**CHAPTER 1**

**INTRODUCTION**

**1.1 OVERVIEW OF THE PROJECT**

This project is aimed at developing an android application to connect the students to an institution. The system is an online application that can be accessed throughout the organization and outside as well with proper login provided. This system can be used as an application for the students to share and gain knowledge through forums, test their knowledge through aptitude tests, calculate their GPA using GPA calculator, and know about the events happening in the institution.

**1.1.1 Objective**

To develop a system that enables the students to interact with the institution and have knowledge about the happening events in the institution.

**1.2 EXISTING SYSTEM**

There are many problems in the existing system. The current system is not much interactive. It lacks user friendly interfaces and hard to use. And the current system doesn’t include all the necessary features.

**1.3 PROPOSED SYSTEM**

The motive of this android application is to enable connectivity between students and the institution. The students can interact through forums. They can also take up tests which will improve their knowledge. The students can be aware of all the events happening in the institution. They can also calculate their GPA, using GPA Calculator which does the calculations accurately.

**Advantages**

* It connects the students to the institution
* It is highly interactive user interface
* It is easy to use
* Integrated facilities

**1.4 REQUIREMENTS**

**1.4.1 MOBILE REQUIREMENTS**

**1.4.1.1 Software Requirements**

Android SDK Version 4.0

**1.4.1.2 Hardware Requirements**

Processor speed : 600 MHz

RAM : 512 MB

**1.4.2 SERVER REQUIREMENTS**

**1.4.2.1 Software Requirements**

Apache Tomcat

**1.4.2.2 Hardware Requirements**

Processor : Intel I3 Processor

RAM : 2 GB RAM

Hard disk Space : 100GB

**1.4.3 DEVELOPMENT ENVIRONMENT**

**1.4.3.1 Software Requirements**

Eclipse Indigo

Android Development Tools (ADT) Plugin for the Eclipse IDE

MySQL Workbench

**1.4.3.2 Hardware Requirements**

Processor : Intel I3 Processor

RAM : 2 GB RAM

Hard disk Space : 100GB

**CHAPTER 2**

**DETAILED DESCRIPTION**

**2.1 MODULES**

Buzzone Android application includes the following modules.

**2.1.1 User Login**

This module contains Login form where the user can enter the username and password to authenticate the user from database whether the user is registered user or not in order to access this system.

**2.1.2 Registration**

This module contains the Registration form where the new user can enter details such as Register number, First name, Last name, Department, Email Address and password. The registered user alone can access the system.

**2.1.3 Forums**

This module allows the students to actively participate in the discussions and allows them to express their ideas by posting them and also gain knowledge by reading the posts by other members.

**2.1.4 Events**

This module displays all the happening and upcoming events in the institution with correct date and time. The event will be cleared off if it is done.

**2.1.5 Test your App**

This module helps to test your knowledge by answering the questions.

**2.1.6 Suggestion box**

This module enables the students to post their valuable suggestions without any hesitation. These suggestions will be visible only to the admin.

**2.1.7 GPA Calculator**

This module helps the students to calculate their GPA easily.

**2.2 SOFTWARE DESCRIPTION**

**2.2.1 Java**

Java is a general-purpose object-oriented programming language developed by Sun Microsystems of USA in 1991. Originally called Oak by James Gosling, one of the inventors of the language, Java was designed for the development of software for consumer electronic devices like TVs, VCRs, toasters and such other electronic machines. This goal had a string impact on the development team to make the language simple, portable and highly reliable. The Java team, which included Patrick Naughton, discovered that the existing languages like C and C++ had limitations in terms of both reliability and portability. However, they modeled their new language Java on C and C++ but removed a number of features of C and C++ that were considered as sources of problems and thus made Java a really, simple, reliable, portable and powerful language.

**Java Features**

The inventors of Java wanted to design a language, which could offer solution to some of the problems encountered in modern programming. They wanted the language not only to be reliable, portable and distributed but also simple, compact and interactive.

Sun Microsystems officially describes Java with following attributes.

* Platform-Independent and Portable
* Object-Oriented
* Robust and Secure
* Distributed
* Familiar, Simple and Small
* Multithreaded and Interactive
* High Performance

**2.2.2 Android**

Android is a Linux based operating system for mobile device such as smart phones and tablet computer. Android is a software stack for mobile devices that includes an operating system, middleware and key applications. The Android SDK provides the tools and APIs necessary to begin developing applications on the Android platform using the Java programming language.

**Features**

Application framework enabling reuse and replacement of components

* Dalvik virtual machine optimized for mobile devices
* Integrated browser based on the open source WebKit engine.
* Optimized graphics powered by a custom 2D graphics library; 3D graphics based on the OpenGL ES 1.0 specification (hardware acceleration optional)
* SQLite for structured data storage
* Media Support for common audio, video, and still image formats (MPEG$, H.264, MP3, AAC, AMR, JPG, PNG, GIF)
* GSM Telephony (hardware dependent)
* Bluetooth, EDGE, 3G and WiFi (hardware dependent)
* Camera, GPS, compass and accelerometer (hardware dependent)
* Rich developmental environment including a device emulator, tools for debugging, memory and performance profiling, and a plugin for the Eclipse IDE.

**Applications**

Android will ship with a set of core applications including an email client SMS program, calendar, maps, browser, contacts, and others. All applications are written using the Java programming language.

**Application Framework**

By providing an open developmental platform, Android offers developers the developers the ability to build extremely rich and innovative applications. Developers are free to take advantage of the device hardware, access location information, run background services, set alarms, add notifications to the status bar, and much more.

Developers have full access to the same framework APIs used by the core applications. The application architecture is designed to simplify the reuse of components. Any application can publish its capabilities and any other application may then make use of those capabilities (subject to security constraints enforced by the framework). This same mechanism allows components to be replaced by the user.

Underlying all applications is a set of services and systems, including :

* A rich and extensible set of Views that can be used to build an application, including lists, grids, text boxes, buttons, and even an embeddable web browser.
* Content Providers that enable applications to access data from other applications (such as Contacts), or to share their own data.
* A Resource Manager, providing access to non-code resources such as localized strings, graphics, and layout files.
* A Notification Manager that enables all applications to display custom alerts in the status bar.
* An Activity Manager that manages the lifecycle of applications and provides a common navigation back stack.

**Libraries**

Android includes a set of C/C++ libraries used by various components of the Android system. These capabilities are exposed to developers through the Android application framework. Some of the core libraries are listed below:

* System C library – a BSD-derived implementation of the standard C system library (libs), tuned for embedded Linux-based devices.
* Media Libraries – based on Packet Video’s Open CORE, the libraries support playback and recording of many popular audio and video formats, as well as static image files, including MPEG4, H.264, MP3, AAC, AMR, JPG and PNG.
* Surface Manager – manages access to the display subsystem and seamlessly composites 2D and 3D graphic layers from multiple applications.
* LibWebCore – a modern web browser engine which powers both the Android browser and an embeddable web view.
* SGL – the underlying 2D graphics engine.
* 3D libraries – an implementation based on OpenGL ES 1.0 APIs. The libraries either use hardware 3D acceleration (where available) or the included, highly optimized 3D software rasterizer
* FreeType – bitmap and vector font rendering.
* SQLite – a powerful and lightweight relational database engine available to all applications.

**Android Runtime**

Android includes a set of core libraries that provides most of the functionality available in the core libraries of the Java programming language. Every Android application runs in its own process, with its own instance of the Dalvik virtual machine. Dalvik VM executes files in the Dalvik Executable (.dex) format which is optimized for minimal memory footprint.

**2.2.3 MySQL**

MySQL is a [relational database management system](http://en.wikipedia.org/wiki/Relational_database_management_system) (RDBMS), and ships with no [GUI](http://en.wikipedia.org/wiki/Graphical_user_interface) tools to administer MySQL databases or manage data contained within the databases. Users may use the included [command line](http://en.wikipedia.org/wiki/Command_line) tools, or use MySQL "front-ends", desktop software and web applications that create and manage MySQL databases, build database structures, back up data, inspect status, and work with data records. The official set of MySQL front-end tools, [MySQL Workbench](http://en.wikipedia.org/wiki/MySQL_Workbench) is actively developed by Oracle, and is freely available for use.

The official [MySQL Workbench](http://en.wikipedia.org/wiki/MySQL_Workbench) is a free integrated environment developed by MySQL AB, that enables users to graphically administer MySQL databases and visually design database structures. MySQL Workbench replaces the previous package of software, [MySQL GUI Tools](http://en.wikipedia.org/wiki/MySQL_GUI_Tools). Similar to other third-party packages, but still considered the authoritative MySQL front end, MySQL Workbench lets users manage database design & modeling, SQL development (replacing MySQL Query Browser) and Database administration (replacing MySQL Administrator).

MySQL Workbench is available in two editions, the regular [free and open source](http://en.wikipedia.org/wiki/Free_and_open_source_software) *Community Edition* which may be downloaded from the MySQL website, and the proprietary *Standard Edition* which extends and improves the feature set of the Community Edition.

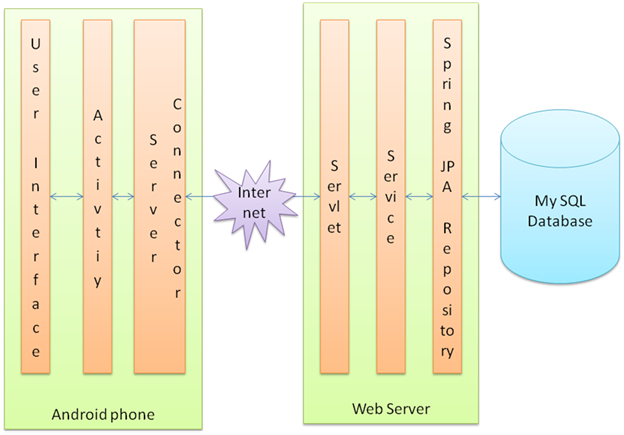
**2.2.4 APACHE TOMCAT SERVER**

**Apache Tomcat** (or simply **Tomcat**, formerly also *Jakarta Tomcat*) is an [open source](http://en.wikipedia.org/wiki/Open_source) [web server](http://en.wikipedia.org/wiki/Web_server) and [servlet](http://en.wikipedia.org/wiki/Java_Servlet) [container](http://en.wikipedia.org/wiki/Web_container) developed by the [Apache Software Foundation](http://en.wikipedia.org/wiki/Apache_Software_Foundation) (ASF). Tomcat implements the [Java Servlet](http://en.wikipedia.org/wiki/Java_Servlet) and the [Java Server Pages](http://en.wikipedia.org/wiki/JavaServer_Pages) (JSP) specifications from [Sun Microsystems](http://en.wikipedia.org/wiki/Sun_Microsystems), and provides a "pure [Java](http://en.wikipedia.org/wiki/Java_(programming_language))" [HTTP](http://en.wikipedia.org/wiki/Hypertext_Transfer_Protocol) [web server](http://en.wikipedia.org/wiki/Web_server) environment for [Java](http://en.wikipedia.org/wiki/Java_(programming_language)) code to run.

**CHAPTER 3**

**DETAILED DESIGN**

**3.1 OVERALL ARCHITECTURE**

****

***Fig 3.1 Overall Architecture Diagram***

**3.2 ANDROID ARCHITECTURE**



***Fig 3.2 Android Architecture Diagram***

**3.3 USECASE DIAGRAM**

****

***Fig 3.3 UseCase Diagram***

**3.4 SEQUENCE DIAGRAMS**

1. **Registration & Login**

****

***Fig 3.4 Sequence Diagram – Registration and Login***

1. **Forums**

****

***Fig3.5 Sequence Diagram - Forums***

1. **Events**

****

***Fig3.6 Sequence Diagram - Events***

1. **Test your Apps**

****

***Fig3.7 Sequence Diagram – Test your Apps***

1. **Suggestion Box**

****

***Fig3.8 Sequence Diagram – Suggestion Box***

1. **GPA calculator**

****

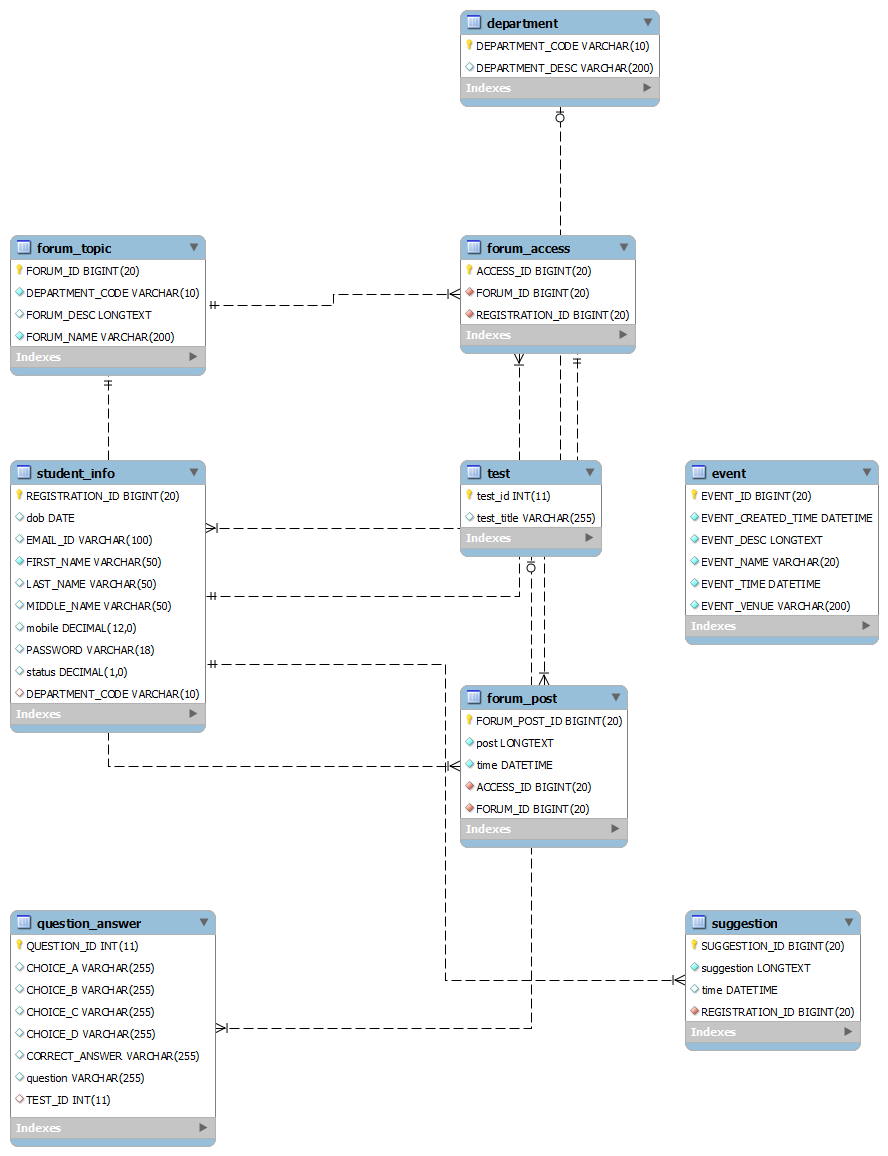
***Fig3.9 Sequence Diagram – GPA Calculator***

**3.5 CLASS DIAGRAM**

****

***Fig3.10 Class Diagram***

**3.6 EER DIAGRAM**

****

***Fig 3.11 EER Diagram***

**CHAPTER 4**

**IMPLEMENTATION AND TEST PLAN**

**4.1 FEASIBILITY STUDY**

The system feasibility can be divided into the following sections.

**4.1.1 Economic Feasibility**

The project is economically feasible as the only cost involved is having a Mobile phone or a Tablet computer with the minimum requirements mentioned earlier. For the users to access the application, the only cost involved will be in getting access to the Internet.

**4.1.3 Technical Feasibility**

To deploy the application, the only technical aspects needed are mentioned below.

Operating Environment : Windows

Platform : Android

Database : MySQL

For users : Internet connection

**4.1.4 Behavioural Feasibility**

The application requires no special technical guidance and all the views available in the application are self explanatory. The users are well guided with warning and failure messages for all the actions taken.

**4.2 TEST PLAN**

**TESTING**

Software testing is an investigation conducted to provide stakeholders with information about the quality of the product or service under test. Software testing also provides an objective, independent view of the software to allow the business to appreciate and understand the risks of software implementation. Test techniques include, but are not limited to, the process of executing a program or application with the intent of finding software bugs.

Software testing can also be stated as the process of validating and verifying that a software program/application/product:

* Meets the business and technical requirements that guided its design and development;
* Works as expected; and
* Can be implemented with the same characteristics.

The project is tested to verify its correctness and efficiency. The test plan includes following test cases.

|  |  |  |  |
| --- | --- | --- | --- |
| **TEST CASE** | **EXPECTED OUTPUT** | **ACTUAL OUTPUT** | **PASS/FAIL** |
| Login after Registration  (before Admin’s approval) | Display the message : “You have not been approved! Contact Admin”. | Display the message : “You have not been approved! Contact Admin”. | PASS |
| Check if the fields are empty | Display the message : “The field cannot be empty” | Display the message : “The field cannot be empty” | PASS |
| Password Validation | Show error if “confirm password” field doesn’t match the “password” field. | Show error if “confirm password” field doesn’t match the “password” field. | PASS |
| Logout  (Security) | The Login page should be displayed after pressing Logout, and pressing back button should not the display the homepage. | The Login page should be displayed after pressing Logout, and pressing back button should not the display the homepage. | PASS |
| Taking up tests | Pressing back button should not display the previously attempted questions. | Pressing back button should not display the previously attempted  questions. | PASS |

***Table 4.1 Test Cases***

**CHAPTER 5**

**CONCLUSION AND FUTURE WORK**

**5.1 CONCLUSION**

* In this application most of the activities are done quickly without any delay.
* Proper attention has been given to make this application user friendly for its operator.
* This application is so simple that even a person who has a little knowledge of technology can handle with it easily.

**5.2 FUTURE SCOPE**

This application can be modified from time to time as per the changing requirement of the user with lesser cost. Also the backend of the application can be changed as per the storage requirement of the application and to provide more security level features.

The application can be modified to provide mailing features to the users to contact the admin and other managers of the application through E-mail.

**APPENDIX A**

**SAMPLE CODING**

**CLIENT SIDE**

**LOGIN PAGE**

package com.saveetha.buzzone.buzzoneapp;

import java.util.ArrayList;

import java.util.List;

import android.app.Activity;

import android.app.AlertDialog;

import android.app.ProgressDialog;

import android.content.DialogInterface;

import android.content.Intent;

import android.os.Bundle;

import android.util.Log;

import android.view.Menu;

import android.view.View;

import android.widget.Button;

import android.widget.EditText;

import android.widget.ProgressBar;

import android.widget.TextView;

import com.saveetha.buzzone.buzzoneapp.common.ServerConnector;

import com.saveetha.buzzone.vo.RequestObject;

import com.saveetha.buzzone.vo.ResponseObject;

import com.saveetha.buzzone.vo.StudentInfo;

public class MainActivity extends Activity {

EditText registerNumber = null;

EditText passwordText = null;

ProgressBar progressBar = null;

TextView statusMessage = null;

private String registerNumberValue;

private String passwordTextValue;

ProgressDialog progDialog = null;

Activity thisActivity = null;

public static StudentInfo loggedinStudent = null;

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_main);

registerNumber = (EditText) findViewById(R.id.register\_no);

passwordText = (EditText) findViewById(R.id.password\_text);

statusMessage = (TextView) findViewById(R.id.status\_message);

Button loginButton = (Button) findViewById(R.id.login\_button);

thisActivity = this;

loginButton.setOnClickListener(new View.OnClickListener() {

public void onClick(View arg0) {

Log.d("MainActivity", "Entered On Click Listener");

Runnable r = new Runnable() {

public void run() {

Log.d("MainActivity", "Entered run");

attemptLogin();

}};

r.run();}});

Button registerButton = (Button) findViewById(R.id.new\_register);

registerButton.setOnClickListener(new View.OnClickListener() {

public void onClick(View v) {

Intent intent = new Intent(MainActivity.this, RegistrationActivity.class);

MainActivity.this.startActivity(intent);

}

});}

protected void attemptLogin() {

registerNumberValue = registerNumber.getText().toString();

passwordTextValue = passwordText.getText().toString();

if ("".equals(passwordTextValue) || registerNumberValue.equals("")) {

AlertDialog.Builder builder = new AlertDialog.Builder(this);

builder.setMessage("Username/Password fields can not be empty")

.setTitle("Login Failed!");

builder.setPositiveButton("ok", null);

AlertDialog dialog = builder.create();

dialog.show();

} else {

RequestObject requestObject = new RequestObject();

requestObject.setRequestType("LOGIN");

StudentInfo studentInfo = new StudentInfo(); studentInfo.setRegistrationId(Long.valueOf(registerNumberValue));

studentInfo.setPassword(passwordTextValue);

Log.d("MainActivity", "atempt Login");

List requestObjectList = new ArrayList();

requestObject.setRequestObjects(requestObjectList);

requestObjectList.add(studentInfo);

ResponseObject responseObject = ServerConnector

.connectToServer(requestObject);

Log.d("MainActivity","AFter connect" + responseObject.getResponseMessage());

if (responseObject.getResponseCode() == 0) {

Log.d("MainActivity","Received success" + responseObject.getResponseMessage());

loggedinStudent = (StudentInfo) responseObject.getResponseObject().get(0);

Intent intent = new Intent(MainActivity.this,

HomepageActivity.class);

MainActivity.this.startActivity(intent);

} else {

AlertDialog.Builder builder = new AlertDialog.Builder(this); builder.setMessage(responseObject.getResponseMessage())

.setTitle("Login Failed!");

builder.setPositiveButton("ok", new DialogInterface.OnClickListener() {

public void onClick(DialogInterface dialog, int id) {

}});

AlertDialog dialog = builder.create();

dialog.show();

registerNumber.setText("");

passwordText.setText("");

}

}

}

public boolean onCreateOptionsMenu(Menu menu) {

getMenuInflater().inflate(R.menu.activity\_main, menu);

return true;

}

}

**HOMEPAGE**

package com.saveetha.buzzone.buzzoneapp;

import android.app.Activity;

import android.app.AlarmManager;

import android.app.PendingIntent;

import android.content.Intent;

import android.os.Bundle;

import android.os.SystemClock;

import android.view.KeyEvent;

import android.view.Menu;

import android.view.View;

import android.widget.Button;

import android.widget.ImageButton;

public class HomepageActivity extends Activity {

ImageButton forumButton = null;

ImageButton eventButton = null;

ImageButton suggestionButton = null;

ImageButton fastFingersButton = null;

Button logoutButton=null;

public boolean onCreateOptionsMenu(Menu menu) {

getMenuInflater().inflate(R.menu.activity\_homepage, menu);

return true;

}

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.home\_page\_final);

forumButton = (ImageButton) findViewById(R.id.colloquiumid);

forumButton.setOnClickListener(new View.OnClickListener() {

public void onClick(View arg0) {

Intent intent = new Intent(HomepageActivity.this,

ForumActivity.class);

HomepageActivity.this.startActivity(intent);

}

});

suggestionButton =(ImageButton) findViewById(R.id.suggestionid);

suggestionButton.setOnClickListener(new View.OnClickListener() {

public void onClick(View v) {

Intent intent= new Intent(HomepageActivity.this,SuggestionActivity.class);

HomepageActivity.this.startActivity(intent);

}

});

eventButton = (ImageButton) findViewById(R.id.eventsid);

eventButton.setOnClickListener(new View.OnClickListener() {

public void onClick(View v) {

Intent intent= new Intent(HomepageActivity.this,NotificationActivity.class);

HomepageActivity.this.startActivity(intent);

}

});

fastFingersButton = (ImageButton) findViewById(R.id.fast\_finger\_id);

fastFingersButton.setOnClickListener(new View.OnClickListener() {

public void onClick(View v) {

Intent intent= new Intent(HomepageActivity.this,QuestionAnswerActivity.class);

HomepageActivity.this.startActivity(intent);

}

});

logoutButton=(Button)findViewById(R.id.logout\_button);

logoutButton.setOnClickListener(new View.OnClickListener() {

public void onClick(View arg0) {

Intent intent = new Intent(HomepageActivity.this,

MainActivity.class);

HomepageActivity.this.startActivity(intent);

}});

}

public boolean onKeyDown(int keyCode, KeyEvent event) {

if (keyCode == KeyEvent.KEYCODE\_BACK && event.getRepeatCount() == 0) {

moveTaskToBack (true);

}

return super.onKeyDown(keyCode, event);

}}

**QUESTIONANSWER ACTIVITY**

package com.saveetha.buzzone.buzzoneapp;

import java.util.ArrayList;

import java.util.List;

import com.saveetha.buzzone.buzzoneapp.common.ServerConnector;

import com.saveetha.buzzone.vo.ForumTopic;

import com.saveetha.buzzone.vo.QuestionAnswer;

import com.saveetha.buzzone.vo.RequestObject;

import com.saveetha.buzzone.vo.ResponseObject;

import com.saveetha.buzzone.vo.Test;

import android.os.Bundle;

import android.app.Activity;

import android.content.Intent;

import android.graphics.Color;

import android.graphics.Paint;

import android.graphics.Paint.Style;

import android.graphics.drawable.Drawable;

import android.graphics.drawable.ShapeDrawable;

import android.graphics.drawable.shapes.RectShape;

import android.view.Menu;

import android.view.View;

import android.view.ViewGroup.LayoutParams;

import android.widget.LinearLayout;

import android.widget.TextView;

public class QuestionAnswerActivity extends Activity {

LinearLayout linearLayout = null;

private static List<Test> tests = null;

public static Test selectedTest = null;

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_question\_answer);

linearLayout = (LinearLayout) findViewById(R.id.testScrollId);

RequestObject requestObject = new RequestObject();

requestObject.setRequestType("GET\_TEST");

ResponseObject responseObject = ServerConnector.connectToServer(requestObject);

if(responseObject.getResponseCode() == 0)

{

tests = (List<Test>) responseObject.getResponseObject().get(0);

for (Test test : tests) {

TextView textView = new TextView(getApplicationContext());

textView.setText(test.getTestTitle());

textView.setTextSize(18);

textView.setTextColor(Color.parseColor("#00ff00"));

textView.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

for (Test test : tests) {

if(String.valueOf(((TextView)v).getText()).contains(test.getTestTitle()))

{

selectedTest = test;

break;

} }

Intent intent = new Intent(QuestionAnswerActivity.this,

QuestionActivity.class); QuestionAnswerActivity.this.startActivity(intent);

}

});

linearLayout.addView(textView);

View breakLine = new View(getApplicationContext());

breakLine.setLayoutParams(new LayoutParams(

LayoutParams.MATCH\_PARENT,

5));

breakLine.setBackgroundDrawable(createLine(Color.parseColor("#0000ff"), 30));

linearLayout.addView(breakLine);

}

}

}

public static Drawable createLine(int color, int height)

{

ShapeDrawable sd = new ShapeDrawable(new RectShape());

sd.setIntrinsicHeight(height);

Paint fgPaintSel = sd.getPaint();

fgPaintSel.setColor(color);

fgPaintSel.setStyle(Style.FILL);

fgPaintSel.setStrokeWidth(height);

return sd;

}

public boolean onCreateOptionsMenu(Menu menu) {

getMenuInflater().inflate(R.menu.activity\_question\_answer, menu);

return true;

}}

**QUESTION ACTIVITY**

package com.saveetha.buzzone.buzzoneapp;

import java.util.ArrayList;

import java.util.List;

import android.app.Activity;

import android.app.AlertDialog;

import android.content.Intent;

import android.os.Bundle;

import android.view.Menu;

import android.view.View;

import android.widget.Button;

import android.widget.RadioButton;

import android.widget.RadioGroup;

import android.widget.TextView;

import com.saveetha.buzzone.buzzoneapp.common.ServerConnector;

import com.saveetha.buzzone.vo.QuestionAnswer;

import com.saveetha.buzzone.vo.RequestObject;

import com.saveetha.buzzone.vo.ResponseObject;

public class QuestionActivity extends Activity {

TextView testTitle = null;

TextView questionText = null;

Button submitButton = null;

private int questionSize = 0;

RadioGroup radioGroup = null;

RadioButton radioButton1= null;

RadioButton radioButton2= null;

RadioButton radioButton3= null;

RadioButton radioButton4= null;

private int count = 0;

private int correctAnswer=0;

private List<QuestionAnswer> questionAnswers = null;

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_question);

testTitle = (TextView) findViewById(R.id.testTitleId); testTitle.setText(QuestionAnswerActivity.selectedTest.getTestTitle());

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

questionText = (TextView) findViewById(R.id.questionText);

radioGroup = (RadioGroup) findViewById(R.id.radioAnswer);

radioButton1=(RadioButton)findViewById(R.id.radio\_choice\_a);

radioButton2=(RadioButton)findViewById(R.id.radio\_choice\_b);

radioButton3=(RadioButton)findViewById(R.id.radio\_choice\_c);

radioButton4=(RadioButton)findViewById(R.id.radio\_choice\_d);

RequestObject requestObject = new RequestObject();

requestObject.setRequestType("GET\_QUESTION");

List requestObjectList = new ArrayList();

requestObject.setRequestObjects(requestObjectList);

requestObjectList.add(QuestionAnswerActivity.selectedTest);

ResponseObject responseObject = ServerConnector.connectToServer(requestObject);

if(responseObject.getResponseCode() == 0)

{

questionAnswers = (List<QuestionAnswer>) responseObject.getResponseObject().get(0);

questionSize = questionAnswers.size(); questionText.setText(questionAnswers.get(count).getQuestion()); radioButton1.setText(questionAnswers.get(count).getChoiceA()); radioButton2.setText(questionAnswers.get(count).getChoiceB()); radioButton3.setText(questionAnswers.get(count).getChoiceC()); radioButton4.setText(questionAnswers.get(count).getChoiceD());

}

submitButton.setOnClickListener(new View.OnClickListener() {

@Override

public void onClick(View v) {

RadioButton selectedButton = (RadioButton) findViewById(radioGroup.getCheckedRadioButtonId());

String choice=null;

if(R.id.radio\_choice\_a == selectedButton.getId())

{

choice = "a";

}

else if(R.id.radio\_choice\_b == selectedButton.getId())

{

choice ="b";

}

else if(R.id.radio\_choice\_c == selectedButton.getId())

{

choice="c";

}

else if(R.id.radio\_choice\_d == selectedButton.getId())

{

choice ="d";

}

QuestionAnswer questionAnswer = questionAnswers.get(count);

if(questionAnswer.getCorrectAnswer().equals(choice))

{

correctAnswer++;

AlertDialog.Builder builder = new AlertDialog.Builder(QuestionActivity.this);

builder.setMessage("Correct Answer!!")

.setTitle("Answer");

builder.setPositiveButton("ok", null);

AlertDialog dialog = builder.create();

dialog.show();

}

else

{

AlertDialog.Builder builder = new AlertDialog.Builder(QuestionActivity.this);

builder.setMessage("Your answer is wrong!, Correct Answer : " + questionAnswers.get(count).getCorrectAnswer())

.setTitle("Answer");

builder.setPositiveButton("ok", null);

AlertDialog dialog = builder.create();

dialog.show();

}

count ++;

if(count<questionSize) {

questionText.setText(questionAnswers.get(count).getQuestion()); radioButton1.setText(questionAnswers.get(count).getChoiceA()); radioButton2.setText(questionAnswers.get(count).getChoiceB()); radioButton3.setText(questionAnswers.get(count).getChoiceC()); radioButton4.setText(questionAnswers.get(count).getChoiceD());

radioGroup.clearCheck();

}

else if(count == questionSize)

{

Intent intent = new Intent(QuestionActivity.this,

TestEndActivity.class);

intent.putExtra("Score", correctAnswer);

intent.putExtra("Totalquestions",questionSize );

QuestionActivity.this.startActivity(intent);

}

}

});

}

public boolean onCreateOptionsMenu(Menu menu) {

// Inflate the menu; this adds items to the action bar if it is present.

getMenuInflater().inflate(R.menu.activity\_question, menu);

return true;

}

}

**SERVER SIDE**

**BuzzoneService.java**

package com.saveetha.buzzone.service;

import java.math.BigDecimal;

import java.util.ArrayList;

import java.util.Date;

import java.util.List;

import org.springframework.stereotype.Service;

import com.saveetha.buzzone.exception.BuzzoneException;

import com.saveetha.buzzone.repository.DepartmentRepository;

import com.saveetha.buzzone.repository.EventRepository;

import com.saveetha.buzzone.repository.ForumAccessRepository;

import com.saveetha.buzzone.repository.ForumPostRepository;

import com.saveetha.buzzone.repository.ForumTopicRepository;

import com.saveetha.buzzone.repository.LibraryBookRepository;

import com.saveetha.buzzone.repository.QuestionAnswerRepository;

import com.saveetha.buzzone.repository.StudentInfoRepository;

import com.saveetha.buzzone.repository.SuggestionRepository;

import com.saveetha.buzzone.repository.TestRepository;

import com.saveetha.buzzone.util.TransformerUtil;

import com.saveetha.buzzone.vo.Department;

import com.saveetha.buzzone.vo.Event;

import com.saveetha.buzzone.vo.ForumAccess;

import com.saveetha.buzzone.vo.ForumPost;

import com.saveetha.buzzone.vo.ForumTopic;

import com.saveetha.buzzone.vo.QuestionAnswer;

import com.saveetha.buzzone.vo.RequestObject;

import com.saveetha.buzzone.vo.ResponseObject;

import com.saveetha.buzzone.vo.StudentInfo;

import com.saveetha.buzzone.vo.Suggestion;

import com.saveetha.buzzone.vo.Test;

public class BuzzoneService {

private DepartmentRepository departmentRepository;

private EventRepository eventRepository;

private ForumAccessRepository forumAccessRepository;

private ForumPostRepository forumPostRepository;

private ForumTopicRepository forumTopicRepository;

private LibraryBookRepository libraryBookRepository;

private StudentInfoRepository studentInfoRepository;

private SuggestionRepository suggestionRepository;

private TestRepository testRepository;

private QuestionAnswerRepository questionAnswerRepository;

public ResponseObject createStudentInfo(RequestObject requestObject)

{

List<Object> requestObjects = requestObject.getRequestObjects();

StudentInfo studentInfo = (StudentInfo) requestObjects.get(0);

ResponseObject responseObject = new ResponseObject();

StudentInfo studentInfoInDB = this.studentInfoRepository

.findOne(studentInfo.getRegistrationId());

if (null != studentInfoInDB) {

if(studentInfoInDB.getStatus().longValue() != 1l )

{

responseObject.setResponseCode(-1);

responseObject.setResponseMessage("Already Registered ! Registration is not active Contact Admin");

}

else

{

responseObject.setResponseCode(-1);

responseObject.setResponseMessage(" Student already registered");

}

}

else{

studentInfo.setStatus(new BigDecimal("0"));

this.studentInfoRepository.save(studentInfo);

responseObject.setResponseCode(0);

responseObject.setResponseMessage("Thanks for registering ! "); }

return responseObject;

}

public ResponseObject validateStudentLogin(RequestObject requestObject)

{

List<Object> requestObjects = requestObject.getRequestObjects();

StudentInfo studentInfo = (StudentInfo) requestObjects.get(0);

StudentInfo studentInfoFromDB = this.studentInfoRepository.findOne(studentInfo.getRegistrationId());

ResponseObject responseObject = new ResponseObject();

if(null == studentInfoFromDB)

{

responseObject.setResponseCode(-1);

responseObject.setResponseMessage("Not a registered student");

}

else if(studentInfoFromDB.getStatus().longValue() != 1l)

{

responseObject.setResponseCode(-1);

responseObject.setResponseMessage("Registration is not active, Contact Admin");

}

else if(studentInfoFromDB.getPassword().equals(studentInfo.getPassword()))

{

responseObject.setResponseCode(0);

responseObject.setResponseMessage("WELCOME");

List responseObjects = new ArrayList();

responseObjects.add(TransformerUtil.transformStudentInfo(studentInfoFromDB));

responseObject.setResponseObject(responseObjects);

}

else

{

responseObject.setResponseCode(-1);

responseObject.setResponseMessage("Incorrect Password");

}

return responseObject;

}

public ResponseObject getDepartmentList(RequestObject requestObject)

{

ResponseObject responseObject = new ResponseObject();

List<Department> listOfDepartments = this.departmentRepository.findAll();

List<Department> listOfNewDept = new ArrayList<Department>();

List responseObjects = new ArrayList();

for (Department department : listOfDepartments)

{ listOfNewDept.add(TransformerUtil.transformDepartment(department));

}

responseObjects.add(listOfNewDept);

responseObject.setResponseCode(0);

responseObject.setResponseMessage("SUCCESS");

responseObject.setResponseObject(responseObjects);

return responseObject;

}

public ResponseObject getEventList(RequestObject requestObject)

{

ResponseObject responseObject = new ResponseObject();

List<Event> listOfEvent = this.eventRepository.getAllFutureEvents(new Date());

List<Event> listOfNewEvents = new ArrayList<Event>();

List responseObjects = new ArrayList();

for (Event event : listOfEvent)

{

listOfNewEvents.add(TransformerUtil.transformEvent(event));

}

responseObjects.add(listOfNewEvents);

responseObject.setResponseCode(0);

responseObject.setResponseMessage("SUCCESS");

responseObject.setResponseObject(responseObjects);

return responseObject;

}

public ResponseObject getForumList(RequestObject requestObject)

{

List<ForumTopic> forumList = this.forumTopicRepository.findAll();

List responseObjects = new ArrayList();

List<ForumTopic> responseForumList = new ArrayList<ForumTopic>();

for (ForumTopic forumTopic : forumList) {

responseForumList.add(TransformerUtil.transformForumTopic(forumTopic)); }

ResponseObject responseObject = new ResponseObject();

responseObjects.add(responseForumList);

responseObject.setResponseCode(0);

responseObject.setResponseMessage("SUCCESS");

responseObject.setResponseObject(responseObjects);

return responseObject;

}

public ResponseObject getForumPosts(RequestObject requestObject) {

List<Object> requestObjects = requestObject.getRequestObjects();

StudentInfo studentInfo = (StudentInfo) requestObjects.get(0);

ForumTopic forumTopic = (ForumTopic) requestObjects.get(1);

List<ForumPost> forumPosts = this.forumPostRepository

.findByForumTopicForumId(forumTopic.getForumId());

List<ForumPost> forumPostsResponse = new ArrayList<ForumPost>();

for (ForumPost forumPost : forumPosts) {

forumPostsResponse.add(TransformerUtil

.transformForumPost(forumPost));

}

List responseObjects = new ArrayList();

ResponseObject responseObject = new ResponseObject();

responseObjects.add(forumPostsResponse);

responseObject.setResponseCode(0);

responseObject.setResponseMessage("SUCCESS");

responseObject.setResponseObject(responseObjects);

return responseObject;

}

public ResponseObject createForumTopic(RequestObject requestObject) {

List<Object> requestObjects = requestObject.getRequestObjects();

StudentInfo studentInfo = (StudentInfo) requestObjects.get(0);

ForumTopic forumTopic = (ForumTopic) requestObjects.get(1);

this.forumTopicRepository.save(forumTopic);

ResponseObject responseObject = new ResponseObject();

responseObject.setResponseCode(0);

responseObject.setResponseMessage("SUCCESS");

return responseObject;

}

public ResponseObject createSuggestion(RequestObject requestObject) {

List<Object> requestObjects = requestObject.getRequestObjects();

StudentInfo studentInfo = (StudentInfo) requestObjects.get(0);

Suggestion suggestion = (Suggestion) requestObjects.get(1);

suggestion.setStudentInfo(studentInfo);

this.suggestionRepository.save(suggestion);

ResponseObject responseObject = new ResponseObject();

responseObject.setResponseCode(0);

responseObject.setResponseMessage("SUCCESS");

return responseObject;

}

public ResponseObject createForumPost(RequestObject requestObject) {

List<Object> requestObjects = requestObject.getRequestObjects();

StudentInfo studentInfo = (StudentInfo) requestObjects.get(0);

ForumTopic forumTopic = (ForumTopic) requestObjects.get(1);

ForumPost forumPost = (ForumPost) requestObjects.get(2);

ForumAccess forumAccessForStudentandForum = this.forumAccessRepository.getForumAccessForStudentandForum(studentInfo.getRegistrationId(), forumTopic.getForumId());

if(forumAccessForStudentandForum == null)

{

forumAccessForStudentandForum = new ForumAccess(); forumAccessForStudentandForum.setForumTopic(forumTopic); forumAccessForStudentandForum.setStudentInfo(studentInfo); this.forumAccessRepository.save(forumAccessForStudentandForum); forumAccessForStudentandForum = this.forumAccessRepository.getForumAccessForStudentandForum(studentInfo.getRegistrationId(), forumTopic.getForumId());

}

forumPost.setForumAccess(forumAccessForStudentandForum);

forumPost.setForumTopic(forumTopic);

this.forumPostRepository.save(forumPost);

ResponseObject responseObject = new ResponseObject();

responseObject.setResponseCode(0);

responseObject.setResponseMessage("SUCCESS");

return responseObject;

}

public ResponseObject getTestList(RequestObject requestObject) {

List<Test> testList = this.testRepository.findAll();

List responseObjects = new ArrayList();

List<Test> responseTestList = new ArrayList<Test>();

for (Test test : testList) {

responseTestList.add(TransformerUtil.transformTest(test));

}

ResponseObject responseObject = new ResponseObject();

responseObjects.add(responseTestList);

responseObject.setResponseCode(0);

responseObject.setResponseMessage("SUCCESS");

responseObject.setResponseObject(responseObjects);

return responseObject;

}

public ResponseObject getQuestionsList(RequestObject requestObject)

{

List<Object> requestObjects = requestObject.getRequestObjects();

Test test = (Test) requestObjects.get(0);

List<QuestionAnswer> questionAnswers = this.questionAnswerRepository.findByTestTestId(test.getTestId());

List responseObjects = new ArrayList();

List<QuestionAnswer> responseQuestionAnswerList = new ArrayList<QuestionAnswer>();

for (QuestionAnswer questionAnswer : questionAnswers) { responseQuestionAnswerList.add(TransformerUtil.transformQuestionAnswer(questionAnswer));

}

ResponseObject responseObject = new ResponseObject();

responseObjects.add(responseQuestionAnswerList);

responseObject.setResponseCode(0);

responseObject.setResponseMessage("SUCCESS");

responseObject.setResponseObject(responseObjects);

return responseObject;

}

public ResponseObject processRequest(RequestObject requestObject)

{

String requestType = requestObject.getRequestType();

ResponseObject responseObject = null;

if("LOGIN".equals(requestType))

{

responseObject = this.validateStudentLogin(requestObject);

}

else if("REGISTER".equals(requestType))

{

responseObject= this.createStudentInfo(requestObject);

}

else if("GET\_DEPT\_LIST".equals(requestType))

{

responseObject= this.getDepartmentList(requestObject);

}

else if("GET\_EVENT".equals(requestType))

{

responseObject= this.getEventList(requestObject);

}

else if("CREATE\_FORUM\_TOPIC".equals(requestType))

{

responseObject = this.createForumTopic(requestObject);

}

else if("GET\_FORUM\_TOPICS".equals(requestType))

{

responseObject = this.getForumList(requestObject);

}

else if("CREATE\_FORUM\_POST".equals(requestType))

{

responseObject = this.createForumPost(requestObject);

}

else if("GET\_FORUM\_POSTS".equals(requestType))

{

responseObject = this.getForumPosts(requestObject);

}

else if("GET\_TEST".equals(requestType))

{

responseObject = this.getTestList(requestObject);

}

else if("GET\_QUESTION".equals(requestType))

{

responseObject = this.getQuestionsList(requestObject);

}

else if("CREATE\_SUGGESTION".equals(requestType))

{

responseObject = this.createSuggestion(requestObject);

}

return responseObject;

}

public DepartmentRepository getDepartmentRepository() {

return departmentRepository;

}

public void setDepartmentRepository(

DepartmentRepository departmentRepository) {

this.departmentRepository = departmentRepository;

}

public EventRepository getEventRepository() {

return eventRepository;

}

public void setEventRepository(EventRepository eventRepository) {

this.eventRepository = eventRepository;

}

public ForumAccessRepository getForumAccessRepository() {

return forumAccessRepository;

}

public void setForumAccessRepository(

ForumAccessRepository forumAccessRepository) {

this.forumAccessRepository = forumAccessRepository;

}

public ForumPostRepository getForumPostRepository() {

return forumPostRepository;

}

public void setForumPostRepository(ForumPostRepository forumPostRepository) {

this.forumPostRepository = forumPostRepository;

}

public ForumTopicRepository getForumTopicRepository() {

return forumTopicRepository;

}

public void setForumTopicRepository(

ForumTopicRepository forumTopicRepository) {

this.forumTopicRepository = forumTopicRepository;

}

public LibraryBookRepository getLibraryBookRepository() {

return libraryBookRepository;

}

public void setLibraryBookRepository(

LibraryBookRepository libraryBookRepository) {

this.libraryBookRepository = libraryBookRepository;

}

public StudentInfoRepository getStudentInfoRepository() {

return studentInfoRepository;

}

public void setStudentInfoRepository(

StudentInfoRepository studentInfoRepository) {

this.studentInfoRepository = studentInfoRepository;

}

public SuggestionRepository getSuggestionRepository() {

return suggestionRepository;

}

public void setSuggestionRepository(

SuggestionRepository suggestionRepository) {

this.suggestionRepository = suggestionRepository;

}

public TestRepository getTestRepository() {

return testRepository;

}

public void setTestRepository(TestRepository testRepository) {

this.testRepository = testRepository;

}

public QuestionAnswerRepository getQuestionAnswerRepository() {

return questionAnswerRepository;

}

public void setQuestionAnswerRepository(

QuestionAnswerRepository questionAnswerRepository) {

this.questionAnswerRepository = questionAnswerRepository;

}};

**BuzzoneRequestHandler.java**

package com.saveetha.buzzone.servlet;

import java.io.IOException;

import java.io.ObjectInputStream;

import java.io.ObjectOutputStream;

import java.util.List;

import javax.servlet.Servlet;

import javax.servlet.ServletConfig;

import javax.servlet.ServletException;

import javax.servlet.ServletRequest;

import javax.servlet.ServletResponse;

import org.springframework.context.ApplicationContext;

import org.springframework.context.support.ClassPathXmlApplicationContext;

import com.saveetha.buzzone.service.BuzzoneService;

import com.saveetha.buzzone.vo.RequestObject;

import com.saveetha.buzzone.vo.ResponseObject;

import com.saveetha.buzzone.vo.StudentInfo;

public class BuzzoneRequestHandler implements Servlet

{

private static ApplicationContext appContext = null;

public void destroy() {

}

public ServletConfig getServletConfig() {

return null;

}

public String getServletInfo() {

return null;

}

public void init(ServletConfig arg0) throws ServletException {

appContext = new ClassPathXmlApplicationContext(

"buzzoneAppContext.xml");

}

public void service(ServletRequest arg0, ServletResponse arg1)

throws ServletException, IOException {

ObjectInputStream objInputStream = new ObjectInputStream(arg0.getInputStream());

Object readObject = null;

ResponseObject responseObject = null;

try {

readObject = objInputStream.readObject();

} catch (ClassNotFoundException e) {

e.printStackTrace();

}

if(readObject instanceof RequestObject)

{

BuzzoneService buzzoneService = (BuzzoneService) appContext.getBean("buzzoneService");

RequestObject requestObject = (RequestObject) readObject;

String requestType = requestObject.getRequestType();

responseObject = buzzoneService.processRequest(requestObject);

}

ObjectOutputStream objStream = new ObjectOutputStream(arg1.getOutputStream());

objStream.writeObject(responseObject);

objStream.flush();

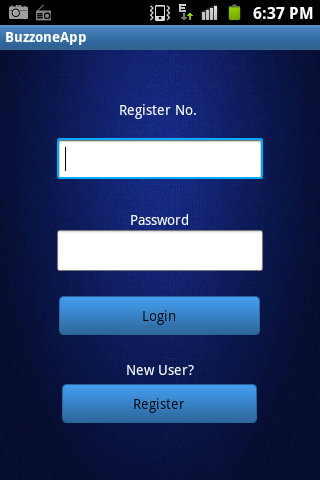
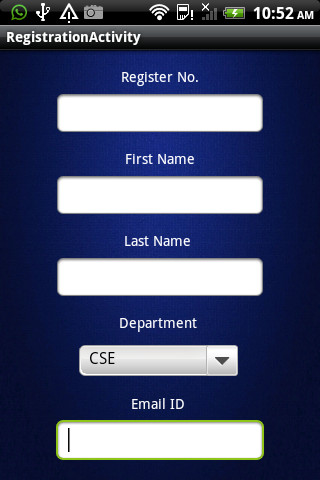
}

}

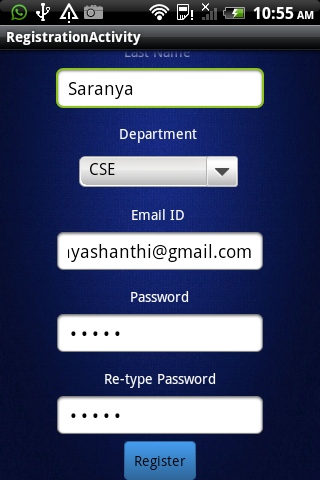
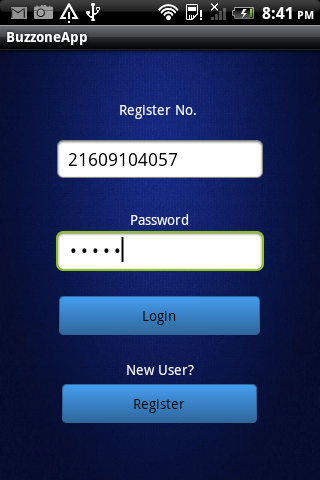
**APPENDIX B**

**OUTPUT SCREENSHOTS**

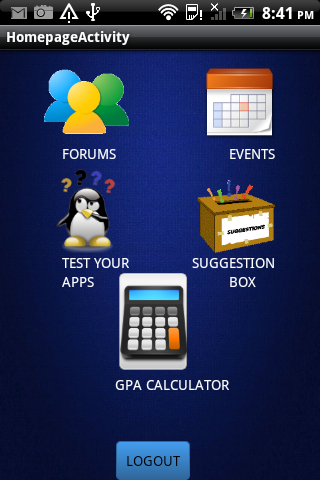
**1. Login Screen 2. Registration Form**

** **

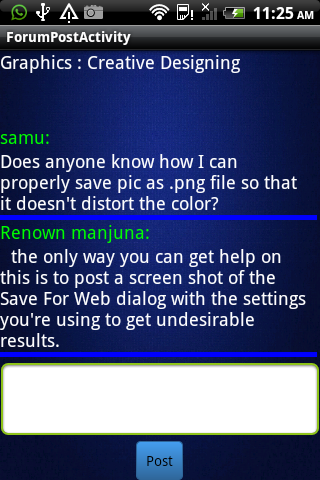
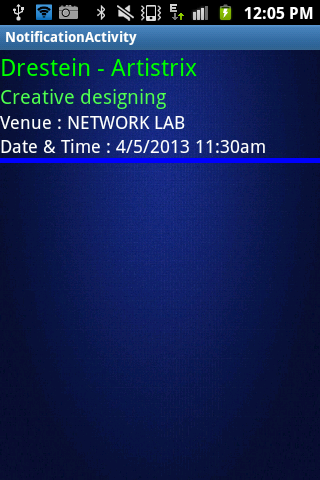
**3. Registration 4. Login**

** **

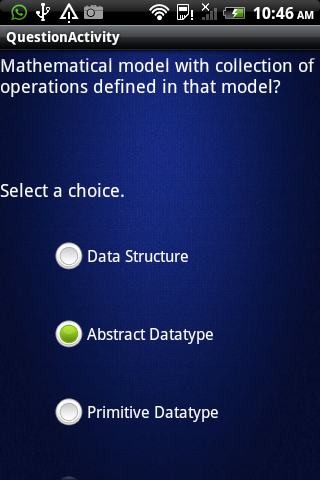
**5. Home Page 6. Forums**

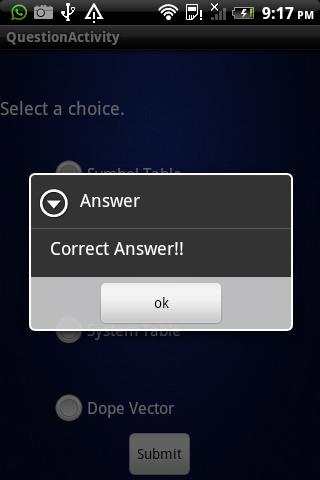
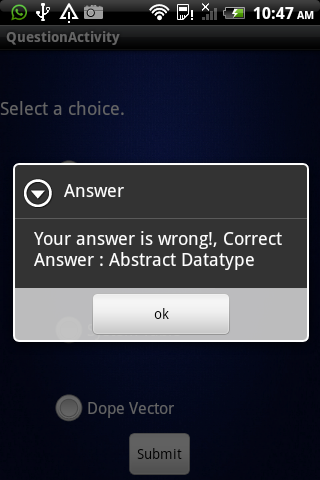
** **

**7. Forum Posts 8. Events Notification**

** **

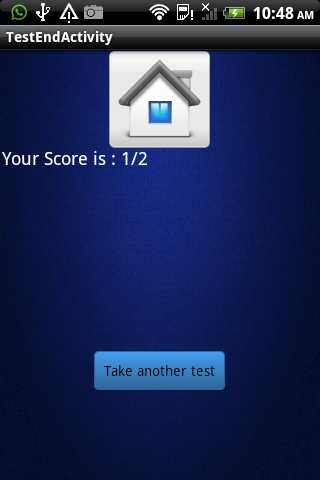
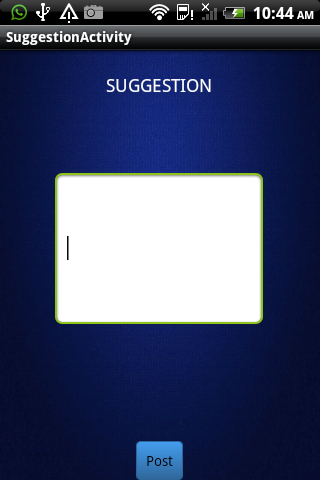
**9. Test your Apps**

****

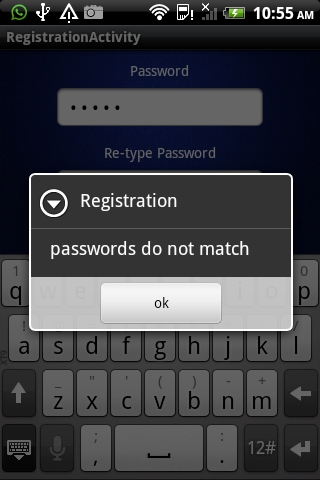
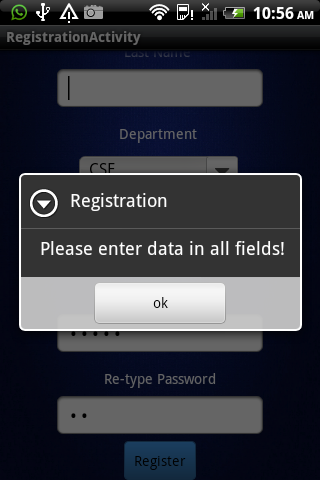
** **

* Alert box Answer displays message “Correct Answer” when the user selects the Correct Answer.
* Alert box Answer displays message “Your Answer is Wrong” along with the Correct Answer when the user selects the Wrong Answer.

**11. Display score 12. Suggestion box**

** **

**TEST CASES FOR REGISTRATION**

* Alert box Registration pops when entered Password and Re-type password fields do not match.
* Alert box Registration pops when all the fields in the Registration Form are not entered.

**REFERENCES**

|  |  |
| --- | --- |
| 1. | *Brett McLaughlin,* ***“*Java & XML data binding”*,*** O'Reilly Media, Inc. |
| 2. | *Craig Larman*, **“Applying UML and Patters”** – An introduction to object-oriented Analysis and Design and iterative Development,Pearson Publications. |
| 3. | *Mark L. Murphy*, **“The Busy Coder's Guide to Android Development”***,* CommonsWare, LLC. |
| 4. | *Nicole B. Ellison*, Department of Telecommunication, Information Studies, and Media Michigan State University, & *Danah m. boyd*, School of Information University of California-Berkeley **“Social Network Sites: Definition, History, and Scholarship”.** |
| 5. | *Schildt*, **“Java2 Complete Reference” –** 3rd edition Tata McGraw Hill Publication. |
| 6. | Android Developer, <http://developer.android.com/> |
| 7. | Forum : <http://stackoverflow.com/> |
|  |  |