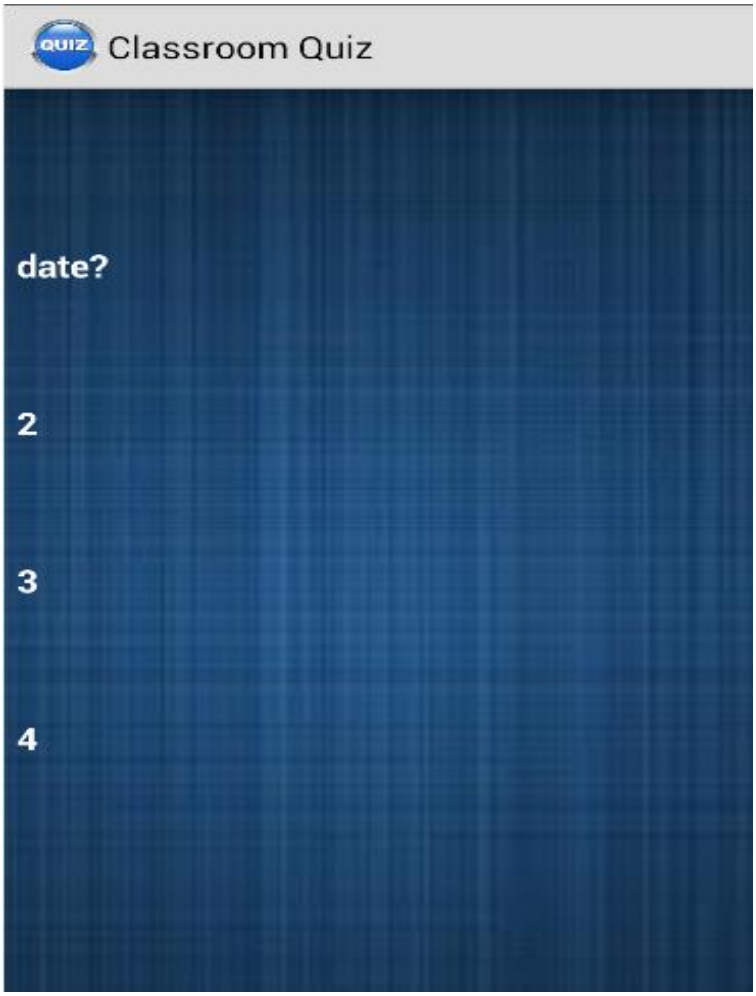

Figure A.2.7 Student question screen

The multiple choice questions are displayed as seen in Figure A.2.7 once the user choses his choice of answer. He is redirected to the screen where he confirms his answer.

Figure A.2.8 Student answer screen-1

Once the user sees his selected choice of answer and he is sure of his answer, he submits the answer.

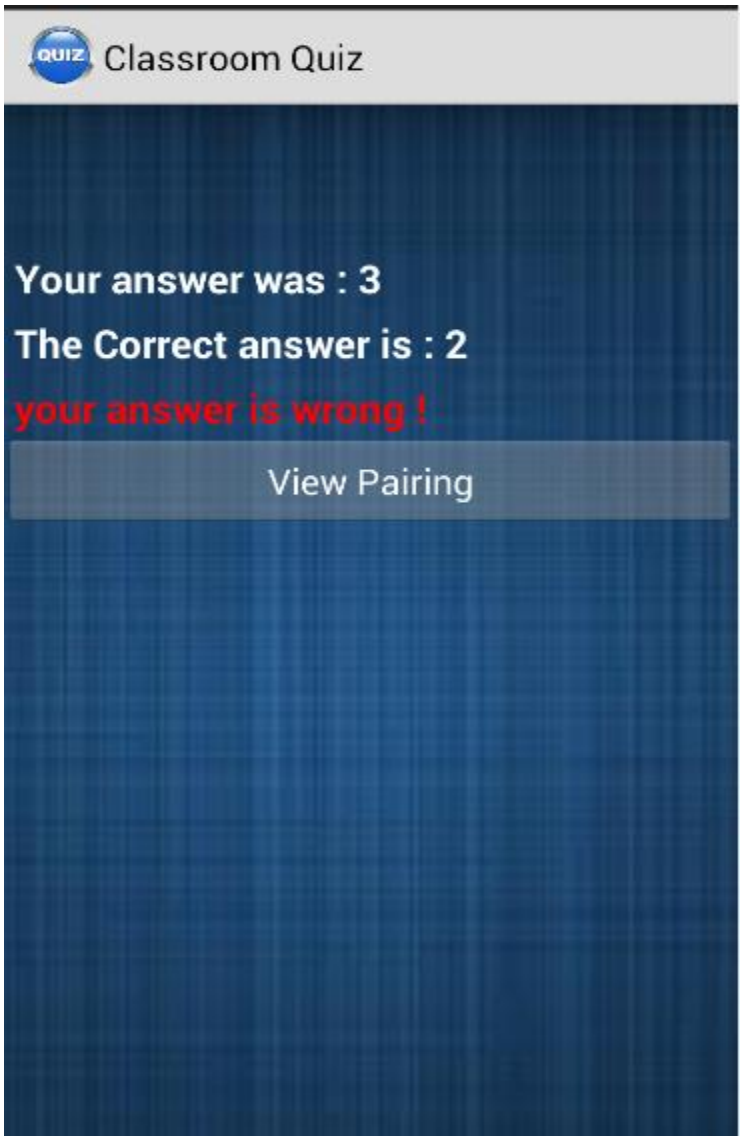

Figure A.2.9 Student answer screen-2

The Application then validates the answer and displays if the answers are correct or wrong.

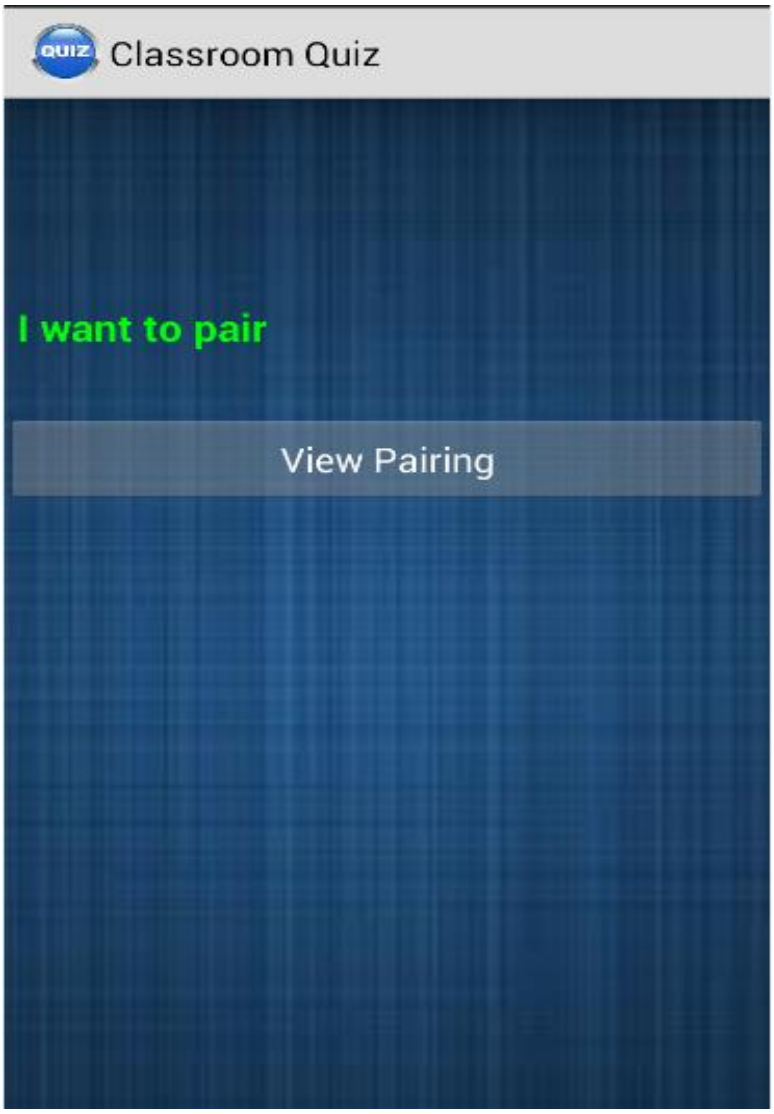

Figure A.2.10 Student teaming decision screen-1

The Figure A.2.10 and Figure A.2.11 displays on the screen based on the instructor's decision if he wants to team students or not .If the instructor decides to team the users then the screen like in Figure A.2.10 would be displayed

Figure A.2.11 Student teaming decision screen-2

If the instructor decides not to team the students then the Figure A.2.11 screen would be displayed

Figure A.2.12 Student team information screen

If the instructor decides to team the students the Figure A.2.12 screen would be displayed containing the teams for the next question.

A.3 How to Set Up?

Client-Side

- Android: Import all files of the android project into the Android Developers Tool (ADT).This application was tested and run with SDK version 9.0 and higher.
- Make sure that the debugger option in the Android device being tested is turned on while debugging the application on the device.
- Or to only deploy the application on the device download and open the .apk file of the project on the device.

Server-Side

- PHP: Import all the PHP files into the server to the path of your directory. In this project CIS department LINUX server was used.

A.4 Pointers to migrate to iOS

As a part of future work migrating this application to iOS could be seen as priority as it is the most popular mobile operating system used. Following are a few hints:

The basic requirements would be:

- A Mac computer running OS X 10.8 (Mountain Lion) or later
- Xcode
- iOS SDK

Apart from that Tools like Java2ObjC could migrate Java code to Objective C. But these tools do not migrate UI, it is only for logic. It means that all UI related code has to be written manually. MySQL C Client Library should be linked to the iOS application. Then the application will be able to connect to the MySQL database and issue SQL directly to it. MySQL server code would remain the same. Apart from that the rest would remain the same as Android.