|  |
| --- |
| Centennial college |
| Education Hub |
| Group 08: NavNisParVip |
|  |
|  |
| **4/17/2014** |

**Submitted to:**

**Prof. Arben Tapia**

**Prepared By:**

**Parimal Patel (300772037)**

**Nisha Dhanwani (300773598)**

**Vipulkumar Patel (300753649)**

**Navdeep Kaur Gill (300772906)**

**Table of Contents**

[Introduction 4](#_Toc386986771)

[Overview of Document 5](#_Toc386986775)

Initial [Gantt Chart 5](#_Toc386986776)

[Most Recent (final) itemized Gantt chart 5](#_Toc386986778)

[Requirements Specification 6-7](#_Toc386986779)

[Functional Requirements 6](#_Toc386986780)

[Non-functional Requirements 7](#_Toc386986781)

[Software Design 8-9](#_Toc386986782)

[Use Case Diagrams: 8](#_Toc386986785)

[Entity Relationship Diagram (ERD) 9](#_Toc386986791)

[UI Design 10-24](#_Toc386986792)

[Test Plan 25-28](#_Toc386986804)

[Introduction 25](#_Toc386986805)

[Goals 25](#_Toc386986806)

[Assumptions 25](#_Toc386986807)

[Risks and Assets 25](#_Toc386986808)

[Scope 25](#_Toc386986809)

[Features to be tested 26](#_Toc386986810)

[Functional Testing 26](#_Toc386986811)

[Non-Functional Testing 26](#_Toc386986812)

[Testing Procedures 27](#_Toc386986814)

[Test Objectives 27](#_Toc386986815)

[Types of Testing 27](#_Toc386986816)

[Testing Tools 28](#_Toc386986817)

[Schedule and Deliverables 28](#_Toc386986818)

[Test Report 29-57](#_Toc386986819)

**Introduction:**

Education Hub is an android application will enact as a bridge of communication between students and faculty. The main concept of this android application is to increase communication between student and expert faculty which can help in better education sharing platform and provide an easy way for education. This application can be accessed by Students and Faculties by their own login.

This application will have two modules.

1. User

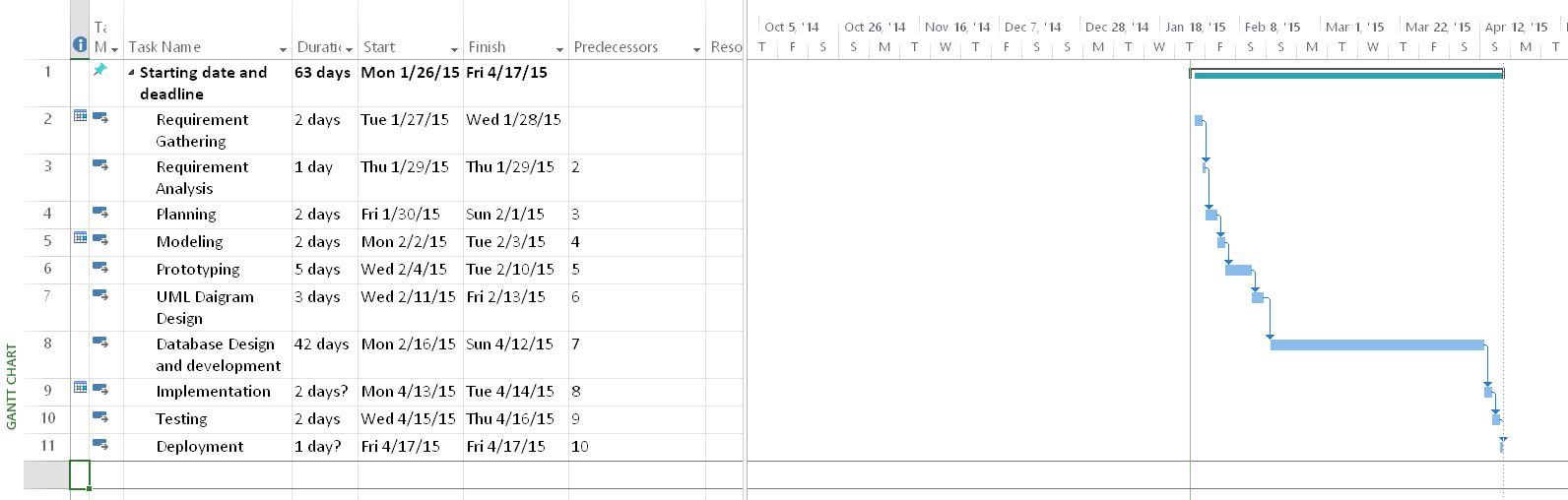
2. Faculty

The following are features of Education Hub:

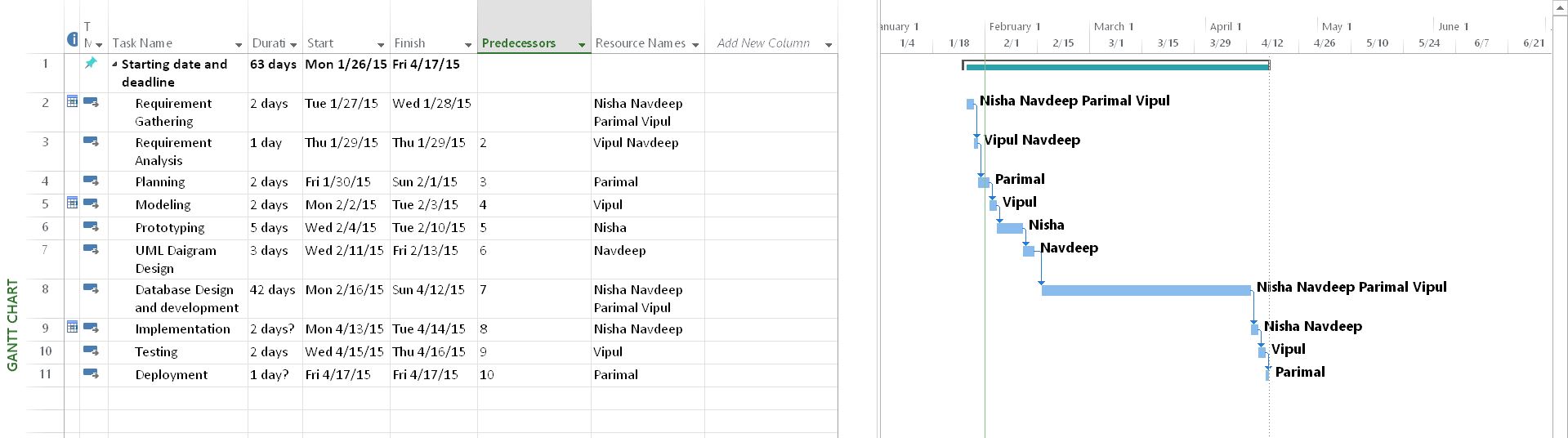
As a student, user can use Student module to access subject materials online, check expert faculty related subject, and find out about other services offered which are not directly related to your studies and more. The services provided to the faculty include viewing and update their personal details, upload the content for a course, view the discussion board and feedbacks to the students, student can send message to faculty.

**Overview of Document:**

Initial Gantt chart:



Final itemized Gantt chart:



# **Requirements Specification**

## Functional Requirements:

Application will act as a bridge between students and faculty for teaching purpose.

**Users for System are:**

**Student:** Student have all access of application as per his roles provided. He can schedule a class, resource sharing, send a message, update his profile, and search for faculty.

**Faculty:** Faculty have all access of application as per his roles provided. He can schedule a class, resource sharing, send a message, update his profile.

## Non-functional Requirements:

**Product Requirements:**

System must have internet connection.

Mobile phone must have Android operating system.

**Usability Requirements:**

User must be registered.

Knowledge of how to operate phone.

**Efficiency Requirements**

**Performance Requirements**

Android OS

Minimum 256MB RAM

**Space Requirement**

Minimum 1MB of free space required.

**Reliability Requirements**

Network Security to prevent unauthorised access and secure confidential user information.

**Portability Requirement**

Can work on any system having Android and internet service.

**Ethical Requirements**

Protection of user’s information

No misuse of confidential data

**Legislative Requirements**

**Privacy Requirements**

User’s information is kept confidential.

User’s data should not be shared with 3rd party.

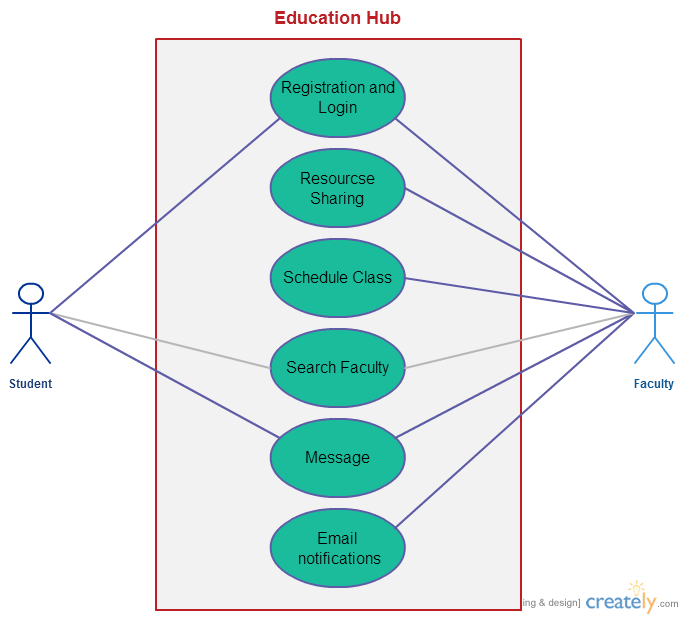
**Safety Requirements**

Username and Password facility is provided to each and every user.

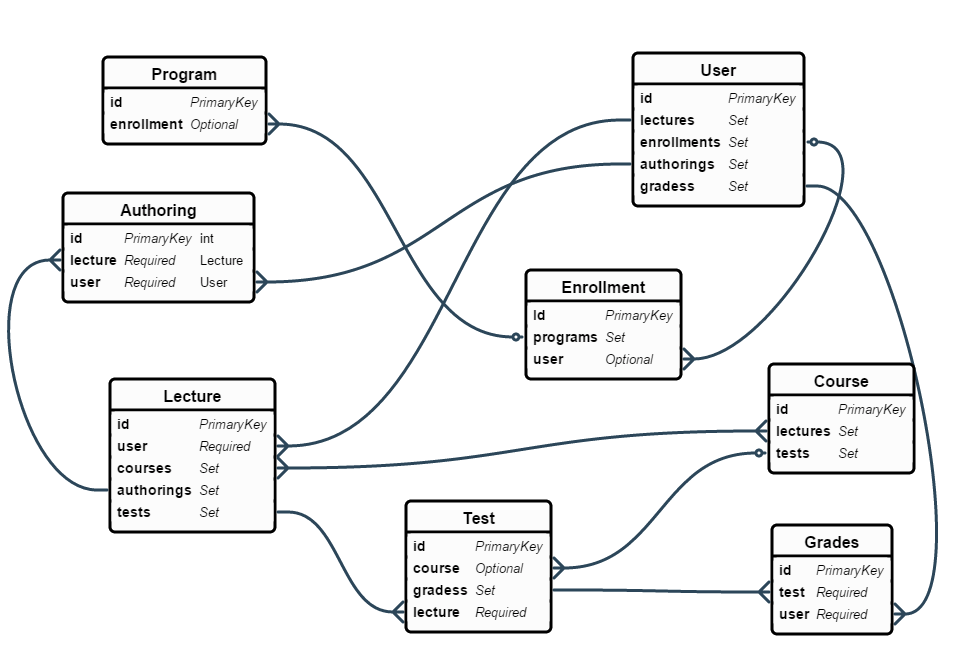
Different Users are provided with different permissions to access information.

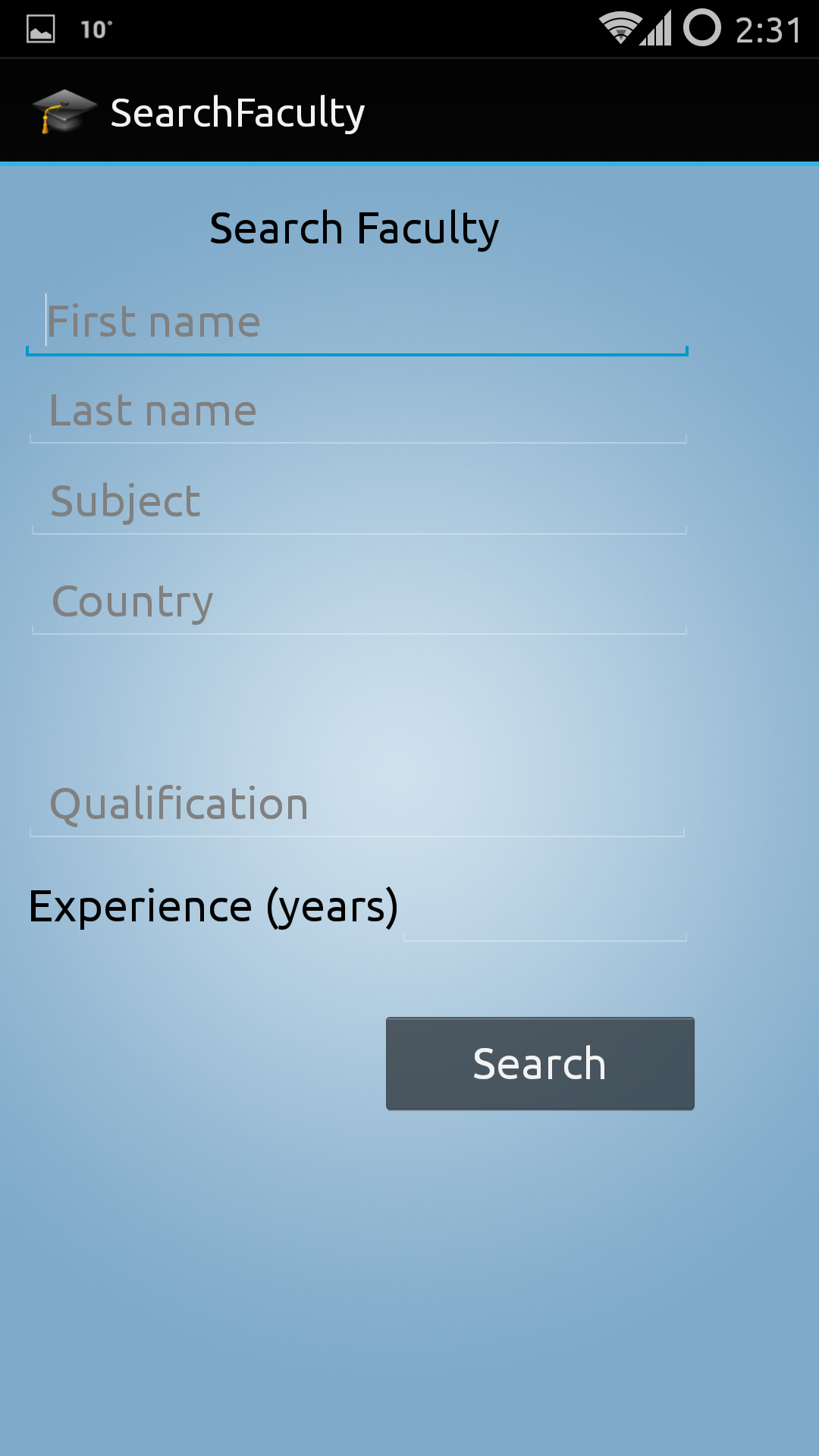
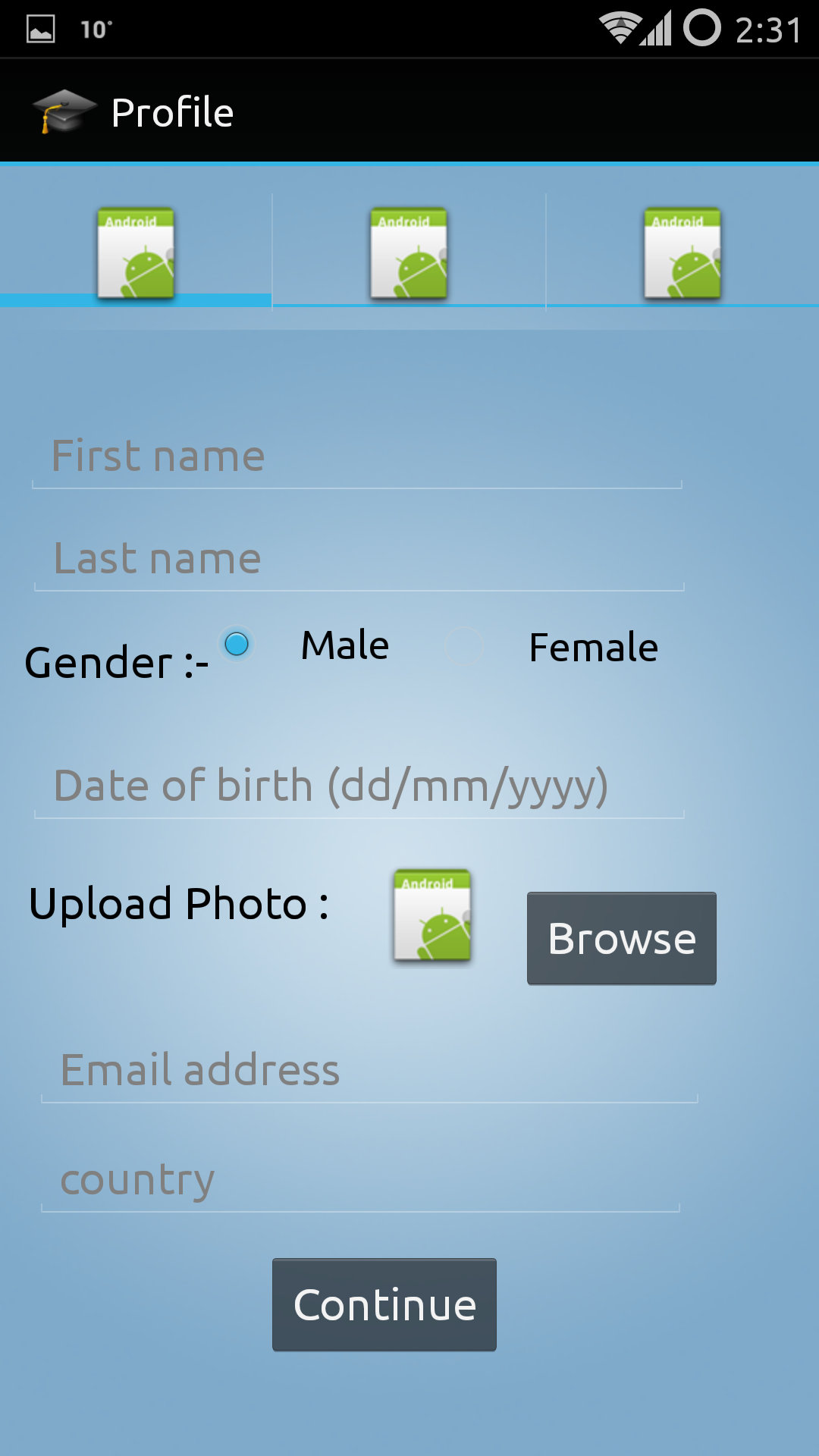
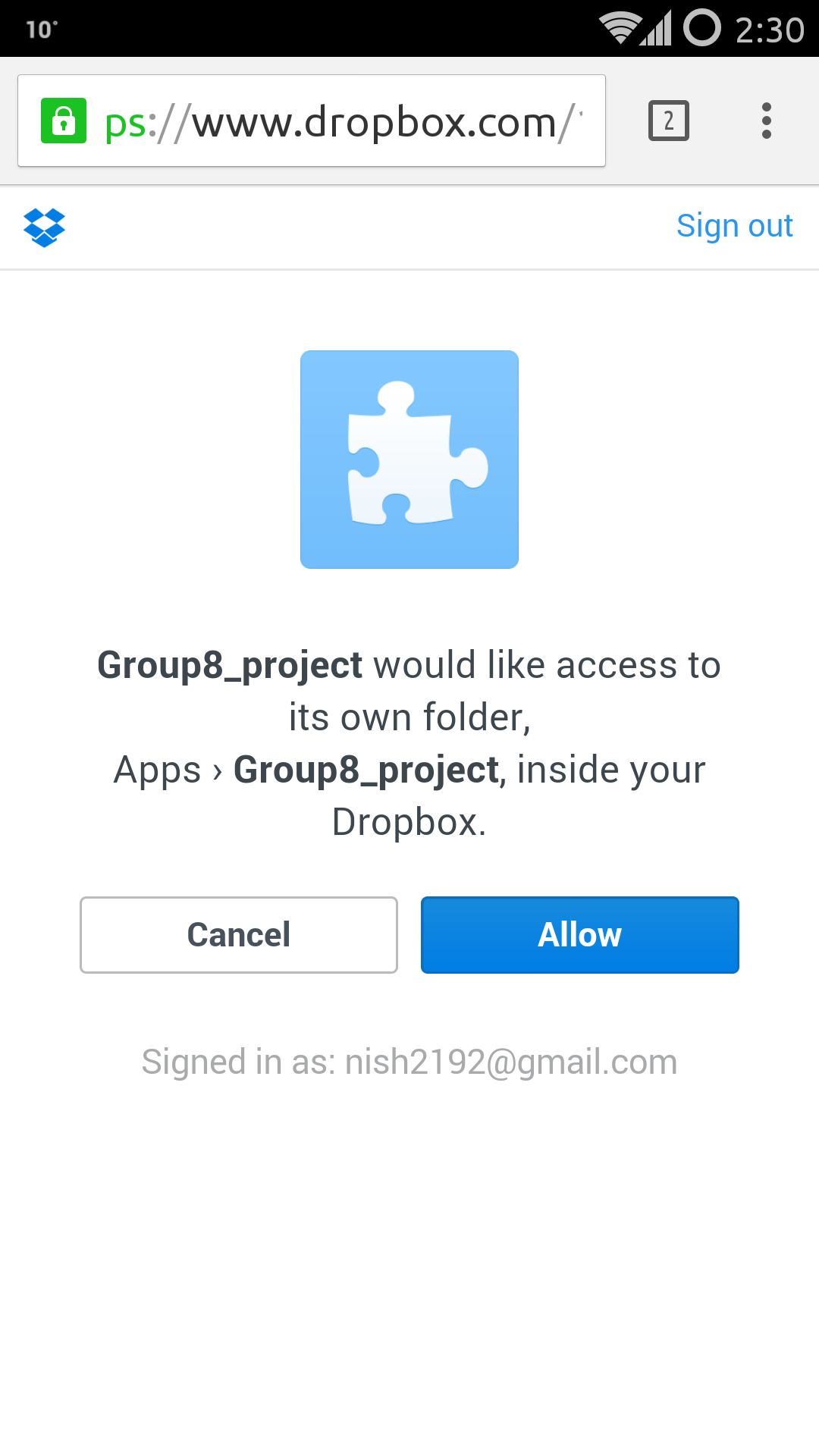
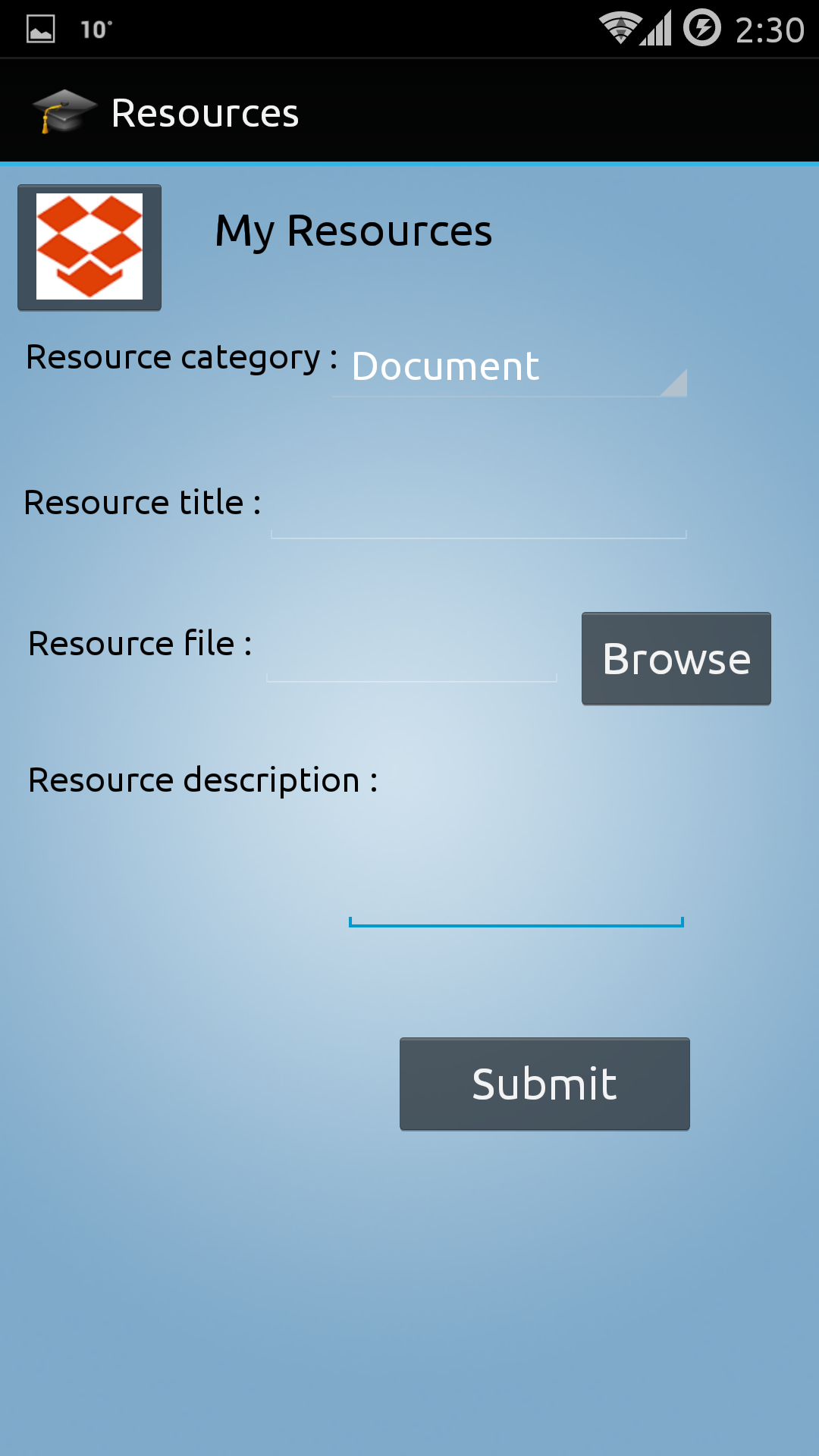
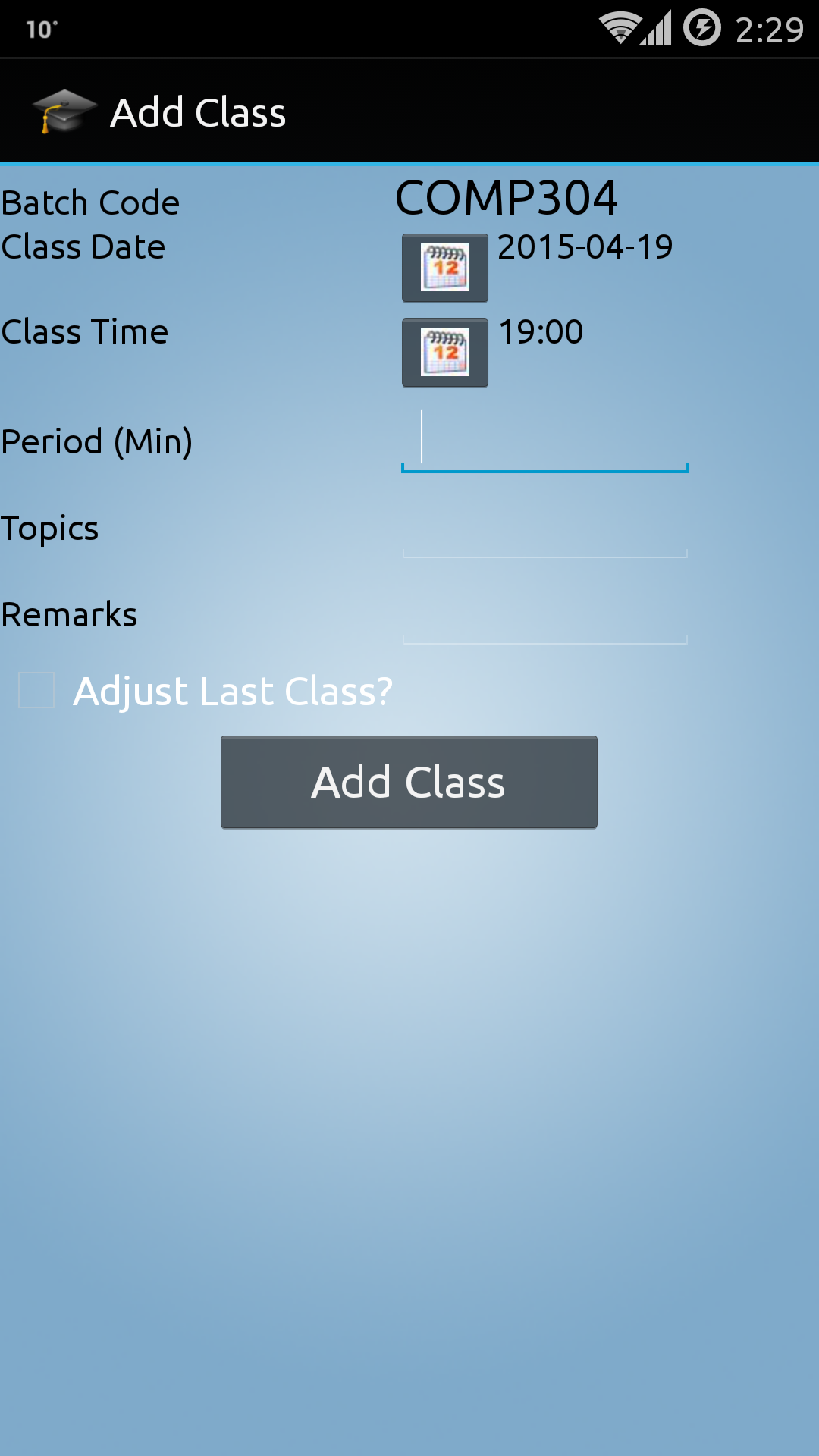
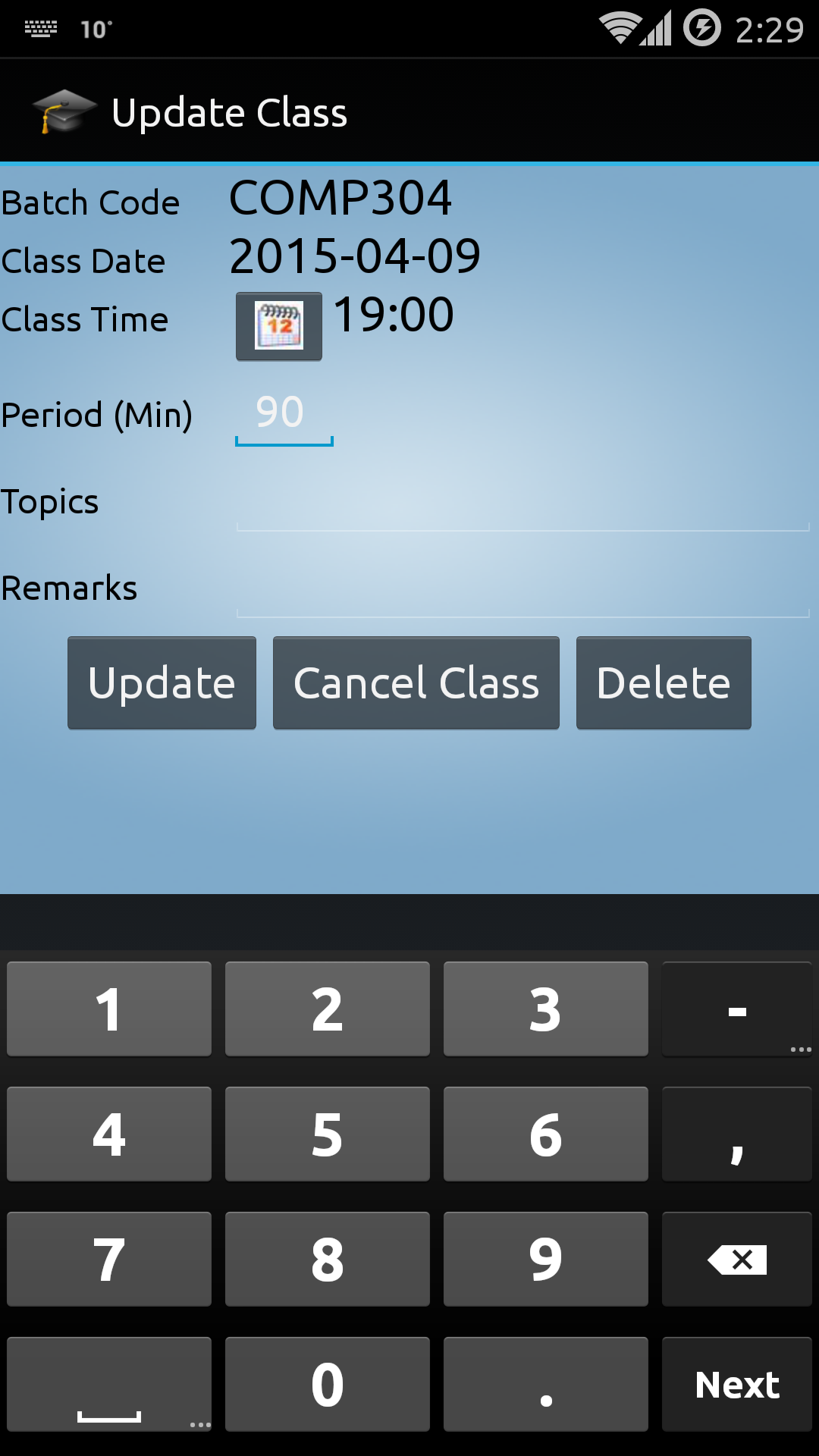
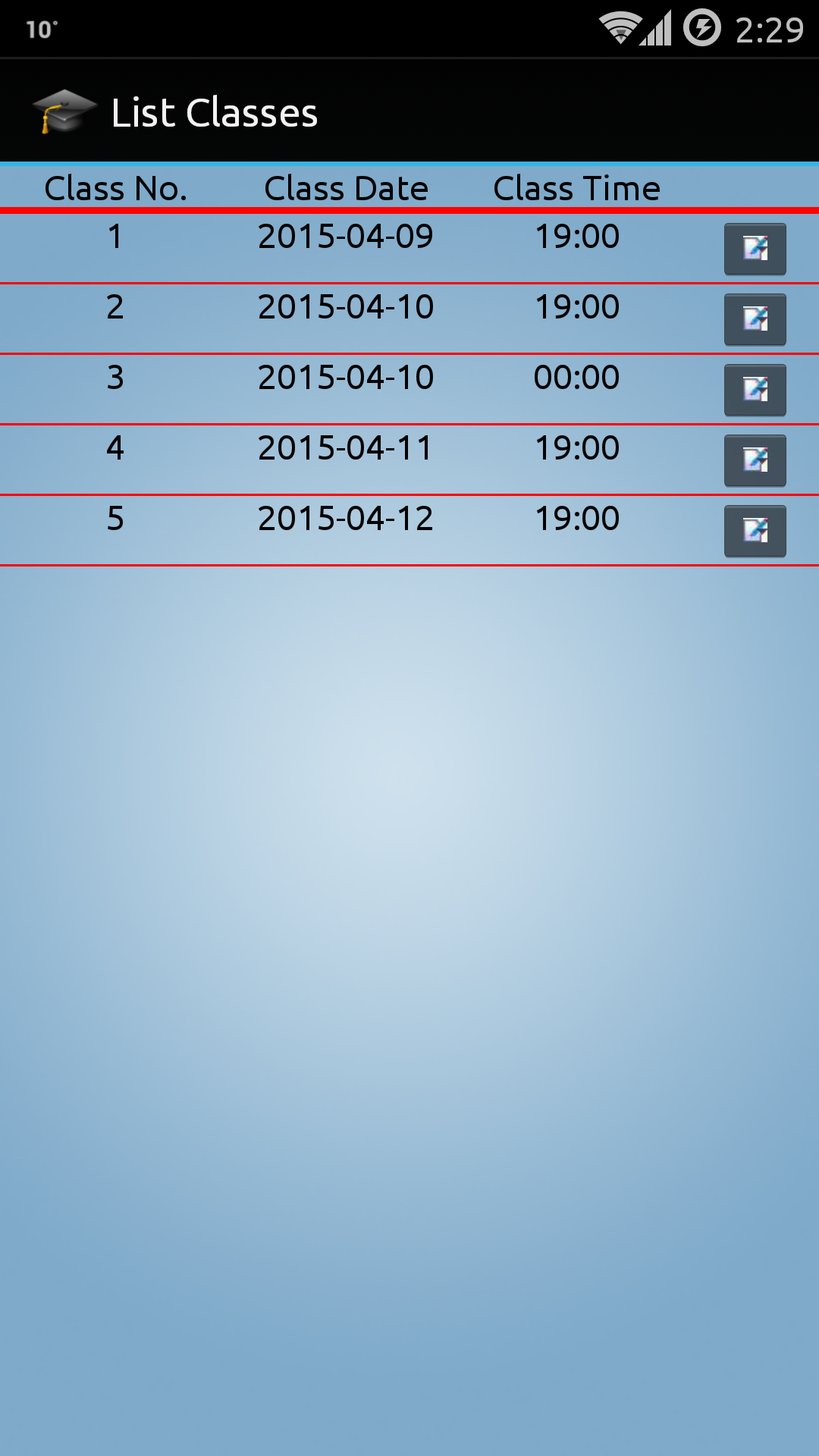
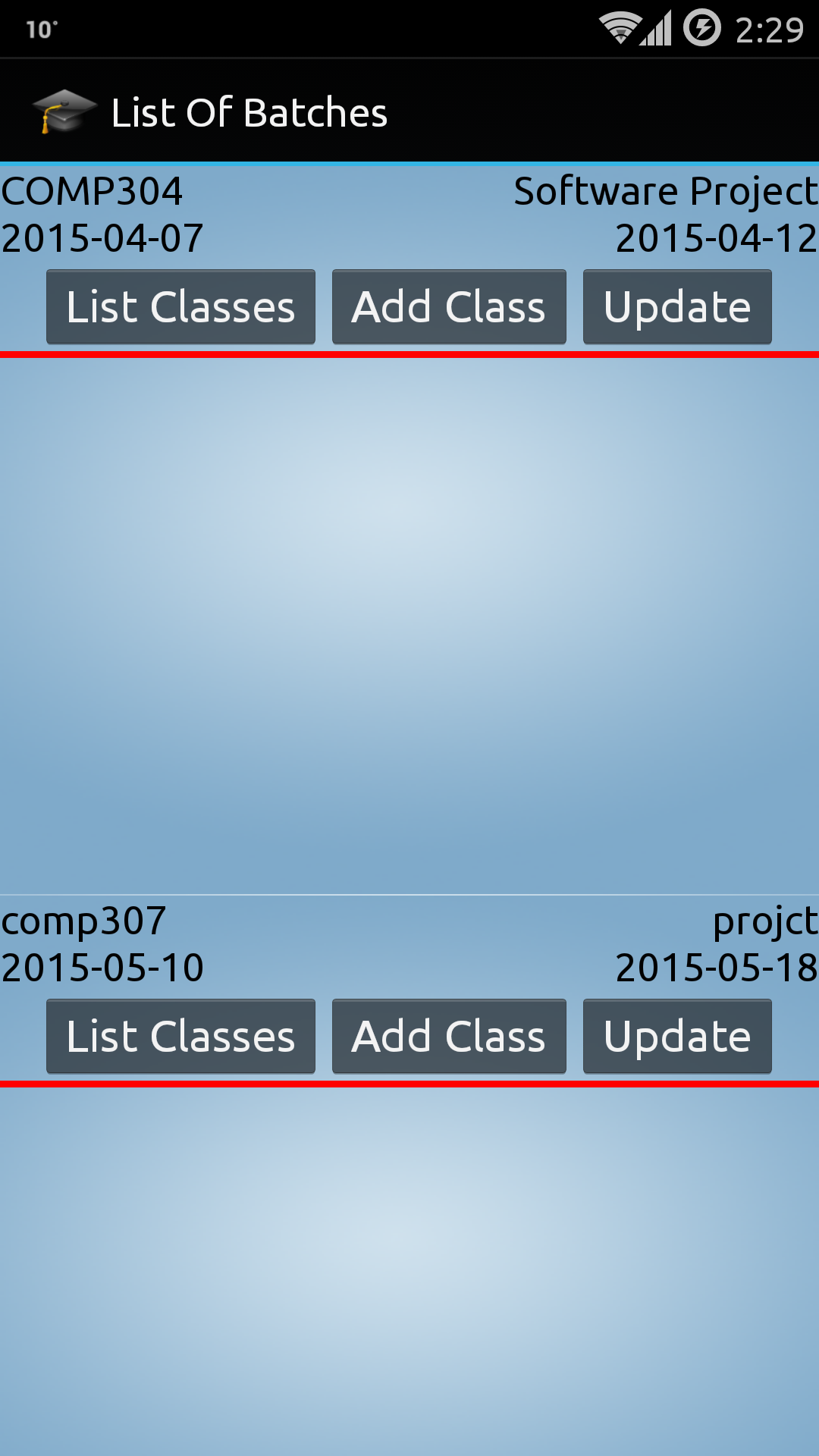
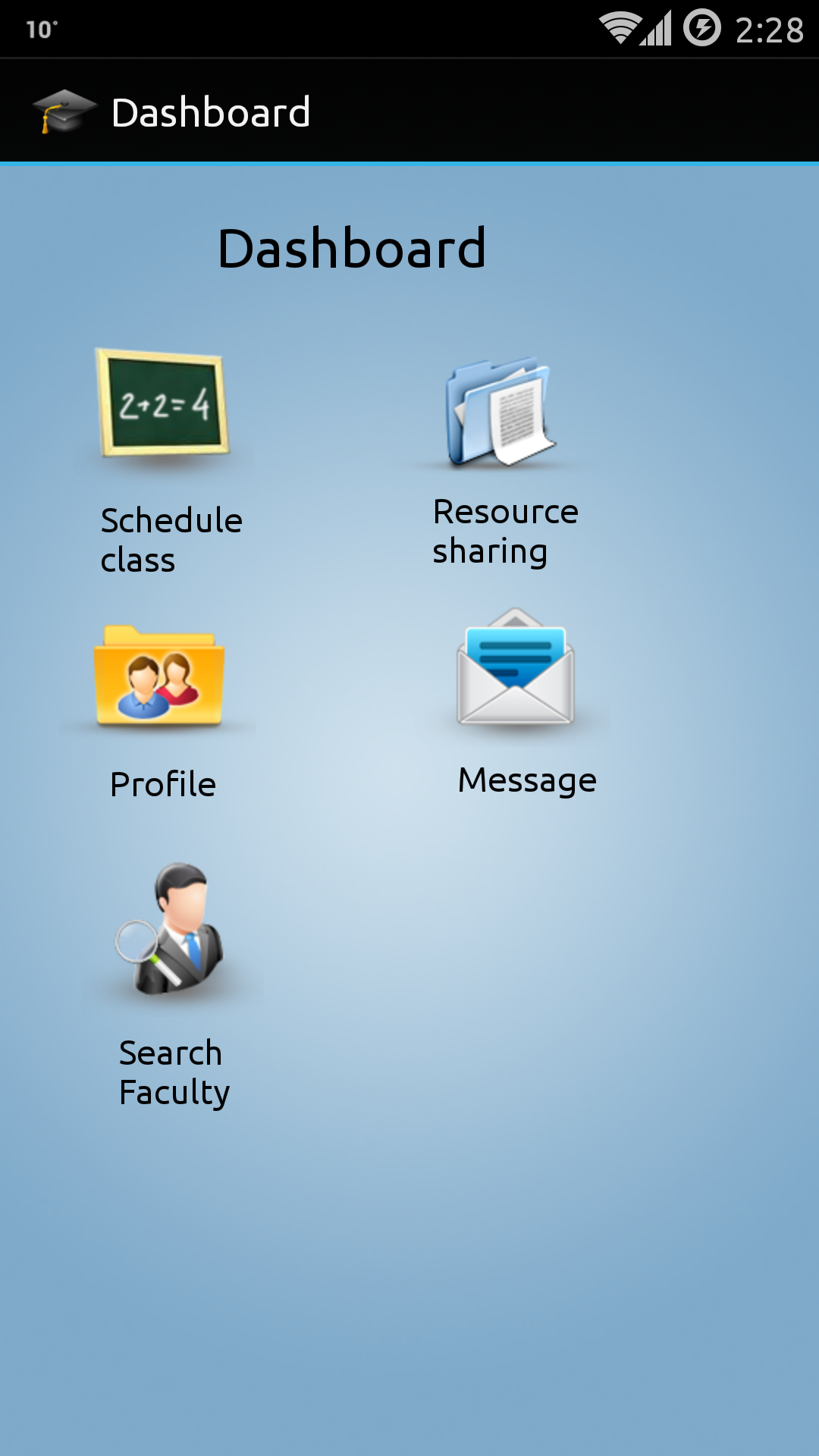
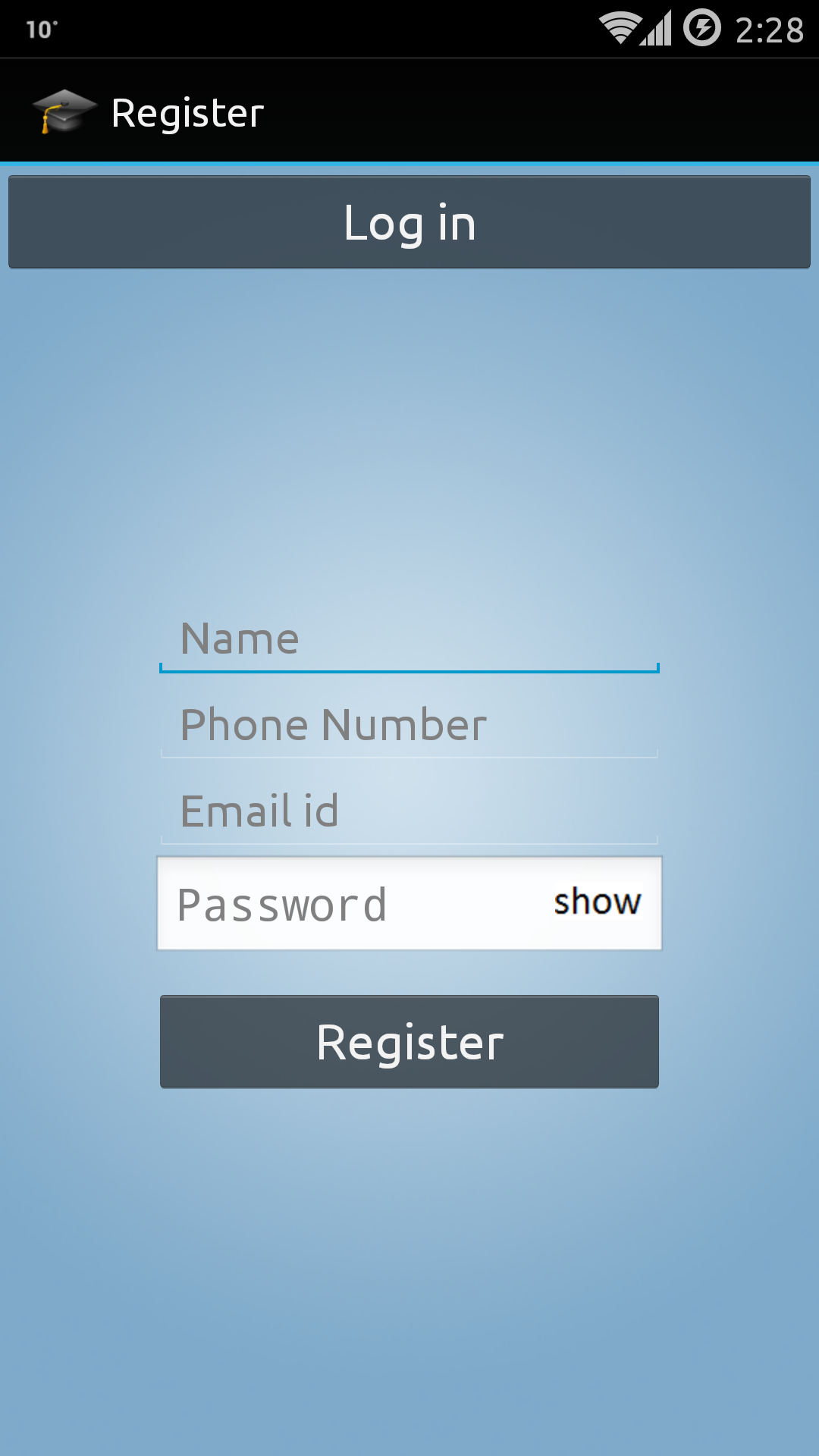
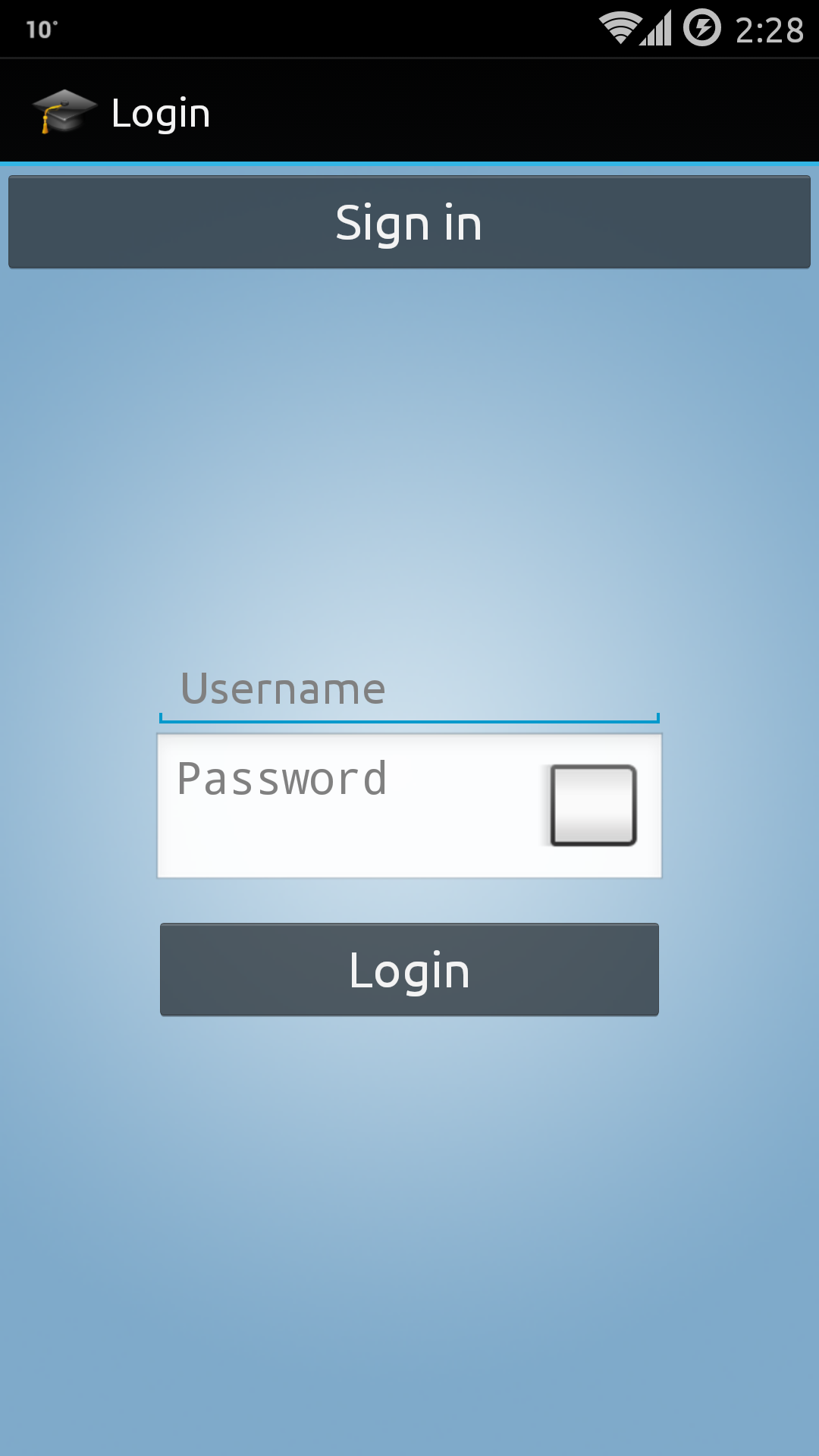
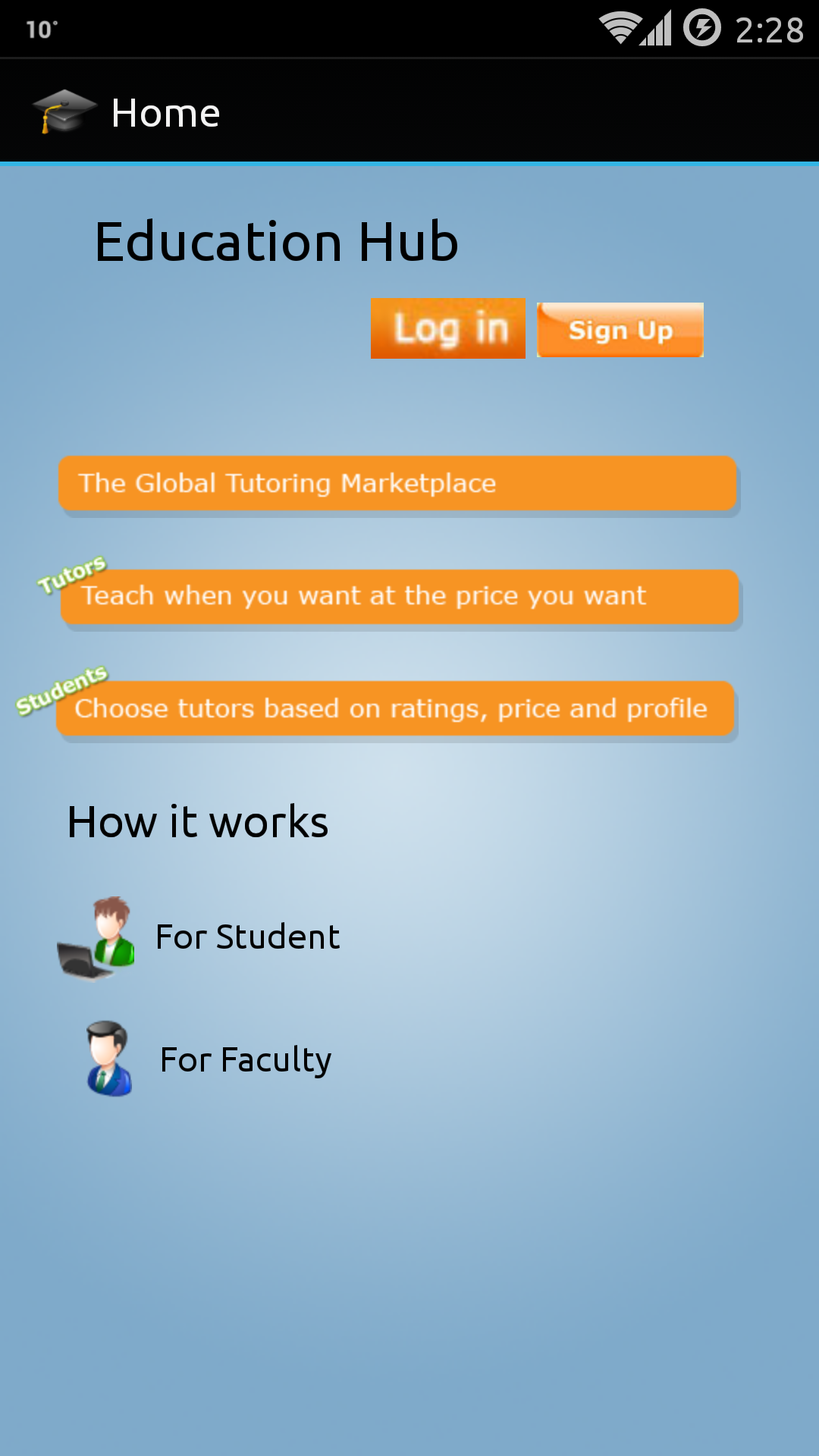
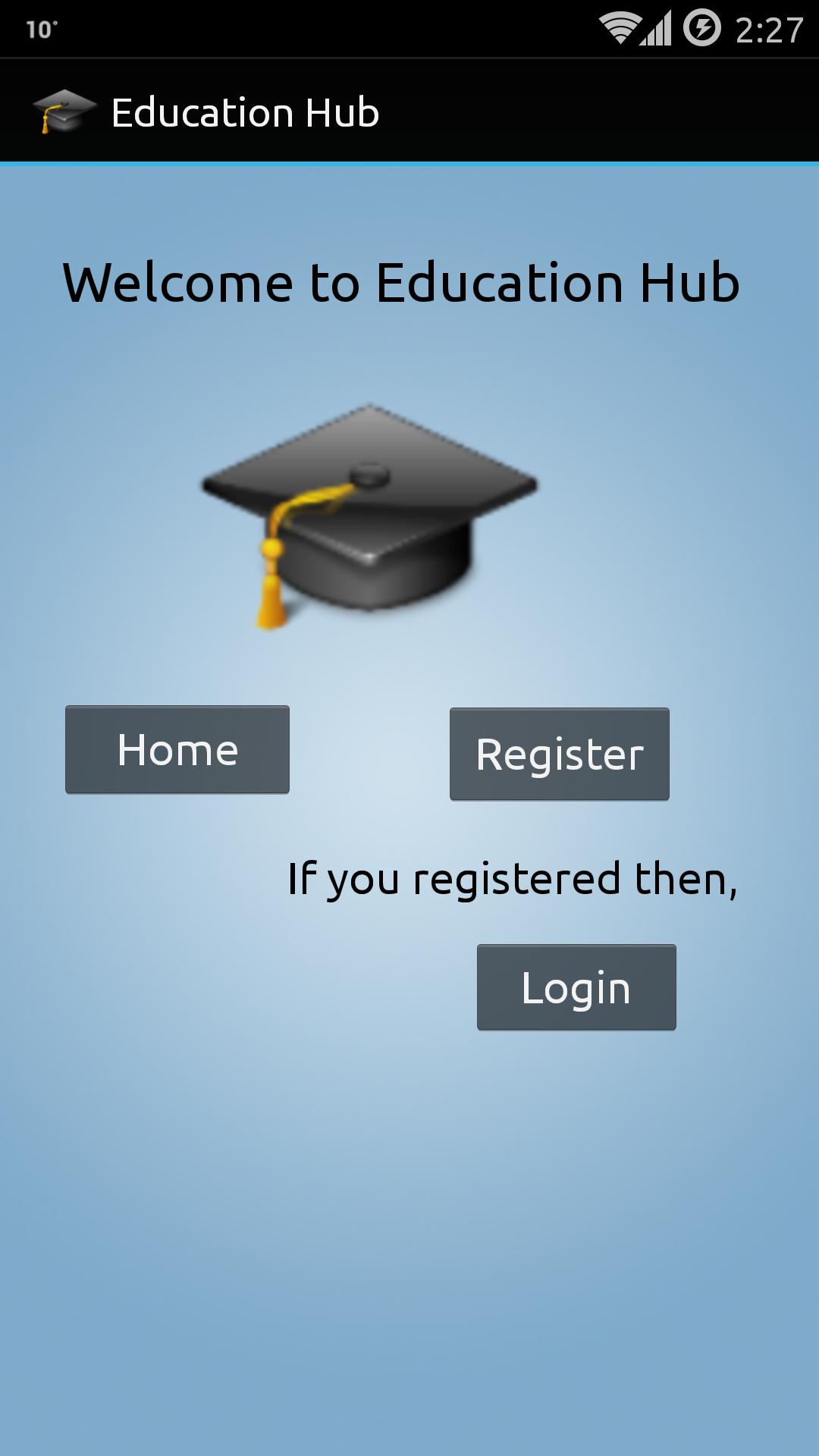
**Diagrams:**

Use case Diagram:

****

Entity-Relationship Diagram:



**UI Design:**

# **Test Plan**

## Introduction:

Test Plan for Education Hub System describes all the following assets

### Goals:

A successful application where different users have different privileges on the basis of their roles to track status of student and faculty. We can check the services provided by faculty to the student of their difficulties in various courses. We have two primary user student and faculty.

### Assumptions:

Assumptions while performing testing are:-

● Testing cycle will start only after finishing the development cycle.

● Unit test coverage and code coverage should be provided at the start of testing cycle.

● No Non-Functional testing will be performed until or unless specified explicitly.

### Risks and Assets:

1. Shared product version
2. Open communications
3. Systems perspective
4. Outline and continuous processes
5. Forward-looking search for uncertainties

## Scope:

Testing will be performed at several points in the software development life cycle as the student registration is constructed. Testing is a very 'dependent' activity. As a result, test planning is a continuing activity performed throughout the system development life cycle. Test plans must be developed for each level of student and faculty login.

## Features to be tested:

### Functional Testing

* Adding/removing users by admin
* View all student and faculty details
* Adding/removing new courses and teaching method
* Updating new event such as online quiz
* View report of student, faculty details

### Non-Functional Testing

* Hardware functions works properly with the system.
* Quality of the software.

## Features not to be tested

* Speed of the software execution.

## Testing Procedures:

Testing is the process of analyzing a software item to detect the differences between existing and required conditions and to evaluate the features of the software item.

### Test Objectives

Today, software must meet your customers' needs in an ever-changing landscape.

### Types of Testing

**Unit testing**

The strategy for unit testing of individual subsystems is described. This includes an indication of the subsystems that will undergo unit tests or the criteria to be used to select subsystems for unit test. Test cases are NOT included here.

**Integration testing**

The integration testing strategy is specified. Describe the tests that will be performed in order to verify the interfaces between the subsystems of the software system. This section includes a discussion of the order of integration of subsystems. Test cases are NOT included here.

**Acceptance testing**

The strategy for testing the software once it has been installed on the user site is specified. This section includes a discussion of the order of acceptance by software function. Test cases are NOT included here.

**Stress testing**

Identify the limits under which the program is expected to perform (memory constraints, disk space constraints, etc.)

**Performance testing**

Refer to the functional requirements that specify acceptable performance. Software developer responsible for developing a particular module and Software Tester specialized in performance testing. Services within the system would be tested to check for performance under peak load.

## Testing Tools

***Android studio and the Android gradle V 1.1:*** *Android studio and the Android gradle V 1.1*is the tool for plugin. This feature is still [considered](http://tools.android.com/tech-docs/unit-testing-support) experimental, we will try to find bugs [and update the bug tracker](https://code.google.com/p/android/issues/entry?template=Android%20Studio%20bug).

**How it works**

Unit tests run on a local JVM on your development machine. Our gradle plugin will compile [source code](http://tools.android.com/tech-docs/unit-testing-support) found in src/test/java and execute it using the usual Gradle testing mechanisms. At runtime, tests will be executed against a modified [version](http://tools.android.com/tech-docs/unit-testing-support) of android.jar where all final [modifiers](http://tools.android.com/tech-docs/unit-testing-support) have been stripped off. This lets you use popular mocking libraries, like Mockito.

You will have to specify your testing [dependencies](http://tools.android.com/tech-docs/unit-testing-support) in the build. Gradle file of your android [module](http://tools.android.com/tech-docs/unit-testing-support).

For example:

dependencies {

  testCompile 'junit:junit:4.12'

  testCompile "org.mockito:mockito-core:1.9.5"

}

**Automated Tests**

* Import automated test projects from eclipse to create managed test cases that are linked to requirements
* Run automated test cases and collect test results in ***Android studio and the Android gradle V 1.1,***

## Schedule and Deliverables:

Below are the deliverables which are to be delivered at the end of testing cycle.

* Acceptance test plan
* System/Integration test plan
* Unit test plans/turnover documentation
* Screen prototypes
* Report mock-ups
* Defect/Incident reports and summaries
* Test logs and turnover report
* Test Summary Report
* Code Coverage Report

**Test Report:**

**Test Cases**

**Test Case #: 1**

**Test Case Name:** Student Registration

**System:** Education Hub

**Short Description:** Test student registration

**Pre-conditions:** The student has valid email-id and phone number.

The system displays the menu for registration.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Step** | **Action** | **Expected System Response** | **Pass/Fail** | **Comment** |
| 1 | Click the **Registration** button | The system displays a registration form |  |  |
| 2 | Enter **Name** | If the name already exist, system displays **Try another name** |  |  |
| 3 | Enter **Email Id** | If the email id already exist, system displays **try another email** |  |  |
| 4 | Select **Signup as Student** | Student selected |  |  |
| 5 | Enter **Phone number** | The system will check validation |  |  |
| 6 | Enter **Password** | The system will check validation |  |  |
| 7 | Re-enter **Password** | The system checks confirm password |  |  |
| 8 | Click **Registration** button | The system displays the **Login** page |  |  |
| 9 | Check **Post-condition 1** |  |  |  |

**Post-condition:** New student is registered and the data is saved in the database.

**Test Case #: 2**

**Test Case Name:** Student Login

**System:** Education Hub

**Short Description:** Test student login and details

**Pre-conditions:** The student has valid username and password.

The system displays the Login page.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Step** | **Action** | **Expected System Response** | **Pass/Fail** | **Comment** |
| 1 | Enter **Username** | If the username is incorrect, system displays **Enter valid username** |  |  |
| 2 | Enter **Password** | If the Password is incorrect, system displays **Enter valid password** |  |  |
| 3 | Select **Login as Student** | The system selects the student |  |  |
| 4 | Click **Login** button | The system displays the **dashboard** |  |  |

**Test Case #: 3**

**Test Case Name:** Send Request to Faculty

**System:** Education Hub

**Short Description:** Student can send request to faculty

**Pre-conditions:** The student has valid username and password.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Step** | **Action** | **Expected System Response** | **Pass/Fail** | **Comment** |
| 1 | Enter **Username** | If the username is incorrect, system displays **Enter valid username** |  |  |
| 2 | Enter **Password** | If the Password is incorrect, system displays **Enter valid password** |  |  |
| 3 | Select **Login as Student** | The system selects the student |  |  |
| 4 | Click **Login** button | The system displays the **dashboard** |  |  |
| 5 | Check **Post-condition 2** |  |  |  |
| 6 | Enter **Number** | The system checks the validation |  |  |
| 7 | Enter **Text** |  |  |  |
| 8 | Click **Send** button | The system displays **the message has sent** |  |  |
| 9 | Check **Post-condition** |  |  |  |

**Test Case #: 4**

**Test Case Name:** Faculty Registration

**System:** Education Hub

**Short Description:** Test Faculty registration

**Pre-conditions:** The Faculty has valid email-id and phone number.

The system displays the menu for registration.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Step** | **Action** | **Expected System Response** | **Pass/Fail** | **Comment** |
| 1 | Click the **Registration** button | The system displays a registration form |  |  |
| 2 | Enter **Name** | If the name already exist, system displays **Try another name** |  |  |
| 3 | Enter **Email Id** | If the email id already exist, system displays **try another email** |  |  |
| 4 | Select **Signup as Faculty** | The system displays Faculty selected |  |  |
| 5 | Enter **Phone number** | The system will check validation |  |  |
| 6 | Enter **Password** | The system will check validation |  |  |
| 7 | Re-enter **Password** | The system checks confirm password |  |  |
| 8 | Click **Registration** button | The system displays the **Login** page |  |  |
| 9 | Check **Post-condition 1** |  |  |  |

**Post-condition:** New faculty member is registered and the data is saved in the database.

**Test Case #: 5**

**Test Case Name:** Faculty Login

**System:** Education Hub

**Short Description:** Test faculty login and details

**Pre-conditions:** The faculty has valid username and password.

The system displays the Login page.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Step** | **Action** | **Expected System Response** | **Pass/Fail** | **Comment** |
| 1 | Enter **Username** | If the username is incorrect, system displays **Enter valid username** |  |  |
| 2 | Enter **Password** | If the Password is incorrect, system displays **Enter valid password** |  |  |
| 3 | Select **Login as Faculty** | The system selects the faculty |  |  |
| 4 | Click **Login** button | The system displays the **dashboard** |  |  |

**Post-condition:** Faculty is able to login now.

**Test Case #: 6**

**Test Case Name:** Response student request

**System:** Education Hub

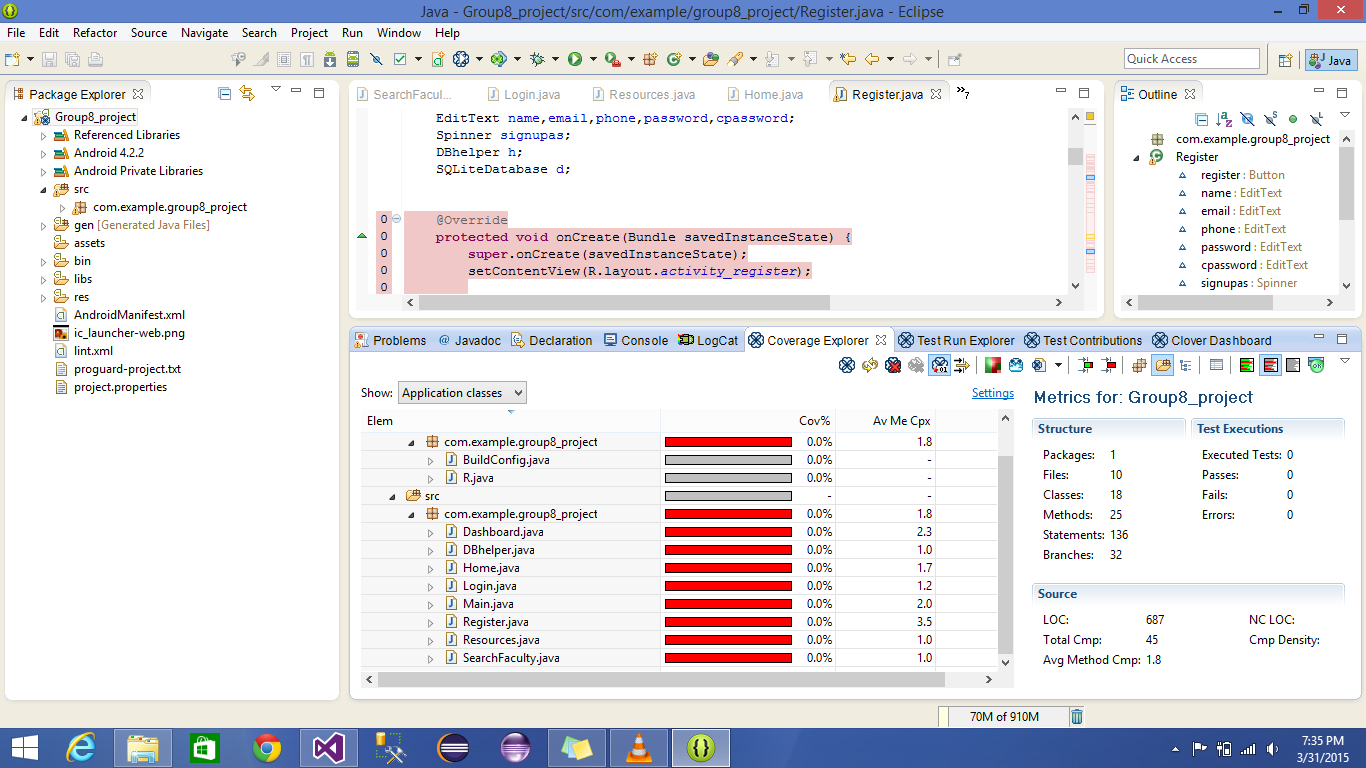
**Short Description:** Faculty can response to student request

**Pre-conditions:** The student has valid username and password.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Step** | **Action** | **Expected System Response** | **Pass/Fail** | **Comment** |
| 1 | Enter **Username** | If the username is incorrect, system displays **Enter valid username** |  |  |
| 2 | Enter **Password** | If the Password is incorrect, system displays **Enter valid password** |  |  |
| 3 | Select **Login as Faculty** | The system selects the student |  |  |
| 4 | Click **Login** button | The system displays the **dashboard** |  |  |
| 5 | Check **Post-condition 2** |  |  |  |
| 6 | Enter **Number** | The system checks the validation |  |  |
| 7 | Enter **Text** |  |  |  |
| 8 | Click **Send** button | The system displays **the message has sent** |  |  |
| 9 | Check **Post-condition** |  |  |  |

**Post-condition:** Student is able to send request to faculty and faculty give the response to student.

**Unit testing result by Clover Application tool**



**Registration page unit test**

package com.example.group8\_project;

import android.support.v7.appcompat.\*;

import android.app.Activity;

import android.content.ContentValues;

import android.content.Intent;

import android.database.Cursor;

import android.database.SQLException;

import android.database.sqlite.SQLiteDatabase;

import android.os.Bundle;

import android.view.Menu;

import android.view.MenuItem;

import android.view.View;

import android.view.View.OnClickListener;

import android.widget.Button;

import android.widget.EditText;

import android.widget.Spinner;

import android.widget.Toast;

public class Register extends Activity {

Button register;

EditText name,email,phone,password,cpassword;

Spinner signupas;

DBhelper h;

SQLiteDatabase d;

@Override[TestMethod]

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_register);

h = new DBhelper(this);

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

name = (EditText)findViewById(R.id.nameReg);

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

phone = (EditText)findViewById(R.id.phoneReg);

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

cpassword = (EditText)findViewById(R.id.conformpassReg);

signupas = (Spinner)findViewById(R.id.spinnerReg);

register.setOnClickListener(new OnClickListener() {

public void onClick(View v) {

// TODO Auto-generated method stub

String Name = name.getText().toString();

String Email = email.getText().toString();

String Phone = phone.getText().toString();

String Password = password.getText().toString();

String Cpassword = cpassword.getText().toString();

String Signupas = (String)signupas.getItemAtPosition(signupas.getSelectedItemPosition());

boolean valid\_username = false;

valid\_username=checkUser(Email);

if(Name.equalsIgnoreCase("")||Email.equalsIgnoreCase("")||Phone.equalsIgnoreCase("")||

Password.equalsIgnoreCase("")||Cpassword.equalsIgnoreCase(""))

{

Toast.makeText(getApplicationContext(), "All Fields Required.", Toast.LENGTH\_SHORT).show();

}

else if(Cpassword.equals(Password) == false)

{

cpassword.setError("Confirm password does not match");

}

else if(!Email.matches("^[\_A-Za-z0-9-]+(\\.[\_A-Za-z0-9-]+)\*@[A-Za-z0-9]+(\\.[A-Za-z0-9]+)\*(\\.[A-Za-z]{2,})$"))

{

email.setError("Pattern of email is wrong");

}

else if(Phone.length()<10)

{

phone.setError("Contact number must be of 10 digit");

}

else if(valid\_username)

{

email.setError("Username exist");

}

else {

d = h.getWritableDatabase();

ContentValues cv =new ContentValues();

cv.put(DBhelper.NAME, Name);

cv.put(DBhelper.EMAIL, Email);

cv.put(DBhelper.PASSWORD, Password);

cv.put(DBhelper.PHONE, Phone);

cv.put(DBhelper.SIGNUPAS, Signupas);

d.insert(DBhelper.SIGNUP, null, cv);

d.close();

Toast.makeText(getApplicationContext(), "Save", 9).show();

Intent i=new Intent(Register.this,Login.class);

startActivity(i);

}

}

});

}

private boolean checkUser(String Email)throws SQLException {

// TODO Auto-generated method stub

d = h.getReadableDatabase();

Cursor c= d.query(DBhelper.SIGNUP, null,DBhelper.EMAIL+"="+"'"+Email+"'", null, null, null, null);

if(c.moveToFirst()){

return true;

}

else{

return false;

}

}

@Override[TestMethod]

public boolean onCreateOptionsMenu(Menu menu) {

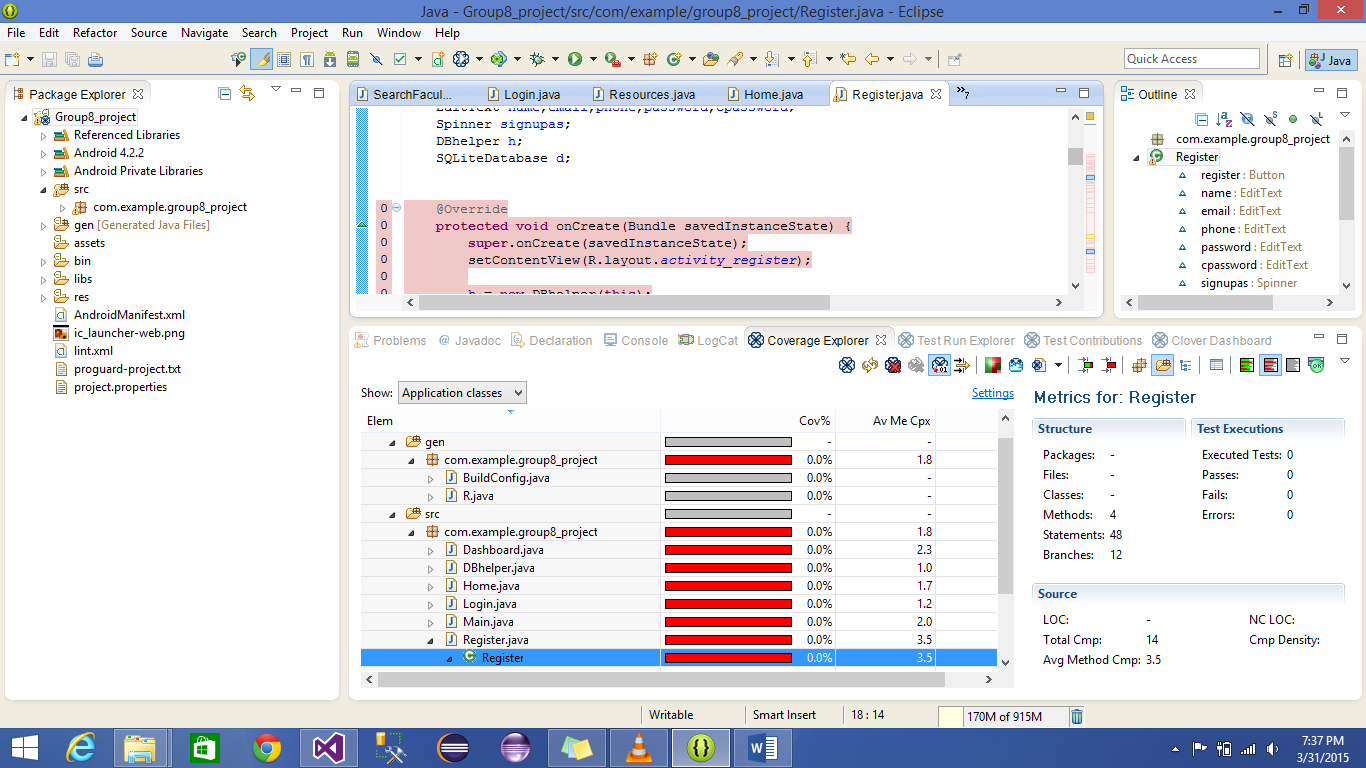
// Inflate the menu; this adds items to the action bar if it is present.

getMenuInflater().inflate(R.menu.register, menu);

return true;

}

}



**Home** **page unit test**

package com.example.group8\_project;

import android.support.v7.appcompat.\*;

import android.app.Activity;

import android.content.Intent;

import android.os.Bundle;

import android.view.Menu;

import android.view.MenuItem;

import android.view.View;

import android.view.View.OnClickListener;

import android.widget.ImageView;

import android.widget.TextView;

public class Home extends Activity implements OnClickListener {

ImageView login,signup ;

@Override[TestMethod]

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_home);

login = (ImageView)findViewById(R.id.loginhome);

signup = (ImageView)findViewById(R.id.signuphome);

login.setOnClickListener(this);

signup.setOnClickListener(this);

}

[TestMethod]

public void onClick(View v) {

// TODO Auto-generated method stub

if(v == login){

Intent i=new Intent(Home.this,Login.class);

startActivity(i);

}

if(v == signup){

Intent i=new Intent(Home.this,Register.class);

startActivity(i);

}

}

@Override[TestMethod]

public boolean onCreateOptionsMenu(Menu menu) {

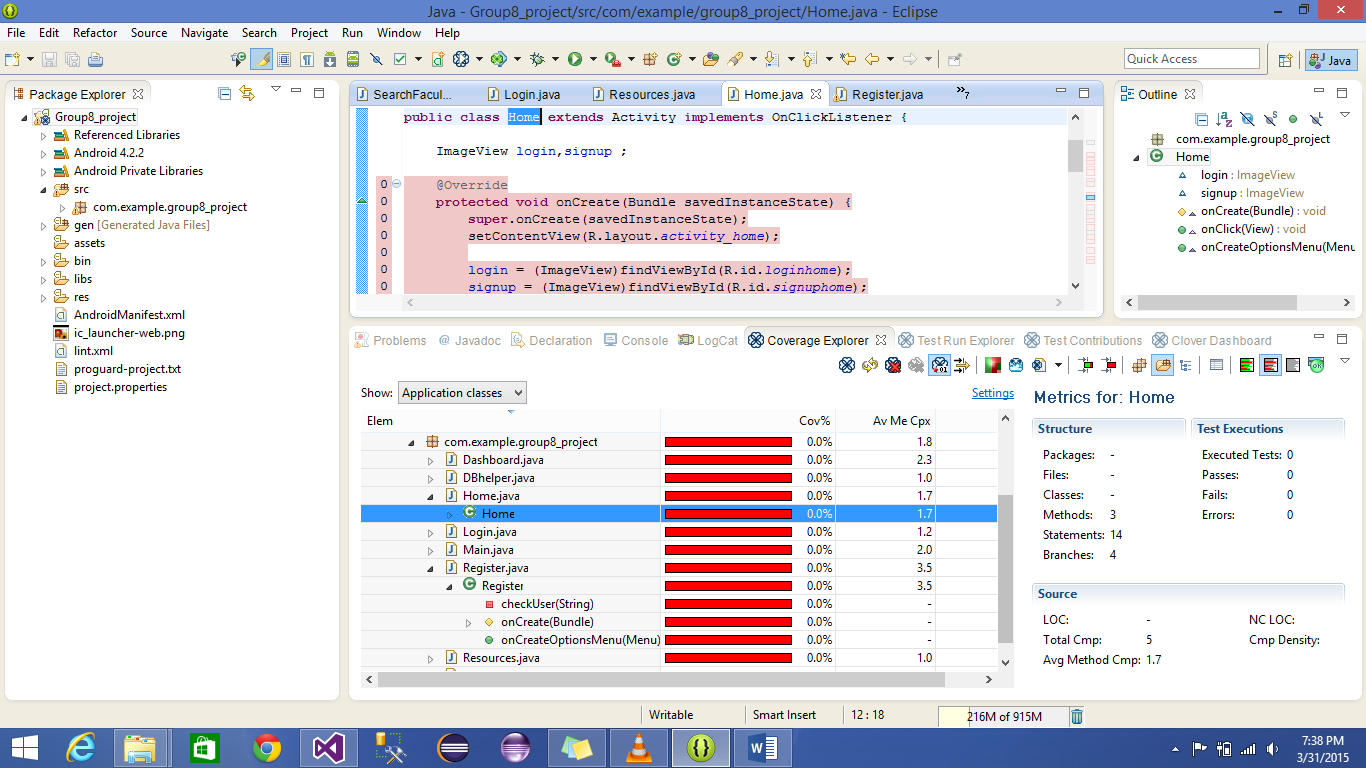
// Inflate the menu; this adds items to the action bar if it is present.

getMenuInflater().inflate(R.menu.home, menu);

return true;

}

}



**Dashboard unit test**

package com.example.group8\_project;

import android.support.v7.appcompat.\*;

import android.app.Activity;

import android.content.Intent;

import android.os.Bundle;

import android.view.Menu;

import android.view.MenuItem;

import android.view.View;

import android.view.View.OnClickListener;

import android.widget.ImageView;

[TestMethod]public class Dashboard extends Activity implements OnClickListener {

ImageView profile,faculty,resource,message,sclass;

@Override[TestMethod]

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_dashboard);

faculty = (ImageView)findViewById(R.id.dashsearchfaculty);

profile = (ImageView)findViewById(R.id.dashprofile);

resource = (ImageView)findViewById(R.id.dashresource);

message = (ImageView)findViewById(R.id.dashmessage);

sclass = (ImageView)findViewById(R.id.dashclass);

faculty.setOnClickListener(this);

profile.setOnClickListener(this);

resource.setOnClickListener(this);

message.setOnClickListener(this);

sclass.setOnClickListener(this);

}

public void onClick(View v) [TestMethod]{

// TODO Auto-generated method stub

if(v == faculty){

Intent i=new Intent(Dashboard.this,SearchFaculty.class);

startActivity(i);

}

if(v == resource){

Intent i=new Intent(Dashboard.this,Resources.class);

startActivity(i);

}

if (v== profile){

Intent i=new Intent(Dashboard.this,Profile.class);

startActivity(i);

}

if(v == message){

}

if(v == sclass){

Intent i=new Intent(Dashboard.this, ListBatchesActivity.class);

startActivity(i);

} }

@Override[TestMethod]

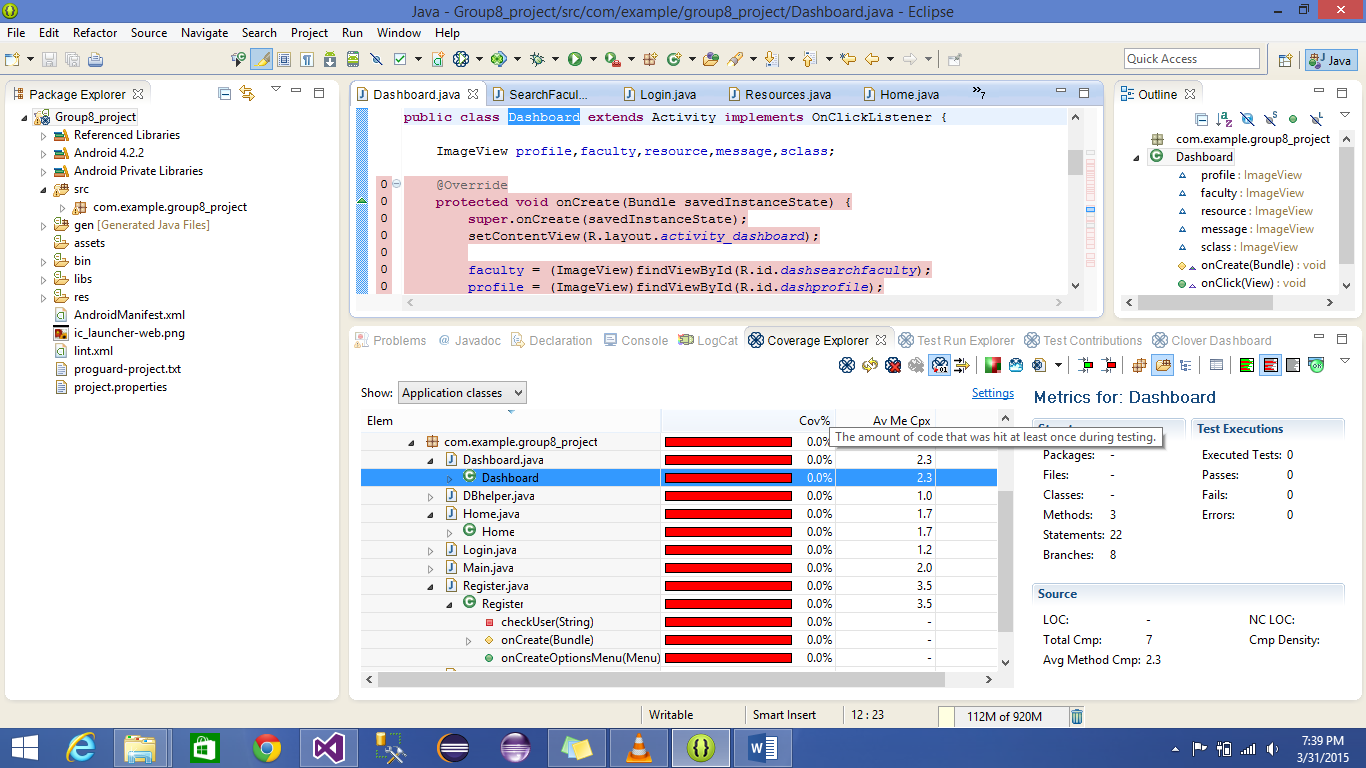
public boolean onCreateOptionsMenu(Menu menu) {

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

getMenuInflater().inflate(R.menu.dashboard, menu);

return true;

} }



**Main page unit test**

package com.example.group8\_project;

import android.support.v7.appcompat.\*;

import android.app.Activity;

import android.content.Intent;

import android.os.Bundle;

import android.view.Menu;

import android.view.MenuItem;

import android.view.View;

import android.view.View.OnClickListener;

import android.widget.Button;

public class Main extends Activity implements OnClickListener [TestMethod]

{

Button login, home, register;

@Override[TestMethod]

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_main);

login=(Button)findViewById(R.id.login1);

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

register=(Button)findViewById(R.id.register1);

login.setOnClickListener(this);

home.setOnClickListener(this);

register.setOnClickListener(this);

}

public void onClick(View v) [TestMethod]

{

// TODO Auto-generated method stub

if(v == login){

Intent i=new Intent(Main.this,Login.class);

startActivity(i);

}

if(v == home){

Intent i=new Intent(Main.this,Home.class);

startActivity(i);

}

if(v == register){

Intent i=new Intent(Main.this,Register.class);

startActivity(i);

}

}

@Override[TestMethod]

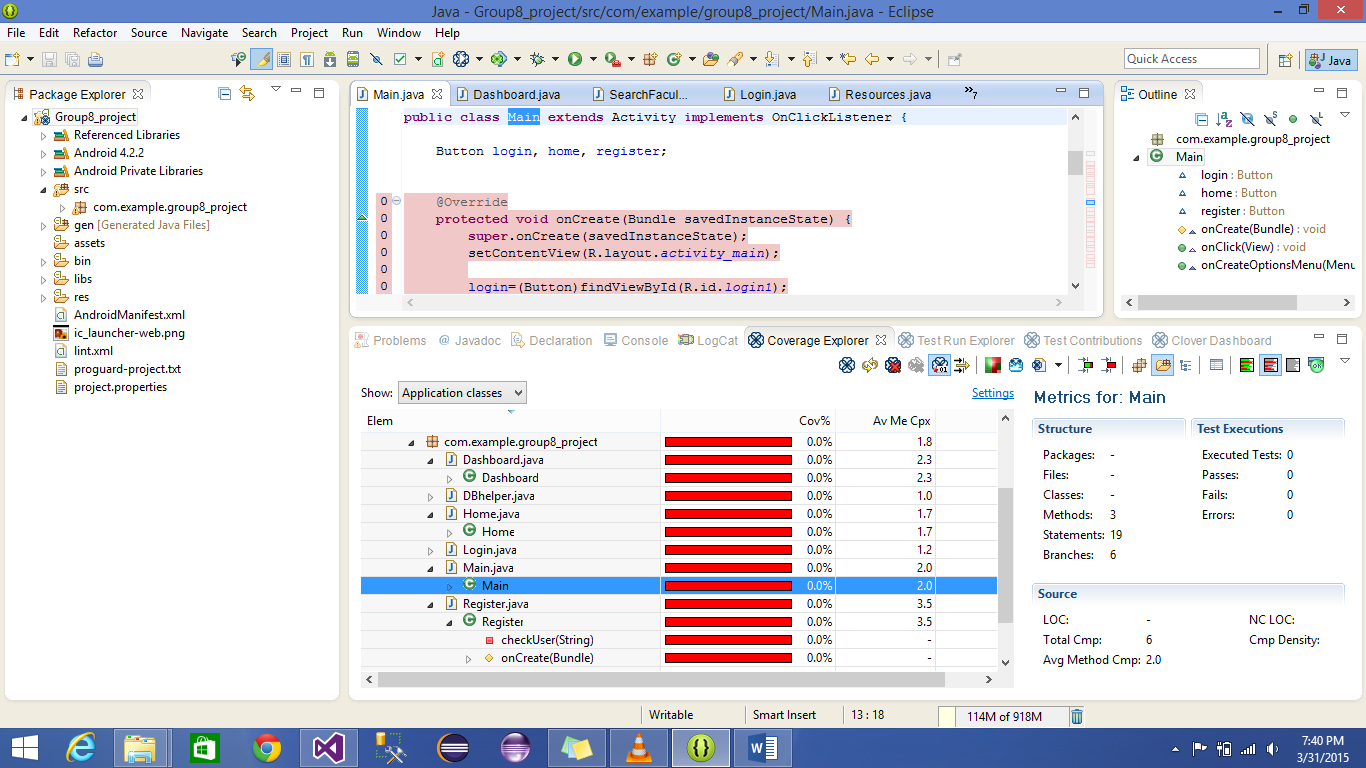
public boolean onCreateOptionsMenu(Menu menu) {

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

getMenuInflater().inflate(R.menu.main, menu);

return true;

}}



**Search faculty test case**

package com.example.group8\_project;

import android.support.v7.appcompat.\*;

import android.app.Activity;

import android.os.Bundle;

import android.view.Menu;

import android.view.MenuItem;

public class SearchFaculty extends Activity {

@Override[TestMethod]

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_search\_faculty);

}

@Override[TestMethod]

public boolean onCreateOptionsMenu(Menu menu) {

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

getMenuInflater().inflate(R.menu.search\_faculty, menu);

return true;

}

}



**Resources unit test**

package com.example.group8\_project;

import android.support.v7.appcompat.\*;

import android.app.Activity;

import android.os.Bundle;

import android.view.Menu;

import android.view.MenuItem;

public class Resources extends Activity {

@Override[TestMethod]

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_resources);

}

@Override[TestMethod]

public boolean onCreateOptionsMenu(Menu menu) {

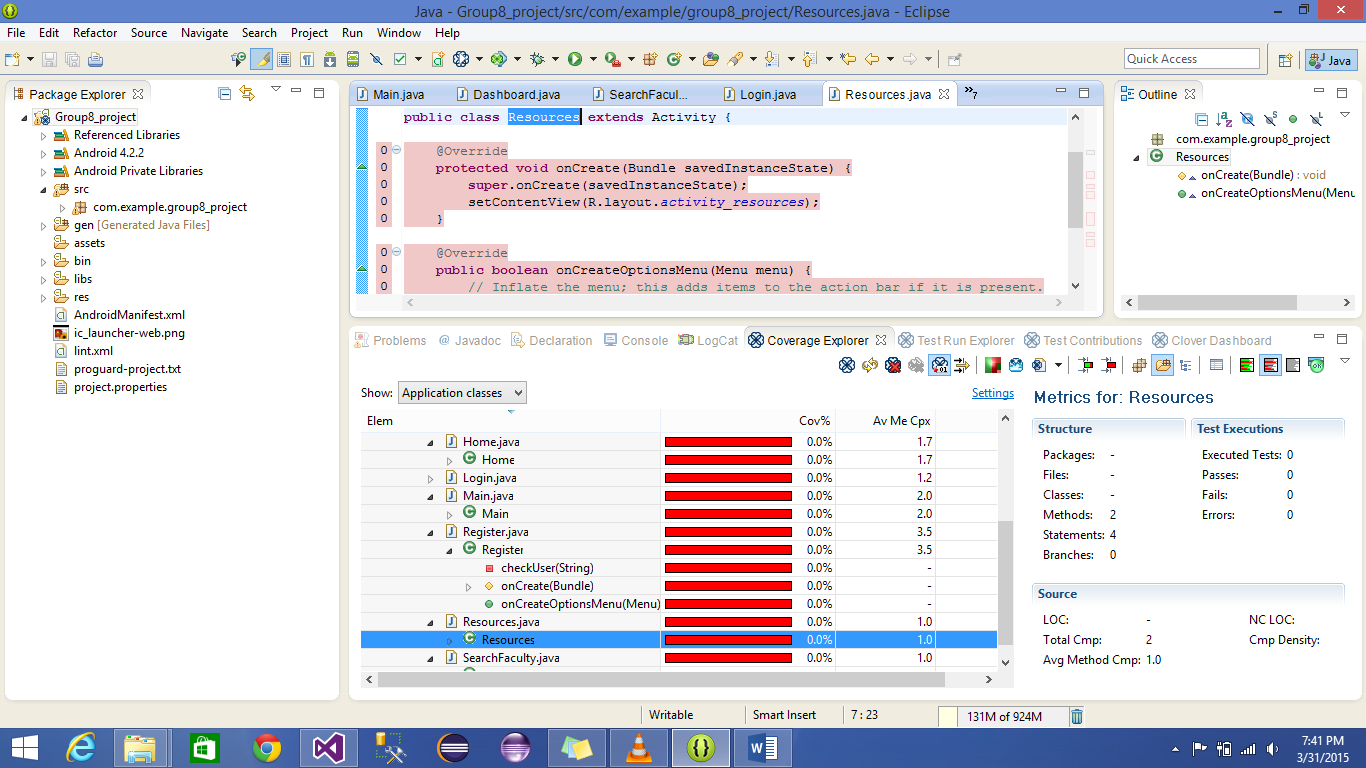
// Inflate the menu; this adds items to the action bar if it is present.

getMenuInflater().inflate(R.menu.resources, menu);

return true;

}

}



**Login page unit test**

package com.example.group8\_project;

import android.support.v7.appcompat.\*;

import android.app.Activity;

import android.content.Intent;

import android.database.Cursor;

import android.database.sqlite.SQLiteDatabase;

import android.os.Bundle;

import android.view.Menu;

import android.view.MenuItem;

import android.view.View;

import android.view.View.OnClickListener;

import android.widget.AdapterView;

import android.widget.Button;

import android.widget.EditText;

import android.widget.Spinner;

import android.widget.Toast;

import android.widget.AdapterView.OnItemSelectedListener;

public class Login extends Activity implements OnClickListener {

Button login;

EditText uname,password;

Spinner sp;

String s1,s2,s3,s4,s5,st;

DBhelper h;

SQLiteDatabase d;

@Override[TestMethod]

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_login);

h = new DBhelper(this);

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

uname = (EditText)findViewById(R.id.usernameLogin);

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

sp = (Spinner)findViewById(R.id.spinnerLogin);

sp.setOnItemSelectedListener(new OnItemSelectedListener() {

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

int arg2, long arg3) {

// TODO Auto-generated method stub

st = arg0.getItemAtPosition(arg2).toString();

}

public void onNothingSelected(AdapterView<?> arg0) {

// TODO Auto-generated method stub

}

});

login.setOnClickListener(this);

}

public void onClick(View v) [TestMethod]

{

// TODO Auto-generated method stub

if(v == login){

Intent i=new Intent(Login.this, Dashboard.class);

startActivity(i);

}

}

@Override[TestMethod]

public boolean onCreateOptionsMenu(Menu menu) {

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

getMenuInflater().inflate(R.menu.login, menu);

return true;

}

}

