**CHAPTER 1**

**INTRODUCTION**

**1.1 OVERVIEW**

The growing impact over smart phones increased the development of mobile application for Interacting with several domains. The proposed system is an Android application to make a provision for online interaction between district Authorities and students. . The purpose is to manifest latent ideas into commercial products that can benefit humanity in general, and help to solve district’s existing social perils in particular. The application is designed specifically for college and university students. The main objective of this project is student can solve different problems as their project by submitting solution in the form of report. The district Authorities can verify the details entered by the students and can validate the report submitted by the corresponding student. In addition district Authorities can broadcast any new upcoming problem statements or projects to the students.

**1.2 PROBLEM STATEMENT**

A portal needs to be developed to facilitate district collector to upload the district specific problems, projects etc. The portal should be accessible to all the institutes so that the students can select the problems as their project work and submit solution in the form of report. The portal should have a provision for online interaction between district Authorities (respective department officials) and the concerned student group/Institute**.**

**1.3 EXISTING SYSTEM**

The existing system which we have is unique initiative to identify new and disruptive digital technology innovations for solving the challenges faced by our country.Through this initiative, govt. departments will directly engage with thousands of technical students and challenge them to build digital solutions to improve their efficiency, plug revenue leakages and corruption.

**1.3.1 DISADVANTAGES OF EXISTING SYSTEM**

* The existing System focus on over all projects and we don’t have any direct interaction with the district authority.
* In the existing system, students can upload their ideas on corresponding projects only for the competition level.
* There is no proper app for submitting student’s project abstract.
* The existing application have more problems on registration process and Participation process.

**1.4 PROPOSED SYSTEM**

The proposed system provide a direct online interaction between students and district authority. The application is designed specifically for college and university students. The main objective of this project is student can solve different problems as their project by submitting solution in the form of report. The students can use this application through their unique username and password by sign in through their personal and academic details, they can view different problem statements and they can enroll themselves with particular project Once the project is done students can submit their project ideas as a report. Since this application facilitate the district collector to upload the specific district problems, projects etc, the students who come up with new ideas and projects can provide a solution to several problems that exist in their own district. This application is compatible with all major mobile platforms and can be used without limitations

**1.4.1** **ADVANTAGES OF THE PROPOSED SYSTEM**

* + It is a facilitator for students with out-of-box ideas, or innovative working prototypes.
  + Creates online interaction between Students and Higher Authority.
  + Online submission of Abstract and Reports.
  + Admin alone visualizes the report.
  + Reports are secured and easily validate by the admin.
  + Provide opportunity for students to provide innovative solutions to their district daunting problems.
  + It is easy portal that provide a new platform to students with ideas and prototypes, which can be transformed into working products.

**CHAPTER -2**

**REQUIREMENTS**

**2.1 TOOLS REQUIREMENTS**

**2.1.1 TOOLS USED**

* FrontEnd : Android studio,Java,XML.
* Back End : Firebase
* Operating System : Windows7/8/10

**2.1.2 MINIMUM REQUIREMENTS OF THE SYSTEM**

* Processor : Intel core i3
* RAM : 4 GB
* HDD : 64 GB
* Monitor : 15’’colour monitor

**2.2 FUNCTIONAL REQUIREMENTS**

* Function to register the students details such as Username, College, Year ,Login ID , Password ,Phone Number.
* Function to login to the student profile with correct login ID and password.
* Function to display several departments related to the existing problems.
* Function that facilitate the student to select a particular department.
* Function to view all the problem statements in the corresponding department by the student.
* Function that facilitate the student to submit the abstract for the problem they choose.
* Function to admin login with appropriate correct password.
* Function that facilitate the admin to view the all abstract submitted by the students from different colleges with their details.

**2.3 NON-FUNCTIONAL REQUIREMENTS**

**2.3.1 Performance**

Performance is measured in terms of the output provided by the application.

Requirement specification plays an important part in the analysis of a system. Only when the

requirement specifications are properly given, it is possible to design a system, which will fit

into required environment. It rests largely in the part of the staffs of the existing system to give the requirement specifications because they are the people who finally use the system.

This is because the requirements have to be known during the initial stages so that the system

can be designed according to those requirements. It is very difficult to change the system once it has been designed and on the other hand designing a system, which does not cater to

the requirements of the user, is of no use. The requirement specification for any system can

be broadly stated as given below:

* The Android Mobile should be able to interface with the server
* The Android Mobile should be accurate
* The Android Mobile should be better than the existing system

The existing system is completely dependent on the user to perform all the

duties.

**2.3.2 Safety and Security Requirements**

* + **User Identification:** The system requires the user to identify himself/herself
    1. **Login ID**: The user and admin who uses the system shall have a Logon ID and Password.
  + **Modification**: Any modification (insert, delete (or) update) for the Database shall besynchronized and done only by the administrator in the ward.
  + **Administrator Rights**: Administrators shall be able to view and modify allinformation and be able to modify them.

**CHAPTER 3**

**MODULES DESCRIPTION**

**3.1 STUDENT INTERFACE MODULE**

In this module we deal with login interface, storage in and retrieval of data from database in the server. Accordingly, depending on whether the student is authenticated, the interface begins. If the student does not have the login Id before, he/she can register with their details by creating a new account. If an essential field is not filled in the form then an error is thrown stating to fill in that field.

After the authentication the ministry for different department module list is displayed where the student can choose the department and corresponding problem statement. Once the student select his/her project he/she can upload the abstract of his project so that the respective details of the student are stored in the database with their abstract.

**3.2 ADMIN FEATURES**

* Admin have their respective user id & password for authenticated login
* Admin can add or remove the problem statements from the list.
* Admin can view all the abstract submissions done by the students with their college name and registration details.

**3.3 Pre-Condition**

* The Admin/district authority should be authenticated
* They should have their user id and password individually

**3.4 Post-Condition**

* Login is successful.

**3.5 Uploading the abstract for problem statement**

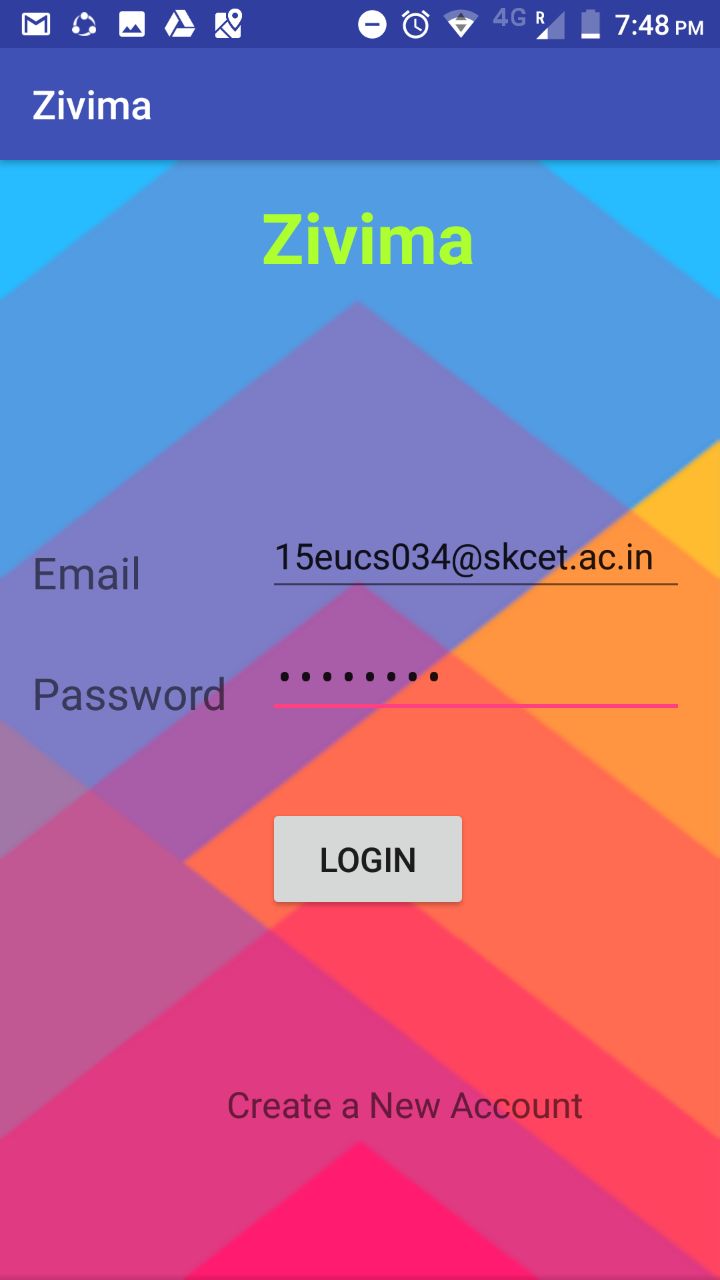
As soon as the student logs in, he/she is asked to select the type of department or ministry, where several problem statements are posted by the district authority. Once the student viewed all the problem statements they can submit their abstract by clicking the attach button for the corresponding problem statement. Once the attachment is submitted the details are stored in respective tables of service in the database. The admin can view and retrieve the details from database to process it and updates the status of application. Admin can view the number of entries in the database for each form.

**CHAPTER 4**

**DESIGN**

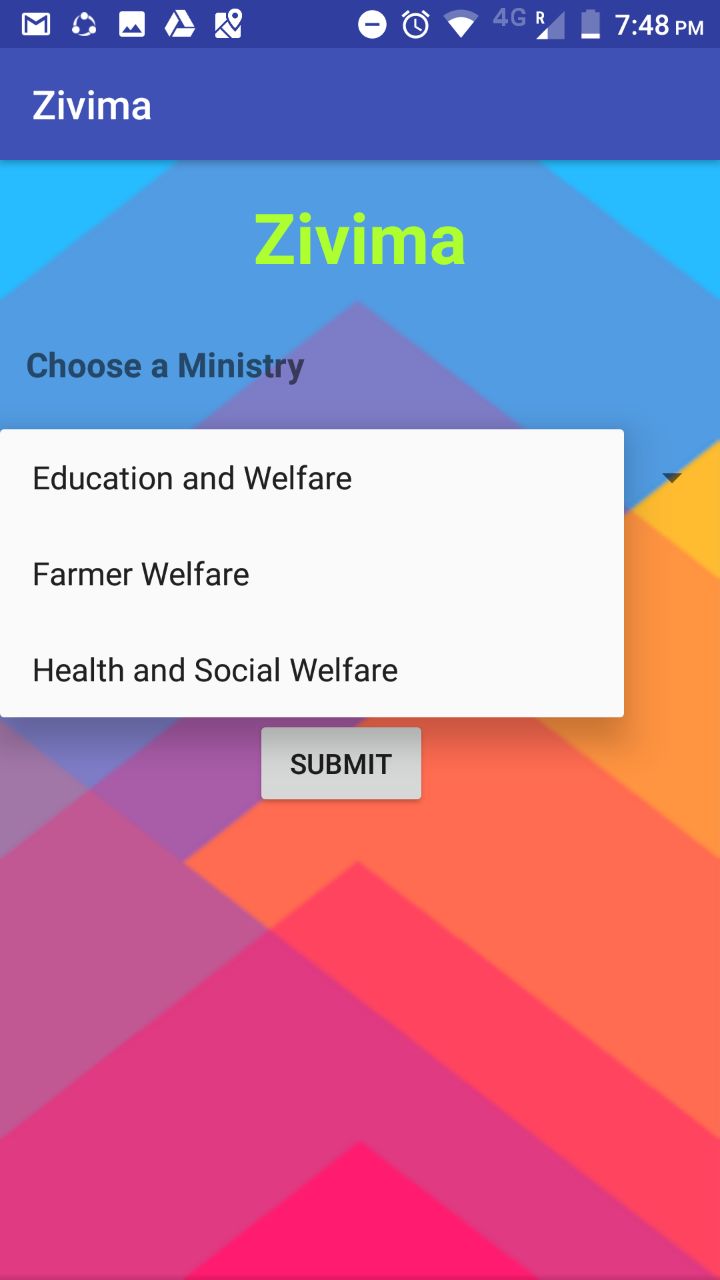
# **4.1 USER INTERFACE MODEL**

* Once the student creates an account he/she can login to the application by entering a correct email ID and password to navigate to the further page.
* If the email ID or password is Invalid it will throw an error message like ‘Authentication Failed’.



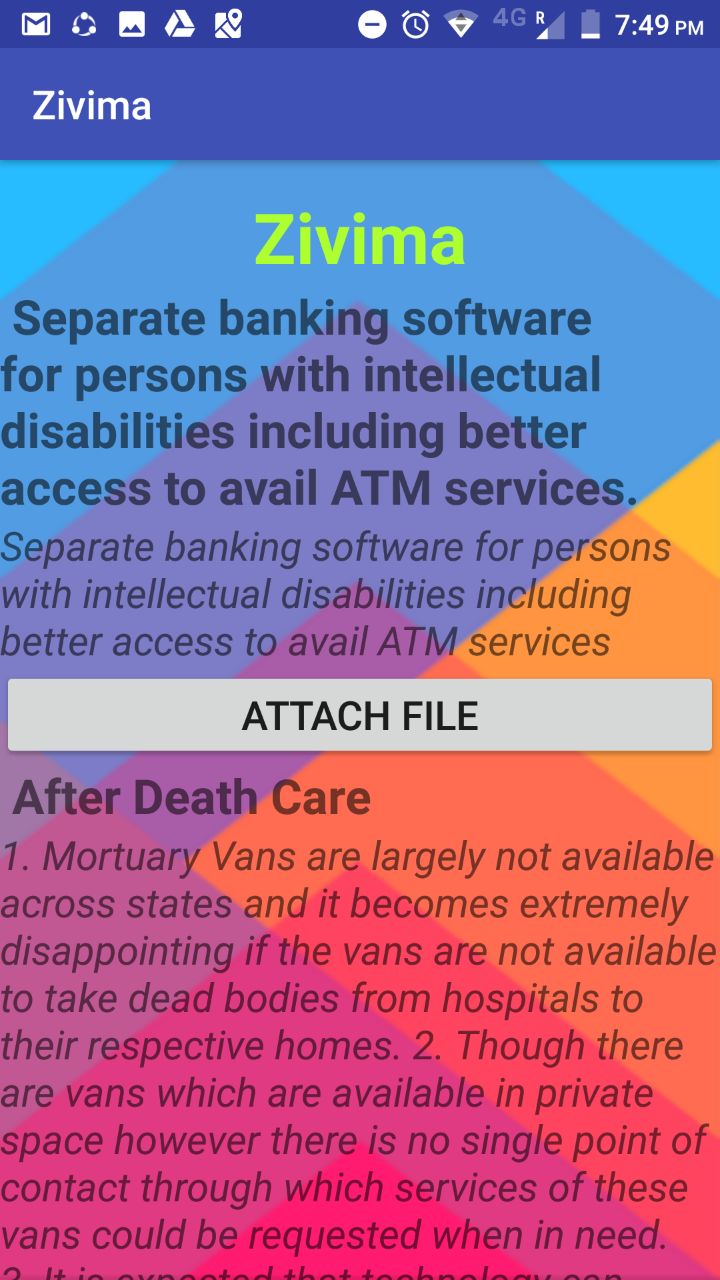
# **Fig . 4.1 Login Page**

* Once the student login in successfully he/she will navigate to the Ministry Selection page.
* Here several departments of problem statements are listed from which student can select their own ministry



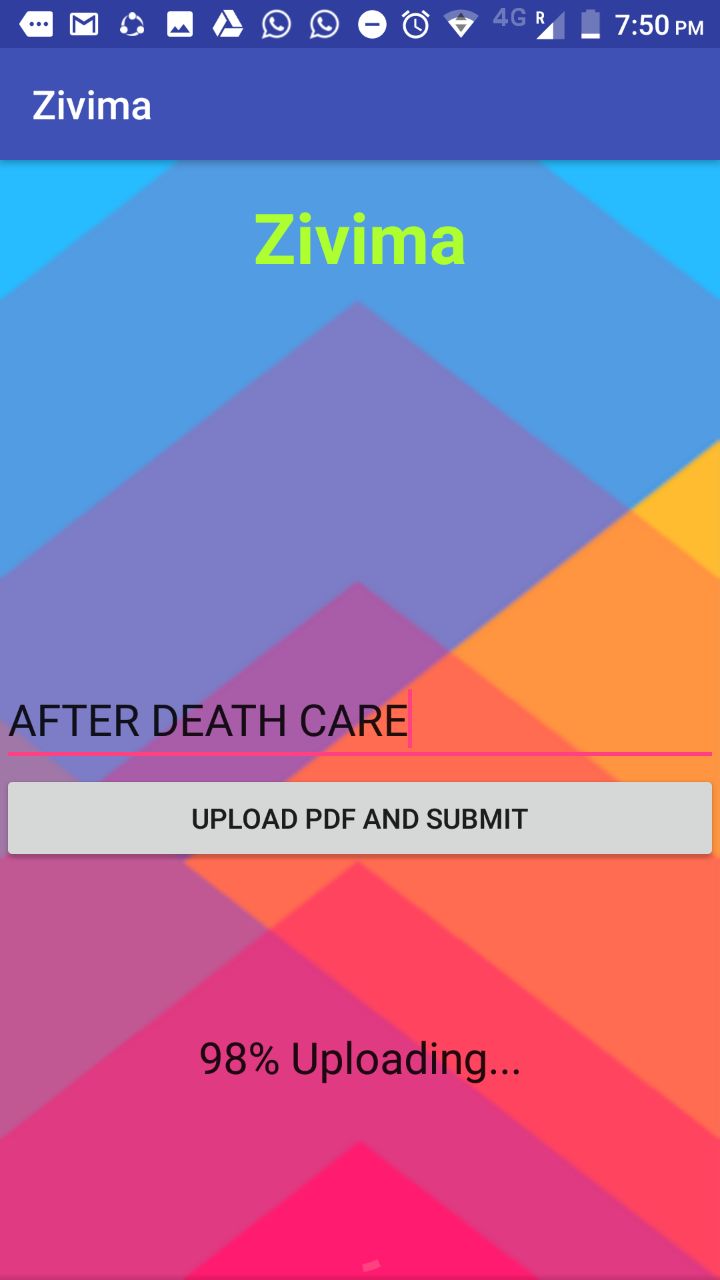
**Fig. 4.2 Ministry selection Page**

* Once the ministry is chosen a list of problem statements will be displayed corresponding to the selected ministry.
* Student can view all the problem statements and select according to their ideas.
* Once they completed the abstract for the chosen prototype they can attach it to database by clicking the attach file button.



**Fig.4.3 Problem Statement Page**

* Student can type the name of their abstract in the Text field and can attach the PDF file by clicking the Upload PDF and Submit Button
* The PDF will be stored in the database with the respective student details which can be viewed and downloaded by the district authority.



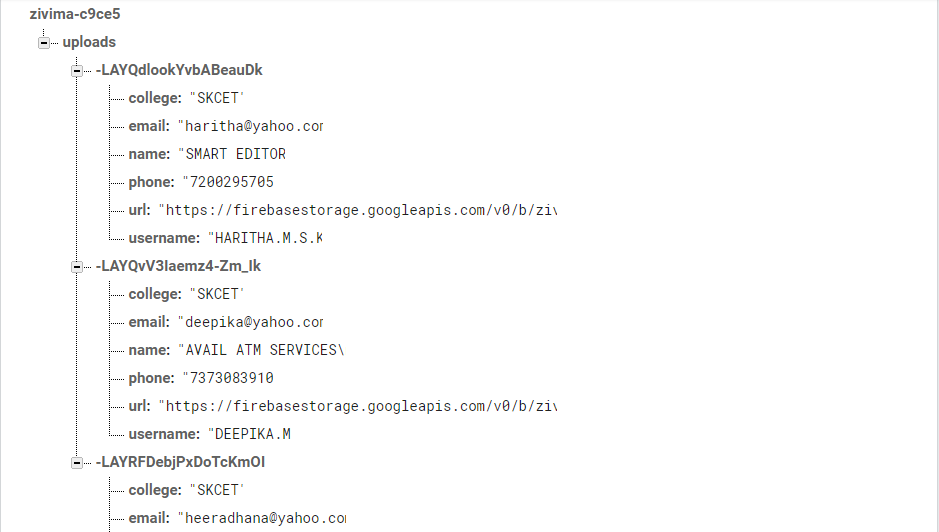
**Fig.4.4 PDF Uploading Page**

**4.2 DATABASE SCHEMA**

* This database schema consists of database named ‘ZIVIMA’ and have three table database, Storage, User.
* These tables have the fields specified in the related forms.

**4.2.1 STUDENT DATABASE TABLE:**

* It is used to store the students details while they create a new account by registration.
* It consists of six fields. They are User name , College Name, Email ID , Phone Number, URL and Abstract Name.



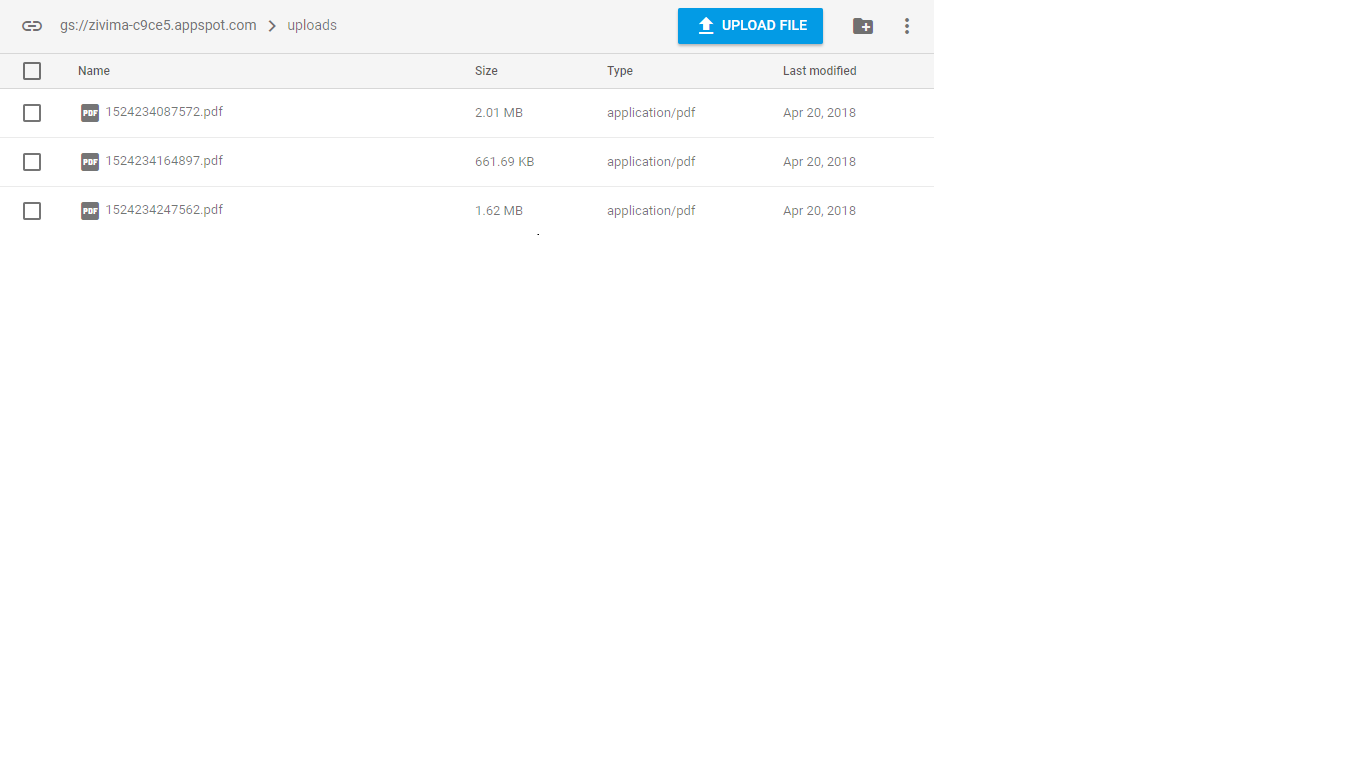
**Fig.4.5 Database Table**

**Table 4.1 Structure of tables of Student Database**

|  |  |  |
| --- | --- | --- |
| **FIELD NAME** | **DATATYPE** | **DESCRIPTION** |
| User Name | VARCHAR | Name of the Student |
| College | VARCHAR | Name of the College |
| Email | VARCHAR | Mail ID of the student |
| Name | VARCHAR | Name of the abstract |
| Phone | INT | Phone Number of the Student |
| URL | VARCHAR | URL of the submission report |

**4.2.2 STORAGE TABLE:**

* It consists of collection of anstract submitted by the students from different colleges.
* This table consists of four fields such as Name, Size,Type,Last Modified.

****

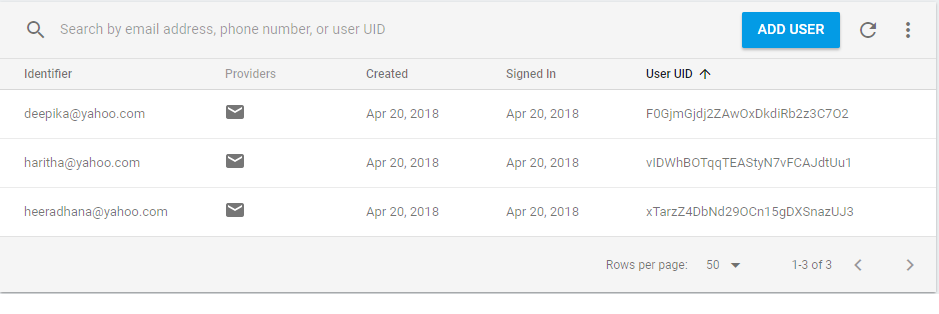
**Fig 4.6 Storage Table**

**Table 4.2 Structure Of Storage table**

|  |  |  |
| --- | --- | --- |
| **FIELD NAME** | **DATATYPE** | **DESCRIPTION** |
| Name | VARCHAR | Name of abstract submitted by the student. |
| Size | VARCHAR | Size of the abstract |
| Type | VARCHAR | Type of the abstract |
| Last Modified | DATE | Date of Submission |

**4.2.3 USER TABLE**

* It consists of list of users i.e students who had registered for the project submission.
* This table consists of Identifier, Created , Signed in, User UID.

****

**Fig 4.7 User Table**

**Table 4.3 Structure Of User Table**

|  |  |  |
| --- | --- | --- |
| **FIELD NAME** | **DATATYPE** | **DESCRIPTION** |
| Identifier | VARCHAR | Student mail ID |
| Created | DATE | Date on which student Create an account |
| Signed in | DATE | Date on which student signed in |
| User UID | VARCHAR | URL of the student account. |

**4.3 ARCHITECTURAL FLOW DIAGRAM**

**4.3.1 STUDENT’S ARCHITECTURAL DIAGRAM**

Main Activity

Selecting Ministries

Student Sign In

If a new User

Student Sign Up

Farmer Welfare Ministry

Health Ministry

Home Science Ministry



Abstract Submission

REAL TIME DATABASE

storage Firebase

**Fig 4.8 Student Architectural Diagram**

Students details along with abstract submitted

Admin Sign In

Main Activity



Download Abstract

Real Time Database

 FIREBASE STORAGE

**Fig.4.9 Admin Architectural Diagram**

**CHAPTER 5**

**IMPLEMENTATION**

**5.1 FRONT END**

* + Java – Android App(Versions 2.3 or higher)
  + Front end is designed by XML.

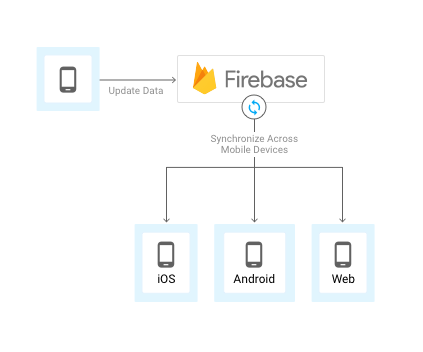
**FEATURES**

* Applications are developed in the Java language using the Android software development kit (SDK).
* The SDK includes a comprehensive set of development tools, including a debugger, software libraries, a handset emulator based on QEMU, documentation, sample code, and tutorials.
* The officially supported integrated development environment (IDE) is Eclipse using the Android Development Tools (ADT) plugin.
* Other development tools are available, including a Native Development Kit for applications or extensions in C or C++, Google App Inventor, a visual environment for novice programmers, and various cross platform mobile web applications frameworks.
* It include tools that interface with Android platform ,such as adb ,fastboot, and systrace.
* These tools are required for Android app development. They are also needed if you want to unlock your device boot loader and flash it with a new system image.

**5.2 BACK END :**

### FIREBASE :

[Firebase](https://firebase.google.com/) is a fully managed platform for building iOS, Android, and web apps that provides automatic data synchronization, authentication services, messaging, file storage, analytics, and more. Starting with Firebase is an efficient way to build or prototype mobile backend services.



**Recommended for:**

* Limiting on-device data storage by storing JSON data in the Firebase Realtime Database and files in Firebase Storage.
* Sending notifications with Firebase Cloud Messaging.
* Automated real-time data synchronization across multiple devices.
* Gracefully handling the offline case.
* Authenticating users through a variety of identity providers.
* Rapid development of a backend service.

**Not recommended for:**

* Apps that need a backend service to modify the synchronized data.

**5.3 CODING FOR THE APPLICATION**

**5.3.1. XML CODE**

**MAIN\_ACTIVITY XML**

*<?*xml version="1.0" encoding="utf-8"*?>*<android.support.constraint.ConstraintLayout xmlns:android="<http://schemas.android.com/apk/res/android>"  
 xmlns:app="<http://schemas.android.com/apk/res-auto>"  
 xmlns:tools="<http://schemas.android.com/tools>"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 android:background="@drawable/back8"  
 tools:context="com.example.admin.zivima.MainActivity">  
  
  
 <RelativeLayout  
 android:layout\_width="368dp"  
 android:layout\_height="495dp"  
 tools:layout\_editor\_absoluteX="8dp"  
 tools:layout\_editor\_absoluteY="0dp">  
 <TextView  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:text="@string/app\_name"  
 android:textSize="35sp"  
 android:textAlignment="center"  
 android:textStyle="bold"  
 android:id="@+id/heading"  
 android:textColor="#ADFF2F"  
 android:layout\_marginLeft="20dp"  
 android:layout\_marginRight="20dp"  
 android:layout\_marginTop="15dp"  
 android:layout\_alignParentTop="true"  
 android:layout\_centerHorizontal="true" />  
 <Button  
 android:id="@+id/button"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:layout\_alignParentBottom="true"  
 android:layout\_alignParentLeft="true"  
 android:layout\_alignParentStart="true"  
 android:layout\_marginBottom="203dp"  
 android:layout\_marginLeft="72dp"  
 android:layout\_marginStart="72dp"  
 android:onClick="GotoSecond"  
 android:text="Admin" />  
  
 <Button  
 android:id="@+id/button2"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:layout\_alignBaseline="@+id/button"  
 android:layout\_alignBottom="@+id/button"  
 android:layout\_marginLeft="55dp"  
 android:layout\_marginStart="55dp"  
 android:layout\_toEndOf="@+id/button"  
 android:layout\_toRightOf="@+id/button"  
 android:onClick="GotoSecond"  
 android:text="Student" />  
 </RelativeLayout>  
</android.support.constraint.ConstraintLayout>

**ADMIN LOGIN**

*<?*xml version="1.0" encoding="utf-8"*?>*<RelativeLayout xmlns:android="<http://schemas.android.com/apk/res/android>"  
 xmlns:app="<http://schemas.android.com/apk/res-auto>"  
 xmlns:tools="<http://schemas.android.com/tools>"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 android:background="@drawable/back8"  
 tools:context="com.example.admin.zivima.AdminLogin">  
 <TextView  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:text="@string/app\_name"  
 android:textSize="35sp"  
 android:textAlignment="center"  
 android:textStyle="bold"  
 android:id="@+id/heading"  
 android:textColor="#ADFF2F"  
 android:layout\_marginLeft="20dp"  
 android:layout\_marginRight="20dp"  
 android:layout\_marginTop="15dp"  
 android:layout\_alignParentTop="true"  
 android:layout\_centerHorizontal="true" />  
 <TextView  
 android:id="@+id/textView"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:layout\_alignBottom="@+id/editText"  
 android:layout\_alignLeft="@+id/textView2"  
 android:layout\_alignStart="@+id/textView2"  
 android:text="Email"  
 android:textSize="22sp"  
 tools:layout\_editor\_absoluteX="30dp"  
 tools:layout\_editor\_absoluteY="163dp" />  
  
 <TextView  
 android:id="@+id/textView2"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:layout\_alignBottom="@+id/editText2"  
 android:layout\_alignParentLeft="true"  
 android:layout\_alignParentStart="true"  
 android:layout\_marginLeft="16dp"  
 android:layout\_marginStart="16dp"  
 android:text="Password"  
 android:textSize="22sp"  
 tools:layout\_editor\_absoluteX="30dp"  
 tools:layout\_editor\_absoluteY="208dp" />  
  
 <EditText  
 android:id="@+id/editText"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:layout\_above="@+id/editText2"  
 android:layout\_alignParentEnd="true"  
 android:layout\_alignParentRight="true"  
 android:layout\_marginBottom="28dp"  
 android:ems="10"  
 android:inputType="textEmailAddress"  
 tools:layout\_editor\_absoluteX="141dp"  
 tools:layout\_editor\_absoluteY="150dp" />  
  
 <EditText  
 android:id="@+id/editText2"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:layout\_alignParentEnd="true"  
 android:layout\_alignParentRight="true"  
 android:layout\_centerVertical="true"  
 android:ems="10"  
 android:inputType="textPassword"  
 tools:layout\_editor\_absoluteX="141dp"  
 tools:layout\_editor\_absoluteY="193dp" />  
  
 <Button  
 android:id="@+id/button3"  
 android:layout\_width="102dp"  
 android:layout\_height="55dp"  
 android:layout\_alignParentBottom="true"  
 android:layout\_centerHorizontal="true"  
 android:layout\_marginBottom="118dp"  
 android:text="Login"  
 android:clickable="true"  
 android:onClick="nextLayout"  
 android:textSize="17sp"  
 tools:layout\_editor\_absoluteX="136dp"  
 tools:layout\_editor\_absoluteY="300dp" />  
</RelativeLayout>

**MININSTRY LIST**

*<?*xml version="1.0" encoding="utf-8"*?>*<RelativeLayout xmlns:android="<http://schemas.android.com/apk/res/android>"  
 xmlns:app="<http://schemas.android.com/apk/res-auto>"  
 xmlns:tools="<http://schemas.android.com/tools>"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 android:background="@drawable/back8"  
 tools:context="com.example.admin.zivima.Ministrieslist">  
 <TextView  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:text="@string/app\_name"  
 android:textSize="35sp"  
 android:textAlignment="center"  
 android:textStyle="bold"  
 android:id="@+id/heading"  
 android:textColor="#ADFF2F"  
 android:layout\_marginLeft="20dp"  
 android:layout\_marginRight="20dp"  
 android:layout\_marginTop="15dp"  
 android:layout\_alignParentTop="true"  
 android:layout\_centerHorizontal="true" />  
 <Spinner  
 android:id="@+id/spinner1"  
 android:layout\_width="fill\_parent"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginTop="73dp"  
 android:entries="@array/Ministries"  
 android:prompt="@string/department\_prompt"  
 android:visibility="visible"  
 android:layout\_below="@+id/heading"  
 android:layout\_alignParentLeft="true"  
 android:layout\_alignParentStart="true" />  
  
 <Button  
 android:id="@+id/button1"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginTop="95dp"  
 android:onClick="goto"  
 android:text="Submit"  
 android:visibility="visible"  
 android:layout\_below="@+id/spinner1"  
 android:layout\_alignLeft="@+id/heading"  
 android:layout\_alignStart="@+id/heading" />  
  
 <TextView  
 android:id="@+id/textView6"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:layout\_alignParentLeft="true"  
 android:layout\_alignParentStart="true"  
 android:textSize="17dp"  
 android:textStyle="bold"  
 android:layout\_below="@+id/heading"  
 android:layout\_marginLeft="13dp"  
 android:layout\_marginStart="13dp"  
 android:layout\_marginTop="29dp"  
 android:text="Choose a Ministry" />  
</RelativeLayout>

**EDUCATION\_MINISTRY:**

*<?*xml version="1.0" encoding="utf-8"*?>*<RelativeLayout xmlns:android="<http://schemas.android.com/apk/res/android>"  
 xmlns:app="<http://schemas.android.com/apk/res-auto>"  
 xmlns:tools="<http://schemas.android.com/tools>"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 android:background="@drawable/back8"  
 tools:context="com.example.admin.zivima.Education">  
 <TextView  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:text="@string/app\_name"  
 android:textSize="35sp"  
 android:textAlignment="center"  
 android:textStyle="bold"  
 android:id="@+id/heading"  
 android:textColor="#ADFF2F"  
 android:layout\_marginLeft="20dp"  
 android:layout\_marginRight="20dp"  
 android:layout\_marginTop="15dp"  
 android:layout\_alignParentTop="true"  
 android:layout\_centerHorizontal="true" />  
 <EditText  
 android:id="@+id/editTextFileName"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:layout\_centerVertical="true"  
 android:hint="Enter a name for your file"  
 android:textAppearance="@style/Base.TextAppearance.AppCompat.Large" />  
  
 <Button  
 android:id="@+id/buttonUploadFile"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:layout\_below="@id/editTextFileName"  
 android:text="Upload PDF And Submit" />  
<LinearLayout  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:layout\_alignParentBottom="true"  
 android:orientation="vertical"  
 android:id="@+id/linearLayout">  
  
  
 <ProgressBar  
 android:id="@+id/progressbar"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:layout\_gravity="center\_horizontal"  
 android:visibility="gone" />  
  
 </LinearLayout>  
  
 <TextView  
 android:id="@+id/textViewStatus"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:layout\_above="@+id/linearLayout"  
 android:layout\_alignParentLeft="true"  
 android:layout\_alignParentStart="true"  
 android:layout\_marginBottom="49dp"  
 android:text="No file selected"  
 android:textAlignment="center"  
 android:textAppearance="@style/Base.TextAppearance.AppCompat.Large" />  
</RelativeLayout>

**VIEW UPLOADS**

*<?*xml version="1.0" encoding="utf-8"*?>*<RelativeLayout xmlns:android="<http://schemas.android.com/apk/res/android>"  
 xmlns:app="<http://schemas.android.com/apk/res-auto>"  
 xmlns:tools="<http://schemas.android.com/tools>"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 android:background="@drawable/back8"  
 tools:context="com.example.admin.zivima.ViewUploadsActivity">  
 <TextView  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:text="@string/app\_name"  
 android:textSize="35sp"  
 android:textAlignment="center"  
 android:textStyle="bold"  
 android:id="@+id/heading"  
 android:textColor="#ADFF2F"  
 android:layout\_marginLeft="20dp"  
 android:layout\_marginRight="20dp"  
 android:layout\_marginTop="15dp"  
 android:layout\_alignParentTop="true"  
 android:layout\_centerHorizontal="true" />  
 <ListView  
 android:id="@+id/listView"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:layout\_below="@+id/heading"  
 android:layout\_alignParentLeft="true"  
 android:layout\_alignParentStart="true" />  
</RelativeLayout>

**5.3.2 JAVA CODE**

**MAINACTIVITY.JAVA**

**package** com.example.admin.zivima;  
  
**import** android.content.Intent;  
**import** android.support.v7.app.AppCompatActivity;  
**import** android.os.Bundle;  
**import** android.view.View;  
**import** android.widget.Button;  
  
**public class** MainActivity **extends** AppCompatActivity {  
  
 @Override  
 **protected void** onCreate(Bundle savedInstanceState) {  
 **super**.onCreate(savedInstanceState);  
 setContentView(R.layout.***activity\_main***);  
 }  
 **public void** GotoSecond(View view){  
 String button\_text;  
 button\_text=((Button) view).getText().toString();  
 **if**(button\_text.equals(**"Student"**)){  
 Intent intent = **new** Intent(**this**,Main2Activity.**class**);  
 startActivity(intent);  
 }  
 **else if**(button\_text.equals(**"Admin"**)){  
 Intent intent = **new** Intent(**this**,AdminLogin.**class**);  
 startActivity(intent);  
 }  
 }  
}

**ADMIN LOGIN**

**package** com.example.admin.zivima;  
  
**import** android.content.Intent;  
**import** android.support.v7.app.AppCompatActivity;  
**import** android.os.Bundle;  
**import** android.view.View;  
**import** android.widget.Button;  
**import** android.widget.EditText;  
**import** android.widget.TextView;  
**import** android.widget.Toast;  
  
**public class** AdminLogin **extends** AppCompatActivity {  
 EditText **login**,**pwd**;  
 @Override  
 **protected void** onCreate(Bundle savedInstanceState) {  
 **super**.onCreate(savedInstanceState);  
 setContentView(R.layout.***activity\_admin\_login***);  
 }  
 **public void** nextLayout(View view){  
 **login**=(EditText) findViewById(R.id.***editText***);  
 **pwd**=(EditText) findViewById(R.id.***editText2***);  
 String loginid=**login**.getText().toString();  
 String pass=**pwd**.getText().toString();  
 String button\_text;  
 button\_text=((TextView) view).getText().toString();  
 **if**(button\_text.equalsIgnoreCase(**"Login"**)){  
 **if**(loginid.equals(**"**[**admin@skcet.ac.in**](mailto:admin@skcet.ac.in)**"**)&& pass.equals(**"admin"**)) {  
 Intent intent = **new** Intent(**this**, ViewUploadsActivity.**class**);  
 startActivity(intent);  
 }  
 **else**{  
 Toast.*makeText*(**this**,**"Incorrect Password or login ID"**,Toast.***LENGTH\_SHORT***).show();  
 }  
 }  
 }  
}

**STUDENT LOGIN**

**package** com.example.admin.zivima;  
  
**import** android.app.ProgressDialog;  
**import** android.nfc.Tag;  
**import** android.support.annotation.NonNull;  
**import** android.support.v7.app.AppCompatActivity;  
**import** android.os.Bundle;  
**import** android.content.Intent;  
**import** android.text.TextUtils;  
**import** android.util.Log;  
**import** android.widget.Button;  
**import** android.widget.EditText;  
**import** android.widget.TextView;  
  
**import** android.view.View;  
**import** android.widget.Toast;  
  
  
**import** com.google.android.gms.tasks.OnCompleteListener;  
**import** com.google.android.gms.tasks.Task;  
**import** com.google.firebase.auth.AuthResult;  
**import** com.google.firebase.auth.FirebaseAuth;  
**import** com.google.firebase.auth.FirebaseUser;  
  
**public class** Main2Activity **extends** AppCompatActivity {  
  
 EditText **etEmail**,**etPassword**;  
 **public static final** String ***TAG*** = Main2Activity.**class**.getSimpleName();  
 **private** FirebaseAuth **mAuth**;  
 **private** FirebaseAuth.AuthStateListener **mAuthListener**;  
 @Override  
 **protected void** onCreate(Bundle savedInstanceState) {  
  
  
  
 **super**.onCreate(savedInstanceState);  
 setContentView(R.layout.***activity\_main2***);  
  
 **mAuth** = FirebaseAuth.*getInstance*();  
  
 **mAuthListener** = **new** FirebaseAuth.AuthStateListener() {  
 @Override  
 **public void** onAuthStateChanged(@NonNull FirebaseAuth firebaseAuth) {  
 FirebaseUser user = firebaseAuth.getCurrentUser();  
 **if** (user != **null**) {  
 *// User is signed in* Log.*d*(***TAG***, **"onAuthStateChanged:signed\_in:"** + user.getUid());  
 } **else** {  
 *// User is signed out* Log.*d*(***TAG***, **"onAuthStateChanged:signed\_out"**);  
 }  
 *// ...* }  
 };  
  
  
 }  
  
 @Override  
 **public void** onStart() {  
 **super**.onStart();  
 **mAuth**.addAuthStateListener(**mAuthListener**);  
 }  
  
 @Override  
 **public void** onStop() {  
 **super**.onStop();  
 **if** (**mAuthListener** != **null**) {  
 **mAuth**.removeAuthStateListener(**mAuthListener**);  
 }  
 }  
 **public void** signIn(){  
 **etEmail** = (EditText) findViewById(R.id.***editText***);  
 **etPassword** = (EditText) findViewById(R.id.***editText2***);  
 String email=**etEmail**.getText().toString();  
 String password=**etPassword**.getText().toString();  
 **int** c=1;  
 **mAuth**.signInWithEmailAndPassword(email,password)  
 .addOnCompleteListener(**this**, **new** OnCompleteListener<AuthResult>() {  
 @Override  
 **public void** onComplete(@NonNull Task<AuthResult> task) {  
 Log.*d*(***TAG***, **"signInWithEmail:onComplete:"** + task.isSuccessful());  
  
 *// If sign in fails, display a message to the user. If sign in succeeds  
 // the auth state listener will be notified and logic to handle the  
 // signed in user can be handled in the listener.* **if** (!task.isSuccessful()) {  
 Log.*w*(***TAG***, **"signInWithEmail:failed"**, task.getException());  
 Toast.*makeText*(Main2Activity.**this**, **"Authentication falied"**,  
 Toast.***LENGTH\_SHORT***).show();  
  
 }  
  
 **else**{  
 Toast.*makeText*(Main2Activity.**this**,  
 **"Logged Successfully"**,  
 Toast.***LENGTH\_SHORT***).show();  
 startActivity(**new** Intent(getApplicationContext(),Ministrieslist.**class**));  
 }  
 }  
  
 });  
 }  
 **public void** Page(View view){  
 String button\_text;  
 button\_text=((Button) view).getText().toString();  
 **if**(button\_text.equals(**"Login"**)) {  
 *//Intent intent = new Intent(this, Main4Activity.class);* signIn();  
 }  
 }  
 **public void** Register(View view){  
 String button\_text;  
 button\_text=((TextView) view).getText().toString();  
 **if**(button\_text.equals(**"Create a New Account"**)){  
 Intent intent = **new** Intent(**this**,Main3Activity.**class**);  
 startActivity(intent);  
 }  
 }  
}

**REGISTRATIO N LAYOUT**

**package** com.example.admin.zivima;  
  
**import** android.app.ProgressDialog;  
**import** android.content.Intent;  
**import** android.nfc.Tag;  
**import** android.support.annotation.NonNull;  
**import** android.support.v4.util.LogWriter;  
**import** android.support.v7.app.AppCompatActivity;  
**import** android.os.Bundle;  
**import** android.text.TextUtils;  
**import** android.util.Log;  
**import** android.view.View;  
**import** android.widget.Button;  
**import** android.wi

dget.EditText;  
**import** android.widget.Toast;  
  
**import** com.google.android.gms.tasks.OnCompleteListener;  
**import** com.google.android.gms.tasks.Task;  
**import** com.google.firebase.FirebaseNetworkException;  
**import** com.google.firebase.auth.AuthResult;  
**import** com.google.firebase.auth.FirebaseAuth;  
**import** com.google.firebase.auth.FirebaseAuthActionCodeException;  
**import** com.google.firebase.auth.FirebaseAuthInvalidCredentialsException;  
**import** com.google.firebase.auth.FirebaseAuthInvalidUserException;  
**import** com.google.firebase.auth.FirebaseUser;  
**import** com.google.firebase.database.DatabaseError;  
**import** com.google.firebase.database.DatabaseReference;  
**import** com.google.firebase.database.FirebaseDatabase;  
  
**import** java.util.logging.Logger;  
  
**public class** Main3Activity **extends** AppCompatActivity {  
 EditText **etEmail**,**etPassword**,**uname**,**col**,**pnum**;  
  
 **public static final** String ***TAG*** = Main3Activity.**class**.getSimpleName();  
 **private** FirebaseAuth **mAuth**;  
 **private** FirebaseAuth.AuthStateListener **mAuthListener**;  
  
 **private** DatabaseReference **mDatabase**;  
 @Override  
 **protected void** onCreate(Bundle savedInstanceState) {  
  
  
  
 **super**.onCreate(savedInstanceState);  
 setContentView(R.layout.***activity\_main3***);  
 **mDatabase** = FirebaseDatabase.*getInstance*().getReference();  
 **mAuth** = FirebaseAuth.*getInstance*();  
 **uname**=(EditText)findViewById(R.id.***editText***);  
 **col**=(EditText)findViewById(R.id.***editText1***);  
 **pnum**=(EditText)findViewById(R.id.***editText4***);  
 **mAuthListener** = **new** FirebaseAuth.AuthStateListener() {  
 @Override  
 **public void** onAuthStateChanged(@NonNull FirebaseAuth firebaseAuth) {  
 FirebaseUser user = firebaseAuth.getCurrentUser();  
 **if** (user != **null**) {  
 *// User is signed in* Log.*d*(***TAG***, **"onAuthStateChanged:signed\_in:"** + user.getUid());  
 } **else** {  
 *// User is signed out* Log.*d*(***TAG***, **"onAuthStateChanged:signed\_out"**);  
 }  
 *// ...* }  
 };  
  
  
 }  
  
 @Override  
 **public void** onStart() {  
 **super**.onStart();  
 **mAuth**.addAuthStateListener(**mAuthListener**);  
 }  
  
 @Override  
 **public void** onStop() {  
 **super**.onStop();  
 **if** (**mAuthListener** != **null**) {  
 **mAuth**.removeAuthStateListener(**mAuthListener**);  
 }  
 }  
 **public void** createAccount(){  
 **etEmail** = (EditText) findViewById(R.id.***editText7***);  
 **etPassword** = (EditText) findViewById(R.id.***editText3***);  
 String email=**etEmail**.getText().toString();  
 String password=**etPassword**.getText().toString();  
 **mAuth**.createUserWithEmailAndPassword(email,password)  
 .addOnCompleteListener(**this**, **new** OnCompleteListener<AuthResult>() {  
 @Override  
 **public void** onComplete(@NonNull Task<AuthResult> task) {  
 Log.*d*(***TAG***, **"createUserWithEmail:onComplete:"** + task.isSuccessful());  
  
 *// If sign in fails, display a message to the user. If sign in succeeds  
 // the auth state listener will be notified and logic to handle the  
 // signed in user can be handled in the listener.* **if** (!task.isSuccessful()) {  
 Toast.*makeText*(Main3Activity.**this**, **"Authentication falied"**,  
 Toast.***LENGTH\_SHORT***).show();  
  
  
 }  
 **else**{  
Toast.*makeText*(Main3Activity.**this**,  
 **"Registered Successfully"**,  
 Toast.***LENGTH\_SHORT***).show();  
 Intent intend=**new** Intent(getApplicationContext(),Ministrieslist.**class**);  
 intend.putExtra(**"Editname"**,**uname**.getText().toString().toUpperCase());  
 intend.putExtra(**"Editcollege"**,**col**.getText().toString().toUpperCase());  
 intend.putExtra(**"Editpnum"**,**pnum**.getText().toString());  
 startActivity(intend);  
 }  
 *// ...* }  
 });  
 }  
 **public void** Ministry(View view){  
 String button\_text;  
 button\_text=((Button) view).getText().toString();  
 **if**(button\_text.equals(**"Register"**)) {  
  
 createAccount();  
 }  
 }  
}

**MINISTRY LIST**

**package** com.example.admin.zivima;  
  
**import** android.content.Intent;  
**import** android.support.v7.app.AppCompatActivity;  
**import** java.util.ArrayList;  
**import** java.util.List;  
**import** android.app.Activity;  
**import** android.os.Bundle;  
**import** android.view.View;  
**import** android.view.View.OnClickListener;  
**import** android.widget.AdapterView;  
**import** android.widget.ArrayAdapter;  
**import** android.widget.Button;  
**import** android.widget.Spinner;  
**import** android.widget.Toast;  
  
**public class** Ministrieslist **extends** AppCompatActivity {  
 Spinner **spinner**;  
 String[] **SPINNERVALUES** = {**"Education and Welfare"**,**"Farmer Welfare"**,**"Health and Social Welfare"**};  
 String **SpinnerValue**;  
 Button **GOTO**;  
 Intent **intent**;  
 @Override  
 **protected void** onCreate(Bundle savedInstanceState) {  
 **super**.onCreate(savedInstanceState);  
 setContentView(R.layout.***activity\_ministrieslist***);  
  
 **spinner** =(Spinner)findViewById(R.id.***spinner1***);  
  
 **GOTO** = (Button)findViewById(R.id.***button1***);  
  
 ArrayAdapter<String> adapter = **new** ArrayAdapter<String>(Ministrieslist.**this**, android.R.layout.***simple\_list\_item\_1***, **SPINNERVALUES**);  
  
 **spinner**.setAdapter(adapter);  
  
 *//Adding setOnItemSelectedListener method on spinner.* **spinner**.setOnItemSelectedListener(**new** AdapterView.OnItemSelectedListener() {  
 @Override  
 **public void** onItemSelected(AdapterView<?> parent, View view, **int** position, **long** id) {  
 **SpinnerValue** = (String)**spinner**.getSelectedItem();  
 }  
  
 @Override  
 **public void** onNothingSelected(AdapterView<?> parent) {  
  
 }  
 });  
 **GOTO**.setOnClickListener(**new** View.OnClickListener() {  
  
 @Override  
 **public void** onClick(View v) {  
 *//* ***TODO Auto-generated method stub* switch**(**SpinnerValue**){  
  
 **case "Education and Welfare"**:  
 **intent** = **new** Intent(Ministrieslist.**this**, Main4Activity.**class**);  
 **intent**.putExtra(**"Editname"**,getIntent().getStringExtra(**"Editname"**));  
 **intent**.putExtra(**"Editcollege"**,getIntent().getStringExtra(**"Editcollege"**));  
 **intent**.putExtra(**"Editpnum"**,getIntent().getStringExtra(**"Editpnum"**));  
 startActivity(**intent**);  
 **break**;  
  
 **case "Farmer Welfare"**:  
 **intent** = **new** Intent(Ministrieslist.**this**, FarmerWelfare.**class**);  
 **intent**.putExtra(**"Editname"**,getIntent().getStringExtra(**"Editname"**));  
 **intent**.putExtra(**"Editcollege"**,getIntent().getStringExtra(**"Editcollege"**));  
 **intent**.putExtra(**"Editpnum"**,getIntent().getStringExtra(**"Editpnum"**));  
 startActivity(**intent**);  
 **break**;  
  
 **case "Health and Social Welfare"**:  
 **intent** = **new** Intent(Ministrieslist.**this**, Health.**class**);  
 **intent**.putExtra(**"Editname"**,getIntent().getStringExtra(**"Editname"**));  
 **intent**.putExtra(**"Editcollege"**,getIntent().getStringExtra(**"Editcollege"**));  
 **intent**.putExtra(**"Editpnum"**,getIntent().getStringExtra(**"Editpnum"**));  
 startActivity(**intent**);  
 **break**;  
  
  
 }  
 }  
 });  
 }  
}

**FARMER WELFARE MINISTRY**

**package** com.example.admin.zivima;  
  
**import** android.content.Intent;  
**import** android.support.v7.app.AppCompatActivity;  
**import** android.os.Bundle;  
**import** android.view.View;  
**import** android.widget.TextView;  
  
**public class** FarmerWelfare **extends** AppCompatActivity {  
  
 @Override  
 **protected void** onCreate(Bundle savedInstanceState) {  
 **super**.onCreate(savedInstanceState);  
 setContentView(R.layout.***activity\_farmer\_welfare***);  
 }  
 **public void** submit(View view){  
 String button\_text;  
 button\_text=((TextView) view).getText().toString();  
 **if**(button\_text.equalsIgnoreCase(**"Attach File"**)){  
 Intent intent = **new** Intent(**this**,Education.**class**);  
 intent.putExtra(**"Editname"**,getIntent().getStringExtra(**"Editname"**));  
 intent.putExtra(**"Editcollege"**,getIntent().getStringExtra(**"Editcollege"**));  
 intent.putExtra(**"Editpnum"**,getIntent().getStringExtra(**"Editpnum"**));  
 startActivity(intent);  
 }  
 }  
}

**HEALTH MINISTRY**

**package** com.example.admin.zivima;  
  
**import** android.content.Intent;  
**import** android.support.v7.app.AppCompatActivity;  
**import** android.os.Bundle;  
**import** android.view.View;  
**import** android.widget.TextView;  
  
**public class** Health **extends** AppCompatActivity {  
  
 @Override  
 **protected void** onCreate(Bundle savedInstanceState) {  
 **super**.onCreate(savedInstanceState);  
 setContentView(R.layout.***activity\_health***);  
 }  
 **public void** submit(View view){  
 String button\_text;  
 button\_text=((TextView) view).getText().toString();  
 **if**(button\_text.equalsIgnoreCase(**"Attach File"**)){  
 Intent intent = **new** Intent(**this**,Education.**class**);  
 intent.putExtra(**"Editname"**,getIntent().getStringExtra(**"Editname"**));  
 intent.putExtra(**"Editcollege"**,getIntent().getStringExtra(**"Editcollege"**));  
 intent.putExtra(**"Editpnum"**,getIntent().getStringExtra(**"Editpnum"**));  
 startActivity(intent);  
 }  
 }  
}

**UPLOAD PDF LAYOUT**

**package** com.example.admin.zivima;  
**public class** Upload {  
 **public** String **name**;  
 **public** String **url**;  
 **public** String **username**;  
 **public** String **college**;  
 **public** String **phone**;  
**public** Upload() {  
 }  
  
 **public** Upload(String name, String url) {  
 **this**.**name** = name;  
 **this**.**url** = url;  
 }  
 **public** Upload(String name,String url,String username,String college,String phone){  
 **this**.**name**=name;  
 **this**.**url**=url;  
 **this**.**username**=username;  
 **this**.**college**=college;  
 **this**.**phone**=phone;  
 }  
 **public** String getUsername(){  
 **return username**;  
 }  
 **public** String getCollege() {  
 **return college**;  
 }  
 **public** String getPhone() {  
 **return phone**;  
 }  
 **public** String getName() {  
 **return name**;  
 }  
  
 **public** String getUrl() {  
 **return url**;  
 }  
}

**VIEW LAYOUTS**

**package** com.example.admin.zivima;  
  
**import** android.content.Intent;  
**import** android.net.Uri;  
**import** android.support.v7.app.AppCompatActivity;  
**import** android.os.Bundle;  
**import** android.view.View;  
**import** android.widget.AdapterView;  
**import** android.widget.ArrayAdapter;  
**import** android.widget.ListView;  
  
**import** com.google.firebase.database.DataSnapshot;  
**import** com.google.firebase.database.DatabaseError;  
**import** com.google.firebase.database.DatabaseReference;  
**import** com.google.firebase.database.FirebaseDatabase;  
**import** com.google.firebase.database.ValueEventListener;  
  
**import** java.util.ArrayList;  
**import** java.util.List;  
  
**public class** ViewUploadsActivity **extends** AppCompatActivity {  
 *//the listview* ListView **listView**;  
  
 *//database reference to get uploads data* DatabaseReference **mDatabaseReference**;  
  
 *//list to store uploads data* List<Upload> **uploadList**;  
 @Override  
 **protected void** onCreate(Bundle savedInstanceState) {  
 **super**.onCreate(savedInstanceState);  
 setContentView(R.layout.***activity\_view\_uploads***);  
  
 **uploadList** = **new** ArrayList<>();  
 **listView** = (ListView) findViewById(R.id.***listView***);  
  
  
 *//adding a clicklistener on listview* **listView**.setOnItemClickListener(**new** AdapterView.OnItemClickListener() {  
 @Override  
 **public void** onItemClick(AdapterView<?> adapterView, View view, **int** i, **long** l) {  
 *//getting the upload* Upload upload = **uploadList**.get(i);  
  
 *//Opening the upload file in browser using the upload url* Intent intent = **new** Intent(Intent.***ACTION\_VIEW***);  
 intent.setData(Uri.*parse*(upload.getUrl()));  
 startActivity(intent);  
 }  
 });  
  
 *//getting the database reference* **mDatabaseReference** = FirebaseDatabase.*getInstance*().getReference(Constants.***DATABASE\_PATH\_UPLOADS***);  
  
 *//retrieving upload data from firebase database* **mDatabaseReference**.addValueEventListener(**new** ValueEventListener() {  
 @Override  
 **public void** onDataChange(DataSnapshot dataSnapshot) {  
 **for** (DataSnapshot postSnapshot : dataSnapshot.getChildren()) {  
 Upload upload = postSnapshot.getValue(Upload.**class**);  
  
 **uploadList**.add(upload);  
 }  
  
 String[] uploads = **new** String[**uploadList**.size()];  
 **for** (**int** i = 0; i < uploads.**length**; i++) {  
 uploads[i] = **"USERNAME : "**+**uploadList**.get(i).getUsername()+**"\nCOLLEGE : "**+**uploadList**.get(i).getCollege()+**"\nABSTRACT TITLE : "**+**uploadList**.get(i).getName()+**"\nPHONE NUMBER : "**+**uploadList**.get(i).getPhone();  
 }  
  
  
 *//displaying it to list* ArrayAdapter<String> adapter = **new** ArrayAdapter<String>(getApplicationContext(), android.R.layout.***simple\_list\_item\_1***, uploads);  
 **listView**.setAdapter(adapter);  
 }  
  
 @Override  
 **public void** onCancelled(DatabaseError databaseError) {  
  
 }  
 });  
 }  
}

**USER LAYOUT**

**package** com.example.admin.zivima;**public class** User {  
 **private** String **username**=**""**;  
 **private** String **college**=**""**;  
 **private** String **phone\_num**=**""**;  
  
 **public** String getUsername() {  
 **return username**;  
 }  
  
 **public void** setUsername(String username) {  
 **this**.**username** = username;  
 }  
  
 **public void** setCollege(String college) {  
 **this**.**college** = college;  
 }  
  
 **public void** setPhone\_num(String phone\_num) {  
 **this**.**phone\_num** = phone\_num;  
 }  
  
 **public** String getCollege() {  
  
 **return college**;  
 }  
  
 **public** String getPhone\_num() {  
 **return phone\_num**;  
 }  
  
 **public** User()  
 {  
  
 }  
  
}

**CHAPTER 6**

**TESTING**

**6.1 INTRODUCTION**

Software testing is a critical element of software quality assurance and represents the ultimate review of specification, design and coding. In fact, testing is the one step in the software engineering process that could be viewed as destructive rather than constructive. A strategy for software testing integrates software test case design methods into a well-planned series of steps that result in the successful construction of software. Testing is the set of activities that can be planned in advance and conducted systematically. The underlying motivation of program testing is to affirm software quality with methods that can economically and effectively apply to both strategic to both large and small-scale systems.

**6.2 STRATEGIC APPROACH TO SOFTWARE TESTING**

The software engineering process can be viewed as a spiral. Initially system engineering defines the role of software and leads to software requirement analysis where the information domain, functions, behaviour, performance, constraints and validation criteria for software are established. Moving inward along the spiral, we come to design and finally to coding. To develop computer software we spiral in along streamlines that decrease the level of abstraction on each turn.

A strategy for software testing may also be viewed in the context of the spiral. Unit testing begins at the vertex of the spiral and concentrates on each unit of the software as implemented in source code. Testing progress is done by moving outward along the spiral to integration testing, where the focus is on the design and the construction of the software architecture. Talking another turn on outward on the spiral we encounter validation testing where requirements established as part of software requirements analysis are validated against the software that has been constructed. Finally we arrive at system testing, where the software and other system elements are tested as a whole.

**6.3 UNIT TESTING**

Unit testing focuses verification effort on the smallest unit of software design, the module. The unit testing we have is white box oriented and some modules the steps are conducted in parallel.

**6.3.1 WHITE BOX TESTING**

To follow the concept of white box testing we have tested each form. We have created independently to verify that Data flow is correct and all conditions are exercised to check their validity. All loops are executed on their boundaries.

This type of testing ensures that

* All independent paths have been exercised at least once
* All logical decisions have been exercised on their true and false sides
* All loops are executed at their boundaries and within their operational bounds
* All internal data structures have been exercised to assure their validity paths.

**6.3.2 CONDITIONAL TESTING**

In this part of the testing each of the conditions were tested to both true and false aspects. And all the resulting paths were tested so that each path that may generate on a particular condition is traced to uncover any possible errors.

**6.3.3 DATA FLOW TESTING**

This type of testing selects the path of the program according to the location of definition and use of variables. This kind of testing was used only when some local variable were declared. The definition-use chain method was used in this type of testing. These were particularly useful in nested statements.

**6.3.4 LOOP TESTING**

In this type of testing all the loops are tested to all the limits possible. The following exercise was adopted for all loops:

* All the loops were tested at their limits, just above them and just below them. All the loops were skipped at least once.
* For nested loops test the inner most loop first and then work outwards.
* For concatenated loops the values of dependent loops were set with the help of connected loop.
* Unstructured loops were resolved into nested loops or concatenated loops and tested as above.
* Each unit has been separately tested by the development team itself and all the input have been validated.

**6.4 TEST CASE**

A test case, in software engineering, is a set of conditions or variables under ~which a tester will determine whether an application, software system or one of its features is working as it was originally established for it to do. The mechanism for determining whether a software program or system has passed or failed such a test is known as a *test oracle*. In some settings, an oracle could be a requirement or use case, while in others it could be a heuristic. It may take many test cases to determine that a software program or system is considered sufficiently scrutinized to be released. Test cases are often referred to as *test* *scripts*, particularly when written - when they are usually collected intotest suites.

**Table6.1 Brief Test Case Description**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **TEST** | **TEST** **DESCRIPTION** | **INPUT** | **EXPECTED RESULT** | **TEST RESULT** |
| 1 | Test the application when login page is opened. |  | System should display login screen with fields User ID and Password with Login button. | **Pass** |
| 2 | Test the application when the student clicks the login button after entering the correct details. | Email Id : user ID  Password : \*\*\*\*\*\*\* | The system should navigate to the page that displays the list of mininstry. | **Pass** |
| 3 | Test the System when the student clicks the login button without entering the correct details. |  | The System should show error indicating wrong user ID and password. | **Pass** |
| 4 | Test the functionality of 'User ID' field | User mailID | The system should accept the input and should not throw any errors. | **Pass** |
| 5 | Test the functionality of 'Password' field | Password  \*\*\*\*\*\*\* | The system should accept the input and should not throw any errors. | **Pass** |
| 6 | Test the system when the student choose the ministry. |  | The system must navigate to the problem statements corresponding to the chosen ministry. | **Pass** |
| 7 | Test the System when the student does not select, any required ministry. |  | The system must throw an error | **Pass** |
| 8 | Test the System when the student attach the file. |  | The system should generate the correct output. | **Pass** |
| 9 | Test the System when the student try to upload file without selecting any file. |  | The system must throw an error | **Pass** |

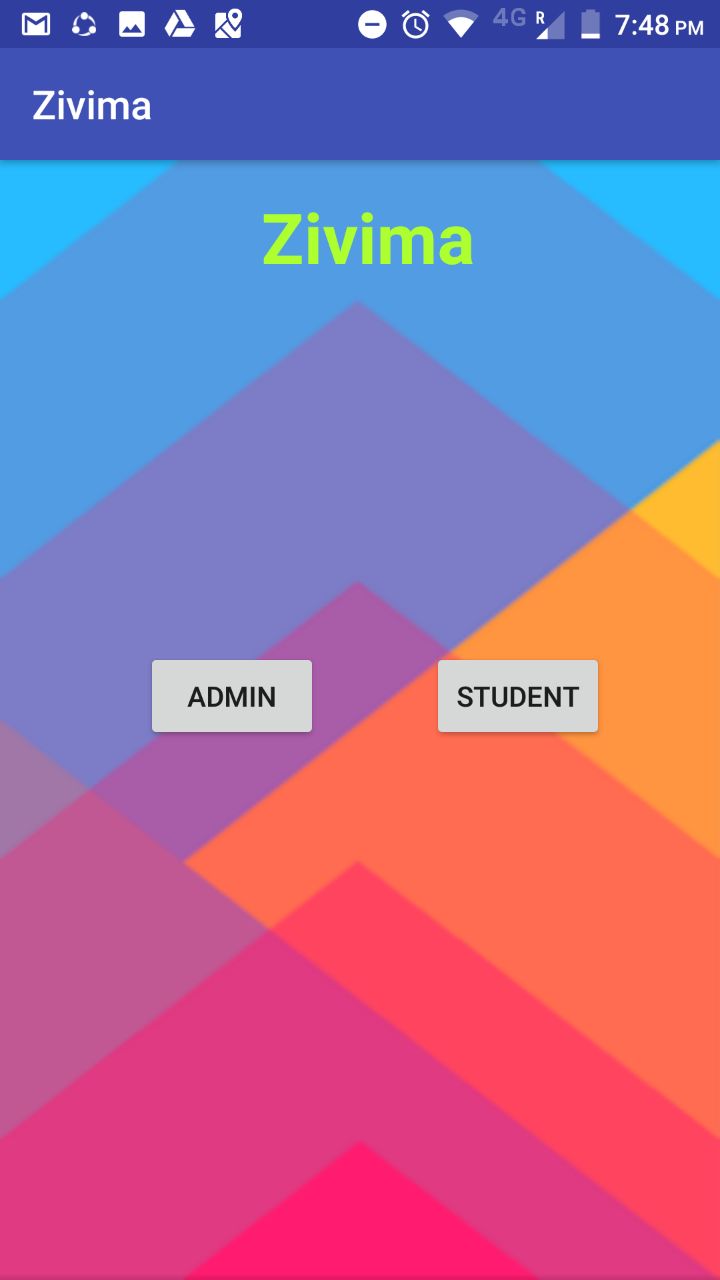
**Table6.2 Test Case Results**

|  |  |  |  |
| --- | --- | --- | --- |
| **S NO.** | **TEST SCENARIO** | **EXPECTED RESULT** | **TEST RESULT** |
| 1 | Username is incorrect  Password is correct | Authentication failed. | Authentication failed. |
| 2 | Username is correct  Password is incorrect | Authentication failed. | Authentication failed. |
| 3 | Both username and password incorrect | Authentication failed. | Authentication failed. |
| 4 | Both username and password correct | Redirect to next page | Redirect to next page |
| 5 | Retrieve Data  (if any problem) | Fetch Data | Couldn’t get data |
| 6 | Retrieve Data | Fetch Data | Data received at the front end |

**CHAPTER 7**

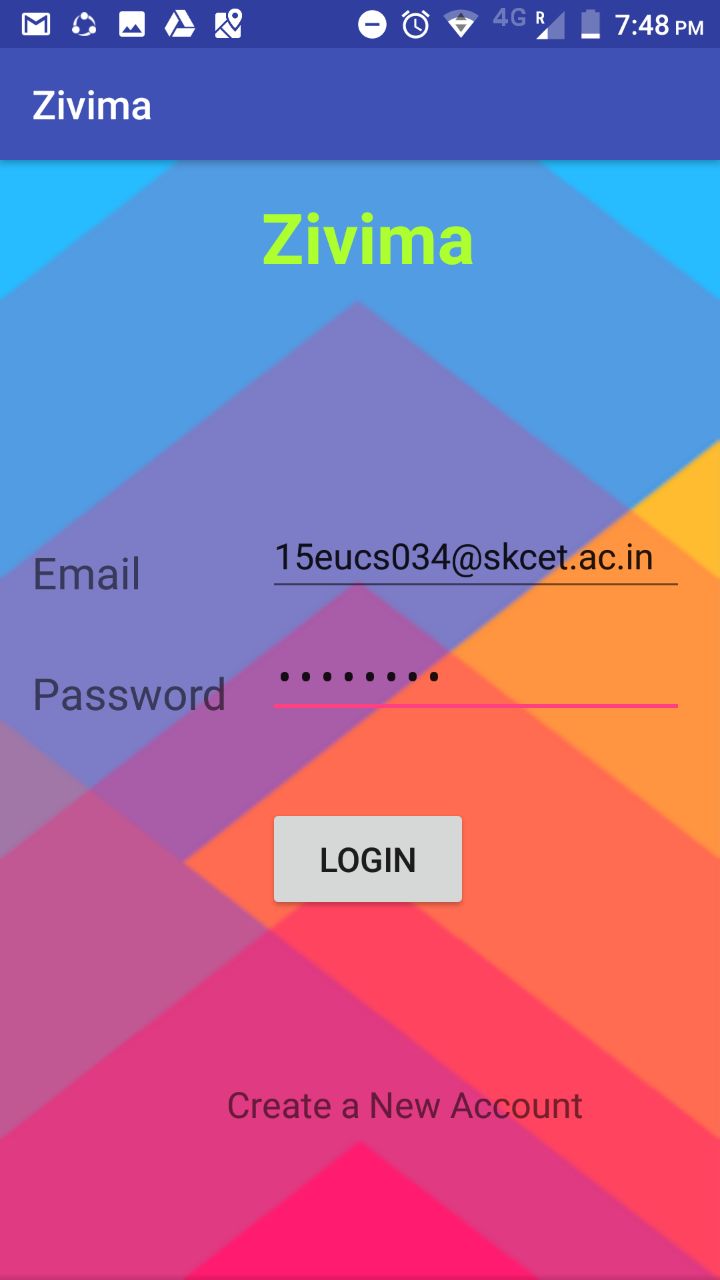
**SCREENSHOTS**

**7.1 HOME PAGE**

****

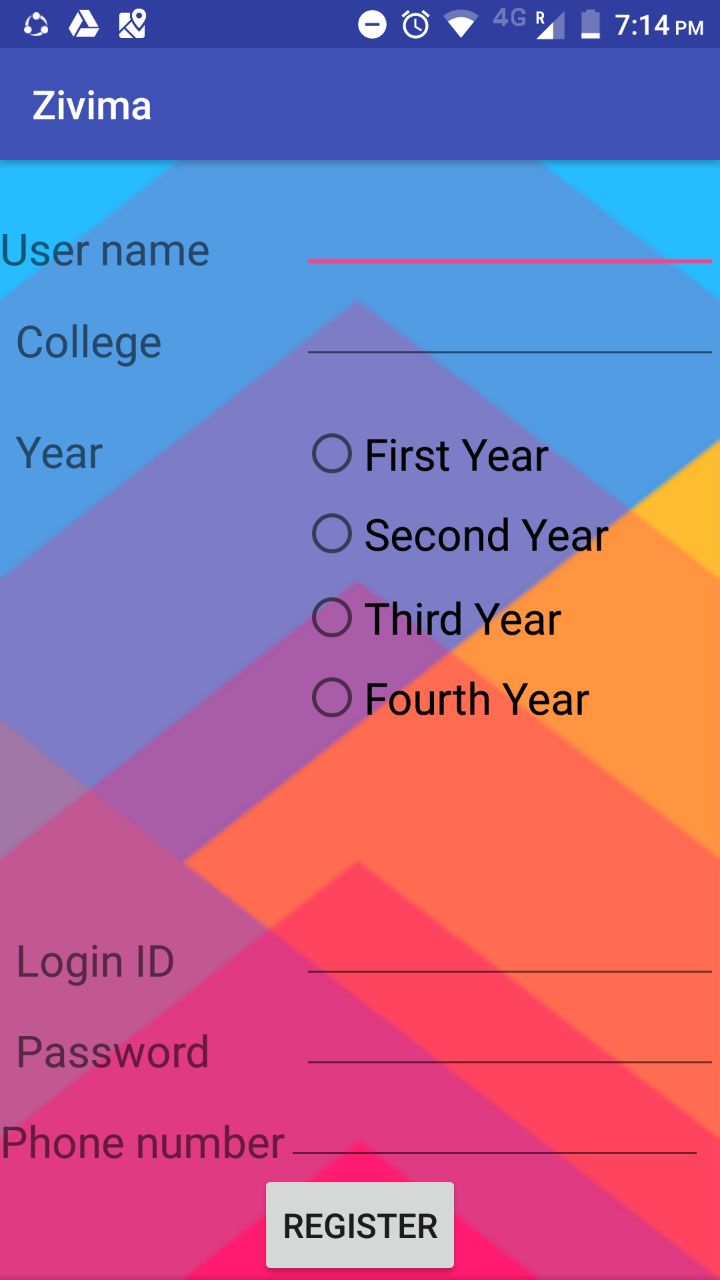
**Fig 7.1**

**7.2 STUDENT LOGIN PAGE**

****

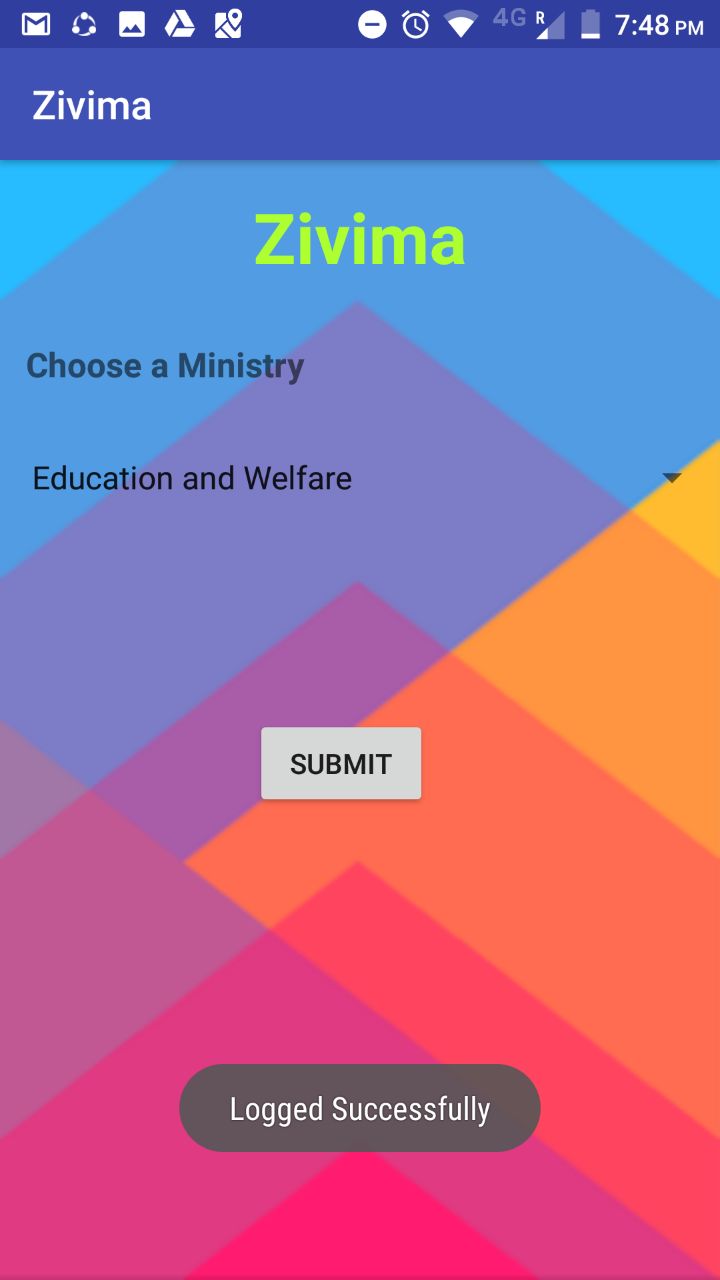
**Fig 7.2**

**7.3 STUDENT REGISTRATION PAGE**

****

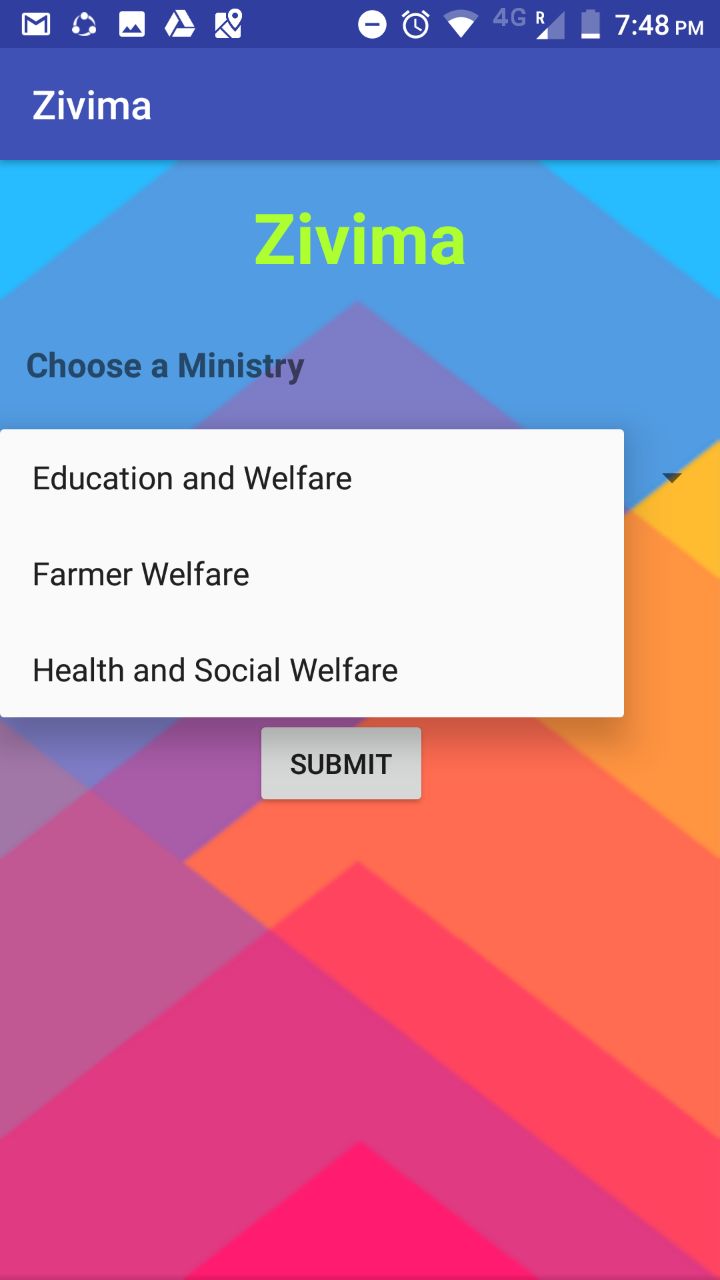
**Fig 7.3**

**7.4 AUTHENTICATION FOR SUCCESSFUL LOGIN**

****

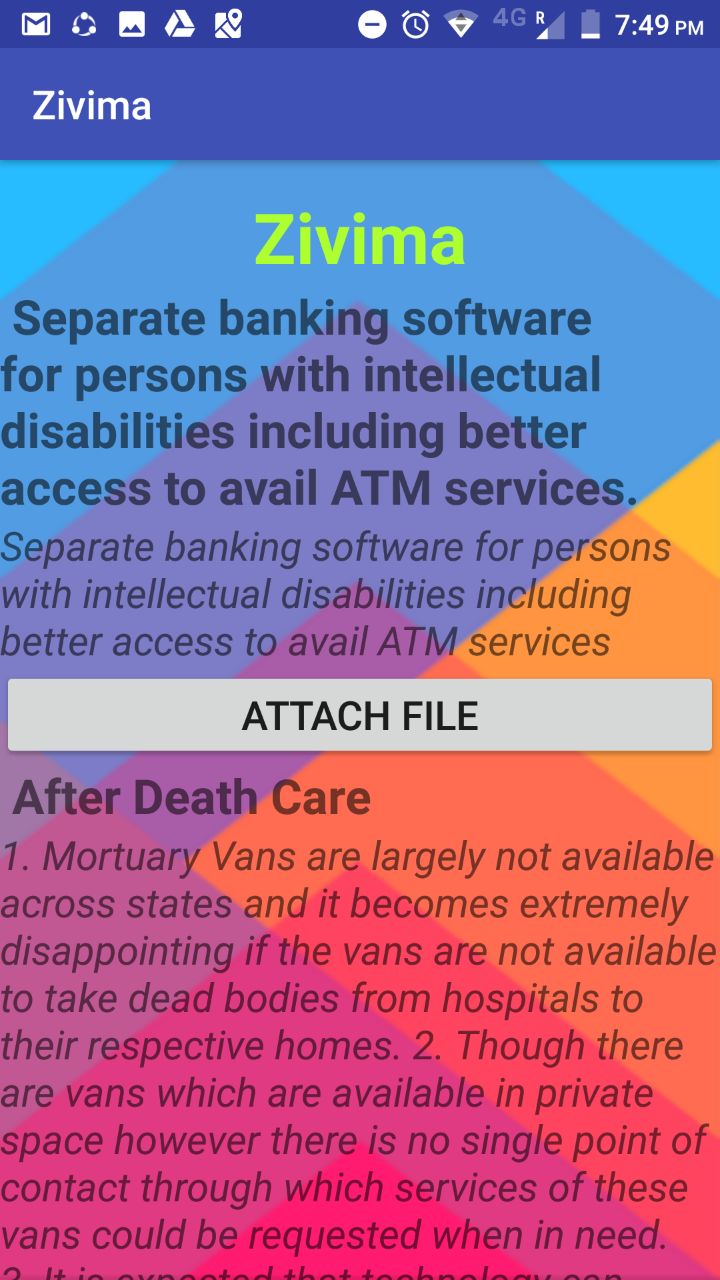
**Fig 7.4**

**7.5 SELECTING MINISTRY PAGE**

****

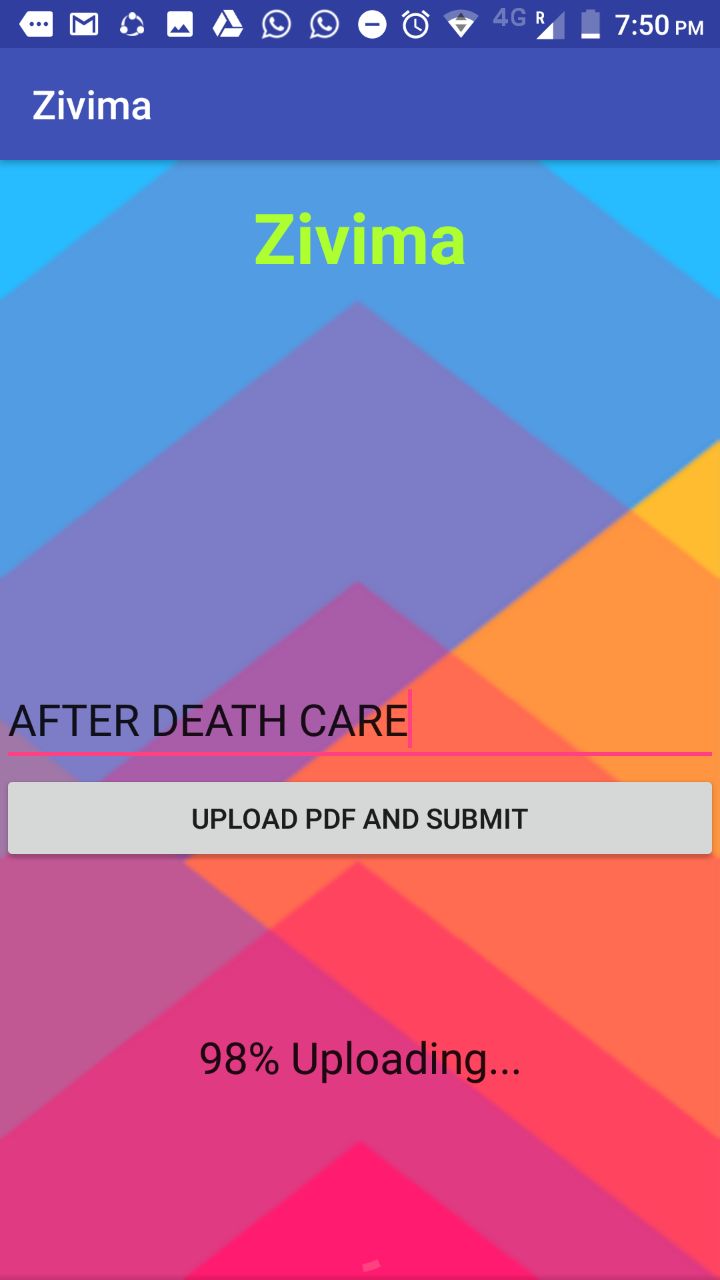
**Fig 7.5**

**7.6 PROBLEM STATEMENT PAGE**

****

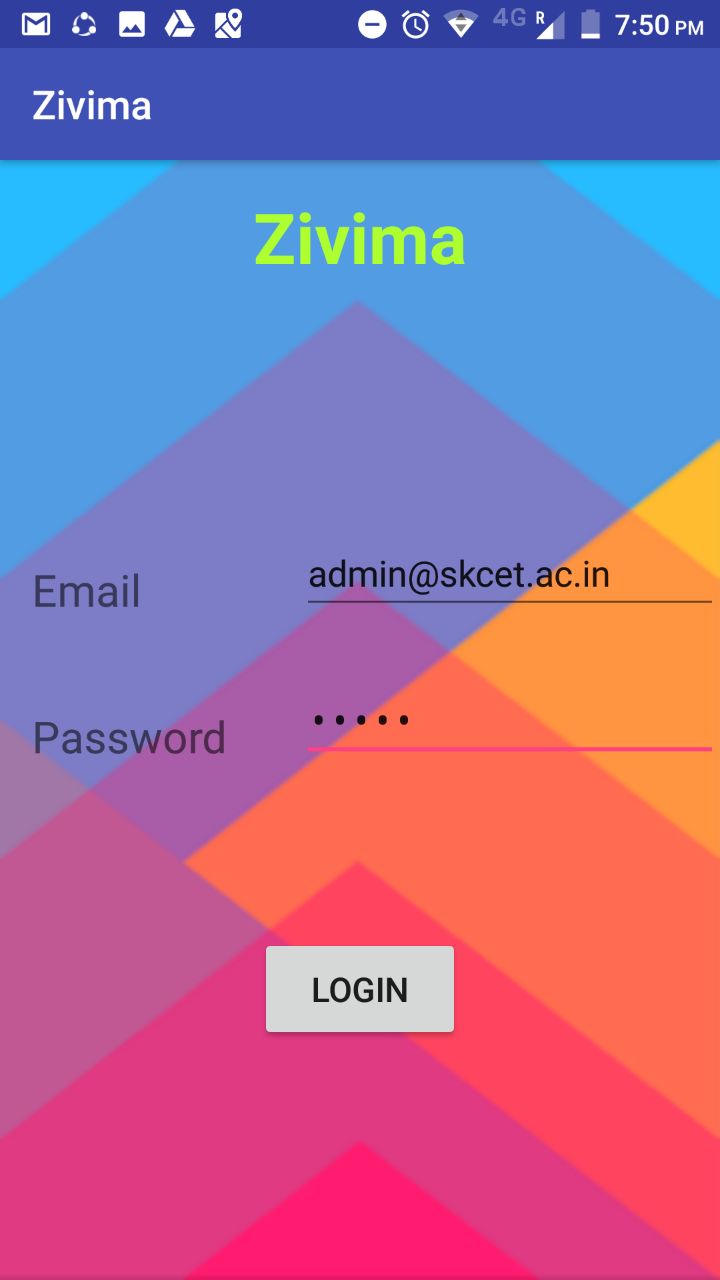
**Fig 7.6**

**7.7 UPLOADING PDF PAGE**

****

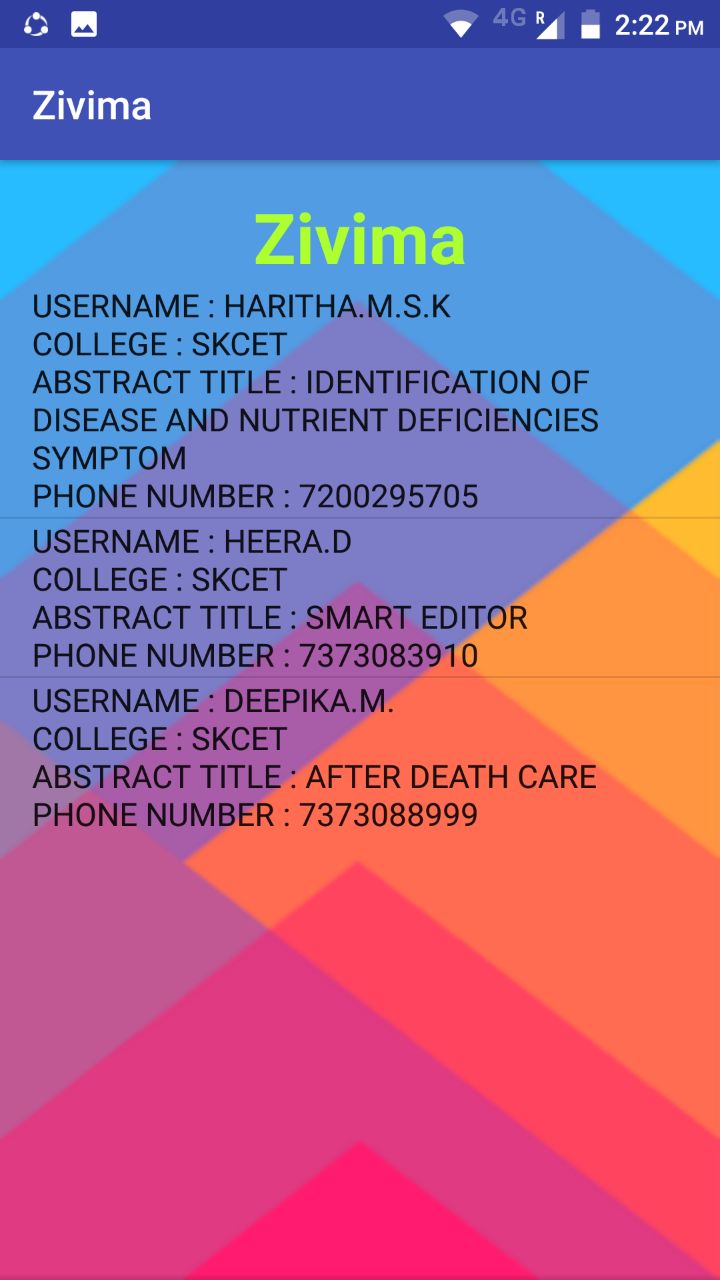
**Fig 7.7**

**7.8 ADMIN LOGIN PAGE**

****

**Fig 7.8**

**7.9 SUBMISSIONS PAGE**

****

**Fig 7.9**

**CHAPTER 8**

**CONCLUSION**

**8.1 FUTURE ENHANCEMENT :**

* Initially it is implemented in District wise.
* Further based on Success will expand throughout INDIA by connecting higher authorities of INDIA.
* Will send notifications to the student for the authentication of their project.

**8.2 CONCLUSION**

Thus the proposed system **(ZIVIMA)** provides a platform to the students from our district with ideas and prototypes, which can be transformed into working products. It is an independent, non-profit entity. It is a facilitator for people with out-of-box ideas, or innovative working prototypes. The purpose is to manifest latent ideas into commercial products that can benefit humanity in general, and help solve district’s existing social perils in particular. Thus **ZIVIMA** is a very good platform to contribute your innovative ideas for our country.

**CHAPTER 9**

**REFERENCES**

**WEBSITES**

* <https://www.tutorialspoint.com/android/android_studio.htm>
* <https://developer.android.com/guide/index.html>
* <https://developer.android.com/training/basics/firstapp/index.html>
* https://www.androidauthority.com/android-studio-tutorial-beginners-637572/
* <https://www.simplifiedcoding.net/firebase-storage-example/>
* <https://stackoverflow.com/questions/14399119/attaching-a-file-of-any-type-in-android-application>
* <https://www.lynda.com/Android-Studio-training-tutorials/6597-0.html>
* <https://in.udacity.com/course/new-android-fundamentals--ud851>