1. INTRODUCTION

1.1 Overview of the project

The knowledge sharing and acquiring community application provides great opportunity for students and experts (students / teachers / professionals) to share their expertise and for acquiring knowledge in the single platform by using android application with network (by using internet). Here the community can be described as group of people such as students, teachers, professors, experts, professionals, etc,. There are many Question and Answer community platforms for sharing and acquiring knowledge, but the main purpose developing this project is to provide expert role for a student for answering the questions of other users (mainly students). By this the knowledge which has gained by the student is going to be useful for sharing to others, by this he / she can gain several factors like confidence, stability, performance over the subject will be enhanced.

For increasing response time of getting answers we are introducing In-app notifications concept which is an expert got intimated through In-app notification if a student posts any question related to the stream which had by the expert. Due to this functionality the student will get immediate response from the expert. This is a native application which is useful especially for under-graduate students of any college or university.

1.1 Existing System

There are many online forums, question and answer communities where students can share and acquire knowledge regarding to education, politics, emerging technologies, ..etc. But coming to education point of view, these forums and Q & A communities are not providing efficient service to under graduate level students. Most of these platforms are providing their services through websites.

1.2 Proposed System

We are developing a knowledge driven community application for this online platform which facilitates collaboration among people. We have chosen Android Platform for the development of this application. We are designing the application in such a way that every individual can share and acquire knowledge (in terms of education) effectively. Here new users can play, which means probability of getting response to the users is more. Here any individual can engage the role of student and subject matter expert (**SME/teacher**) simultaneously based on the circumstances.

2. LITERATURE SURVEY

2.1 Problem Statement

The main reason for developing this particular application is to make effective learning and teaching between people in a single platform. There are many Question and Answer communities where student can acquire knowledge. But the problem with the previous system is, a student can't find any subjective-tags i.e, particular subject from particular stream for posting a question. Due to this under-graduate level students are facing difficulties while posting questions. Current system is lack in terms of notifications like **In-app notifications**, **SMS**, ..etc. The answers which were posted by experts are difficult to understand for students without proper explanation in terms of design, ..etc.

2.2 Problem Solution

Here, we are providing particular streams and their subjective tags where any individual can select them before posting a question. We are providing real-time notifications like In-app notifications while mobile is connected to the internet for increasing the efficiency of getting answers in-time.

If an individual posted a question in a particular subject, then immediately it will notify to the expert through In-app notifications whose expertisation related to the question was asked. In the same way, if the question was answered by an expert, then it will notify to the student who has asked that particular question through In-app notifications. Here, we are giving a chance for the student/individual to play the role of an expert in particular subjects which they are expertised.

2.3 Survey

Literature review is a text of a scholarly paper, which includes the current knowledge including substantive findings, as well as theoretical and methodological contributions to a particular logic. It consists of ideas, theories and examples which are taken from outside world for the development of project. Literature reviews are secondary sources, and do not report new or original experimental work. These are a basis for research in nearly every academic field.

Community based question and answering (CQA) sites and applications achieve knowledge sharing among community users through a participatory platform where users can ask and answer questions for each other (Shah, et al., 2009). Since the first CQA site appeared in Korea in 2002, these services have been developing at a very fast pace in many parts of the world.

For example, Yahoo! Answers, as by far the most widely used CQA site, had a reported 62 million unique visitors per month in the United States alone in 2010 (Gazan, 2011). As iResearch's latest survey results show (iResearch), the total number of user accesses on Chinese CQA platforms in April 2010 was more than 2.26 billion times, and it is 1.4 times larger than that of the same period of the year before. All these demonstrate that CQA services have expanded to meet a wide range of people's information needs, and more importantly, CQA services have been viewed as rapidly developing social collaboration platforms (Shah, et al., 2009).

Community based Q & A sites and applications

Community driven Question Answering (CQA) websites like Stack Overflow, Quora and Yahoo! Answers are popular contemporary genre of websites on the Internet. CQA websites follow a standard Q&A format where a user asks a question on a problem she faces; while other users (who may have some prior expertise) respond with their answers on the question. Effectively, CQA websites follow a crowd sourced model in which the knowledge of experts is exploited to form a large scale knowledge base on variety of topics. Stack Exchange is a platform which provides libraries to deploy topic-based community powered Q&A websites.

Stack Exchange is a growing network of thematic question-answering websites with each website dedicated to a specific field of expertise. It consists of 107 CQA websites with 4.1M users, 7.3M questions, 13.2 M answers and 8.5M visits per day.1 CQA websites on Stack Exchange span across different orthogonal themes like Technology (Web Applications, Game Development), Culture (Travel, Christianity), Arts (Photograph, Scientific Fiction) and Sciences (Mathematics, Physics).
Stack Overflow is a programming based CQA and the most popular Stack Exchange website consisting of 7.1M questions, 12M answers and 2M+ registered users on its website.

Got ideas by doing research on Stack Overflow

Stack Overflow is the first and most popular Stack Exchange website which caters to the benefits of professional programmers and programming enthusiasts. It is a free and open Q&A website where users can ask programming related questions. Stack Overflow maintains a strong emphasis on question-answer based format of the site and strongly discourages discussion or chit-chat. The community strongly discourages questions which could generate chit-chat, opinions, polls etc. and employs elected moderators to ensure content quality maintenance.

Stack Overflow also encourages question askers to post details about the problem they face and recommend that questions contain source code. In addition, it is encouraged to mention details about previously tried but unsuccessful solutions along with the actual question. In particular, questions on the topics which contain specific programming problems, software algorithms, coding techniques and software development tools are recommended and considered fit for its Q&A format. An intricate community based voting process is followed to reward users for good quality questions and answers. Relevant, technically challenging and good question-answers are rewarded by the community with votes. Similarly, answers which address the problem encountered by the original question can be voted accepted. This voting process allows post owners to earn a reputation which is a reflection of their contribution worth to the Stack Overflow community.

Conversely, the same voting process can lead to penalties on the post owner’s reputation due to low quality posts like wrong answers, spam and advertisements. Badges (the online equivalent of medals) are awarded to users as incentives to highlight special achievements based on community participation. This community based reputation reward process helps to ensure a reasonable degree of high quality content on the website and weed out low quality content. Stack Overflow is a free, open website to all users and therefore, maintenance of content quality on such a large scale social collaborative platform is a challenge. In this thesis, we specifically look into the aspect of question quality on Stack Overflow.

Why study Stack Overflow?

Stack Overflow is a popular collaborative Q&A website used by programmers all over the world to seek answers to programming related questions. Stack Overflow is a free and open website and has 2M+ registered users with 7.1M questions and 12M answers. The underlying theme of Stack Overflow is programming-related topics and the target audience are software developers, maintenance professionals and programmers. Besides being a question-answer website, Stack Overflow has evolved into a knowledge base for programming related tasks. Therefore, Stack Overflow has attracted increasing attention from different research communities like software engineering, human computer interaction, social computing and data mining.

Researchers have mined the Stack Overflow knowledge-base to extract unique insights for various core and ancillary programming tasks like building crowd sourced API documentation, API usage obstacles, innovation diffusion via URL link sharing, mobile development issues, improvement of bug tracking systems and programming topic trends. Nasehi et al. analyze questions on Stack Overflow to understand the quality of a code example. They find nine attributes of good questions like concise code, links to extra resources and inline documentation. Wang and Godfrey analyze iOS and Android developer questions on Stack Overflow to detect API usage obstacles. They used topic models to find a set API classes on iOS and Android documentation which were difficult for developers to understand.

Asaduzzaman et al. analyze unanswered questions on Stack Overflow and use a machine learning classifier to predict such questions. They observe certain characteristics of unanswered questions on Stack Overflow and use a machine learning classifier to predict such questions. They observe certain characteristics of unanswered questions which include vagueness, homework questions etc. Allamanis and Sutton perform a topic modeling analysis on Stack Overflow questions to combine topics, types and code. They find that programming languages are a mixture of concepts and questions on Stack Overflow are concerned with the code example rather than the application domain. Stack Overflow has made its entire data publicly available for research purposes. Considering the structure, content and availability of data Stack Overflow is a very good potential testbed to perform our experiments.

2.4 Technologies used

Android

JAVA

PHP

MySQL

Android

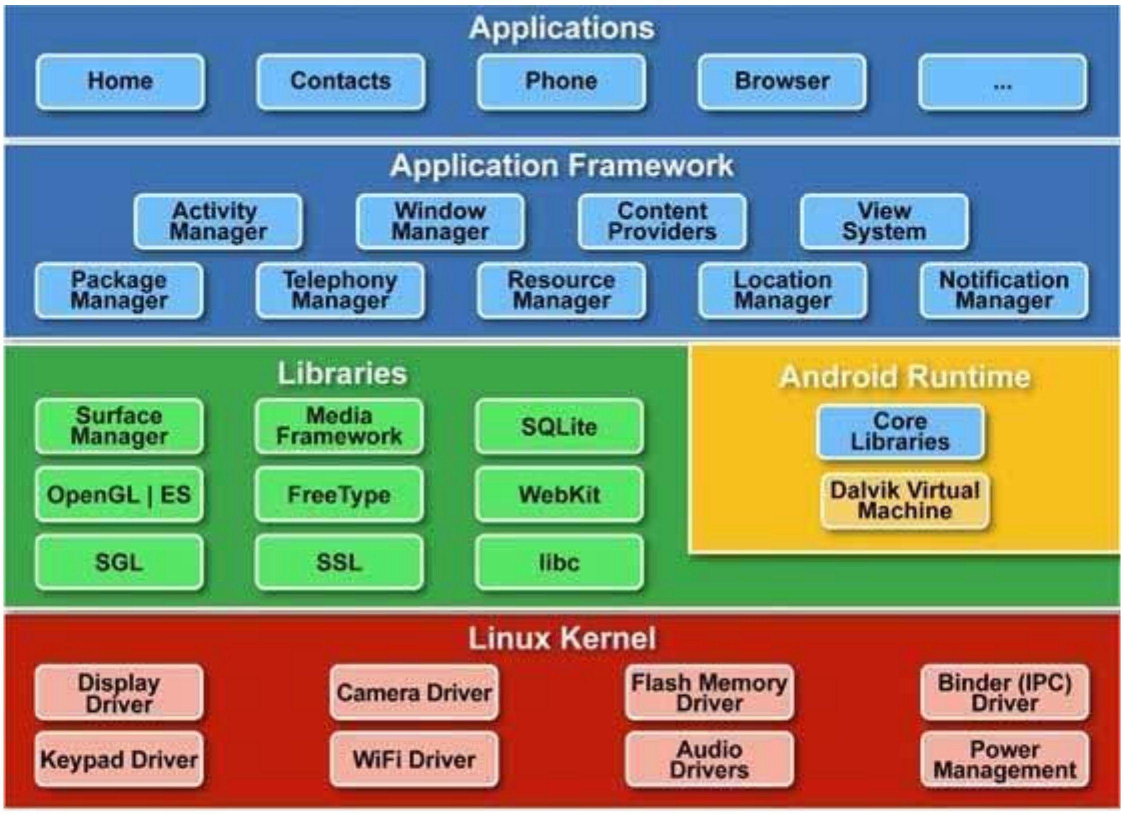
Android is a mobile operating system developed by open handset alliance led by Google. Android is based on modified version of the Linux-kernel and primarily for touch screen mobile devices such as smart phones and tablets. It is completely open-source operating system. The code is freely available and can be modified by several developers as needed to create custom mobile solutions.

Andy Rubin founded Android Incorporation in Palo Alto, California in October, 2003. Google acquired Android on 17th August 2005. Since then it is a subsidiary of Google Incorporation. Google formed Open Handset alliance (OHA) on 5th November 2007. The Open Handset Alliance is a group of 84 companies who have come together to accelerate innovation in mobile and offer consumers a richer, less expensive, and better mobile experience.

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 develop applications on the Android platform using the Java programming language.

Some of the important features of Android Studio are Gradle-based build support, Android-specific refactoring and quick fixes, Lint tools to catch performance, usability, version compatibility and other problems, Support for building Android Wear Apps and Android Virtual Device (Emulator) to run and debug Apps in Android Studio.

Android Software Stack



Why Android?

1. Open Source
2. Larger Developer and Community Reach
3. Increased Marketing
4. Inter App Integration
5. Reduced Cost of Development
6. Higher Success Ratio
7. Rich Development environment

JAVA

Java technology is both a programming language and an application software development platform.

Java as programming language

Java is a general-purpose computer programming language that is concurrent, class-based and object-oriented. It is intended to let application developers **“Write Once Run Anywhere”** (WORA), i.e, the compiled Java code can run on all platforms that support java without need for recompilation. Hence, Java is Platform Independent language. Java applications are typically compiled to bytecode that can run on any JVM (Java Virtual Machine) regardless of computer architecture.

Java as software platform

Java is a set of computer software and specifications developed by sun microsystems, which was later acquired by the oracle corporation that provides a system for developing application software and deploying it in cross-platform computing environment. The most common types of programs written in Java are applets (which can be embedded in HTML).

PHP

PHP- Hypertext preprocessor, formerly known as personal homepage. It is a server-side scripting language and a powerful tool for making dynamic and interactive web-pages. It is used as gateway between application and the database of the server. It is a widely used, open source scripting language. PHP scripts are executed on the server. PHP can generate dynamic page content. PHP can perform CRUD operations on the server. PHP can collect form data. It can send and receive cookies. It can add, delete, modify data in your database. It can be used to control user-access. It can encrypt data.

MySQL

MySQL is an open-source relational database management system. Many applications are using MySQL database. MySQL is one of the best RDBMS being used for developing web-based software applications. MySQL is a database system that runs on a server. MySQL uses standard SQL. The data in a MySQL database is stored in table format. A table is a collection of columns and rows.

MySQL Database

MySQL is released under an open-source license. So you have nothing to pay to use it. It is a very powerful program in its own right. It handles a large subset of the functionality of the most expensive and powerful database packages. It uses a standard form of the well-known SQL data language. It works on many operating systems and with many languages including PHP, PERL, C, C++, JAVA, etc. It works very quickly and works well even with large data sets. It is very friendly with PHP, the most appreciated language for web development. It supports large databases, up to 50 million rows or more in a table. The default file size limit for a table is 4GB. It is customizable. The open-source GPL license allows programmers to modify the MySQL software to fit their own specific environments.

3. SYSTEM ANALYSIS

System analysis is the process of studying a procedure or business in order to identify its goals and purposes and create systems and procedures that will achieve them in an efficient way. Another view sees system analysis as a problem-solving technique that breaks down a system into its component pieces for the purpose of the studying how well those component parts work and interact to accomplish their purpose. The field of system analysis relates closely to requirements analysis or to operations research. It is also "an explicit formal inquiry carried out to help a decision maker identify a better course of action and make a better decision than she might otherwise have made.

3.1 Feasibility Study

A project feasibility study is a comprehensive report that examines in detail analysis of a given project. It also takes into consideration of Plan, Processes, People and Power which are used in development of the project. The goal is to determine whether the project should go ahead, be redesigned, or else abandoned altogether. Our project is feasible to develop based on android platform.

Feasibility study is based on economic, technological, legal and scheduling factors:

Technical feasibility.

Legal feasibility.

Operational feasibility.

Schedule feasibility.

Resource feasibility.

Technical Feasibility

This assessment is based on an outline design of system requirements to determine whether the company has the technical expertise to handle completion of the project. Our team has the technical expertise in JAVA, PHP, MySQL..etc., for the development of this project.

Legal Feasibility

Legal feasibility determines whether the proposed system conflicts with legal requirements, in terms of providing security. In this project we are providing login id and password for the users for secure login. We are using active and automatic moderators for controlling wrongly misplaced words / statements.

Operational Feasibility

Operational feasibility is the measure of how well a proposed system solves the problem and how it satisfies the requirements identified in the requirement analysis phase of system development. As discussed in the problem solution (1.5), we are providing subjective-tags, real notifications through SMS..etc., for solving the problems in existed system.

Schedule Feasibility

A project will fail if it takes too long to be completed before it is useful. Typically this means estimating how long the system will take to develop, and if it can be completed in a given time period using some methods like payback period. Our project will take 3 months approximately for development.

Resource feasibility

Resource feasibility describes how much time is available to build the new system, type and amount of resources required and some developmental procedures. We have efficient resources in terms of hardware and software like computer, mobile and Android Studio, MySQL..etc., respectively.

3.2 Software Requirements Specifications

A SRS is a description of a software system to be developed. It lays out functional and non-functional requirements and may include a set of use cases that describe user interactions that the software must provide. SRS establishes the basis for an agreement between customers and suppliers on what the software product is to do as well as it is not expected to do.

3.2.1 Functional Requirements:

Functional Requirements defines the functions of a system. A function is described as a set of inputs, the behavior, and outputs.

Functional requirements are categorized into 3 parts:

Input

Behavior

Output

Input:

Here users can post questions by selecting particular streams and their subjects.

Behavior:

Firstly, the posted question is saved in the database. Secondly, it redirects the question to the expert related to subject which the question is asked.

Output:

The expert posts the answers to the questions asked by the user.

3.2.2 Non-Functional Requirements:

Non-functional requirements specifies criteria that can be used to judge the operation of a system, rather than specific behaviors. It imposes constraints on the design. Some of our project are given below:

Accessibility

Portability

Performance

Privacy

Security

Accessibility:

There is need of data connection or internet. Since this app is used and useful mostly in outdoor environment. By connecting mobile to the internet user can access the app and uses the functionalities of the app.

Portability:

The app works on all versions of android.

Performance:

The expert will get instance response in the form of message after posting a question by the student and for posting an answer vice-versa. For getting messages, we are using **Google Firebase Cloud Messaging (FCM)** service for intimation of messages to the users (students and experts). By this, performance of app will be increased in terms of response time.

Privacy:

We are giving privacy by providing login id and password for every user.

Security:

We are providing security to the database by using authentication key and the key is only known to administrator.

3.3 System Requirements Specifications

3.3.1 Software Requirements (Minimal)

Operating System: Windows XP

Technologies: JAVA, PHP, Android

IDE: Android Studio

Database: MySQL

Toolkit: Android mobile phone

Front-end: XML (for setting layout of the app)

Back-end: JAVA, PHP, MySQL

3.3.2 Hardware Requirements (Minimal)

Processor: Pentium

Speed: 1.8GHz

RAM: 4GB

Hard Disk: 40GB for Android studio tools (SDK’s, API’s….)

4. SYSTEM DESIGN

4.1 Introduction to UML Diagrams

System design is the process of defining the architecture, modules, interfaces, and data for a system to satisfy specified requirements. We can use UML diagrams for describing the logical and physical design of a system.

There are two types of UML diagrams:

1. Structural diagrams
2. Behavioral diagrams

4.2 UML Diagrams

Structural diagram:

Structural diagram show the **static structure** of the system and its parts on different abstraction and implementation **levels** and how they are related to each other. The elements in a structure diagram represent the meaningful concepts of a system, and may include abstract, real world and implementation concepts.

Types of structural diagrams:

1. Class diagram
2. Component diagram
3. Deployment diagram
4. Object diagram etc,.

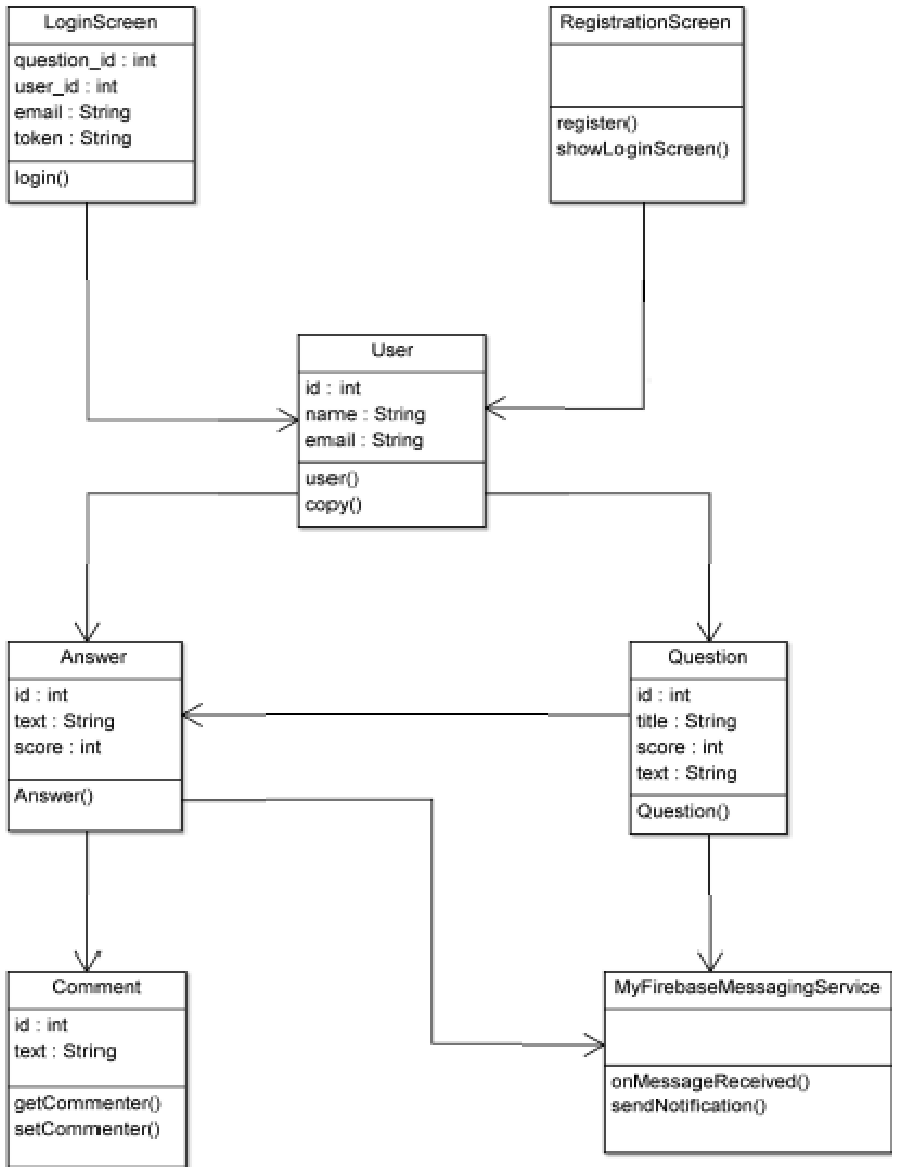
Behavioral diagram:

Behavioral diagram show the **dynamic behavior** of the objects in a system, which can be described as a series of changes to the system over time.

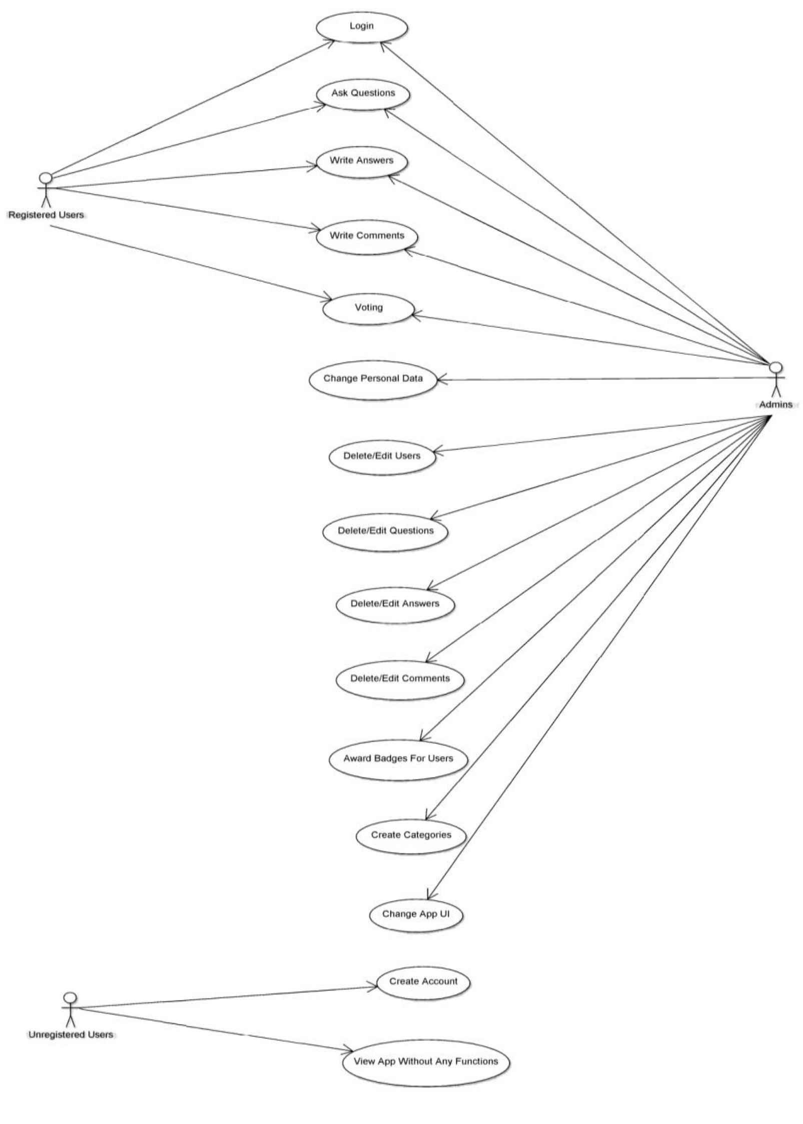
Types of behavioral diagrams:

1. Use-case diagram
2. Activity diagram
3. Sequence diagram
4. State-chart diagram etc,.

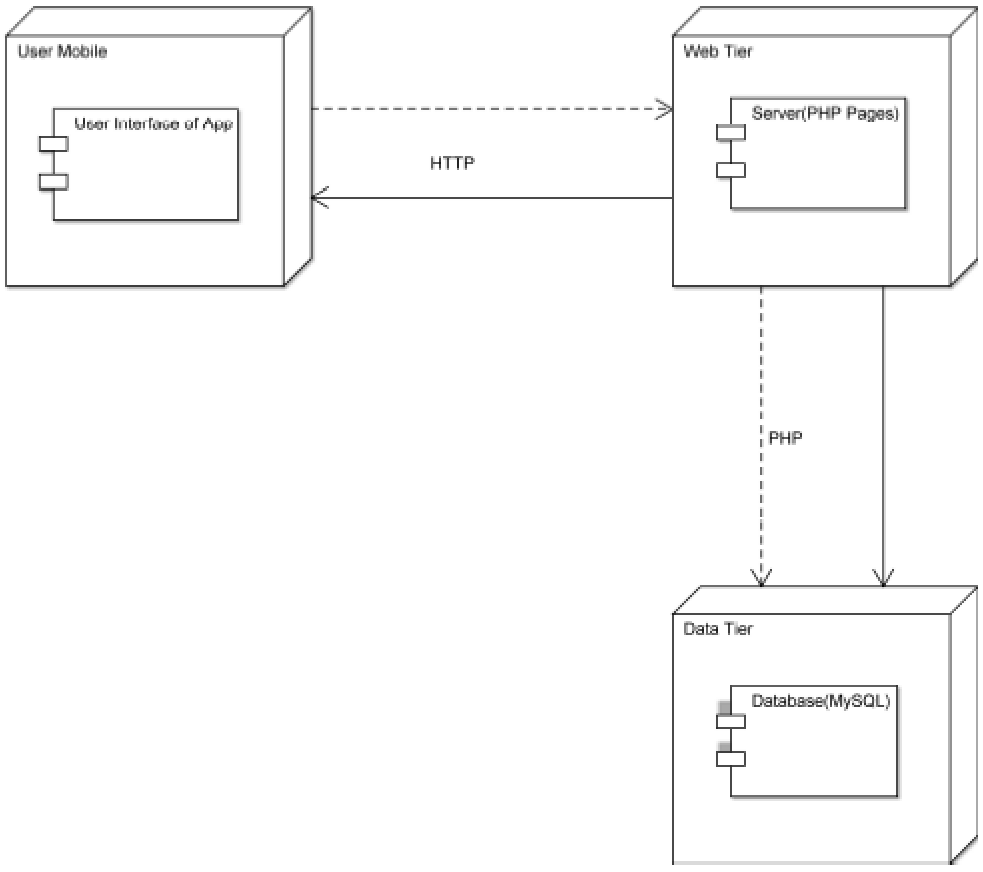
4.1.1 Class Diagram



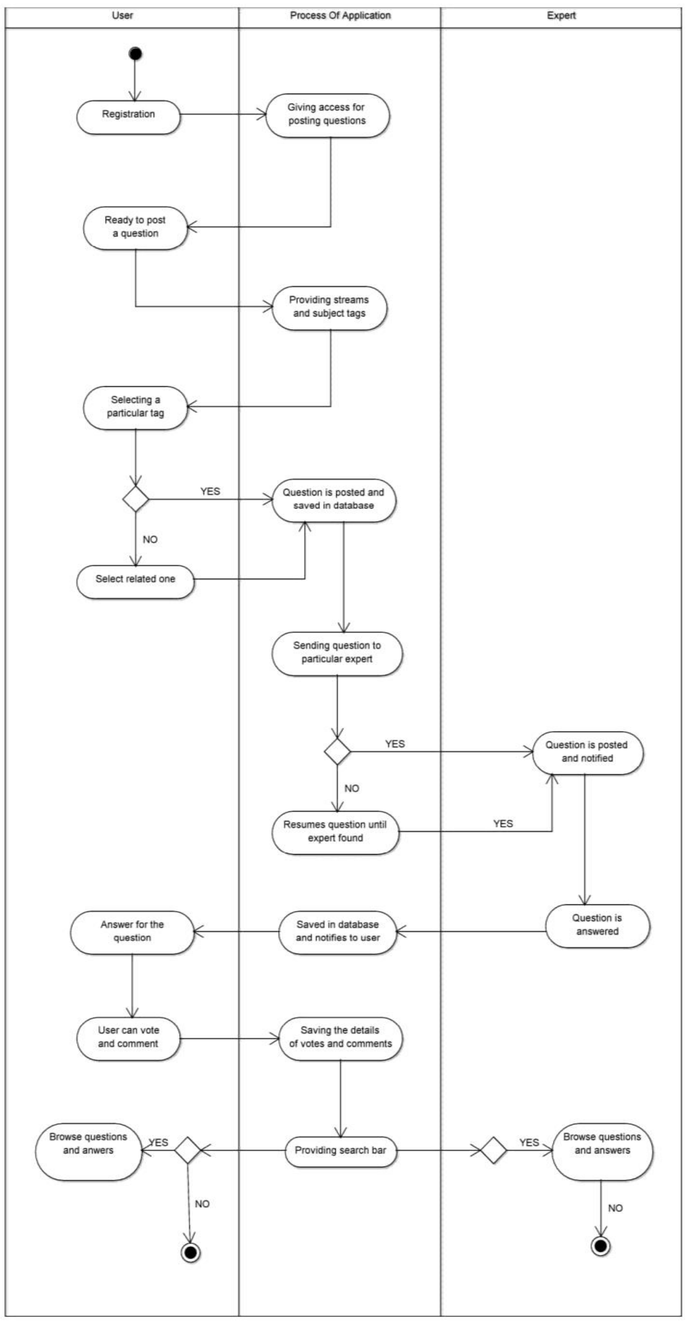
4.1.2 Use-Case Diagram



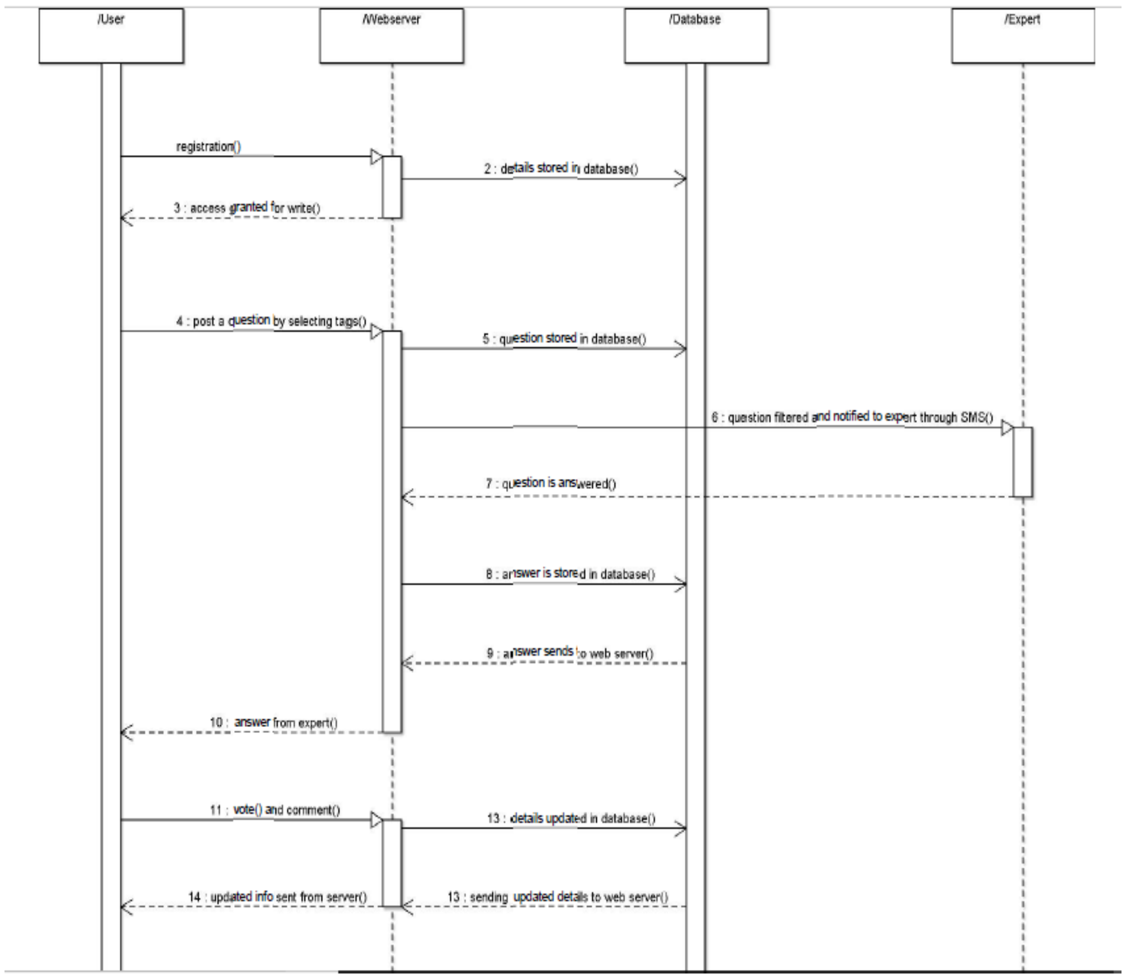
4.1.3 Deployment Diagram



4.1.4 Activity Diagram


User
Process Of Application
Expert
Registration
Giving access for posting questions
Ready to post a question
Providing streams and subject tags
Selecting a particular tag
YES
NO
Question is posted and saved in database
Select related one
Sending question to particular expert
YES
NO
Question is posted and notified
Resumes question until expert found
YES
Question is answered
Saved in database and notifies to user
Answer for the question
User can vote and comment
Saving the details of votes and comments
Providing search bar
Browse questions and anwers
YES
NO
YES
Browse questions and answers
NO

4.1.5 Sequence Diagram


/User
/Webserver
/Database
/Expert
registration()
2 : details stored in database()
3 : access granted for write()
4 : post a question by selecting tags()
5 : question stored in database()
6 : question filtered and notified to expert through SMS()
7 : question is answered()
8 : answer is stored in database()
9 : answer sends to web server()
10 : answer from expert()
11 : vote() and comment()
13 : details updated in database()
14 : updated info sent from server()
13 : sending updated details to web server()

5. IMPLEMENTATION

5.1 Source Code

Android Manifest file

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="learn.teach.QnA"
    android:versionCode="2">

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

    <application
        android:name="learn.teach.QnA.App"
        android:allowBackup="true"
        android:icon="@drawable/qnalogo"
        android:label="@string/app_name"
        android:supportsRtl="true"
        android:theme="@style/MyMaterialTheme">
        <activity
            android:name="learn.teach.QnA.LoginScreen"
            android:screenOrientation="portrait">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>

        </activity>
        <activity
            android:name="learn.teach.QnA.RegistrationScreen"
            android:screenOrientation="portrait" />
        <activity android:name="learn.teach.QnA.MenuActivity" />
        <activity android:name="learn.teach.QnA.QuestionsActivity" >

        </activity>
        <activity android:name="learn.teach.QnA.LogoutActivity" />
        <activity android:name="learn.teach.QnA.QuestionDetailsActivity" />
        <activity android:name="learn.teach.QnA.NewQuestion" />
        <activity android:name="learn.teach.QnA.EditQuestionActivity" />
        <activity android:name="learn.teach.QnA.EditAnswerActivity" />
        <activity android:name="learn.teach.QnA.AnswerDetails" />

        <service android:name="learn.teach.QnA.MyFirebaseMessagingService">
            <intent-filter>
                <action android:name="com.google.firebase.MESSAGING_EVENT"
/>
            </intent-filter>
        </service>

        <activity android:name="learn.teach.QnA.ExpertActivity" />

        <!--
     Set custom default icon. This is used when no icon is set for incoming notification
messages.
     See README(https://goo.gl/l4GJaQ) for more.
        -->
        <meta-data

android:name="com.google.firebase.messaging.default_notification_icon"
            android:resource="@drawable/qnalogo" />
        <!--
             Set color used with incoming notification messages. This is used when no
color is set for the incoming
             notification message. See README(https://goo.gl/6BKBk7) for more.
        -->
        <meta-data

android:name="com.google.firebase.messaging.default_notification_color"
            android:resource="@color/colorAccent" />

        <activity android:name="learn.teach.QnA.QuestionsByMe" />
        <activity android:name="learn.teach.QnA.ProfileActivity"></activity>

    </application>
    </manifest>
```

Java Files

LoginScreen.java

```
package learn.teach.QnA;

import android.app.Activity;
import android.content.Context;
import android.content.Intent;
import android.database.Cursor;
import android.database.sqlite.SQLiteDatabase;
import android.net.ConnectivityManager;
import android.net.NetworkInfo;
import android.os.Bundle;
import android.os.StrictMode;
import android.support.v7.app.AppCompatActivity;
import android.util.Log;
import android.view.View;
import android.widget.EditText;
import android.widget.ProgressBar;
import android.widget.TextView;
import android.widget.Toast;
import com.google.firebase.messaging.FirebaseMessaging;

import java.io.InputStream;
import java.util.HashMap;

public class LoginScreen extends AppCompatActivity {
    final String TAG = "LoginScreen";
    SQLiteDatabase database;
    String email = "";
    String token = "";
    String status, category, subcategory;
    int question_id;
    int user_id;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        if (getIntent().getExtras() != null) {
            for (String key : getIntent().getExtras().keySet()) {
                String value = getIntent().getExtras().getString(key);
                Log.d(TAG, "Key: " + key + " Value: " + value);
            }
        }
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_login_screen);


        int SDK_INT = android.os.Build.VERSION.SDK_INT;
        if (SDK_INT > 8) {
            StrictMode.ThreadPolicy policy = new StrictMode.ThreadPolicy.Builder()
                    .permitAll().build();
            StrictMode.setThreadPolicy(policy);
        }
        final TextView text = (TextView) findViewById(R.id.loginStatus);
        database = getBaseContext().openOrCreateDatabase("sqlite-QnA.db",
MODE_PRIVATE, null);
        //verifyTokenIfPresent();
        Thread thread = new Thread(new Runnable() {

            public void run() {
                String textToSetForView = "";
                if (isConnected()) {
                    textToSetForView = "Connected";
                } else {
                    textToSetForView = "No Internet Connection";
                }
                final String textVal = textToSetForView;
                runOnUiThread(new Runnable() {

                    public void run() {
                        text.setText(textVal);
                    }
                });
            }
        });
        thread.start();
    }

    public boolean isConnected() {
        ConnectivityManager connMgr = (ConnectivityManager)
getSystemService(Activity.CONNECTIVITY_SERVICE);
        NetworkInfo networkInfo = connMgr.getActiveNetworkInfo();
        return networkInfo != null && networkInfo.isConnected();
    }


    public void verifyTokenIfPresent() {
        try {
```

```
Cursor query = database.rawQuery(
        "SELECT * FROM users", null);
if (query.moveToFirst()) {
    String email = query.getString(0);
    String uid = query.getString(1);
    String category = query.getString(2);
    String subcategory = query.getString(3);
    String status = query.getString(4)
   String token = query.getString(5);
   HashMap<String, String> valuePairs = new HashMap<String, String>();
    valuePairs.put("email", email);
    valuePairs.put("uid", uid);
    valuePairs.put("category", category);
    valuePairs.put("subcategory", subcategory);
    valuePairs.put("status", status);
    valuePairs.put("token", token);
    HttpPostRequest post = new HttpPostRequest(
            Config.baseURL + "user/token.php", valuePairs);
    if (post.code == 200) {
        if(post.responseText.contains("success")) {
            Config.email = email;
            Config.uid = uid;
            Config.category= category;
            Config.subcategory = subcategory;
            Config.status = status;
            Config.token = token;
            System.out.println("here 1");
            goToMenuActivity();
        }
        else
            System.out.println("Error1 "+post.responseText);
    } else {
        System.out.println("Error verifying login token");
    }
```

```
            } else {
                System.out.println("No token in database");
                return;
            }
        } catch (Exception exc) {
            //exc.printStackTrace();
            System.out.println("verify token: no user local tb");
        }

    }

    public void createToastMessage(final String message) {
        runOnUiThread(new Runnable() {
            @Override
            public void run() {
                Toast.makeText(getApplicationContext(), message,
Toast.LENGTH_LONG).show();
            }
        });

    }

    public void login(View view) {

        final String email2 = ((EditText)
findViewById(R.id.editText)).getText().toString();
        final String pwd=((EditText)
findViewById(R.id.editText2)).getText().toString();
        if(email2.equals("") && pwd.equals(""))
        {
            Toast.makeText(this, "please enter your email and password",
Toast.LENGTH_SHORT).show();
            return;
        }
    if(email2.equals(""))
    {
      Toast.makeText(this, "please enter your email",
Toast.LENGTH_SHORT).show();
      return;
    }

    if(pwd.equals(""))
    {
      Toast.makeText(this, "please enter your password",
Toast.LENGTH_SHORT).show();
      return;
    }


    InputStream inStream = null;
    String result = "";
    final ProgressBar progressBar = (ProgressBar)
findViewById(R.id.loginProgressBar);
    progressBar.setVisibility(View.VISIBLE);
    Thread thread = new Thread(new Runnable() {
      @Override
      public void run() {
        try {

          HashMap<String, String> valuePairs = new HashMap<String, String>();
          email = ((EditText) findViewById(R.id.editText)).getText().toString();
          final String password = ((EditText)
findViewById(R.id.editText2)).getText().toString();
if(email=="" || password=="")
  return;
          final String fEmail = email;
          valuePairs.put("email", email);
```

```
                valuePairs.put("password", password);
                final HttpPostRequest post =
                      new HttpPostRequest(Config.baseURL + "user/login.php",
valuePairs);
                runOnUiThread(new Runnable() {
                   @Override
                   public void run() {
                      TextView text = (TextView) findViewById(R.id.loginStatus);
                      try {

                         //final JSONObject json = new JSONObject(post.responseText);
                         runOnUiThread(new Runnable() {
                            @Override
                            public void run() {
                               try {

createToastMessage(post.responseText);//.getString("status"));
                               } catch (Exception exc) {
                                  exc.printStackTrace();
                               }

                               progressBar.setVisibility(View.INVISIBLE);
                            }
                         });
                         if (post.code == 200 && post.responseText.contains("success"))
{
                            database.execSQL("DROP TABLE IF EXISTS users");
                            database.execSQL("CREATE TABLE users(email TEXT,
uid INTEGER, category TEXT, subcategory TEXT, status TEXT, token
TEXT)");
                            String[]values=post.responseText.split("-");
if(post.responseText.contains("student"))
{
   database.execSQL(
      "INSERT INTO users VALUES('" + email + "', '" +
            values[1]+ "', '', '', 'student', '')");

}
else
{
   database.execSQL(
      "INSERT INTO users VALUES('" + email + "', '" +
            values[1]+ "', '" +values[2]+ "', '" +
            values[3]+ "', '" +values[4]+ "', '" +
            values[5]+ "')");

   FirebaseMessaging.getInstance().subscribeToTopic(values[2]);
}



                        goToMenuActivity();
                     }
                  } catch (Exception exc) {
                     text.setText("Error!!"+exc.toString());
                     exc.printStackTrace();
                     System.out.println("Response: " + post.responseText);
                     runOnUiThread(new Runnable() {
                        @Override
                        public void run() {
                           progressBar.setVisibility(View.INVISIBLE);
                        }
                     });
                  }
               }
            });

         } catch (Exception exc) {
            System.out.println("Exception");
```

```
                exc.printStackTrace();

                runOnUiThread(new Runnable() {
                    @Override
                    public void run() {
                        Toast.makeText(LoginScreen.this, Config.internetError,
Toast.LENGTH_SHORT).show();
                        progressBar.setVisibility(View.INVISIBLE);
                    }
                });
            }


        }

    });
    thread.start();


}

public void getCredentials() {
    database = openOrCreateDatabase("sqlite-QnA.db",
Context.MODE_PRIVATE, null);
    Cursor query = database.rawQuery("SELECT * FROM users", null);
    if (query.moveToFirst()) {
        email = query.getString(0);
        user_id = query.getInt(1);
        category= query.getString(2);
        subcategory = query.getString(3);
        status = query.getString(4);
        token= query.getString(5);
    } else {
        System.out.println("No token in database");
```

```
            return;
        }
        database.close();
    }


    public void showRegistrationActivity(View view) {
        Intent intent = new Intent(LoginScreen.this, RegistrationScreen.class);
        System.out.println("Switching!");
        startActivity(intent);
        finish();
    }
    public void showQuestionsActivity(View view) {
        Intent intent = new Intent(LoginScreen.this, QuestionsActivity.class);
        System.out.println("Switching!");
        startActivity(intent);
        finish();
    }
    public void goToMenuActivity() {
        try {
            /////////////SendFirebaseToken task = new SendFirebaseToken(
            ////////////        getApplicationContext());
            //////////////task.execute();
            Intent intent = new Intent(LoginScreen.this, MenuActivity.class);
            System.out.println("Switching!!");
            startActivity(intent);
            finish();
        }
        catch (Exception exc) {
            TextView text = (TextView) findViewById(R.id.loginStatus);
            text.setText("Error!"+exc.toString());
            exc.printStackTrace();
        }
    }
}
```

RegistrationScreen.java

```
package learn.teach.QnA;
import android.app.Activity;
import android.content.Intent;
import android.net.ConnectivityManager;
import android.net.NetworkInfo;
import android.os.Bundle;
import android.support.design.widget.Snackbar;
import android.view.View;
import android.widget.EditText;
import android.widget.ProgressBar;
import android.widget.TextView;
import android.widget.Toast;
import android.widget.ToggleButton;

import com.google.firebase.messaging.FirebaseMessaging;
import com.jaredrummler.materialspinner.MaterialSpinner;

import java.util.HashMap;

public class RegistrationScreen extends Activity {
    ToggleButton tg1;
    MaterialSpinner spinnercat, spinnersubcat;
    private Activity activity;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_register);
        activity = this;

        spinnersubcat = (MaterialSpinner) findViewById(R.id.spinnersubcat);
        spinnercat = (MaterialSpinner) findViewById(R.id.spinnermaincat);
        spinnercat.setItems("Mechanical", "Civil", "ECE", "EEE", "CSE");
        spinnersubcat.setItems("m1", "m2");

        spinnercat.setOnItemSelectedListener(new
MaterialSpinner.OnItemSelectedListener<String>() {
            @Override public void onItemSelected(MaterialSpinner view, int position,
long id, String item) {
                Snackbar.make(view, "Clicked " + item,
Snackbar.LENGTH_LONG).show();

                switch(item)
                {
                    case "Mechanical": spinnersubcat.setItems("m1", "m2"); break;
                    case "Civil": spinnersubcat.setItems("c1", "c2"); break;
                    case "ECE": spinnersubcat.setItems("e1", "e2"); break;
                    case "EEE": spinnersubcat.setItems("ee1", "ee2"); break;
                    case "CSE": spinnersubcat.setItems("cs1", "cs2"); break;
                }

            }
        });

        tg1=(ToggleButton)findViewById(R.id.toggleButton);
        tg1.setOnClickListener(new View.OnClickListener() {

            @Override
            public void onClick(View v) {
                // TODO Auto-generated method stub
                if (tg1.isChecked()) {
                    spinnercat.setVisibility(View.VISIBLE);
                    spinnersubcat.setVisibility(View.VISIBLE);
                    //Toast.makeText(activity, "on", Toast.LENGTH_SHORT).show();
                } else {
                    spinnercat.setVisibility(View.INVISIBLE);
                    spinnersubcat.setVisibility(View.INVISIBLE);
                    //Toast.makeText(activity, "off", Toast.LENGTH_SHORT).show();
                }
            }
        });



        final TextView text = (TextView) findViewById(R.id.registrationStatus);
        Thread thread = new Thread(new Runnable() {

            public void run() {
                String textToSetForView = "";
                if (isConnected()) {
                    textToSetForView = "Connected";
                } else {
                    textToSetForView = "No Internet Connection";
                }
                final String textVal = textToSetForView;
                runOnUiThread(new Runnable() {

                    public void run() {
                        text.setText(textVal);
                    }
                });
            }
        });
        thread.start();
    }

    public boolean isConnected() {
        ConnectivityManager connMgr = (ConnectivityManager)
getSystemService(Activity.CONNECTIVITY_SERVICE);
        NetworkInfo networkInfo = connMgr.getActiveNetworkInfo();
```

```
        return networkInfo != null && networkInfo.isConnected();
    }

    public void showLoginScreen(View view) {
        Intent intent = new Intent(RegistrationScreen.this, LoginScreen.class);
        startActivity(intent);
        System.out.println("Switching to login screen");
    }

    public void createToastMessage(final String message) {
        runOnUiThread(new Runnable() {
            @Override
            public void run() {
                Toast.makeText(getApplicationContext(), message,
Toast.LENGTH_LONG).show();
            }
        });

    }

    public void register(View view) {

        final String email = ((EditText)
findViewById(R.id.emailAddressRegister)).getText().toString();
        final String pwd=((EditText)
findViewById(R.id.passwordRegister)).getText().toString();

        if(email=="" || pwd=="")
        {
            Toast.makeText(activity, "plz enter email and pwd",
Toast.LENGTH_SHORT).show();
            return;
        }
        Thread thread = new Thread(new Runnable() {
            @Override
            public void run() {
                TextView text = ((TextView) findViewById(R.id.registrationStatus));
                final ProgressBar progressBar = (ProgressBar)
findViewById(R.id.registrationProgressBar);
                runOnUiThread(new Runnable() {
                    @Override
                    public void run() {
                        progressBar.setVisibility(View.VISIBLE);
                    }
                });
                try {
                    HashMap<String, String> valuePairs = new HashMap<>();
                    valuePairs.put("email", email);
                    valuePairs.put("password", pwd);
                    valuePairs.put("confirm_password", ((EditText)
findViewById(R.id.confirmPasswordRegister)).getText().toString());
                    valuePairs.put("name", ((EditText)
findViewById(R.id.nameRegister)).getText().toString());
                    if (tg1.isChecked())
                    {
                        valuePairs.put("category", ((MaterialSpinner)
findViewById(R.id.spinnermaincat)).getText().toString());
                        valuePairs.put("subcategory", ((MaterialSpinner)
findViewById(R.id.spinnersubcat)).getText().toString());
                        valuePairs.put("status", "expert");
                    }
                    else
                    {
                        valuePairs.put("category", "");
                        valuePairs.put("subcategory", "");
                        valuePairs.put("status", "student");
                    }
                    HttpPostRequest post = new HttpPostRequest(
                Config.baseURL + "user/register.php", valuePairs);
            if (post.code == 200) {
                if(post.responseText.contains("success")) {
                    createToastMessage("Success! Please return to the login page
and login." + post.responseText);
                    try
                    {

FirebaseMessaging.getInstance().subscribeToTopic(valuePairs.get("category"));
                    }
                    catch(Exception expp){
                        Toast.makeText(activity, "err "+expp.toString(),
Toast.LENGTH_SHORT).show();
                    }
                }
                else
                    createToastMessage(post.responseText);
            } else if (post.code == 400) {
                createToastMessage(post.responseText);
            } else {
                createToastMessage("An error occurred! Please try again.");
            }

        } catch (Exception exc) {
            exc.printStackTrace();
            createToastMessage(Config.internetError);
        } finally {
            runOnUiThread(new Runnable() {
                @Override
                public void run() {
                    progressBar.setVisibility(View.INVISIBLE);
                }
            });
        }
```

```
            }
        });
        thread.start();
    }
}
```

Question.java

```
package learn.teach.QnA;
public class Question {
    int id;
    String title;
    String text;
    User asker;
    int likes;
    int dislikes;
    int score;

    public String getCategory() {
        return category;
    }

    public void setCategory(String category) {
        this.category = category;
    }

    String category;

    public String getSubcategory() {
        return subcategory;
    }

    public void setSubcategory(String subcategory) {
        this.subcategory = subcategory;
    }

    String subcategory;
    boolean userLiked = false;

    public Question(int id, String title, String text, User asker, int likes, int dislikes,
int score, boolean userLiked, String category, String subcategory)
    {
        this.id = id;
        this.title = title;
        this.text = text;
        this.asker = asker;
        this.likes = likes;
        this.dislikes = dislikes;
        this.score = score;
        this.userLiked = userLiked;
        this.category = category;
        this.subcategory= subcategory;
    }

    public int getId() {
        return id;
    }

    public String getTitle() {
        return title;
    }

    public void setTitle(String title) {
        this.title = title;
    }

    public String getText() {
        return text;
```

```
}

public void setText(String text) {
   this.text = text;
}

public User getAsker() {
   return asker;
}

public void setAsker(User asker) {
   this.asker = asker;
}

public int getLikes() {
   return likes;
}

public void setLikes(int likes) {
   this.likes = likes;
}

public int getDislikes() {
   return dislikes;
}

public void setDislikes(int dislikes) {
   this.dislikes = dislikes;
}

public int getScore() {
   return score;
}
```

```
    public void setScore(int score) {
        this.score = score;
    }

    public boolean isUserLiked() {
        return userLiked;
    }

    public void setUserLiked(boolean userLiked) {
        this.userLiked = userLiked;
    }
}
```

Answer.java

```
package learn.teach.QnA;
public class Answer {
    int id;
    String text;
    User answerer;
    int likes;
    int dislikes;
    int score;

    public Answer(int id, String text, User answerer, int likes, int dislikes, int score) {
        this.id = id;
        this.text = text;
        this.answerer = answerer;
        this.likes = likes;
        this.dislikes = dislikes;
        this.score = score;
    }

    public int getId() {
```

```
    return id;
}

public void setId(int id) {
    this.id = id;
}

public String getText() {
    return text;
}

public void setText(String text) {
    this.text = text;
}

public User getAnswerer() {
    return answerer;
}

public void setAnswerer(User answerer) {
    this.answerer = answerer;
}

public int getLikes() {
    return likes;
}

public void setLikes(int likes) {
    this.likes = likes;
}

public int getDislikes() {
    return dislikes;
}
```

```
    public void setDislikes(int dislikes) {
        this.dislikes = dislikes;
    }

    public int getScore() {
        return score;
    }

    public void setScore(int score) {
        this.score = score;
    }
}
```

FirebaseMessagingService.java

```
package learn.teach.QnA;
import android.app.NotificationManager;
import android.app.PendingIntent;
import android.content.Context;
import android.content.Intent;
import android.media.RingtoneManager;
import android.net.Uri;
import android.support.v4.app.NotificationCompat;
import android.util.Log;
import android.widget.Toast;
import com.google.firebase.messaging.FirebaseMessagingService;
import com.google.firebase.messaging.RemoteMessage;
public class MyFirebaseMessagingService extends FirebaseMessagingService {

    private static final String TAG = "MyFirebaseMsgService";

    @Override
    public void onMessageReceived(RemoteMessage remoteMessage) {
        try {
            String from=remoteMessage.getFrom();
            String body="-";
            try{
            body=remoteMessage.getNotification().getBody();}catch(Exception exp){}
            //Displaying data in log
            //It is optional
            Log.d(TAG, "From:=> " + from);
            Log.d(TAG, "Notification Message Body: " + body);

            //int actionId = Integer.valueOf(remoteMessage.getData().get("objectId"));
            //Log.d("PushNotification", String.valueOf(actionId));
            //Calling method to generate notification
            sendNotification(from+":"+body, 1);//actionId);
            Toast.makeText(this, "::"+ from, Toast.LENGTH_SHORT).show();
        }
        catch(Exception exp) {}
    }

    //This method is only generating push notification
    //It is same as we did in earlier posts
    private void sendNotification(String messageBody, int actionId) {
        Intent intent = new Intent(getApplicationContext(), AnswerDetails.class);
        intent.putExtra("actionId", actionId);
        int uniqueInt = (int) (System.currentTimeMillis() & 0xfffffff);
        PendingIntent pendingIntent =
PendingIntent.getActivity(getApplicationContext(), uniqueInt, intent,
                PendingIntent.FLAG_CANCEL_CURRENT);

        Uri defaultSoundUri =
RingtoneManager.getDefaultUri(RingtoneManager.TYPE_NOTIFICATION);
        NotificationCompat.Builder notificationBuilder = new
NotificationCompat.Builder(this)
```

```
                .setSmallIcon(R.mipmap.ic_launcher)
                .setContentTitle("QnA Notification")
                .setContentText(messageBody)
                .setAutoCancel(true)
                .setSound(defaultSoundUri)
                .setContentIntent(pendingIntent);

        NotificationManager notificationManager =
                (NotificationManager)
getSystemService(Context.NOTIFICATION_SERVICE);

        notificationManager.notify(actionId, notificationBuilder.build());

    }
}
```

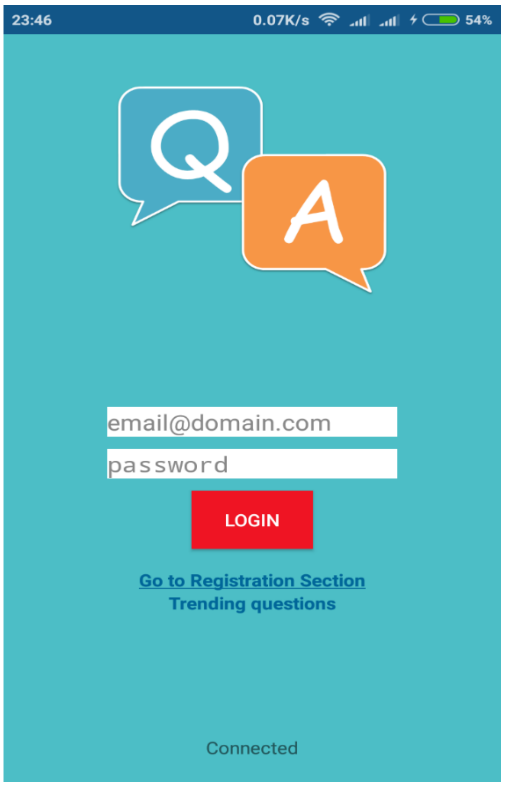
Gradle Script

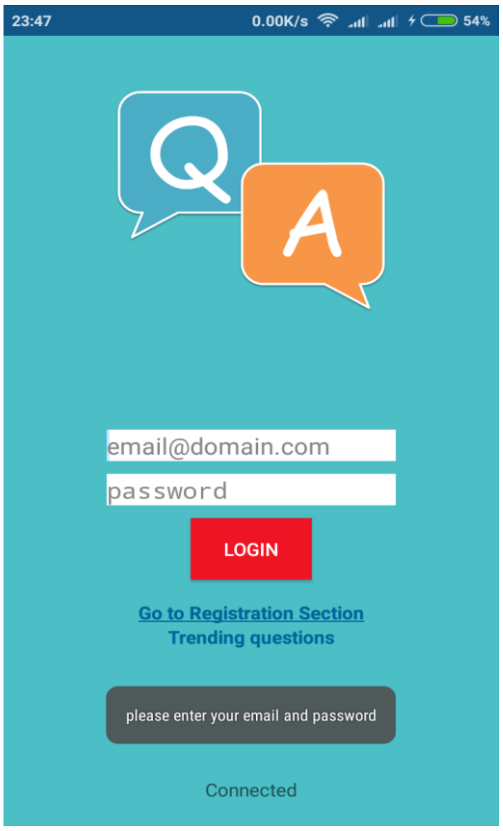
Build.gradle file

```
apply plugin: 'com.android.application'

android {
    compileSdkVersion 25
    buildToolsVersion '26.0.2'

    def versionPropsFile = file('version.properties')
    def versionBuild

    /*Setting default value for versionBuild which is the last incremented value stored
in the file */
    if (versionPropsFile.canRead()) {
        def Properties versionProps = new Properties()
        versionProps.load(new FileInputStream(versionPropsFile))
        versionBuild = versionProps['VERSION_BUILD'].toInteger()
    } else {
        throw new GradleException("Could not read version.properties!")
    }

    /*Wrapping inside a method avoids auto incrementing on every gradle task run.
Now it runs only when we build apk*/
    ext.autoIncrementBuildNumber = {

        if (versionPropsFile.canRead()) {
            def Properties versionProps = new Properties()
            versionProps.load(new FileInputStream(versionPropsFile))
            versionBuild = versionProps['VERSION_BUILD'].toInteger() + 1
            versionProps['VERSION_BUILD'] = versionBuild.toString()
            versionProps.store(versionPropsFile.newWriter(), null)
        } else {
            throw new GradleException("Could not read version.properties!")
        }
    }

    task increaseVersion() << {
        autoIncrementBuildNumber()

    }
    tasks.whenTaskAdded { task ->
        if (task.name == 'assembleRelease') {
            task.finalizedBy 'increaseVersion'
        }
    }

    defaultConfig {
        applicationId "learn.teach.QnA"
        minSdkVersion 19
        targetSdkVersion 24
        versionCode versionBuild
        versionName "1.0." + versionBuild
    }
    buildTypes {
        release {
            minifyEnabled false
            proguardFiles getDefaultProguardFile('proguard-android.txt'), 'proguard-
rules.pro'
        }
    }

    useLibrary 'org.apache.http.legacy'
}
repositories {
    mavenCentral()
}
dependencies {
    compile fileTree(dir: 'libs', include: ['*.jar'])
    compile group: 'org.apache.httpcomponents', name: 'httpclient-android',
version: '4.3.5.1'
    compile group: 'com.squareup.okhttp3', name: 'logging-interceptor', version:
'3.8.0'
    compile('com.mikepenz:fastadapter:2.5.2@aar') {
        transitive = true
    }
    compile 'com.android.support:appcompat-v7:25.3.1'
    compile 'com.android.support:design:25.3.1'
    compile 'com.squareup.picasso:picasso:2.5.2'
    compile 'com.nineoldandroids:library:2.4.0'
    compile 'com.daimajia.slider:library:1.1.5@aar'
    compile 'com.google.firebase:firebase-messaging:9.6.0'
    compile 'com.jaredrummler:material-spinner:1.1.0'
    compile 'com.github.jd-alexander:LikeButton:0.2.0'
    compile 'com.squareup.okhttp3:okhttp:3.8.0'
    compile 'com.mikepenz:fastadapter-commons:2.5.2@aar'
    compile 'com.mikepenz:fastadapter-extensions:2.5.2@aar'
    compile 'com.mikepenz:materialize:1.0.1@aar'
    compile 'uk.co.chrisjenx:calligraphy:2.3.0'
    compile 'com.android.volley:volley:1.0.0'
    compile 'com.android.support.constraint:constraint-layout:1.0.2'
    testCompile 'junit:junit:4.12'
    compile 'com.github.clans:fab:1.6.4'
}
apply plugin: 'com.google.gms.google-services'
```

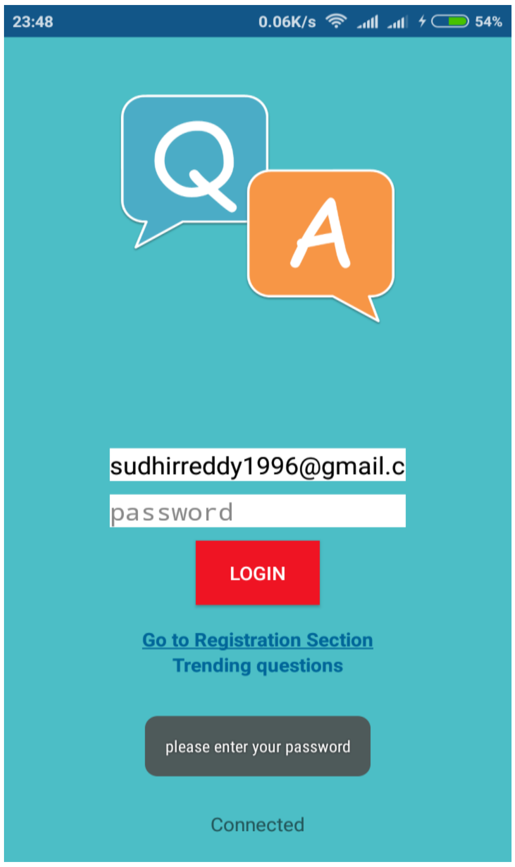
5.2 Output Screens

Login Activities





please enter your email and password



23:48 0.06K/s 54%

sudhirreddy1996@gmail.c

password

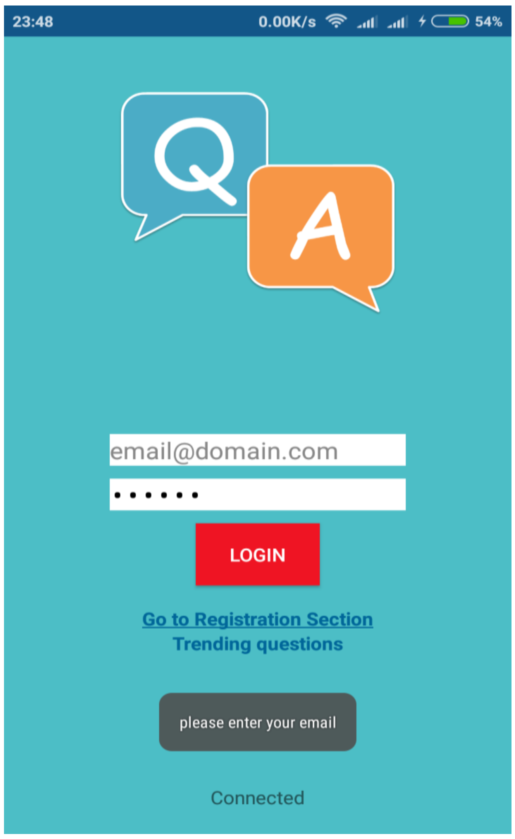
LOGIN

Go to Registration Section

Trending questions

please enter your password

Connected



23:48 0.00K/s 54%

email@domain.com

••••••

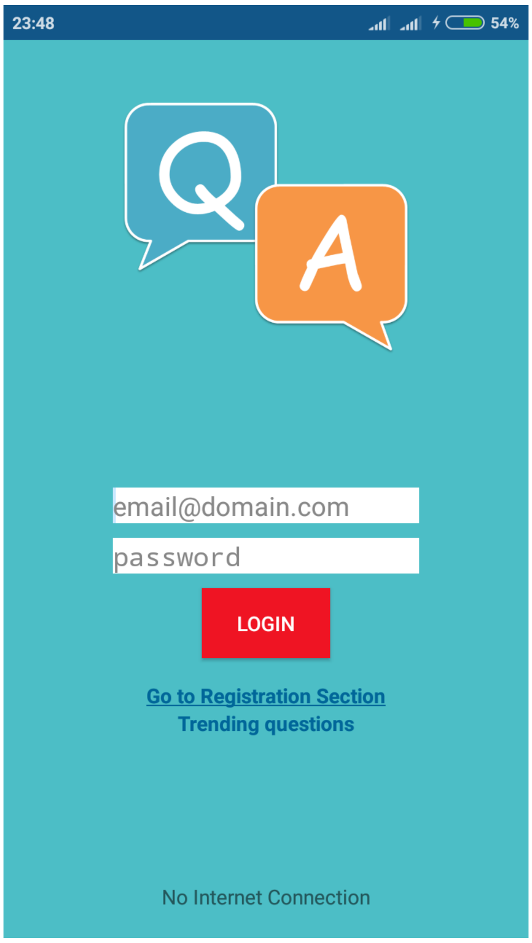
LOGIN

Go to Registration Section

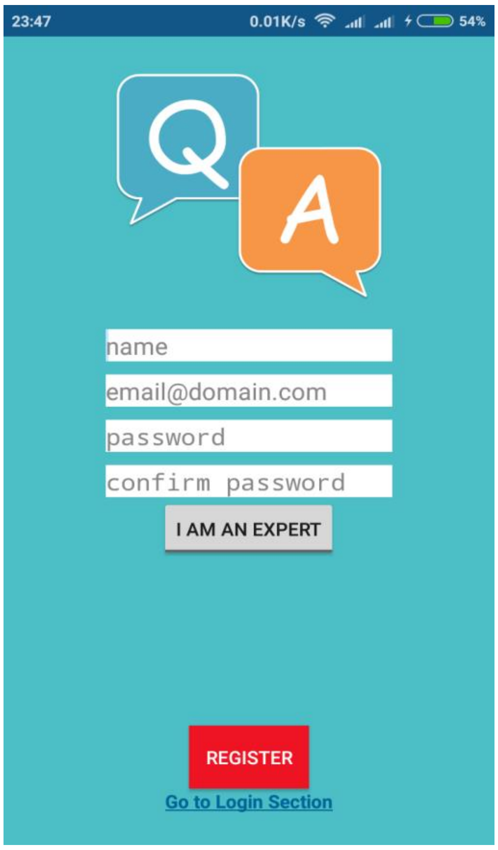
Trending questions

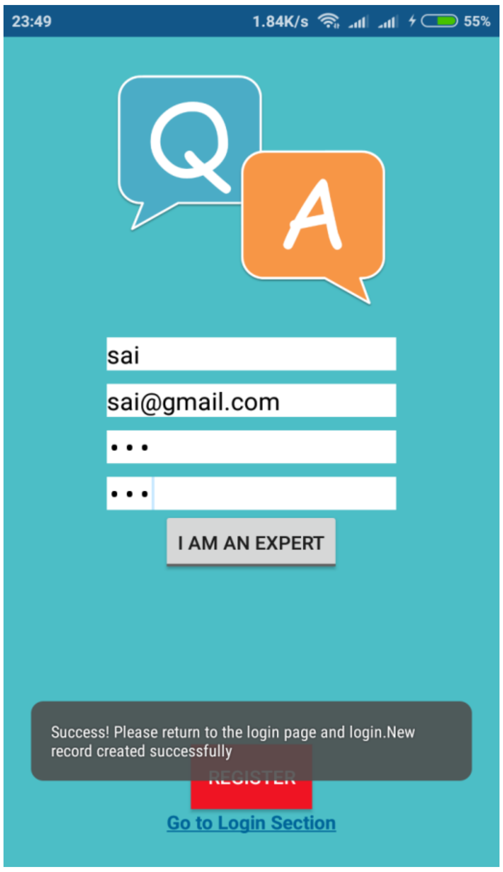
please enter your email

Connected

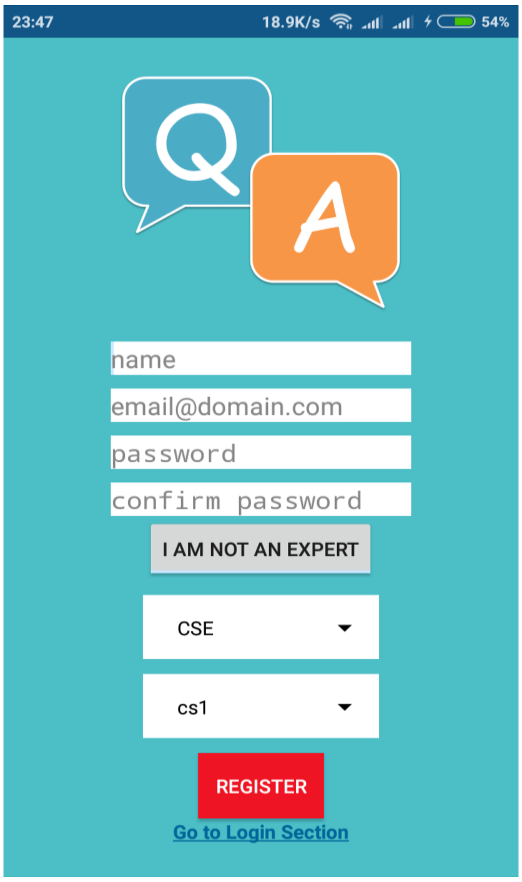

23:48
54%
email@domain.com
password
LOGIN
Go to Registration Section
Trending questions
No Internet Connection

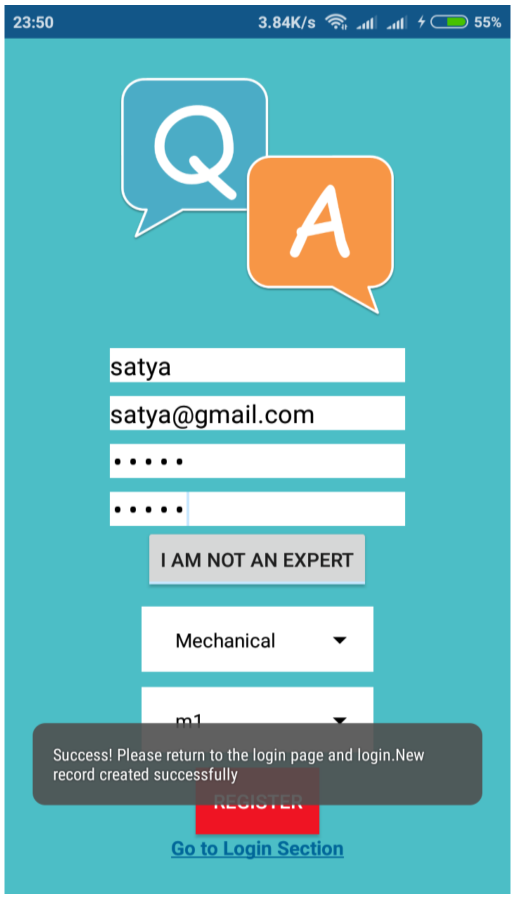
Registration Activities





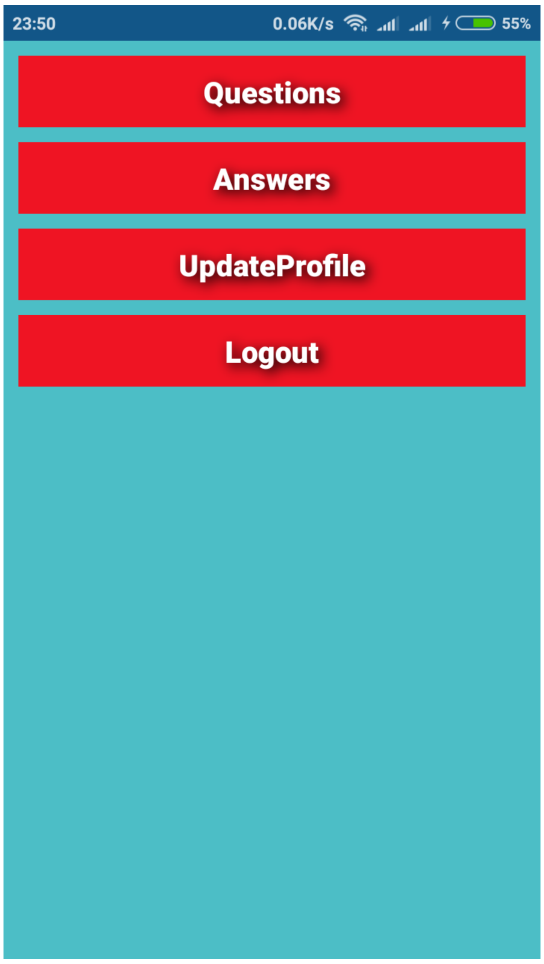
Success! Please return to the login page and login.New record created successfully



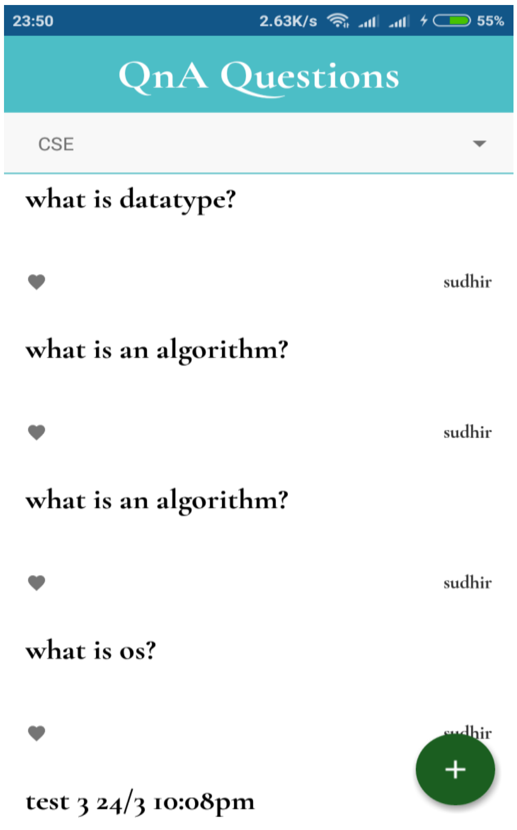


Success! Please return to the login page and login.New record created successfully

Menu Activity



Question Posting Activities


23:50 2.63K/s 55%

QnA Questions

CSE

what is datatype?

sudhir

what is an algorithm?

sudhir

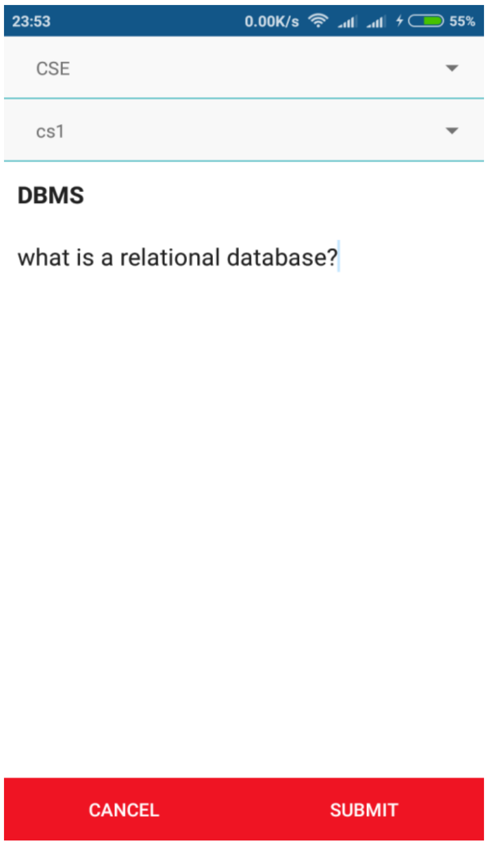
what is an algorithm?

sudhir

what is os?

sudhir

test 3 24/3 10:08pm


23:53 0.00K/s 55%

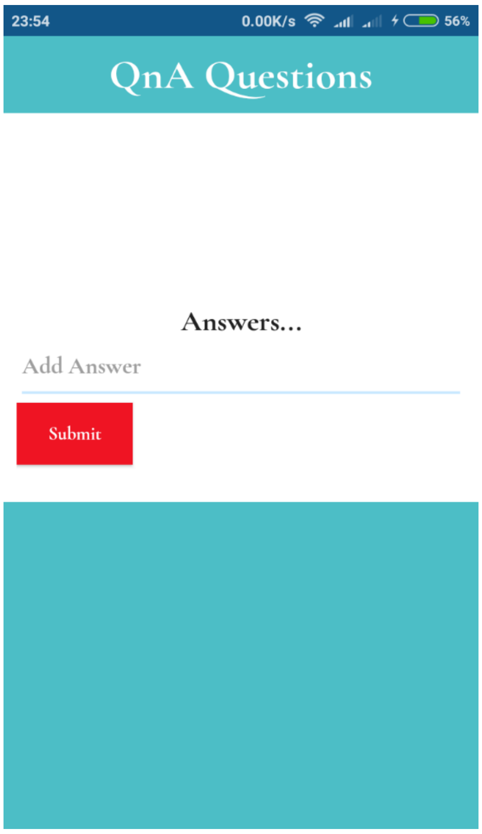
CSE

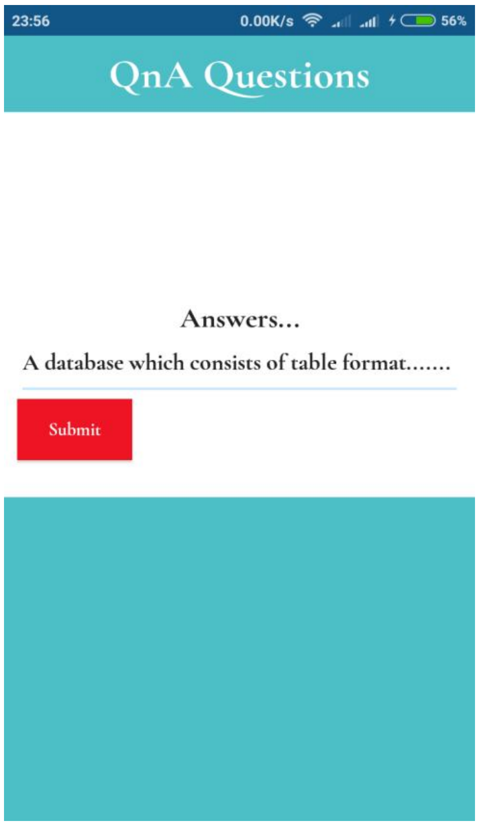
cs1

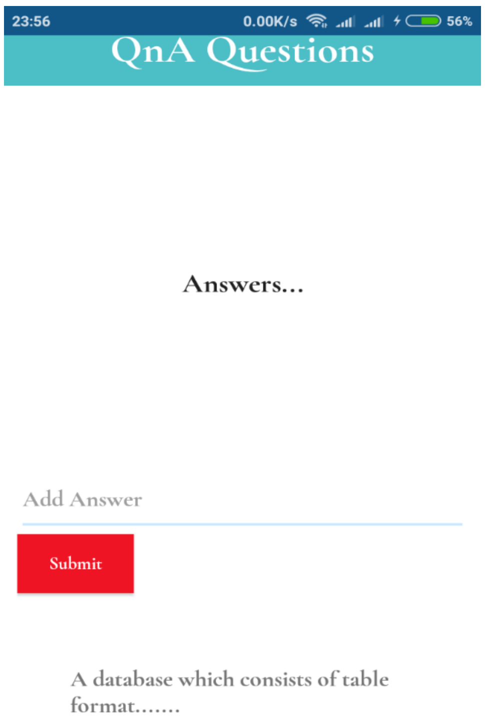
DBMS

what is a relational database?

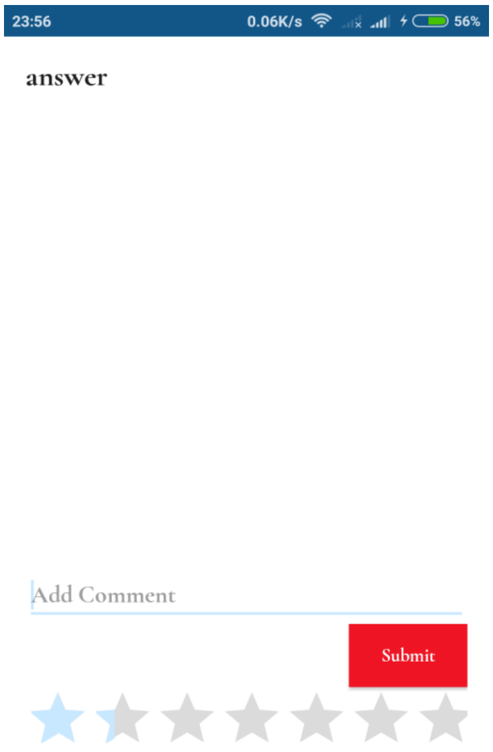
Answer Activities



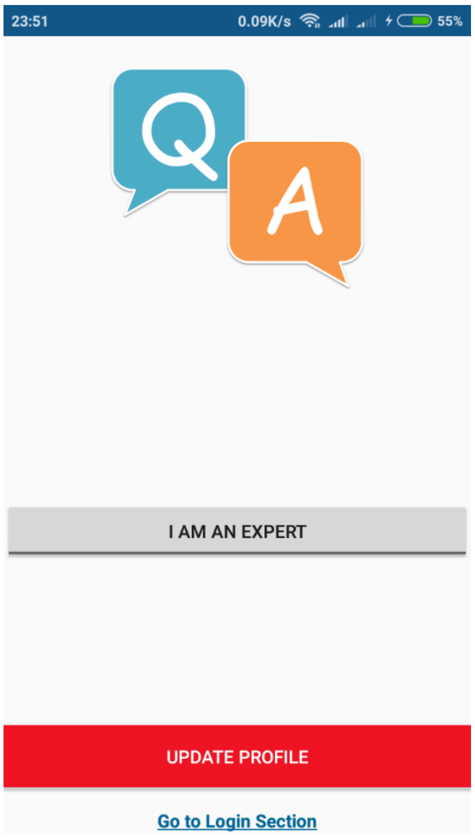


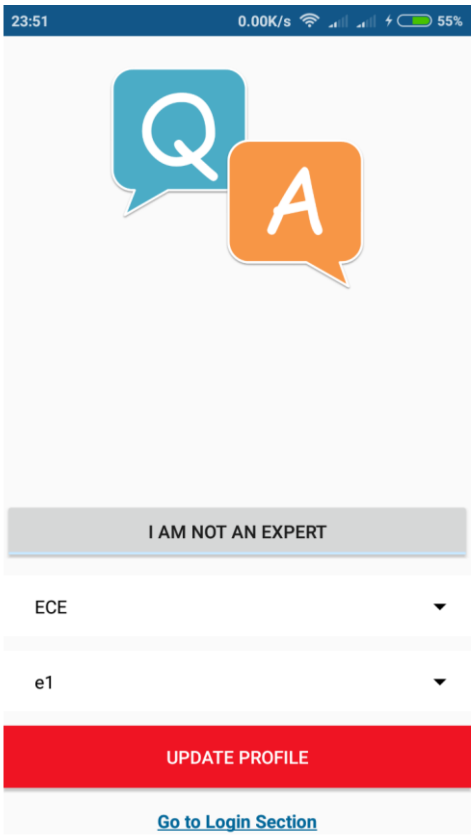


A database which consists of table format.......

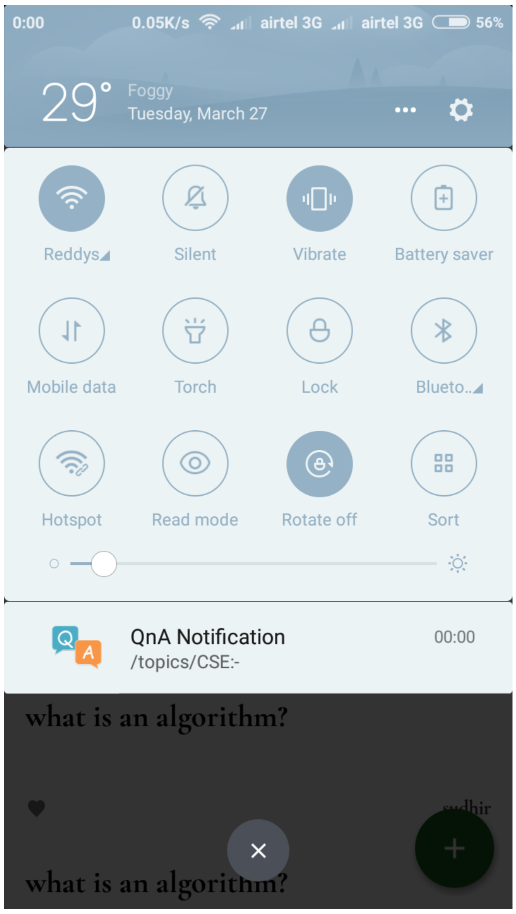


Update Profile Activities





Notification Activity



QnA Notification 00:00
/topics/CSE:-

what is an algorithm?

sudhir

what is an algorithm?

6. SYSTEM TESTING

6.1 Types of Testing

Software testing is an investigation conducted to provide stakeholders with information about the quality of the product or service under test. Software testing methods are traditionally divided into white- and black-box testing. These two approaches are used to describe the point of view that a test engineer takes when designing test cases. White-box testing (source code) tests internal structures or workings of a program, as opposed to the functionality exposed to the end-user.

Techniques used in white-box testing include:

API testing – testing of the application using public and private APIs

Code coverage – creating tests to satisfy some criteria of code coverage

Fault injection methods – intentionally introducing faults to gauge the efficacy of testing strategies

Mutation testing methods

Static testing methods

Black-box testing treats the software as a "black box", examining functionality without any knowledge of internal implementation, without seeing the source code. The testers are only aware of what the software is supposed to do, not how it does it.

Mobile application testing is a process by which application software developed for handheld mobile devices is tested for its functionality, usability and consistency. Mobile application testing can be automated or manual type of testing

6.2 Types of Mobile Application Testing

1. Functional Testing: Functional testing ensures that the application is working as per the requirements. Most of the test conducted for this is driven by the user interface and call flows.

2. Laboratory Testing: Laboratory testing, usually carried out by network carriers, is done by simulating the complete wireless network. This test is performed to find out any glitches when a mobile application uses voice and/or data connection to perform some functions.

3. Performance Testing: This testing process is undertaken to check the performance and behavior of the application under certain conditions such as low battery, bad network coverage, low available memory, simultaneous access to application's server by several users and other conditions. Performance of an application can be affected from two sides: application's server side and client's side. Performance testing is carried out to check both.

4. Memory Leakage Testing: Memory leakage happens when a computer program or application is unable to manage the memory it is allocated resulting in poor performance of the application and the overall slowdown of the system. As mobile devices have significant constraints of available memory, memory leakage testing is crucial for the proper functioning of an application

5. Interrupt Testing: An application while functioning may face several interruptions like incoming calls or network coverage outage and recovery. The different types of interruptions are:

1. Incoming and Outgoing SMS and MMS
2. Incoming and Outgoing calls
3. Incoming Notifications
4. Battery Removal
5. Cable Insertion and Removal for data transfer
6. Network outage and recovery
7. Media Player on/off
8. Device Power cycle

An application should be able to handle these interruptions by going into a suspended state and resuming afterwards.

6. Usability testing: Usability testing is carried out to verify if the application is achieving its goals and getting a favorable response from users. This is important as the usability of an application is its key to commercial success (it is nothing but user friendliness). Another important part of usability testing is to make sure that the user experience is uniform across all devices. This section of testing hopes to address the key challenges of the variety of mobile devices and the diversity in mobile platforms/OS, which is also called device fragmentation. One key portion of this type of usability testing is to be sure that there are no major errors in the functionality, placement, or sizing of the user interface on different devices.

7. Installation testing: Certain mobile applications come pre-installed on the device whereas others have to be installed from the store. Installation testing verifies that the installation process goes smoothly without the user having to face any difficulty. This testing process covers installation, updating and uninstalling of an application

8. Certification Testing: To get a certificate of compliance, each mobile device needs to be tested against the guidelines set by different mobile platforms. The Certified Mobile Application Tester (CMAT) certification exam is offered by the Global Association for Quality Management (GAQM) via Pearson Vue Testing Center worldwide to benefit the Mobile Application Testing Community.

9. Security Testing: To check for vulnerabilities to hacking, authentication and authorization policies, data security, session management and other security standards.

10. Location Testing: Connectivity changes with network and location, but you can't mimic those fluctuating conditions in a lab. Only in Country non automated testers can perform comprehensive usability and functionality testing.

11. Outdated Software Testing: Not everyone regularly updates their operating system. Some Android users might not even have access to the newest version. Professional Testers can test outdated software.

12. Load Testing: When many users all attempt to download, load, and use your app or game simultaneously, slow load times or crashes can occur causing many customers to abandon your app, game, or website. In-country human testing done manually is the most effective way to test load.

Android Automated Testing

Android testing is based on JUnit. In general, a JUnit test is a method whose statements test a part of the application. You organize test methods into classes called test cases. You can further organize these classes into test suites.

In JUnit, you build one or more test classes and use a test runner to execute them. In Android, you use Android Studio (or the Android Plugin for Gradle) to build one or more test source files into an Android test app. From your testing environment, you can run your test in one of the following ways:

On your local machine: Compile the test classes and execute them locally on the Java Virtual Machine (JVM) using the JUnit test runner. On a device or emulator: Install the test app and the app under test to a physical device or emulator, and then execute your tests using an Android-specific test runner (such as Android Junit Runner).

Instrumentation

Android instrumentation is a set of control methods or hooks in the Android system. These hooks control an Android component independently of its normal lifecycle. They also control how Android loads applications. Normally, an Android component runs in a lifecycle that the system determines. For example, an Activity object's lifecycle starts when an Intent activates the Activity. The system calls the object's onCreate() method, on then the onResume() method. When the user starts another application, the system calls the onPause() method. If the Activity code calls the finish() method, the system calls the onDestroy() method. The Android framework API does not provide a way for your code to invoke these callback methods directly, but you can do so using instrumentation.

The system runs all the components of an application in the same process. You can allow some components, such as content providers, to run in a separate process, but you can't force an application to run in the same process as another application that is already running. Instrumentation can load both a test package and the app under test into the same process. Since the application components and their tests are in the same process, your tests can invoke methods in the components, and modify and examine fields in the components.

Android Testing Support Library APIs

The Android Testing Support Library provides a set of APIs that allow you to quickly build and run test code for your apps, including JUnit 4 and functional user interface (UI) tests. The library includes the following instrumentation-based APIs that are useful when you want to automate your tests:

Android Junit Runner: JUnit 4-compatible test runner for Android

Espresso: UI testing framework; suitable for functional UI testing within an app

UI Automator: UI testing framework; suitable for cross-app functional UI testing across system and installed apps

Assertion classes

Because Android Testing Support Library APIs extend JUnit, you can use assertion methods to display the results of tests. An assertion method compares an actual value returned by a test to an expected value, and throws an AssertionException if the comparison test fails. Using assertions is more convenient than logging, and provides better test performance. To simplify your test development, we recommend that you use the Hamcrest library, which lets you create more flexible tests using the Hamcrest matcher APIs. Monkey and Monkeyrunner

The SDK provides two tools for functional-level application testing

The UI/Application Exerciser Monkey, usually called "monkey", is a command-line tool that sends pseudo-random streams of keystrokes, touches, and gestures to a device. You run it with the Android Debug Bridge (ADB) tool. You use it to stress-test your application and report back errors that are encountered. You can repeat a stream of events by running the tool each time with the same random number seed.

The monkeyrunner tool is an API and execution environment for test programs written in Python. The API includes functions for connecting to a device, installing and uninstalling packages, taking screenshots, comparing two images, and running a test package against an application. Using the API, you can write a wide range of large, powerful, and complex tests. You run programs that use the API with the monkeyrunner command-line tool.

6.3 Test Cases

Test Case ID: 1

Test Case Description: User launches the app

Test Conditions:

Step No.	Step Description	Test Data	Expected Result	Overall Result
1	User launches the application by pressing app icon		Main activity window must be displayed	success

Test Case ID: 2

Test Case Description: Registration Screen

Test Conditions:

Step No.	Step Description	Test Data	Expected Result	Overall Result
1	User can register as student and expert by filling the details	Details are stored in the database	Toast message must be shown as registration success	success

Test Case ID: 3

Test Case Description: Login Screen

Test Conditions:

Step No.	Step Description	Test Data	Expected Result	Overall Result
1	User can login into the application by using login credentials	Details are verified and validated from database	Toast message must be shown as login success	
2	After successful login		Process activity must be displayed with some Buttons having functionalities	success
3	Buttons		i.Questions ii.Answers iii.Update profile iv.Logout	

Test Case ID: 4

Test Case Description: Questions Activity

Test Conditions:

Step No.	Step Description	Test Data	Expected Result	Overall Result
1	User have to press Question button		Question activity window must be displayed	
2	Existed Questions	Questions are selected from the database	Existed questions must be displayed in ranking order	
3	User have to press post button(+) for posting questions		Question posting activity must be displayed	success
4	User have to select categories and sub-categories of a subject before posting a question		Categories and sub-categories must be displayed as drop-down list / spinner	
5	For posting a question, user have to press submit button	Questions are stored in the database	Toast message must be shown as question posted successfully	

Test Case ID: 5

Test Case Description: Answers Activity

Test Conditions:

Step No.	Step Description	Test Data	Expected Result	Overall Result
1	User must be logged in as expert before answering a question			success
2	User have to press Answer button		Answer activity window must be displayed	
3	Existed Answers	Answers are selected from the database	Existed answers must be displayed in ranking order	
4	For posting the answer, expert must had selected the category of question that has asked			
5	For posting a answer, user have to press submit button	Answers are stored in the database	Toast message must be shown as answer posted successfully	

Test Case ID: 6

Test Case Description: Updating the profile of user

Test Conditions:

Step No.	Step Description	Test Data	Expected Result	Overall Result
1	User have to press Update profile button		Update profile activity window must be displayed	success
2	After updating the details, user have to press update button	Details are updated in the database	Toast message must be shown as details updated successfully	

Test Case ID: 7

Test Case Description: Logout from the application

Test Conditions:

Step No.	Step Description	Test Data	Expected Result	Overall Result
1	User have to press logout button		Toast message must be shown as logged out successfully	success

7. CONCLUSION & FUTURE ENHANCEMENTS

Conclusion

Our hypothesis was that knowledge sharing through community platforms like android applications, websites,..etc, are the best ways to share and acquire knowledge throughout the world. By this multiple number of users (students and experts) from different places can be present in single platform in our project. So, that the queries of students can be easily solved by experts. For increasing response time we are providing In-app notifications, by this students can get a immediate response from the experts.

Future Enhancements

- We will categorize experts based on sub-streams also.
- We are going to add experts of different streams from different universities and organizations for increasing reliability and quality of answers.
- We will provide various category of materials of different streams in the form of PDF's, DOC's, etc., which are useful for the users as reference.
- We will provide best search engine for searching questions and answers.
- We will provide adding picture functionality while posting and answering the question.

8. BIBLIOGRAPHY

1. Java The Complete Reference-Herbert Schildt, ninth edition
2. Software Engineering-K.K.Agarawal, New Age Publications
3. Android Programming-Bill Phillips
4. Android Programming for Beginners-John Hortan

9. REFERENCES

- **https://developer.android.com/index.html**
- **https://stackoverflow.com/**
- **https://www.quora.com/**
- **http://www.vogella.com/tutorials/Android/article.html**
- **https://codelabs.developers.google.com/?cat=Android**
- **http://www.xmlnews.org/docs/xml-basics.html**
- **https://code.tutsplus.com/tutorials/a-beginners-guide-to-http-and-rest--net-16340**
- **https://www.tutorialspoint.com/android/android_studio.htm**
- **https://www.w3schools.in/category/android-tutorial/**
- **https://www.w3schools.com/php/default.asp**