MINI PROJECT

(2020-2021)

Secure E-Learning Chat Application

REPORT

Institute Of Engineering and Technology

Team Members

Shalini Kumari
(181500649)
Ritik Srivastava
(181500578)
Adarsh Kumar
(181500037)
Shashank Agrawal
(181500652)
Vanshika Agrawal
(181500780)

-Supervised By
Name- Mr. Akash Chaudhary
Training and Placement Dept.
Department of Computer Engineering & Applications

ACKNOWLEDGEMENT

If words are considered as a symbol of approval and token of appreciation then let the words play the heralding role expressing my gratitude. The satisfaction that accompanies that the successful completion of any task would be incomplete without the mention of people whose ceaseless cooperation made it possible, whose constant guidance and encouragement crown all efforts with success. We are grateful to teacher **Mr. Akash Chaudhary** for the guidance, inspiration and constructive suggestions that helpful us in the preparation of this project.

LETTER OF TRANSMITTAL

November 25 2020,

Professor. Akash Choudhry

G.L.A University

Mathura, Utter Pradesh

Dear Sir,

We would like to take the pleasure of presenting the final documentation of our project Chat Application. The purpose of the Document is to inform you about the use, operation, maintenance, or design of software or hardware through the use of manuals, listings, diagrams, and other hard- or soft-copy written and graphic materials. We have learnt a lot of new and unique things in the path of completing this project. We tried our best to use the guidelines and counselling provided by you to make this documentation as informative as possible. We express our heartiest gratitude, for providing us with the opportunity to complete this documentation on our Documentation of Software Engineering course. We would be greatly obliged if you contact us for any explanation or any query about the report as and when deemed necessary.

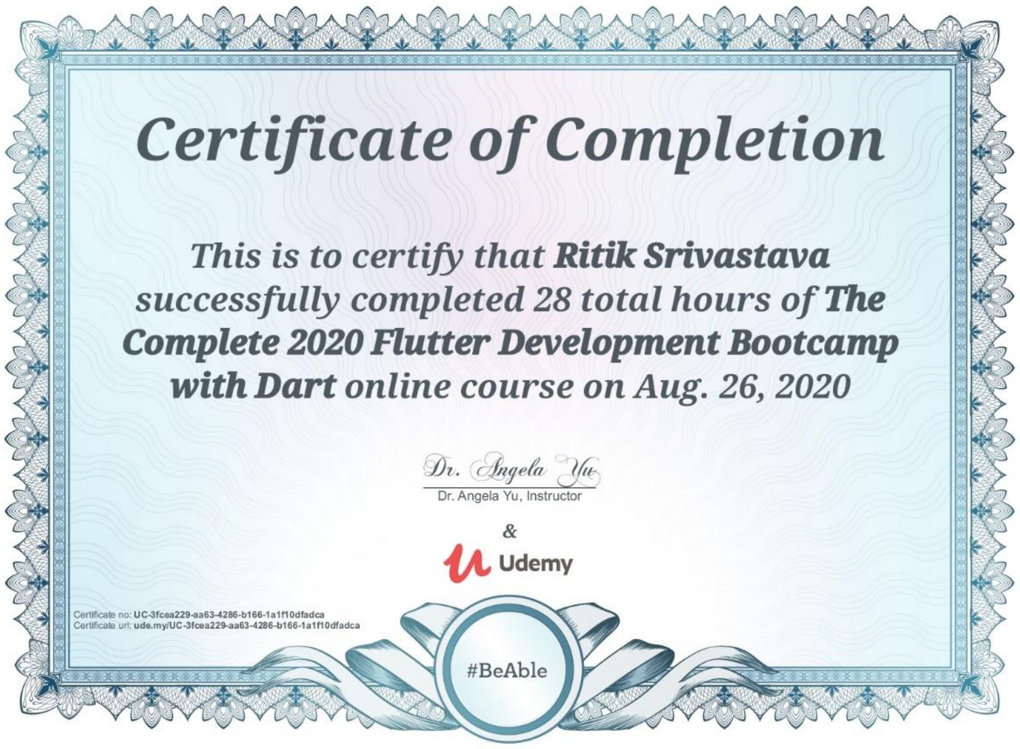
Sincerely Yours,

Team Comperio,

Comperio181500@gmail.com

Teacher's Signature & Date

CERTIFICATES



Certificate of Completion

This is to certify that **Ritik Srivastava** successfully completed 28 total hours of **The Complete 2020 Flutter Development Bootcamp with Dart** online course on Aug. 26, 2020

Dr. Angela Yu

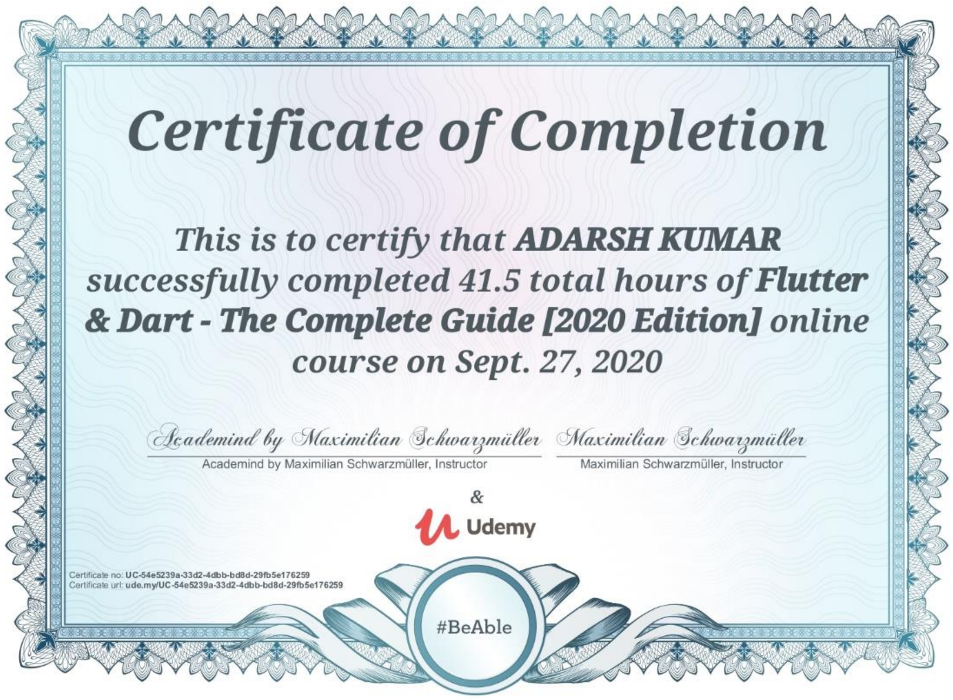
Dr. Angela Yu, Instructor

&

Udemy

Certificate no: **UC-3fcea229-aa63-4286-b166-1a1f10dfadca**
Certificate url: **ude.my/UC-3fcea229-aa63-4286-b166-1a1f10dfadca**

#BeAble



Certificate of Completion

This is to certify that ***ADARSH KUMAR*** *successfully completed 41.5 total hours of* ***Flutter & Dart - The Complete Guide [2020 Edition]*** *online course on Sept. 27, 2020*

Academind by Maximilian Schwarzmüller — Academind by Maximilian Schwarzmüller, Instructor

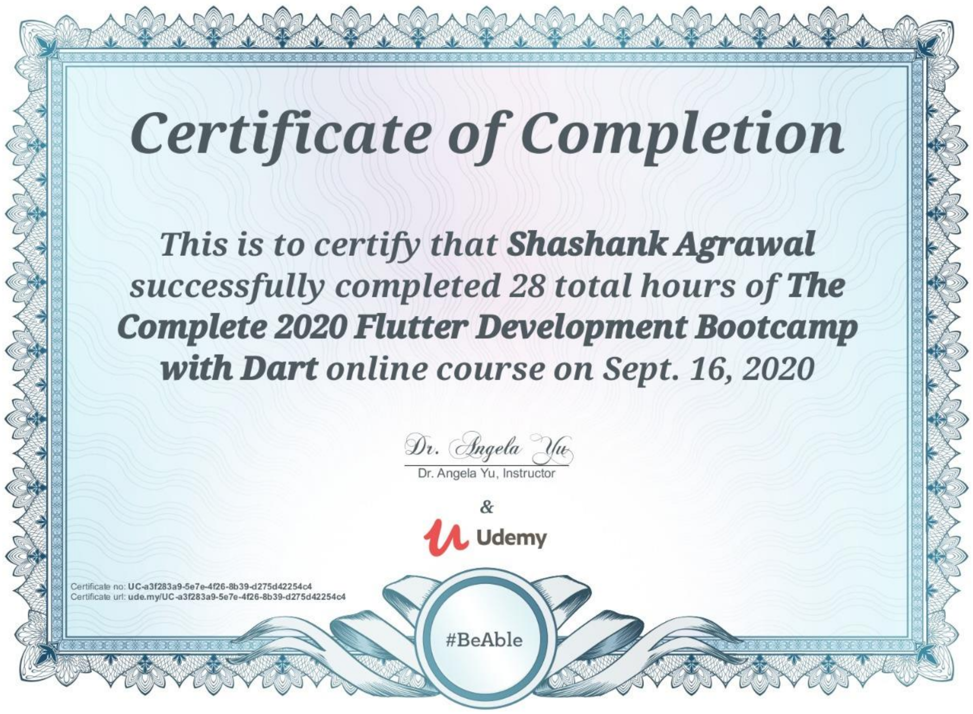
Maximilian Schwarzmüller — Maximilian Schwarzmüller, Instructor

&

Udemy

Certificate no: **UC-54e5239a-33d2-4dbb-bd8d-29fb5e176259**
Certificate url: **ude.my/UC-54e5239a-33d2-4dbb-bd8d-29fb5e176259**

#BeAble



Certificate of Completion

This is to certify that ***Shashank Agrawal*** *successfully completed 28 total hours of* ***The Complete 2020 Flutter Development Bootcamp with Dart*** *online course on Sept. 16, 2020*

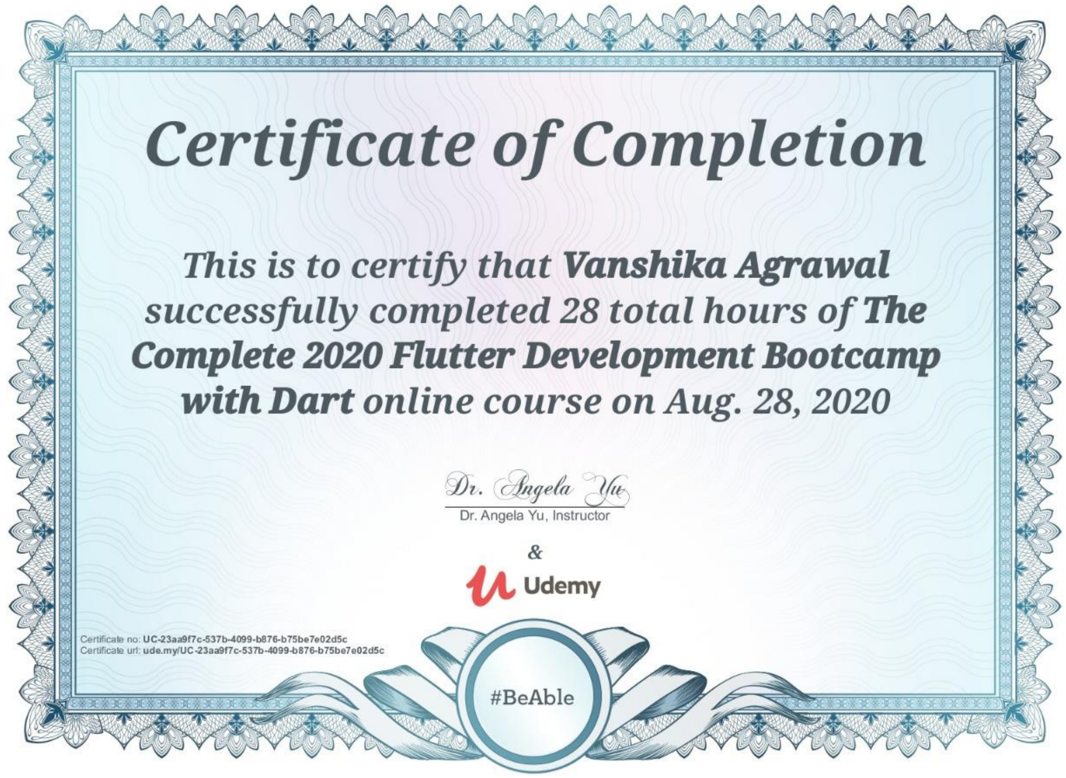
Dr. Angela Yu — Dr. Angela Yu, Instructor

&

Udemy

Certificate no: **UC-a3f283a9-5e7e-4f26-8b39-d275d42254c4**
Certificate url: **ude.my/UC-a3f283a9-5e7e-4f26-8b39-d275d42254c4**

#BeAble



Certificate of Completion

*This is to certify that **Vanshika Agrawal** successfully completed 28 total hours of **The Complete 2020 Flutter Development Bootcamp with Dart** online course on Aug. 28, 2020*

Dr. Angela Yu

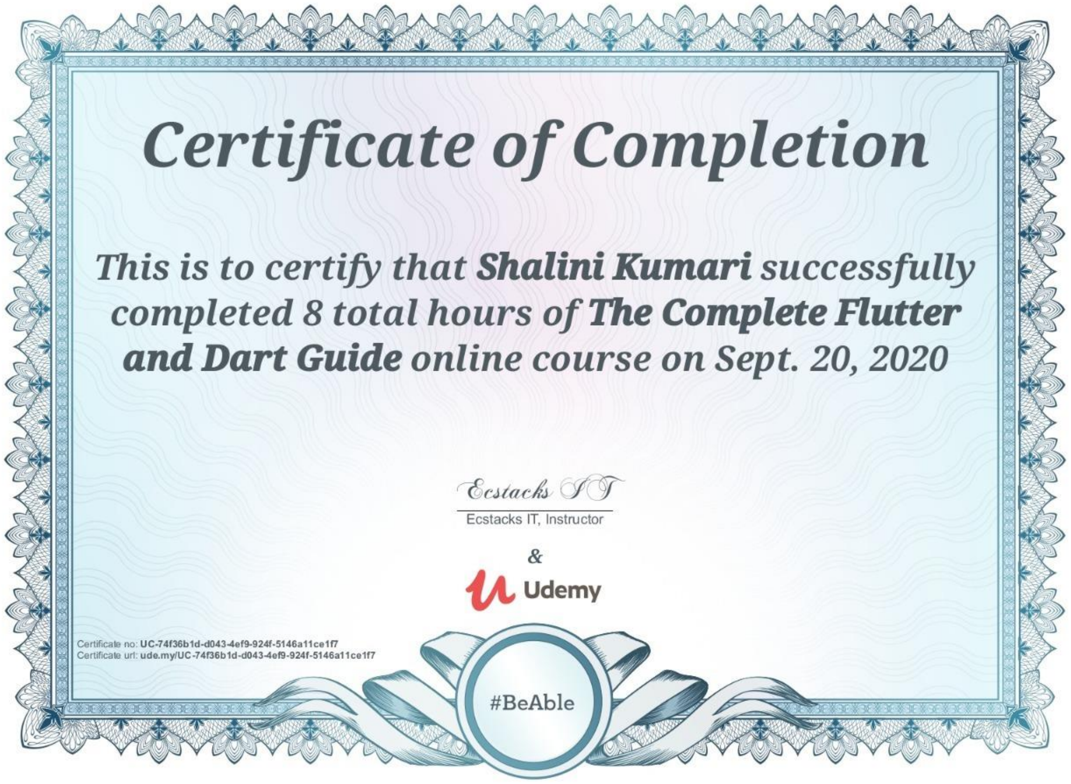
Dr. Angela Yu, Instructor

&

Udemy

Certificate no: **UC-23aa9f7c-537b-4099-b876-b75be7e02d5c**
Certificate url: **ude.my/UC-23aa9f7c-537b-4099-b876-b75be7e02d5c**

#BeAble



Certificate of Completion

*This is to certify that **Shalini Kumari** successfully completed 8 total hours of **The Complete Flutter and Dart Guide** online course on Sept. 20, 2020*

Ecstacks IT

Ecstacks IT, Instructor

&

Udemy

Certificate no: **UC-74f36b1d-d043-4ef9-924f-5146a11ce1f7**
Certificate url: **ude.my/UC-74f36b1d-d043-4ef9-924f-5146a11ce1f7**

#BeAble

CONTENT

Abstract of the project e-chat Application:

The purpose of e-chat Application is to automate the existing manual system by the help of computerized equipment's and full-fledged computer software, fulfilling their requirements, so that their valuable data/information can be stored for a longer period with easy accessing and manipulation of the same. The software and hardware are easily available and easy to work with.

E-chat Application, can lead to error free, secure, reliable and fast management system. It can assist the user to concentrate on their other activities rather to concentrate on the recording keeping. Thus it will help organization can maintain computerized records without redundant entries. That means that one need not be distracted by information that is not relevant, while being able to reach the information.

The aim is to automate its existing manual system by the help of computerized equipment and full-fledged computer software, fulfilling their requirement, so that their valuable data/information can be stored for a longer period with easy accessing and manipulation of the same. Basically the project describe how to manage for good performance and better services for clients.

Introduction to e-chat Application:

Online e-chat application is a form of communication that utilizes computer programs that allow for two-way conversations between users in real time (events that occur in cyberspace at the same speed that they would occur in real life).

Typically the users will connect to a chat server using a chat client and meet in a chat room. Once the users are in the same chat room, they can converse with one another by typing messages into a window where all of the other users in the chat room can see the message. The user can also see all of the messages entered by the other users. Conversations are then carried on by reading the messages entered by the other users in the chat room and responding to them.

Purpose:

However, the purpose of this project is to develop an e-chat application. The objective of this process is as follows:

1. To develop an instant messaging solution to enable users to seamlessly communicate with each other And solve their doubts.
2. The project should be very easy to use enabling even a novice person to use it.

How does online chat work?

The technologies that allows users to converse in real time via typed messages are the chat servers and the chat clients. These special servers allow the messages to be quickly received and posted so that the chat clients can receive the messages and display them in the window on the users' computers. Fortunately, most chat clients are relatively inexpensive shareware programs, and there are a number of chat servers that let the user connect to them for free. Online chat could be an alternative to meeting in a traditional classroom. While all of the participants would still be able to freely converse with one another, they would carrying on the conversation from the comfort of their own home, thus avoiding the expense and inconvenience of traveling to another place.

Problem Statement:

As we all know about the current pandemic situation due to COVID-19, because of which we all are locked at our homes for so many months and don't know what future situation will be. Especially for students, this situation seems very difficult to handle.

So many students are slowly developing big confusions and distractions in their brains. But many are doing e-learning courses and online classes to develop their skills, but then also there are many questions that are always inside them and they don’t have anyone to guide them. So, to clear all these questions and put their cars back on track our app can really help. This app will connect students directly with the teachers of their fields and will help them to solve their queries.

Reason:

In the past few years, chat rooms have become a popular way to support a forum for n-way conversation or discussion among a set of people with interest in a common topic. Chat applications range from simple, text-based ones to entire virtual worlds with exotic graphics. But also, many of us still have many questions in our brains.

The reasons behind this maybe are:

1. There is no authority to check the mentors' guidance.
2. They have to rely only on their faculty and their guidance.
3. Delay in the response due to a bunch of emails to be filtered by them to give any reply to anyone.
4. The questions can be missed out by the faculty due to a lot of messages.
5. Fail to recall the consultation after a respective time.

So by taking all these points under consideration we proposed to build a project which can solve these issues, like:

1. If a reply is not given within an allotted time the ranking of the teacher will decrease.
2. Huge and the wide network is created by this app having all university teachers on a single platform.

Objective:

The main objective of the project is to make an E-learning chat mobile application at the university level where queries of students could be solved by teachers which can be achieved by the following:

- Login by University E-mail Id.
- Verifying teachers.
- Creating a chat-box.

PROJECT SUMMARY

Project Background:

The previous work of this already exists. The similar application can be found on the project either Android market. This project will focus on providing high quality usability experiences to users mainly following Googles user interface guideline. Experiments the application will be tested on a test group to improve the usability quality based on the user’s feedback.

Functional and Non Functional Requirements:

Functional Requirements:

1. User Registration

   User must be able to register for the application through a valid email address. On installing the application, user must be prompted to register their email address. If user skips this step, application should close. The users email address will be the unique identifier of his/her account on Chat Application.
2. Adding New Professors

   The application should detect all the users on the app. If any of the professors have user accounts with Chat Application, those professors must be searched and a message them to add to your list.
3. Send Message

   User should be able to send instant message to any contact on his/her Chat Application contact list. User should be notified when message is successfully delivered to the recipient by displaying a tick sign next to the message sent.

Non Functional Requirements:

1. Privacy

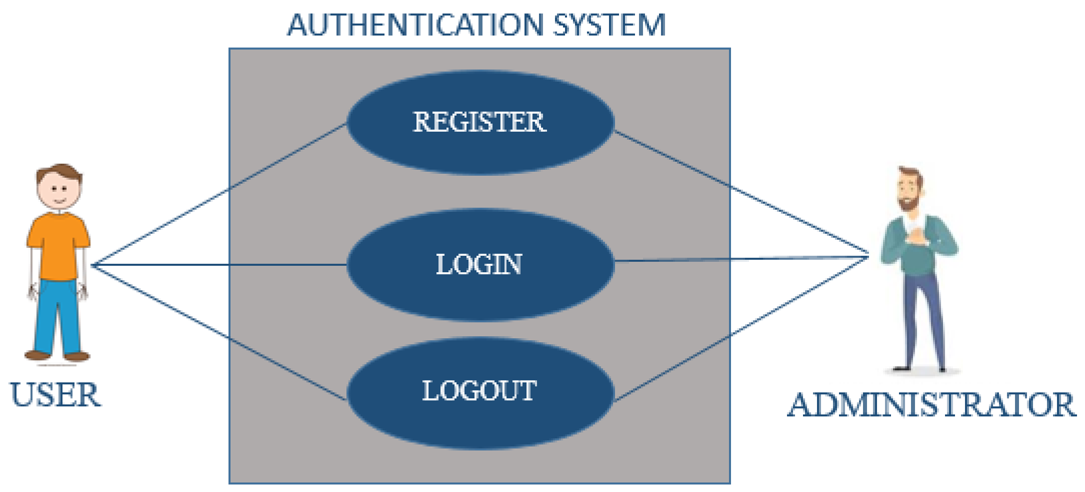
   Messages shared between users should be encrypted to maintain privacy.
2. Robustness

   In case user's device crashes, a backup of their chat history must be stored on remote database servers to enable recoverability.
3. Performance

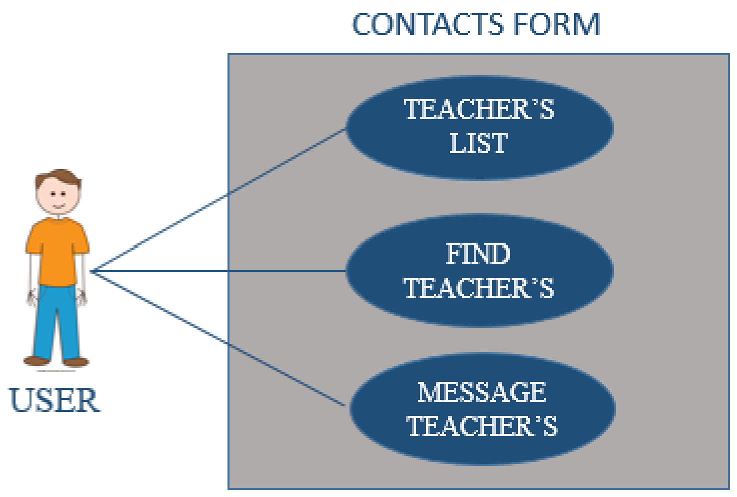
   Application must be lightweight and must send messages instantly.

Use case Diagram:

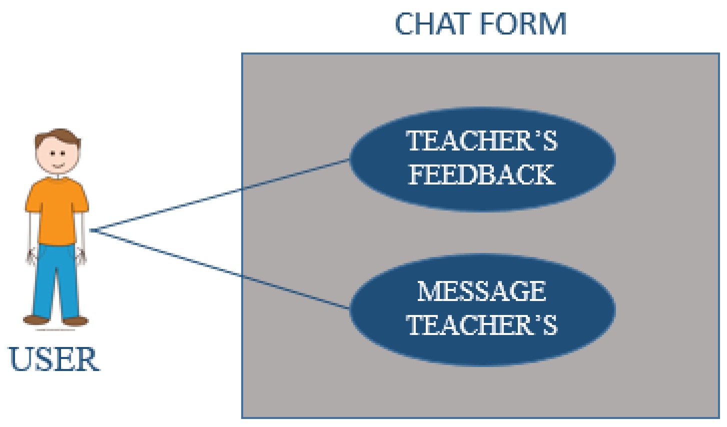
Authentication System-



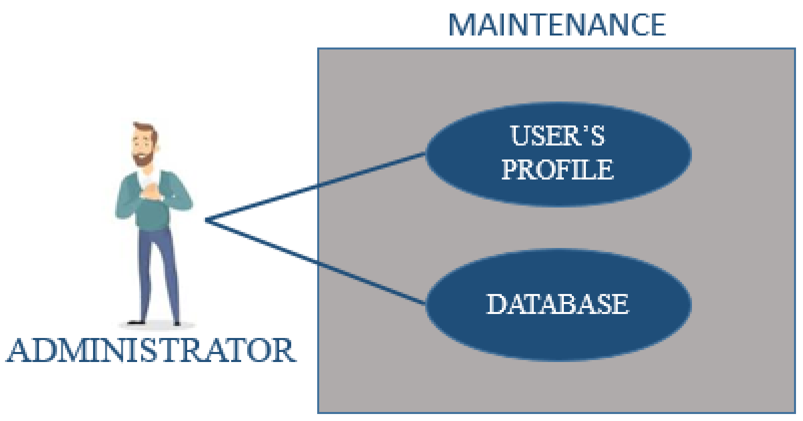
Contact Form-



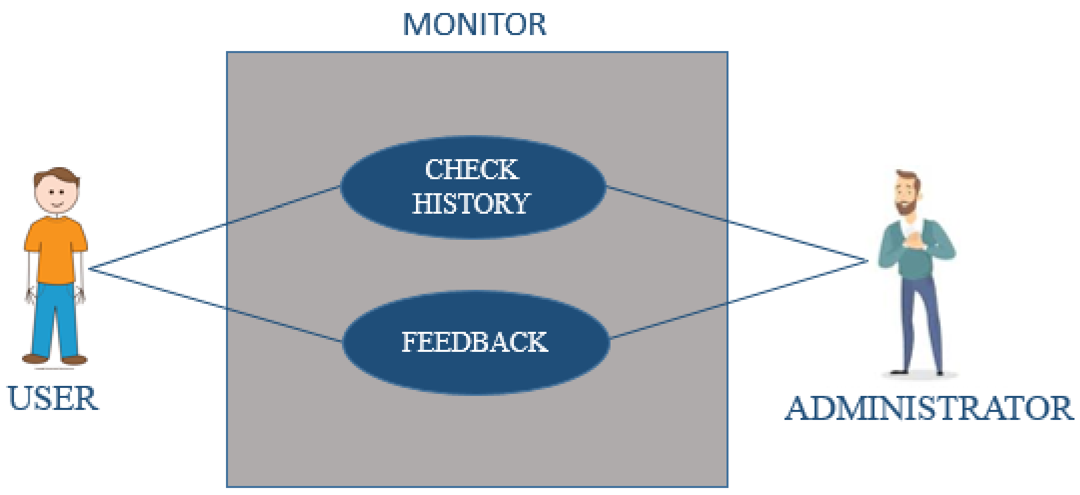
Chat Form-



Maintenance-

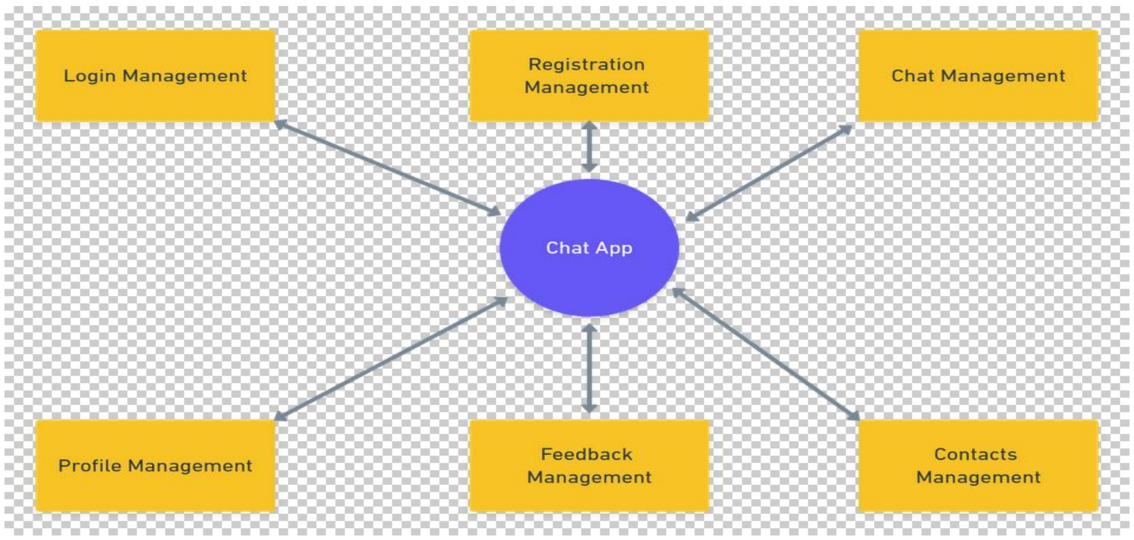


Monitor-

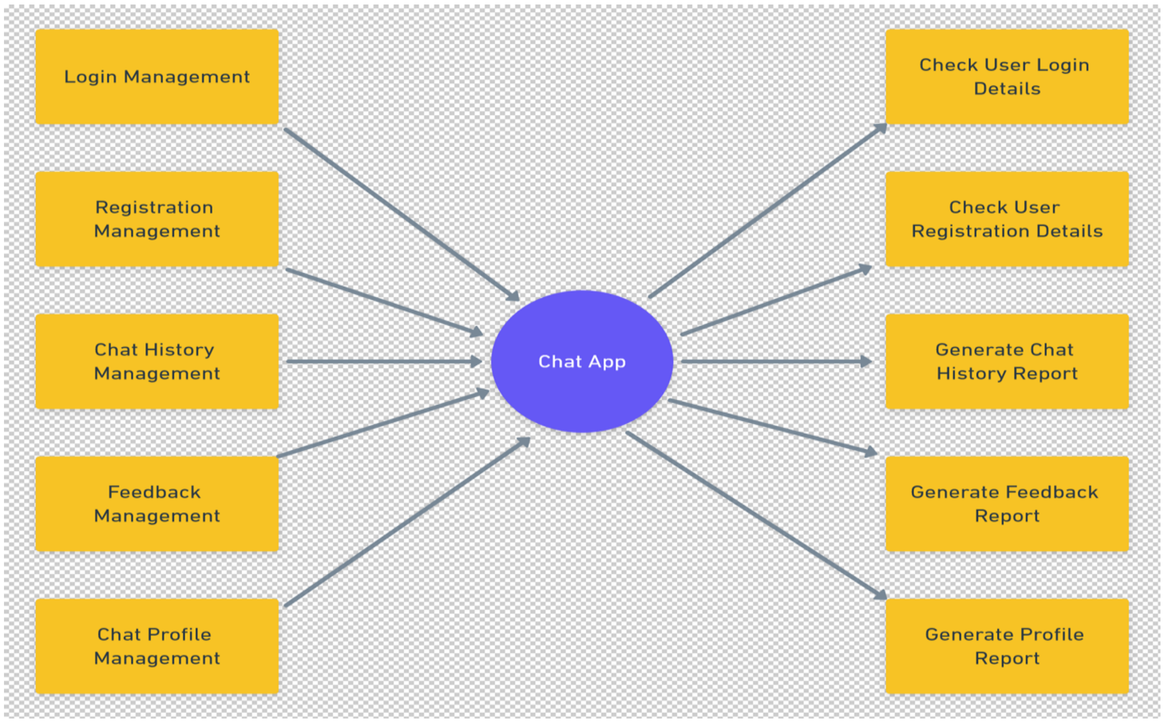


Data Flow Diagram:

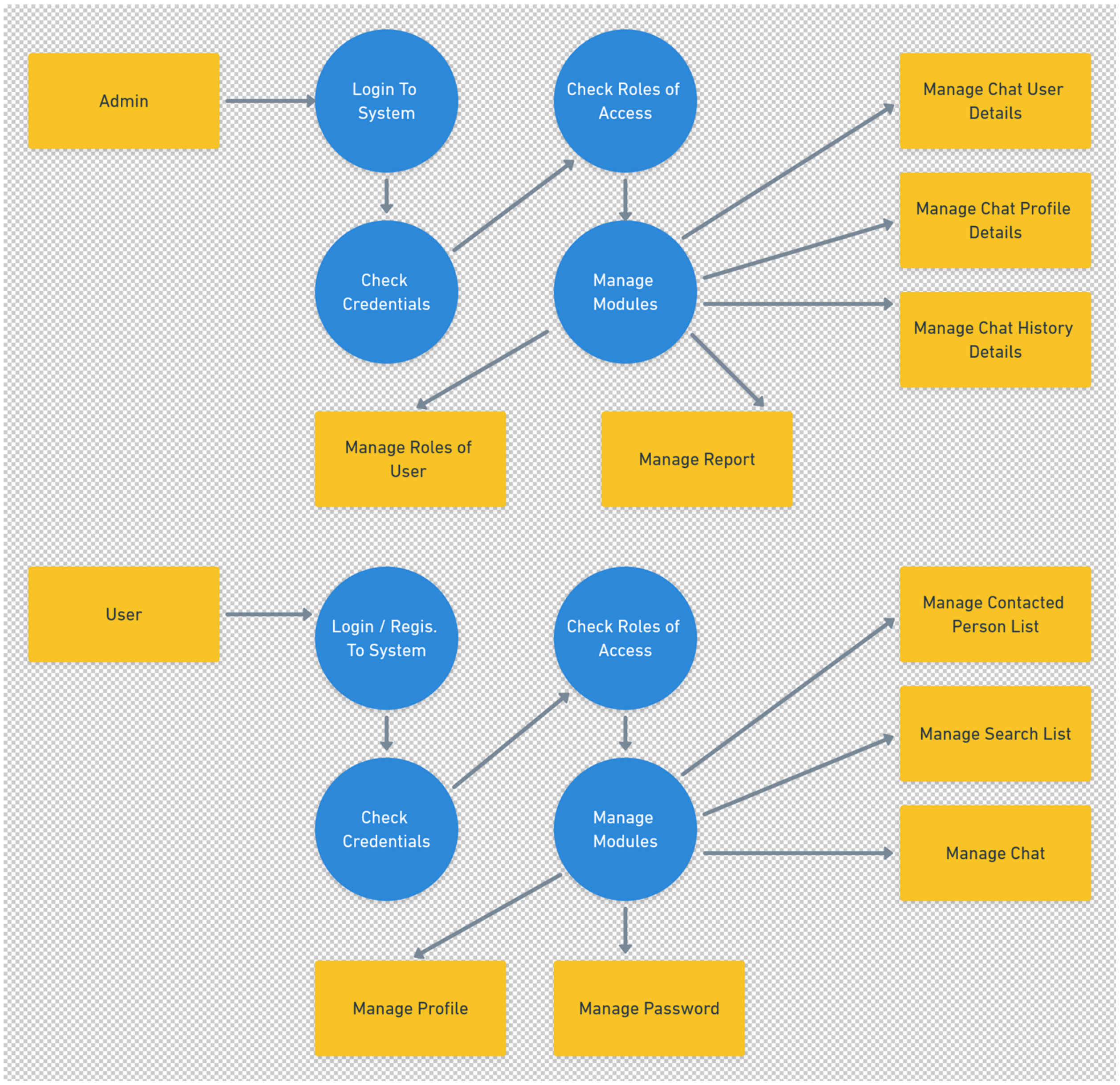
Level 0 DFD



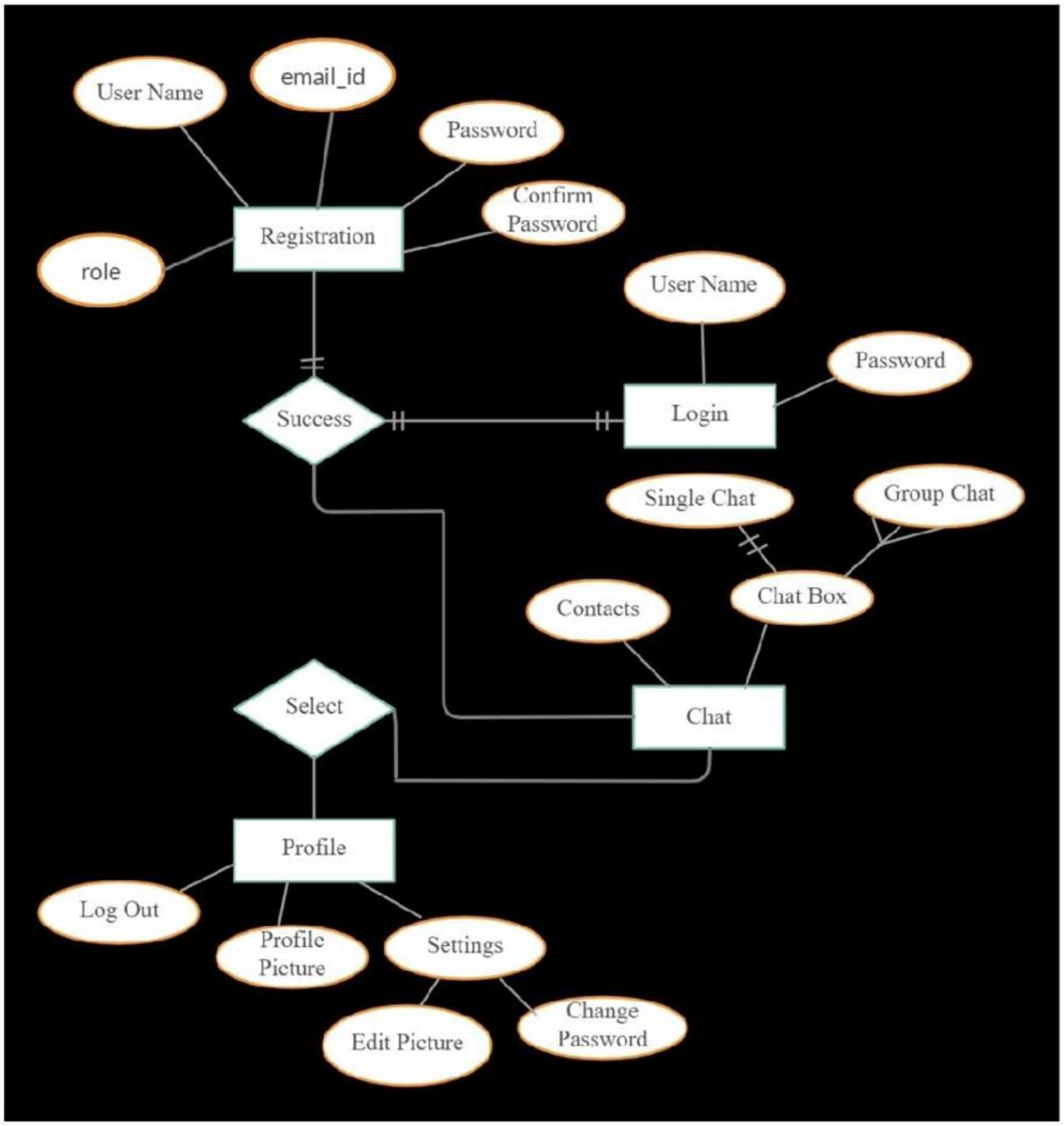
Level 1 DFD



Level 2 DFD


Admin
Login To System
Check Roles of Access
Manage Chat User Details
Check Credentials
Manage Modules
Manage Chat Profile Details
Manage Chat History Details
Manage Roles of User
Manage Report
User
Login / Regis. To System
Check Roles of Access
Manage Contacted Person List
Check Credentials
Manage Modules
Manage Search List
Manage Chat
Manage Profile
Manage Password

ER Diagram:


User Name
email_id
Password
Confirm Password
role
Registration
User Name
Password
Success
Login
Single Chat
Group Chat
Contacts
Chat Box
Select
Chat
Profile
Log Out
Profile Picture
Settings
Edit Picture
Change Password

Future Scope:

The application can be extended to cover a wide range of intra-organizational as well as extra-organizational issues. Apart from doubt sessions related to studies, other areas like management and mental wellness could also be touched upon.

It could evolve as a single platform for knowledge sharing within an organization. Content moderation can be done using the sentimental analysis to ensure the code of conduct. Extending it to Web Support.

Requirements:

1. Hardware:
   Ram required- 8GB or above
   Processor- core i5 or above

2. Software:
   Linux or Windows Operating System
   Android Studio or Vs-code
   Android or IOS Emulator or Physical Smartphone.

3. Technology Used:
   Flutter 1.20
   Dart Programming Language
   Firebase

Methodology:

The idea of this methodology is to develop an e-Learning system that favours the competitive position of the organization that uses it and is adaptable to suit the student's needs. It refers to the mode and manner in which they produce the content that will be used in training.

In this application students and the faculty can sign up using their university email id and help each other to solve the queries they have. Students can broadcast or can directly message to a particular faculty for their queries. Students can also tag their messages to a particular faculty or experts in that field. To differentiate between students and faculty we will provide verified tags to the faculty using this application and for that, the faculty just have to make a request on the verification request tab.

Project Summary:

The user interacts with the tool using GUI. The GUI Operates in two forms, Contacts Forms and Chat forms. The contacts forms contains list of all friend and the Chat form will be used to chat with friends.

Project Deliverable's-

1. An android app
2. Documentation
3. Readme File

Hardware Interface-

• Android Phone
• 128 MB minimum RAM Required
• Internet or LAN Connections
• Processor with Speed of 500 MHz

Constraints and Limitations-

The system must be connected with internet. User only can use or install this app on android devices. This app doesn't have video and audio calling system. We may add it in later version.

Modules:	**Description:**
Login/Sign Up Screen	Users can sign up with their university email id and specify their role as student or faculty.
Contact Screen	Users can see the list of contacts they messaged previously or got a message from.
Chat Screen	Users can see the chats/messages they send or receive.
User Profile Screen	Users can see their information here.
Verification Screen	Faculty can send a verification request from this page.
Feedback Screen	The students can give their feedback or any suggestion regarding any topic.

Testing:

The Flutter framework provides comprehensive support for Flutter automated testing of mobile apps. Manually testing mobile apps can be hard, especially if your app has many features. Automated tests help to ensure that your app performs correctly before you publish it while maintaining your feature and bug-fix velocity.

Automated testing falls into a few categories:

- A unit testing tests a single function, method, or class.
- A widget test (in other UI frameworks referred to as component test) tests a single widget.
- An Integration Test tests a complete app or a large part of an app.

Contributions:

Today people know how to use smart devices and these devices are a huge part of their life. Hence, no time is better than now to introduce this technology (educational mobile app) via their mobile devices. These mobile learning platforms are very appealing and will have the total attention of the learners.

E-learning is a new education concept by using internet technology, it delivers the digital content, provides a learner-orient environment for the teachers and students. E-learning promotes the construction of life-long learning opinions and learning society

.

E-Learning has completely transformed the way in which learning is imparted to students. Unlike traditional chalk and board methods of teaching, e-learning makes learning simpler, easier, and more effective.

E-Learning mobile app allows users to learn by using podcasts, videos and other mechanisms on their smartphone. Hence, they make the process of learning quite flexible for learners.

E-Learning mobile app allows the student to directly be in touch with the subject professors. Hence they can clear their doubts quickly. Often students suffer from lack of collaboration, however, these mobile learning apps offer them collaborative learning via online learning community app.

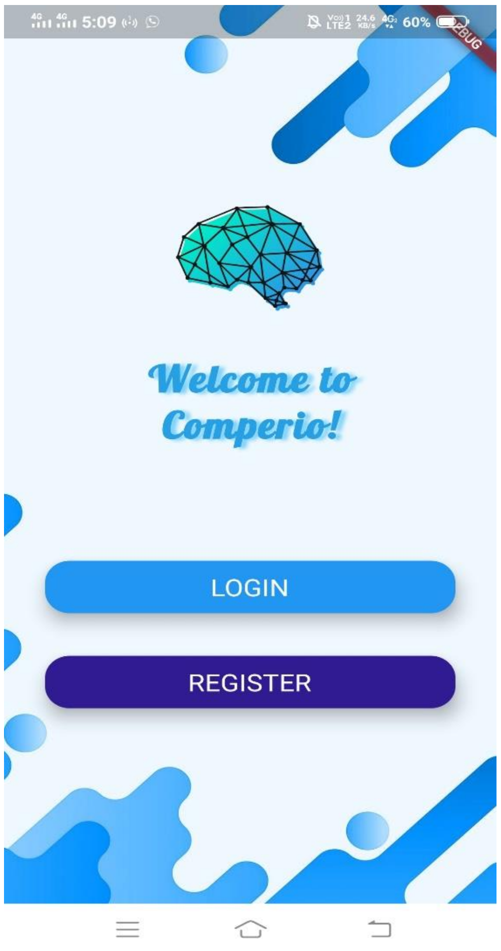
E-learning app ensure higher engagement, as learning is easily accessible by the students at any time and any place. One of the best benefits of e-learning is multi-device support.

Project description and code:

Welcome Screen:

This Screen has two buttons. For a new user, there is an option to register and create a new account and login option for an already registered user.

```dart
import 'package:comperio/constants.dart';
import 'package:comperio/screen/login_screen.dart';
import 'package:comperio/screen/registration_screen.dart';
import 'package:flutter/material.dart';
import 'package:flutter/services.dart';

class WelcomeScreen extends StatefulWidget {
  final String id = 'WelcomeScreen';

  @override
  _WelcomeScreenState createState() => _WelcomeScreenState();
}

class _WelcomeScreenState extends State<WelcomeScreen>
    with TickerProviderStateMixin {
  AnimationController controller;
  AnimationController controller1;
  AnimationController controller2;

  @override
  void initState() {
    super.initState();
    controller = AnimationController(
      duration: Duration(seconds: 3),
      vsync: this,
    );
    controller.forward();
    controller.addListener(() {
      setState(() {});
    });
    controller1 = AnimationController(
      duration: Duration(milliseconds: 50),
      vsync: this,
      lowerBound: 0.0,
      upperBound: 0.1,
    );

    controller1.addListener(() {
      setState(() {});
    });

    controller2 = AnimationController(
      duration: Duration(milliseconds: 50),
      vsync: this,
      lowerBound: 0.0,
      upperBound: 0.1,
    );

    controller2.addListener(() {
      setState(() {});
    });
  }

  @override
  void dispose() {
    controller.dispose();
    controller1.dispose();
    controller2.dispose();
    super.dispose();
  }

  Future<bool> _onWillPop() {
    return showDialog(
          context: context,
          builder: (context) => AlertDialog(
            title: Text('Are you sure?'),
            content: Text('Do you want to exit App'),
            actions: <Widget>[
              FlatButton(
                onPressed: () => Navigator.of(context).pop(false),
                child: Text('No'),
              ),
              FlatButton(
                onPressed: () => SystemNavigator.pop(),
                /*Navigator.of(context).pop(true)*/
                child: Text('Yes'),
              ),
            ],
          ),
        ) ??
        false;
  }

  @override
  Widget build(BuildContext context) {
    double scale1 = 1 + controller1.value;
    double scale2 = 1 + controller2.value;

    return WillPopScope(
      onWillPop: _onWillPop,
      child: Container(
        decoration: BoxDecoration(
          color: Colors.white,
          image: DecorationImage(
            image: AssetImage('images/WelcomeScreen3.jpg'),
            fit: BoxFit.cover,
          ),
        ),
        constraints: BoxConstraints.expand(),
        child: Scaffold(
          backgroundColor: Colors.transparent,
          body: Container(
            child: Column(
              mainAxisAlignment: MainAxisAlignment.spaceEvenly,
              children: <Widget>[
                Container(
                  padding: EdgeInsets.only(left: 30.0, right:
30.0),
                  child: Column(
                    children: <Widget>[
                      Hero(
                        tag: 'logo',
                        child: Container(
                          height: controller.value * 100,
                          child: Image(
                            image: AssetImage(
                              'images/comperio-logo.png',
                            ),
                            height: 110.0,
                            width: 110.0,
                          ),
                        ),
                      ),
                      SizedBox(
                        height: 30.0,
                      ),
                      Center(
                        child: Text(
                          'Welcome to',
                          style: KWelcomeTextStyle,
                        ),
                      ),
                      Center(
                        child: Text(
                          'Comperio!',
                          style: KWelcomeTextStyle,
                        ),
                      ),
                      SizedBox(
                        height: 90.0,
                      ),
                      ButtonTheme(
                        minWidth: 300.0,
                        child: GestureDetector(
                          onTapDown: onTapDown1,
                          onTapUp: onTapUp1,
                          onTapCancel: onTapCancel1,
                          child: Transform.scale(
                            scale: scale1,
                            child: RaisedButton(
                                                    elevation: 10.0,
                                                    // splashColor: Colors.green,
                                                    shape: RoundedRectangleBorder(
                                                      borderRadius:
                      BorderRadius.circular(18.0),
                                                    ),
                                                    onPressed: () {
                                                      Navigator.pushNamed(context,
                      LoginScreen().id);
                                                    },
                                                    padding: EdgeInsets.only(top: 10.0,
                      bottom: 10.0),
                                                    color: Colors.blue,
                                                    textColor: Colors.white,
                                                    child: Text(
                                                      'Login'.toUpperCase(),
                                                      style: TextStyle(fontSize: 20),
                                                    ),
                                                  ),
                                                ),
                                              ),
                                            ),
                                            SizedBox(
                                              height: 30.0,
                                            ),
                                            ButtonTheme(
                                              minWidth: 300.0,
                                              child: GestureDetector(
                                                onTapDown: onTapDown2,
                                                onTapUp: onTapUp2,
                                                onTapCancel: onTapCancel2,
                                                child: Transform.scale(
                                                  scale: scale2,
                                                  child: RaisedButton(
                                                    elevation: 10.0,
                                                    // splashColor: Colors.green,
                                                    shape: RoundedRectangleBorder(
                                                      borderRadius:
                      BorderRadius.circular(18.0),
                                                    ),
                                                    onPressed: () {
                                                      Navigator.pushNamed(
                                                          context,
                      RegistrationScreen().id);
                                                    },
                                                    padding: EdgeInsets.only(top: 10.0,
                      bottom: 10.0),

                                                    color: Color(0xff311b92),
                                                    textColor: Colors.white,
                                                    child: Text(
                                                      'Register'.toUpperCase(),
                                                      style: TextStyle(fontSize: 20),
                                                    ),
                                                  ),
                                                ),
                                              ),
                                            ),
                                          ],
                                        ),
                                      ),
                                    ],
                                  ),
                                ),
                              ),
                            ),
                          );
                        }

                        onTapUp1(TapUpDetails details) {
                          controller1.reverse();
                        }

                        onTapDown1(TapDownDetails details) {
                          controller1.forward();
                        }

                        onTapCancel1() {
                          controller1.reverse();
                        }
                        onTapUp2(TapUpDetails details) {
                          controller2.reverse();
                        }

                        onTapDown2(TapDownDetails details) {
                          controller2.forward();
                        }

                        onTapCancel2() {
                          controller2.reverse();
                        }
                      }
```



Registration Screen:

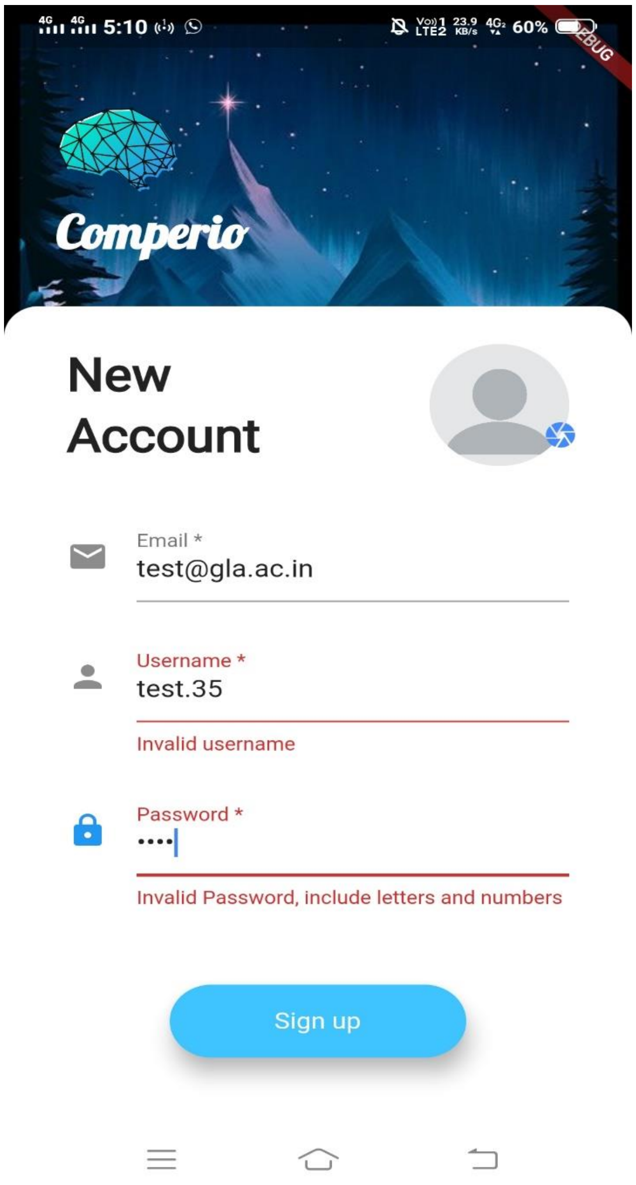
On this screen, the user will input the details such as username, email address, and password. There is an option to upload a display picture too. For all the fields' validation will be performed.

```
import 'dart:io';
import 'package:cloud_firestore/cloud_firestore.dart' show
FirebaseFirestore;
import 'package:comperio/constants.dart';
import 'package:comperio/regexValidator.dart';
import 'package:comperio/screen/contacted_person_screen.dart';
import 'package:comperio/screen_app_logo.dart';
import 'package:firebase_auth/firebase_auth.dart';
import 'package:firebase_storage/firebase_storage.dart';
import 'package:flutter/cupertino.dart';
import 'package:flutter/material.dart';
import 'package:flutter/widgets.dart';
import 'package:image_picker/image_picker.dart';
import 'package:modal_progress_hud/modal_progress_hud.dart';
import 'package:path/path.dart';
import 'package:rich_alert/rich_alert.dart';

import '../helper_functions.dart';

class RegistrationScreen extends StatefulWidget {
  final String id = 'RegistrationScreen';

  @override
  _RegistrationScreenState createState() =>
_RegistrationScreenState();
}

class _RegistrationScreenState extends State<RegistrationScreen> {
  final GlobalKey<FormState> _formKey = GlobalKey<FormState>();
  bool _autoValidate = false;
  bool showSpinner = false;
  final _auth = FirebaseAuth.instance;
  String email;
  String username;
  String password;
  String role;
  File _image;
  double feedbackValue;
  double feedbackRate;

  // String feedback = "0";
  String url =
      'https://firebasestorage.googleapis.com/v0/b/comperio-
1071d.appspot.com/o/default-profile.webp?alt=media&token=c737b18e-
9625-4b0d-8d7c-8ef5794486f3';

  Future getImages() async {
    PickedFile pickedFile =
        await ImagePicker().getImage(source: ImageSource.gallery);
    var image = File(pickedFile.path);

    setState(() {
      _image = image;
      // print('Image path $_image');
    });
  }

  void _addToDatabase(String userName, String dpUrl) {
    print('dpurl is: $dpUrl');
    List<String> splitList = username.split(" ");
    List<String> indexList = [];

    for (int i = 0; i < splitList.length; i++) {
      for (int y = 1; y < splitList[i].length + 1; y++) {
        indexList.add(splitList[i].substring(0, y).toLowerCase());
      }
    }
    // print(indexList);

FirebaseFirestore.instance.collection('users').doc(username).set({
      'username': userName,
      'searchKeywords': indexList,
      'profileURL': dpUrl,
      'email': email,
      'role': role,
      'feedbackValue': 0.0,
      'feedbackRate': 0.0,
    });
  }

  Future uploadPic(BuildContext context) async {
    String fileName = basename(_image.path);
    StorageReference firebaseStorageRef =
        FirebaseStorage.instance.ref().child(fileName);
    StorageUploadTask uploadTask =
firebaseStorageRef.putFile(_image);
    var dowUrl = await (await
uploadTask.onComplete).ref.getDownloadURL();
    url = dowUrl.toString();
    print(url);
    StorageTaskSnapshot taskSnapshot = await uploadTask.onComplete;
    setState(() {
      print("Profile Picture uploaded");
      // Scaffold.of(context)
      //     .showSnackBar(SnackBar(content: Text('Profile Picture
Uploaded')));
    });
  }

  assignRole() {
    if (email.contains('_')) {
      setState(() {
        role = "Student";
      });
    } else {
      setState(() {
        role = "Professor";
      });
    }
  }

  bool _validateInputs() {
    if (_formKey.currentState.validate()) {
//    If all data are correct then save data to out variables
      _formKey.currentState.save();

      return true;
    } else {
//    If all data are not valid then start auto validation.
      setState(() {
        _autoValidate = true;
      });
      return false;
    }
  }

  @override
  Widget build(BuildContext context) {
    return Container(
      decoration: BoxDecoration(
        image: DecorationImage(
          image: AssetImage('images/app-background.jpg'),
          fit: BoxFit.cover,
        ),
      ),
      constraints: BoxConstraints.expand(),
      child: Scaffold(
        resizeToAvoidBottomInset: true,
        resizeToAvoidBottomPadding: true,
        backgroundColor: Colors.transparent,
        body: ModalProgressHUD(
          inAsyncCall: showSpinner,
          child: Container(
            child: Column(
              crossAxisAlignment: CrossAxisAlignment.start,
              children: <Widget>[
                Container(
                  padding: EdgeInsets.only(
                      top: 60.0, left: 30.0, right: 30.0, bottom:
30.0),
                  child: Column(
                    crossAxisAlignment: CrossAxisAlignment.start,
                    children: <Widget>[
                      ScreenAppLogo(),
                      Text(
                        'Comperio',
                        style: KAppNameTextStyle,
                      ),
                    ],
                  ),
                ),
                Flexible(
                  fit: FlexFit.tight,
                  child: Container(
                    width: double.infinity,
                    padding: EdgeInsets.only(left: 20.0, right:
20.0),
                    decoration: BoxDecoration(
                      color: Colors.white,
                      borderRadius: BorderRadius.only(
                        topLeft: Radius.circular(20.0),
                        topRight: Radius.circular(20.0),
                      ),
                    ),
                    child: SingleChildScrollView(
                      child: Padding(
                        padding: const EdgeInsets.only(
                            left: 16.0, right: 16.0, top: 25,
bottom: 16.0),
                        child: Column(
                          mainAxisSize: MainAxisSize.min,
                          crossAxisAlignment:
CrossAxisAlignment.stretch,
                          children: <Widget>[
                            Row(
                              mainAxisAlignment:
MainAxisAlignment.spaceBetween,
                              children: [
                                Column(
                                  crossAxisAlignment:
CrossAxisAlignment.start,
                                  children: [
                                    Text(
                                      'New',
                                      style: KCardTextStyle,
                                    ),
                                    SizedBox(
                                      height: 5.0,
                                    ),
                                    Text(
                                      'Account',
                                      style: KCardTextStyle,
                                    ),
                                  ],
                                ),
                                Container(
                                  child: Stack(
                                    alignment: Alignment.center,
                                    overflow: Overflow.visible,
                                    children: [
                                      Container(
                                        width: 80.0,
                                        height: 80.0,
                                        child: CircleAvatar(
                                          radius: 40.0,
                                          backgroundColor:
Colors.black12,

                                          child: ClipOval(
                                            child: SizedBox(
                                              width: 80.0,
                                              height: 80.0,
                                              child: (_image !=
null)

                                                  ? Image.file(
                                                      _image,
                                                      fit:
BoxFit.fill,

                                                    )
                                                  : Image.asset(

'images/default-profile.jpg',

                                                      fit:
```

```
BoxFit.fill,
                                                    ),
                                                ),
                                              ),
                                            ),
                                          ),
                                          Positioned(
                                            top: -1,
                                            right: -30,
                                            bottom: -40,
                                            left: 40,
                                            child: IconButton(
                                              icon: Icon(
                                                Icons.camera,
                                                color:
Colors.blueAccent,
                                                size: 20,
                                              ),
                                              onPressed: () {
                                                getImages();
                                              },
                                            ),
                                          ),
                                        ],
                                      ),
                                    ),
                                  ],
                                ),
                                SizedBox(
                                  height: 30.0,
                                ),
                                Flexible(
                                  fit: FlexFit.loose,
                                  child: Form(
                                    key: _formKey,
                                    autovalidate: _autoValidate,
                                    child: Column(
                                      mainAxisAlignment:

MainAxisAlignment.spaceEvenly,
                                      children: <Widget>[
                                        TextFormField(
                                          keyboardType:
TextInputType.emailAddress,
                                          decoration: const
InputDecoration(
                                            icon: Icon(Icons.email),
                                            hintText: 'Enter your
Email',
                                            hintStyle: TextStyle(
                                              color: Colors.grey,
                                            ),
                                            labelText: 'Email *',
                                          ),
                                          validator:
RegexValidator.validateEmail,
                                          onSaved: (String value) {
                                            // This optional block of
code can be used to run
                                            // code when the user saves
the form.
                                            email = value;
                                          },
                                          // onChanged: (String value)
{
                                          //   // This optional block
of code can be used to run
                                          //   // code when the user
saves the form.
                                          //   email = value;
                                          // },
                                        ),
                                        SizedBox(
                                          height: 20.0,
                                        ),
                                        TextFormField(
                                          decoration: const
InputDecoration(
                                            icon: Icon(Icons.person),
                                            hintText: 'Enter Username',
                                            hintStyle: TextStyle(
                                              color: Colors.grey,
                                            ),
                                            labelText: 'Username *',
                                          ),
                                          validator:

RegexValidator.validateUserName,
                                          onSaved: (String value) {
                                            username = value;
                                          },
                                          // onChanged: (String value)
{
                                          //   // This optional block
of code can be used to run
                                          //   // code when the user
saves the form.
                                          //   username = value;
                                          // },
                                        ),
                                        SizedBox(
                                          height: 20.0,
                                        ),
                                        TextFormField(
                                          decoration: const
InputDecoration(
                                            icon: Icon(Icons.lock),
                                            hintText: 'Enter the
password',
                                            hintStyle: TextStyle(
                                              color: Colors.grey,
                                            ),
                                            labelText: 'Password *',
                                          ),
                                          obscureText: true,
                                          validator:

RegexValidator.validatePassword,
                                          onSaved: (String value) {
                                            password = value;
                                          },
                                          // onChanged: (String value)
{
                                          //   // This optional block
of code can be used to run
                                          //   // code when the user
saves the form.
                                          //   password = value;
                                          // },
                                        ),
                                        SizedBox(
                                          height: 50.0,
                                        ),
                                        Padding(
                                          padding: const
EdgeInsets.symmetric(
                                            horizontal: 30.0,
                                          ),
                                          child: RaisedButton(
                                            elevation: 10.0,
                                            shape: StadiumBorder(),
                                            color:
Colors.lightBlueAccent,
                                            onPressed: () async {
                                              bool isShowSpinner =
                                                  _validateInputs();
                                              setState(() {
                                                showSpinner =
isShowSpinner;
                                              });
                                              try {
                                                // final newUser =
await _auth
                                                //
.createUserWithEmailAndPassword(
                                                //           email:
email,
                                                //           password:
password);
                                                assignRole();
```

```
                                            await _auth
.createUserWithEmailAndPassword(
                                                    email: email,
                                                    password:
password)
                                                .then((newUser) {
                                              if (newUser != null)
{
                                                HelperFunctions
.saveUserLoggedInSharedPreference(
                                                        true);
                                                HelperFunctions
.saveUserNameSharedPreference(
                                                        username);
                                                HelperFunctions
.saveUserEmailSharedPreference(
                                                        email);
                                                HelperFunctions
.saveUserPhotoUrlSharedPreference(
                                                        url);
                                                HelperFunctions
.saveUserRoleSharedPreference(
                                                        role);

Navigator.pushNamed(context,
ContactedPersonScreen().id);
                                              }
                                            });

                                            if (_image != null) {
                                              await
```

```
uploadPic(context);
                                            }


_addToDatabase(username, url);

                                            setState(() {
                                              showSpinner = false;
                                            });
                                            showDialog(
                                                context: context,
                                                builder:
                                                    (BuildContext
context) {

                                                  return

                                                    //uses the

custom alert dialog

                                                    alertTitle:


richTitle("CONGRATS!"),

                                                    alertSubtitle:

richSubtitle(

                                                        "Your

assign role is ${role.toUpperCase()}"),

                                                    alertType:


RichAlertType.SUCCESS,

                                                  );
                                                });
                                          } catch (e) {
                                            print(e.message);
                                            if (e.message ==
                                                "The email address

is already in use by another account.") {

                                              setState(() {
                                                showSpinner =

false;

                                              });
                                              showDialog(
                                                    context: context,
                                                    builder:
                                                        (BuildContext
context) {
                                                      return
RichAlertDialog(
                                                        //uses the
custom alert dialog
                                                        alertTitle:
richTitle("Try Again"),

alertSubtitle: richSubtitle("Already registered with this email
id."),
                                                        alertType:

RichAlertType.WARNING, actions: <Widget>[

RaisedButton(
                                                            color:
Colors.red,
                                                            child:
Text(
                                                              "OK",
                                                              style:
TextStyle(),
                                                            ),

onPressed: () {

Navigator.pop(

context);
                                                            },
                                                          ),
                                                        ],
                                                      );
                                                    });
                                              }
                                            }
                                          },
                                          child: Text(
                                            'Sign up',
                                            style:
KLoginRegistrationButtonStyle,
                                          ),
                                          padding:

EdgeInsets.symmetric(
                                            vertical: 15.0,
                                            horizontal: 60.0,
                                          ),
                                        ),
                                      ),
                                    ],
                                  ),
                                ),
                              ),
                            ],
                          ),
                        ),
                      ),
                    ),
                  ),
                ],
              ),
            ),
          ),
        ),
      );
    }
}
```



New
Account

Email *
test@gla.ac.in

Username *
test.35

Invalid username

Password *
••••

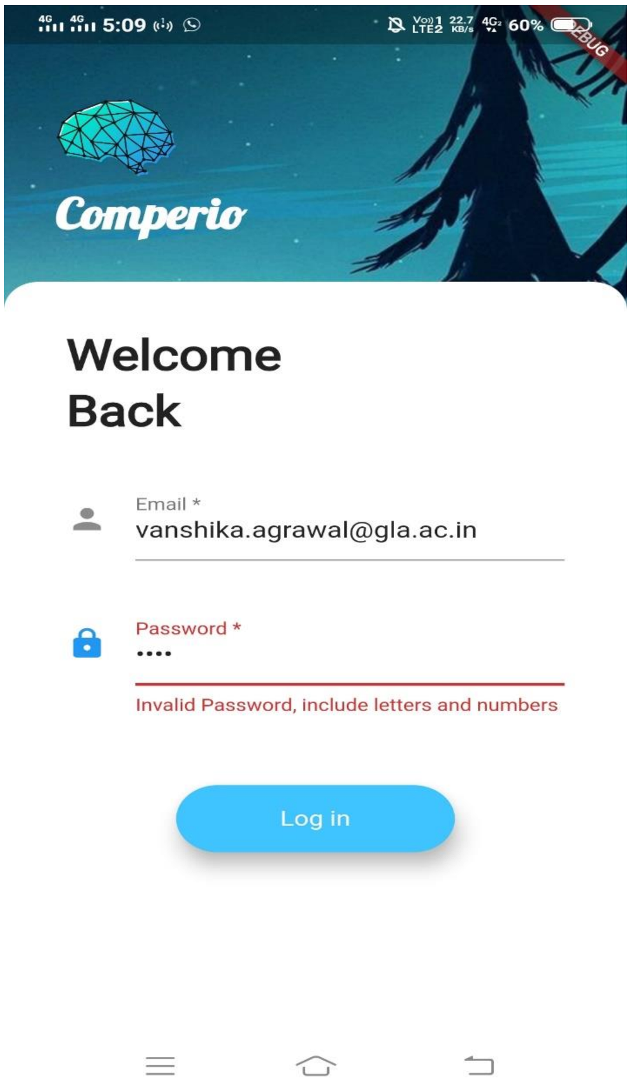
Invalid Password, include letters and numbers

Login Screen:

On this screen, the user will input the details such as email address, and password. When details don't match with registered details dialog box will be prompt.

```
import 'package:cloud_firestore/cloud_firestore.dart' show
FirebaseFirestore;
import 'package:comperio/checkInternet.dart';
import 'package:comperio/constants.dart';
import 'package:comperio/regexValidator.dart';
import 'package:comperio/screen/contacted_person_screen.dart';
import 'package:comperio/screen_app_logo.dart';
import 'package:firebase_auth/firebase_auth.dart';
import 'package:flutter/cupertino.dart';
import 'package:flutter/material.dart';
import 'package:modal_progress_hud/modal_progress_hud.dart';
import 'package:rich_alert/rich_alert.dart';

import '../helper_functions.dart';

class LoginScreen extends StatefulWidget {
  final String id = 'LoginScreen';

  @override
  _LoginScreenState createState() => _LoginScreenState();
}

class _LoginScreenState extends State<LoginScreen> {
  final GlobalKey<FormState> _formKey = GlobalKey<FormState>();
  bool _autoValidate = false;
  bool showSpinner = false;
  final _auth = FirebaseAuth.instance;
  String email;
  String password;

  @override
  void initState() {
    super.initState();
    CheckInternet().checkConnection(context);
  }

  @override
  void dispose() {
    CheckInternet().listener.cancel();
    super.dispose();
  }

  bool _validateInputs() {
    if (_formKey.currentState.validate()) {
//    If all data are  _autoValidate = true;correct then save data
to out variables
      _formKey.currentState.save();
      return true;
    } else {
//    If all data are not valid then start auto validation.
      setState(() {});
      return false;
    }
  }

  @override
  Widget build(BuildContext context) {
    return Container(
      decoration: BoxDecoration(
        color: Colors.lightBlueAccent,
        image: DecorationImage(
          image: AssetImage('images/app-background-2.jpg'),
          fit: BoxFit.cover,
        ),
      ),
      constraints: BoxConstraints.expand(),
      child: Scaffold(
        resizeToAvoidBottomInset: true,
        resizeToAvoidBottomPadding: true,
        backgroundColor: Colors.transparent,
        body: ModalProgressHUD(
          inAsyncCall: showSpinner,
          child: Container(
            child: Column(
              crossAxisAlignment: CrossAxisAlignment.start,
              children: <Widget>[
                Container(
                  padding: EdgeInsets.only(
                      top: 60.0, left: 30.0, right: 30.0, bottom:
30.0),
                  child: Column(
                    crossAxisAlignment: CrossAxisAlignment.start,
                    children: <Widget>[
                      ScreenAppLogo(),
                      Text(
                        'Comperio',
                        style: KAppNameTextStyle,
                      ),
                    ],
                  ),
                ),
                Flexible(
                  fit: FlexFit.tight,
                  child: Container(
                    width: double.infinity,
                    padding: EdgeInsets.symmetric(horizontal:
20.0),

                    decoration: BoxDecoration(
                      color: Colors.white,
                      borderRadius: BorderRadius.only(
                        topLeft: Radius.circular(20.0),
                        topRight: Radius.circular(20.0),
                      ),
                    ),
                    child: Padding(
                      padding: const EdgeInsets.only(
                          left: 16.0, right: 16.0, top: 35, bottom:
16.0),

                      child: Form(
                        key: _formKey,
                        autovalidate: _autoValidate,
                        child: SingleChildScrollView(
                          child: Column(
                            mainAxisSize: MainAxisSize.min,
                            crossAxisAlignment:
CrossAxisAlignment.stretch,
                            children: <Widget>[
                              Text(
                                'Welcome',
                                style: KCardTextStyle,
                              ),
                              SizedBox(
                                height: 5.0,
                              ),
                              Text(
                                'Back',
                                style: KCardTextStyle,
                              ),
                              SizedBox(
                                height: 30.0,
                              ),
                              Flexible(
                                fit: FlexFit.loose,
                                child: Column(
                                  mainAxisAlignment:

MainAxisAlignment.spaceEvenly,
                                  children: <Widget>[
                                    TextFormField(
                                      keyboardType:
TextInputType.emailAddress,
                                      decoration: const
InputDecoration(
                                        icon: Icon(Icons.person),
                                        hintText: 'Enter the
Email',
                                        hintStyle: TextStyle(
                                          color: Colors.grey,
                                        ),
                                        labelText: 'Email *',
                                      ),
                                      validator:
RegexValidator.validateEmail,
                                      onSaved: (String value) {
                                        // This optional block of
code can be used to run
                                        // code when the user saves
the form.
                                        email = value;
                                      },
                                    ),
                                    SizedBox(
                                      height: 30.0,
                                    ),
                                    TextFormField(
                                      decoration: const
InputDecoration(
                                        icon: Icon(Icons.lock),
                                        hintText: 'Enter the
password',
                                        hintStyle: TextStyle(
                                          color: Colors.grey,
                                        ),
                                        labelText: 'Password *',
                                      ),
                                      obscureText: true,
                                      validator:
RegexValidator.validatePassword,
                                      onSaved: (String value) {
                                        // This optional block of
code can be used to run
                                        // code when the user saves
the form.
                                        password = value;
                                      },
                                    ),
                                    SizedBox(
                                      height: 50.0,
                                    ),
                                    Padding(
                                      padding: const
EdgeInsets.symmetric(
                                        horizontal: 30.0,
                                      ),
                                      child: RaisedButton(
                                        elevation: 10.0,
                                        shape: StadiumBorder(),
                                        color:
Colors.lightBlueAccent,
                                        onPressed: () async {
                                          CheckInternet()

.checkConnection(context);

                                          bool isShowSpinner =
                                              _validateInputs();
                                          setState(() {
                                            showSpinner =
isShowSpinner;
                                          });
                                          try {
                                            CheckInternet()

.checkConnection(context);

                                            await _auth

.signInWithEmailAndPassword(
                                                             email: email,
                                                             password:
password)
                                                          .then((newUser)
async {
                                                        if (newUser != null)
{
                                                          var
userInfoSnapshot =
                                                              await
FirebaseFirestore
                                                                  .instance

.collection("users")

.where('email',
isEqualTo: email)
                                                                     .get()

.catchError((e) {

print(e.toString());
                                                            });

                                                            HelperFunctions

.saveUserLoggedInSharedPreference(
                                                                    true);
                                                            HelperFunctions

.saveUserNameSharedPreference(

userInfoSnapshot.docs[0]

.get('username'));
                                                            HelperFunctions

.saveUserEmailSharedPreference(

userInfoSnapshot.docs[0]

.get('email'));
                                                            HelperFunctions

.saveUserPhotoUrlSharedPreference(

userInfoSnapshot.docs[0]

.get('profileURL'));

                                                            HelperFunctions

.saveUserRoleSharedPreference(

userInfoSnapshot.docs[0]
```

```
.get('role'));


Navigator.pushNamed(context,

ContactedPersonScreen().id);
                                                        }
                                                      });
                                                      setState(() {
                                                        showSpinner = false;
                                                      });
                                                    } catch (e) {
                                                      print(e.message);
                                                      if (e.message ==
                                                              "The password
is invalid or the user does not have a password." ||
                                                          e.message ==
                                                              "There is no
user record corresponding to this identifier. The user may have
been deleted.") {
                                                        setState(() {
                                                          showSpinner =

false;

                                                        });
                                                        showDialog(
                                                            context: context,
                                                            builder:
                                                                (BuildContext

context) {

                                                              return

RichAlertDialog(

                                                                //uses the

custom alert dialog

                                                                alertTitle:

richTitle(

                                                                    "Try

Again"),


alertSubtitle: richSubtitle(

                                                                    "Wrong
email address or password."),
                                                                    },
                                                                  ),
                                                                ],
                                                              );
                                                            });
                                                      }
                                                    }
                                                  },
                                                  child: Text(
                                                    'Log in',
                                                    style:
KLoginRegistrationButtonStyle,
                                                  ),
                                                  padding:
EdgeInsets.symmetric(
                                                    vertical: 15.0,
                                                    horizontal: 60.0,
                                                  ),
                                                ),
                                              ),
                                            ],
                                          ),
                                        ),
                                      ],
                                    ),
                                  ),
                                ),
                              ),
                            ),
                          ),
                        ],
                      ),
                    ),
                  ),
                ),
              );
            }
}
```


Comperio
Welcome Back
Email *
vanshika.agrawal@gla.ac.in
Password *
••••
Invalid Password, include letters and numbers
Log in

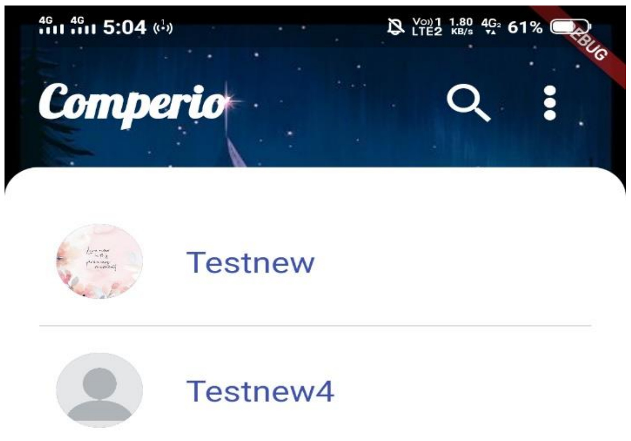
Contact person screen:

This Screen shows the persons list whom messages sent. Top bar has option to search and menu icon which has option to see profile, change password and log out.

```
import 'package:basic_utils/basic_utils.dart';
import 'package:cloud_firestore/cloud_firestore.dart';
import 'package:comperio/app_icons.dart';
import 'package:comperio/choice.dart';
import 'package:comperio/constants.dart';
import 'package:comperio/contact_popup_menu.dart';
import 'package:comperio/helper_functions.dart';
import 'package:comperio/screen/chat_screen.dart';
import 'package:comperio/screen/searchScreen.dart';
import 'package:firebase_auth/firebase_auth.dart';
import 'package:flutter/material.dart';
import 'package:flutter/services.dart';
import 'package:font_awesome_flutter/font_awesome_flutter.dart';

class ContactedPersonScreen extends StatefulWidget {
  final String id = 'ContactedPersonScreen';

  @override
  _ContactedPersonScreenState createState() =>
_ContactedPersonScreenState();
}

List<Choice> choices = const <Choice>[
  const Choice(title: 'Profile', icon: FontAwesomeIcons.user),
  const Choice(title: 'Change Password', icon:
FontAwesomeIcons.key),
  const Choice(title: 'Log out', icon:
FontAwesomeIcons.signOutAlt),
];
String chatRoomId;

class _ContactedPersonScreenState extends
```

```
State<ContactedPersonScreen> {
  final user = FirebaseAuth.instance.currentUser;
  String username;

  // Stream myStream;
  // String url ;

  getUserInfo() async {
    String name = await
HelperFunctions.getUserNameSharedPreference();
    setState(() {
      username = name;
    });
  }

  createChatRoom(
      BuildContext context, String receiver, String url, String
role) {
    List<String> users = [username, receiver];
    String chatRoomId = receiver;
    HelperFunctions.saveChatRoomIdSharedPreference(chatRoomId);
    Map<String, dynamic> chatRoom = {
      "users": users,
      "chatRoomId": chatRoomId,
      "profileUrl": url,
      "role": role,
    };

    FirebaseFirestore.instance
        .collection('users')
        .doc(username)
        .collection("chatRoom")
        .doc(chatRoomId)
        .set(chatRoom)
        .catchError((e) {
      print(e);
    });
    Navigator.pushNamed(context, ChatScreen().id);
  }
  @override
  void initState() {
    super.initState();
    getUserInfo();
  }

  Future<bool> _onWillPop() {
    return showDialog(
          context: context,
          builder: (context) => AlertDialog(
            title: Text('Are you sure?'),
            content: Text('Do you want to exit App'),
            actions: <Widget>[
              FlatButton(
                onPressed: () => Navigator.of(context).pop(false),
                child: Text('No'),
              ),
              FlatButton(
                onPressed: () => SystemNavigator.pop(),
                /*Navigator.of(context).pop(true)*/
                child: Text('Yes'),
              ),
            ],
          ),
        ) ??
        false;
  }

  @override
  Widget build(BuildContext context) {
    return WillPopScope(
      onWillPop: _onWillPop,
      child: Container(
        decoration: BoxDecoration(
          color: Color(0xff311b92),
          image: DecorationImage(
            image: AssetImage('images/app-background.jpg'),
            fit: BoxFit.cover,
          ),
        ),
        constraints: BoxConstraints.expand(),
        child: Scaffold(
          // floatingActionButton: FloatingActionButton(
          //   onPressed: () {
          //     // Add your onPressed code here!
          //   },
          //   child: AppIcons(
          //     iconName: Icons.group_add,
          //     iconSize: 30.0,
          //     colour: Colors.white,
          //   ),
          //   backgroundColor: Color(0xff0d47a1),
          // ),
          resizeToAvoidBottomInset: false,
          resizeToAvoidBottomPadding: false,
          backgroundColor: Colors.transparent,
          body: Column(
            crossAxisAlignment: CrossAxisAlignment.start,
            children: <Widget>[
              Container(
                padding: EdgeInsets.only(
                    top: 40.0, left: 20.0, right: 20.0, bottom:
20.0),
                child: Row(
                  mainAxisAlignment:
MainAxisAlignment.spaceBetween,
                  children: [
                    // CircleAvatar(
                    //   radius: 17.0,
                    //   backgroundColor: Colors.white,
                    // ),
                    // SizedBox(
                    //   width: 7.0,
                    // ),
                    Text(
                      'Comperio',
                      style: KAppNameTextStyle,
                    ),
                    Row(
                      mainAxisAlignment: MainAxisAlignment.end,
                      children: [
                      IconButton(
                        icon: Hero(
                          tag:"logo1",
                          child: AppIcons(
                              iconName: Icons.search,
                              colour: Colors.white,
                              iconSize: 35.0),
                        ),
                        onPressed: () {
                          Navigator.pushNamed(context,
SearchScreen().id);
                        },
                      ),
                        ContactPopupMenu(choices: choices),
                      ],
                    ),
                  ],
                ),
              ),
              Expanded(
                child: Container(
                  padding: EdgeInsets.symmetric(horizontal: 20.0),
                  decoration: BoxDecoration(
                    color: Colors.white,
                    borderRadius: BorderRadius.only(
                      topLeft: Radius.circular(20.0),
                      topRight: Radius.circular(20.0),
                    ),
                  ),
                  child: StreamBuilder<QuerySnapshot>(
                    stream: FirebaseFirestore.instance
                        .collection('users')
                        .doc(username)
                        .collection('chatRoom')
                        .snapshots(),
                    builder: (context, snapshot) {
                      return (snapshot.connectionState ==
                              ConnectionState.waiting)
                          ? Center(child:
```

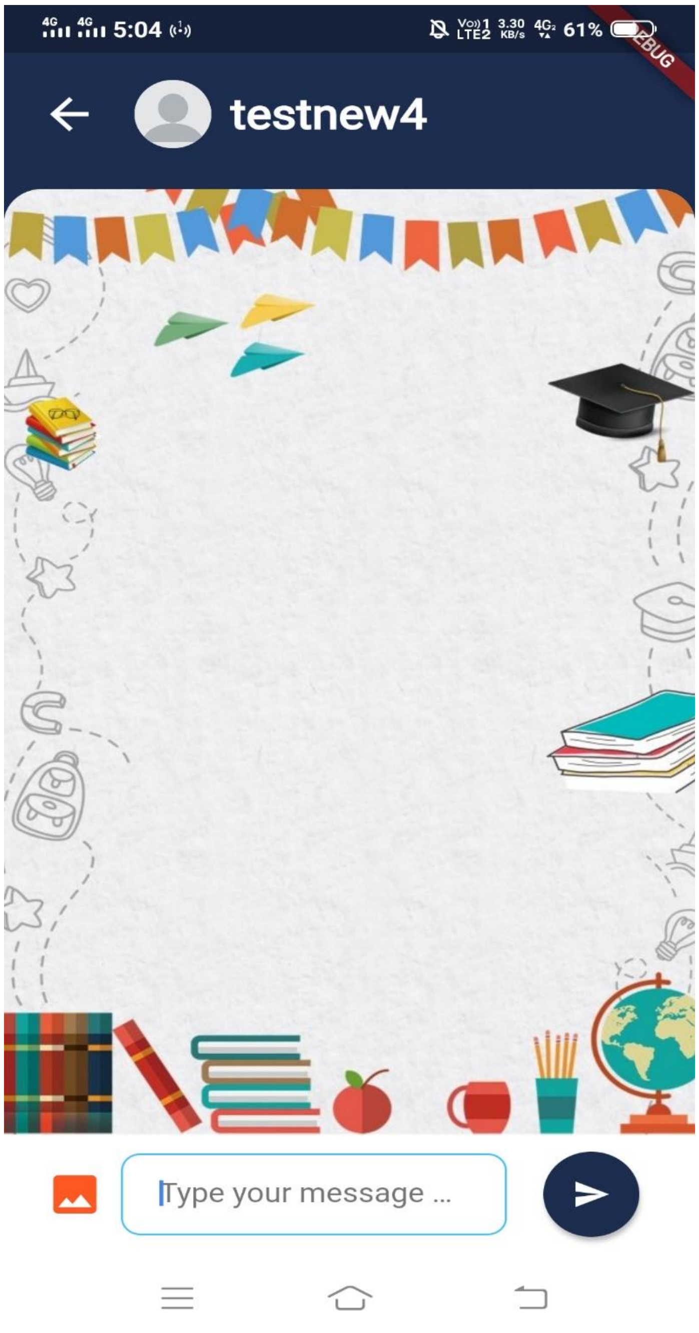
```
CircularProgressIndicator())
                          : ListView.separated(
                              separatorBuilder: (context, index) =>
Divider(
                                    color: Colors.grey,
                                  ),
                              itemCount: snapshot.data.docs.length,
                              itemBuilder: (context, index) {
                                DocumentSnapshot data =
                                    snapshot.data.docs[index];
                                //
print(data.data()['profileUrl']);
                                // print(data.data()['users'][1]);
                                return Padding(
                                  padding: const
EdgeInsets.all(6.0),
                                  child: ListTile(
                                    // padding:
EdgeInsets.all(10.0),
                                    // child: Row(
                                    //   children:


leading:(data.data()['profileUrl'] != null)
                                            ? CircleAvatar(
                                                radius: 25.0,
                                                child: ClipOval(
                                                  child: SizedBox(
                                                    width: 50.0,
                                                    height: 50.0,
                                                    child:
Image.network(

data.data()['profileUrl'],
                                                      width: 50,
                                                      height: 50,
                                                      fit:
BoxFit.fill,
                                                    ),
                                                  ),
                                                ),
                                              )
Image.asset(

'images/default-profile.jpg',
                                                      width: 50,
                                                      height: 50,
                                                      fit:

BoxFit.fill,
                                                    ),
                                                  ),
                                                ),
                                              ),
                                        // SizedBox(
                                        //   width: 25.0,
                                        // ),
                                        title:Text(
                                          StringUtils.capitalize(

data.data()['users'][1]),

                                          style:

KSearchDisplayNameTextStyle,
                                        ),
                                    hoverColor: Colors.blueAccent,
                                    onTap: () {
                                      String user =

data.data()['users'][1];
                                      String url =

data.data()['profileUrl'];
                                      String role =

data.data()['role'];
                                      createChatRoom(context, user,

url, role);
                                    },
                                    //   ],
                                    // ),
                                  ),
                                );
                              });
                    },
                  ),
                ),
              ),
            ],
          ),
        ),
      ),
    );
  }
}
```


5:04
61%
DEBUG
Comperio
Testnew
Testnew4

Chat Screen:

User can type in the message and send images he/she wants to send. For Student to give feedback to professor there is an icon of feedback on top-right corner.

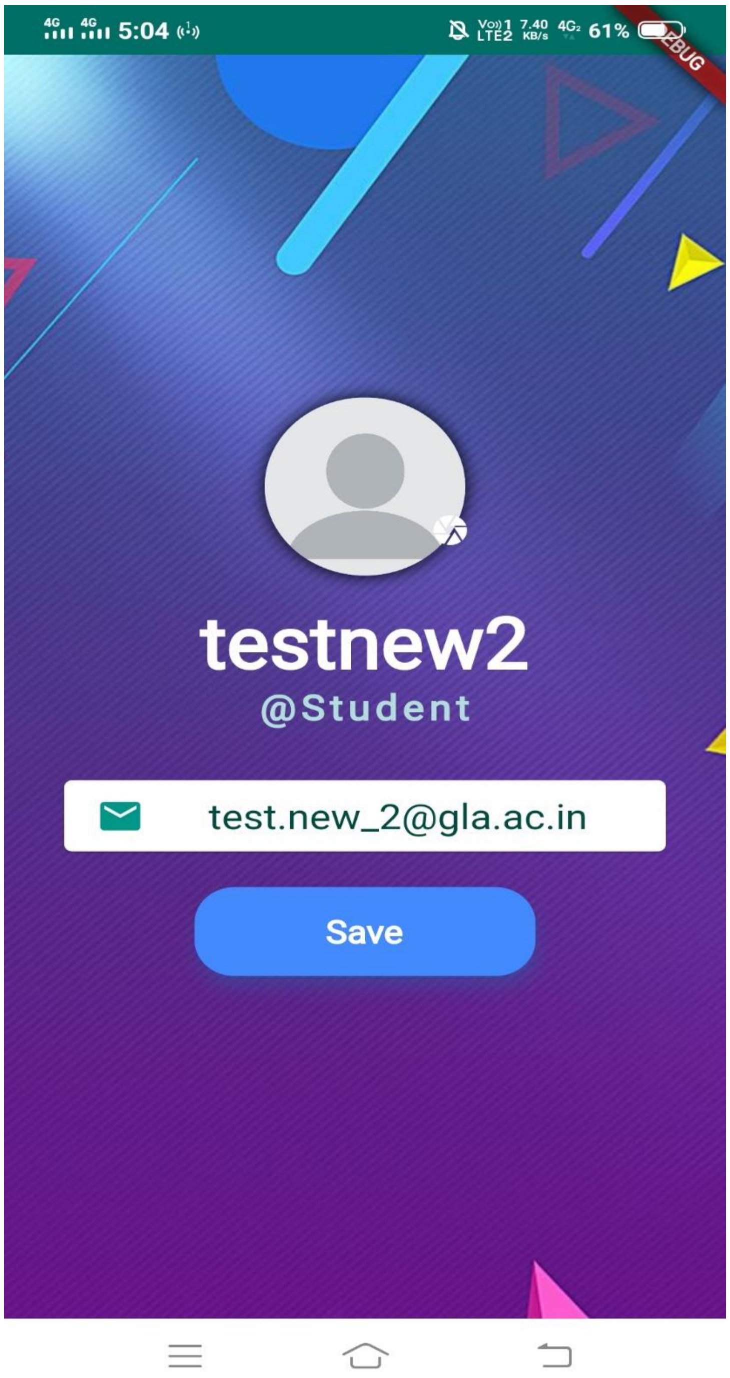
```
import 'package:cloud_firestore/cloud_firestore.dart';
import 'package:comperio/app_icons.dart';
import 'package:comperio/attach_file_components.dart';
import 'package:comperio/constants.dart';
import 'package:comperio/helper_functions.dart';
import 'package:comperio/screen/contacted_person_screen.dart';
import 'package:firebase_auth/firebase_auth.dart';
import 'package:flutter/cupertino.dart';
import 'package:flutter/material.dart';
import 'package:flutter/painting.dart';
import 'package:intl/intl.dart';

import 'feedback_screen.dart';

final _firestore = FirebaseFirestore.instance;
User loggedInUser;
String chatRoomId = "test";
String username = "admin";
String url = "";
String myUrl = "";
String myrole = "";
String receiverRole = "";

class ChatScreen extends StatefulWidget {
  final String id = 'ChatScreen';

  @override
  _ChatScreenState createState() => _ChatScreenState();
}

class _ChatScreenState extends State<ChatScreen> {
  final messageTextController = TextEditingController();
  final _auth = FirebaseAuth.instance;
  String messageText = "";

  getUserName() async {
    String chatId = await
HelperFunctions.getChatRoomIdSharedPreference();
    String myUsername = await
HelperFunctions.getUserNameSharedPreference();
    var receiverInfo =
        await
FirebaseFirestore.instance.collection('users').doc(chatId).get();
    String myUrl = await
HelperFunctions.getUserPhotoUrlSharedPreference();
    String MyRole = await
HelperFunctions.getUserRoleSharedPreference();

    // String pUrl = ;
    setState(() {
      chatRoomId = chatId;
      username = myUsername;
      url = receiverInfo.data()['profileURL'];
      receiverRole = receiverInfo.data()['role'];
      myUrl = myUrl;
      myrole = MyRole;
    });
    print(url);
    print(chatRoomId);
    print(myrole);
  }

  @override
  void initState() {
    super.initState();
    getCurrentUser();
    getUserName();
  }

  void getCurrentUser() async {
    try {
      final user = await _auth.currentUser;
      if (user != null) {
        loggedInUser = user;
        print(loggedInUser.email);
      }
    } catch (e) {
      print(e);
    }
  }

  void messageStream() async {
    await for (var snapshot in _firestore
        .collection('users')
        .doc(username)
        .collection("chatRoom")
        .doc(chatRoomId)
        .collection('Messages')
        .snapshots()) {
      for (var message in snapshot.docs) {
        print(message.data());
      }
    }
  }

  @override
  Widget build(BuildContext context) {
    return Container(
      decoration: BoxDecoration(
        color: Color(0xFF1d2d50),
      ),
      constraints: BoxConstraints.expand(),
      child: Scaffold(
        resizeToAvoidBottomInset: false,
        resizeToAvoidBottomPadding: false,
        backgroundColor: Colors.transparent,
        body: Column(
          crossAxisAlignment: CrossAxisAlignment.start,
          children: <Widget>[
            Container(
              padding: EdgeInsets.only(
                  top: 40.0, left: 10.0, right: 20.0, bottom:
20.0),
              child: Row(
                mainAxisAlignment: MainAxisAlignment.spaceBetween,
                children: [
                  Row(
                    mainAxisAlignment: MainAxisAlignment.start,
                    children: [
                      IconButton(
                        icon: AppIcons(
                          iconName: Icons.arrow_back,
                          iconSize: 30.0,
                          colour: Colors.white,
                        ),
                        onPressed: () {
                          Navigator.pushNamed(
                              context, ContactedPersonScreen().id);
                        },
                      ),
                      SizedBox(
                        width: 10.0,
                      ),
                      CircleAvatar(
                        radius: 20.0,
                        backgroundColor: Colors.white,
                        child: ClipOval(
                          child: SizedBox(
                            width: 40.0,
                            height: 40.0,
                            child: Image.network(
                              url,
                              width: 50,
                              height: 50,
                              fit: BoxFit.fill,
                            ),
                          ),
                        ),
                      ),
                      SizedBox(
                        width: 10.0,
                      ),
                      Text(
                        chatRoomId,
                        style: KUserTextStyle,
                      ),
                      // SizedBox(
                      //   width: 100.0,
                      // ),
                    ],
                  ),
                  (myrole == "Student")
                      ? (receiverRole == "Professor")
                          ? IconButton(
                              alignment: Alignment.centerRight,
                              icon: AppIcons(
                                iconName: Icons.feedback,
                                iconSize: 30.0,
                                colour: Colors.white,
                              ),
                              onPressed: () {
                                Navigator.pushNamed(
                                    context, FeedbackScreen().id);
                              },
                            )
                          : Container()
                      : Container(),
                ],
              ),
            ),
            Expanded(
              child: Container(
                decoration: BoxDecoration(
                  color: Colors.white,
                  image: DecorationImage(
                    image: AssetImage('images/chat_bg8.jpeg'),
                    fit: BoxFit.cover,
                  ),
                  borderRadius: BorderRadius.only(
                    topLeft: Radius.circular(20.0),
                    topRight: Radius.circular(20.0),
                  ),
                ),
                child: Column(
                  mainAxisAlignment:
MainAxisAlignment.spaceBetween,
                  crossAxisAlignment: CrossAxisAlignment.stretch,
                  children: <Widget>[
                    MessagesStream(),
                  ],
                ),
              ),
            ),
            Container(
              padding: EdgeInsets.all(10.0),
              color: Colors.white,
              child: SingleChildScrollView(
                child: Container(
                  margin: EdgeInsets.only(bottom: 5.0, left: 3.0),
                  padding: EdgeInsets.only(
                      bottom:
MediaQuery.of(context).viewInsets.bottom),
                  child: Row(
                    crossAxisAlignment: CrossAxisAlignment.center,
                    children: <Widget>[
                      AttachFileBottomSheet(),
                      Expanded(
                        child: TextField(
                          keyboardType: TextInputType.multiline,
                          maxLines: null,
                          controller: messageTextController,
                          onChanged: (value) {
                            //do something when pressed
                            messageText = value;
                          },
                          decoration: InputDecoration(
                            contentPadding: EdgeInsets.symmetric(
                                vertical: 10.0, horizontal: 20.0),
                            hintText: 'Type your message here...',
                            enabledBorder: OutlineInputBorder(
                              borderRadius:

BorderRadius.all(Radius.circular(12.0)),
                              borderSide: BorderSide(
                                color: Colors.grey,
                              ),
                            ),
                            focusedBorder: OutlineInputBorder(
                              borderRadius:

BorderRadius.all(Radius.circular(10.0)),
                              borderSide:
                                  BorderSide(color:
Colors.lightBlueAccent),
                            ),
                          ),
                        ),
                      ),
                      RaisedButton(
                        padding: EdgeInsets.all(15.0),
                        color: Color(0xFF1d2d50),
                        onPressed: () {
                          //Implement send functionality.
                          messageTextController.clear();
                          if (messageText.isNotEmpty) {
                            _firestore
                                .collection('users')
                                .doc(username)
                                .collection("chatRoom")
                                .doc(chatRoomId)
                                .collection('Messages')
                                .add({
                              'message': messageText,
                              'sender': loggedInUser.email,
                              'Date': DateFormat('dd-MMM-yy hh:mm')
                                  .format(DateTime.now())
                                  .toString(),
                              'fileUrl': " ",
                              'type': " ",
                              'orderDateFormat':
                                  DateFormat('dd-MMM-yy hh:mm:ss')
                                      .format(DateTime.now())
                                      .toString(),
                            });
                            FirebaseFirestore.instance
                                .collection('users')
                                .doc(chatRoomId)
                                .collection("chatRoom")
                                .doc(username)
                                .set({
                              "users": [chatRoomId, username],
                              "chatRoomId": username,
                              "profileUrl": myUrl,
                            });

                            _firestore
                                .collection('users')
                                .doc(chatRoomId)
                                .collection("chatRoom")
                                .doc(username)
                                .collection('Messages')
                                .add({
                              'message': messageText,
                              'sender': loggedInUser.email,
                              'Date': DateFormat('dd-MMM-yy hh:mm')
                                  .format(DateTime.now())
                                  .toString(),
                              'fileUrl': " ",
                              'type': " ",
                              'orderDateFormat':
                                  DateFormat('dd-MMM-yy hh:mm:ss')
                                      .format(DateTime.now())
                                      .toString(),
                            });
                          }

                          setState(() {
                            messageText = "";
                          });
                        },
                        shape: CircleBorder(),
                        child: AppIcons(
                          iconName: Icons.send,
                          iconSize: 20.0,
                          colour: Colors.white,
                        ),
                      ),
                    ],
                  ),
                ),
              ),
            )
          ],
        ),
      ),
    );
  }
}

class MessagesStream extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return StreamBuilder<QuerySnapshot>(
      stream: _firestore
          .collection('users')
          .doc(username)
          .collection("chatRoom")
          .doc(chatRoomId)
          .collection('Messages')
          .orderBy('orderDateFormat')
          .snapshots(),
      builder: (context, snapshot) {
        if (!snapshot.hasData) {
          return Center(
            child: CircularProgressIndicator(
              backgroundColor: Colors.lightBlueAccent,
            ),
          );
        }
        final messages = snapshot.data.docs.reversed;
        List<MessageBubble> messageBubbles = [];
        for (var message in messages) {
          final messageText = message.data()['message'];
          final messageSender = message.data()['sender'];
          final messageDateTime = message.data()['Date'];
          final messageFile = message.data()['fileUrl'];
          final messageType = message.data()['type'];
          final currentUser = loggedInUser.email;

          final messageBubble = MessageBubble(
            sender: messageSender,
            text: messageText,
            dateTime: messageDateTime,
            isMe: currentUser == messageSender,
            messageFileUrl: messageFile,
          );
          messageBubbles.add(messageBubble);
        }
        return Expanded(
          child: ListView(
            reverse: true,
            padding: EdgeInsets.symmetric(
              horizontal: 10.0,
              vertical: 20.0,
            ),
            children: messageBubbles,
          ),
        );
      },
    );
  }
}

class MessageBubble extends StatelessWidget {
  final String sender;
  final String text;
  final bool isMe;
  final String dateTime;
  final String messageFileUrl;
  final String messageType;

  MessageBubble(
      {this.sender,
      this.text,
      this.isMe,
      this.dateTime,
      this.messageFileUrl,
      this.messageType});

  @override
  Widget build(BuildContext context) {
    return Padding(
      padding: EdgeInsets.all(10.0),
      child: Column(
        crossAxisAlignment:
            isMe ? CrossAxisAlignment.end :
CrossAxisAlignment.start,
        children: <Widget>[
          // Text(
          //   sender,
          //   style: TextStyle(
          //     fontSize: 12.0,
          //     color: Colors.black54,
          //   ),
          // ),
          (messageFileUrl == " ")
              ? Container(
                  constraints: BoxConstraints(
                    maxWidth: 2 *
(MediaQuery.of(context).size.width) / 3,
                  ),
                  child: Material(
                    borderRadius: isMe
                        ? BorderRadius.only(
                            topLeft: Radius.circular(30.0),
                            bottomLeft: Radius.circular(30.0),
                            bottomRight: Radius.circular(30.0),
                          )
                        : BorderRadius.only(
                            topRight: Radius.circular(30.0),
                            bottomLeft: Radius.circular(30.0),
                            bottomRight: Radius.circular(30.0),
                          ),
                    elevation: 10.0,
                    color: isMe ? Color(0xff81d4fa) : Colors.white,
                    child: Padding(
                      padding: EdgeInsets.symmetric(
                          vertical: 12.0, horizontal: 20.0),
                      child: Text(
                        text,
                        style: TextStyle(
                          fontSize: 15.0,
                          color: Colors.black,
                        ),
                      ),
                    ),
                  ),
                )
              : AttachFileBottomSheet()
                  .getImageBubble(messageFileUrl, context, isMe),
          Text(
            dateTime,
            style: TextStyle(
              fontSize: 10.0,
              color: Colors.black,
            ),
          ),
        ],
      ),
    );
  }
}
```


5:04
61%
DEBUG
testnew4
Type your message ...

Profile Screen:

This screen shows user details. The user can change the image. After selecting image by clicking on image and clicking on save, image will successfully updated and message prompt.

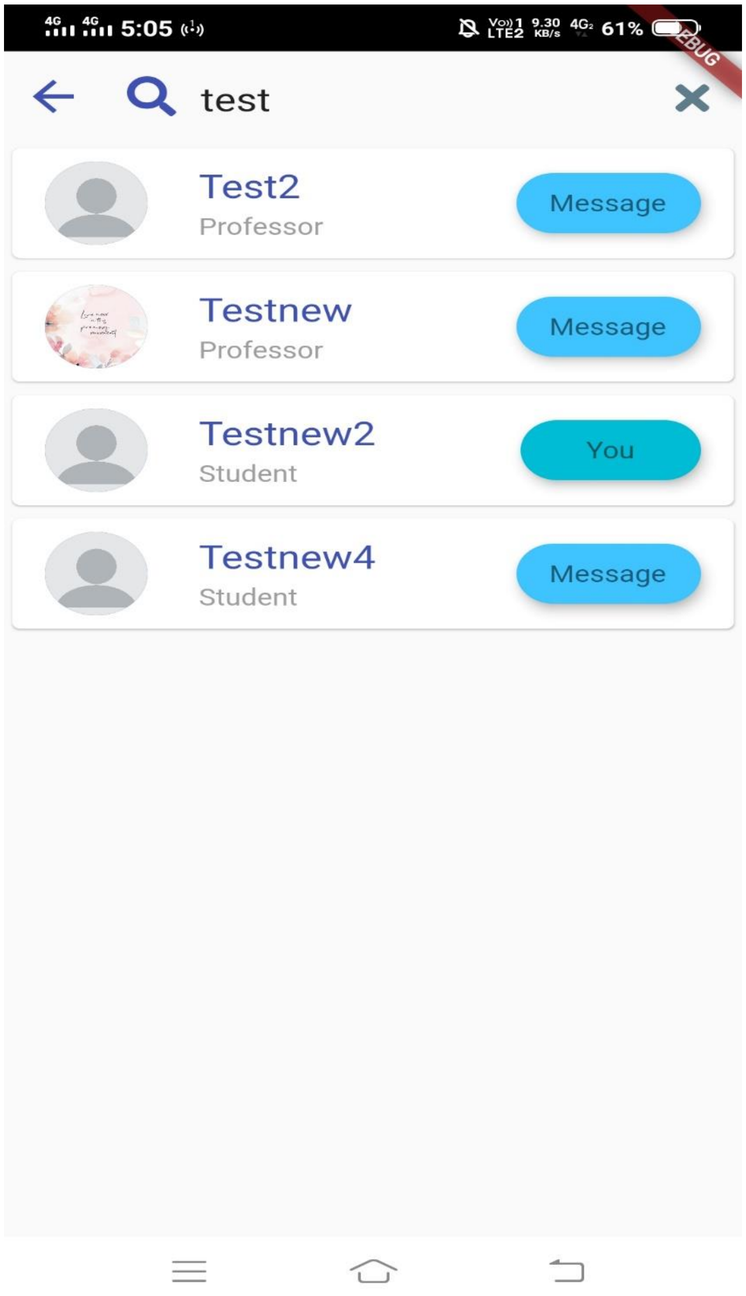
```
// To-Do:
// -when viewing own profile:
// fetch current user name, email and profile photo, provide edit
options,  don't show send message button
// -when viewing other person profile:
// fetch searched result's name, email and profile photo, show send
message button

// -when viewing faculty profile:
// show rating
// -when viewing student profile:
// don't show rating

// - adding rating mechanism

// - show @professor when faculty and @student when student's
profile

import 'dart:io';
import 'package:cloud_firestore/cloud_firestore.dart';
import 'package:comperio/helper_functions.dart';
import 'package:firebase_storage/firebase_storage.dart';
import 'package:flutter/material.dart';
import 'package:image_picker/image_picker.dart';
import 'package:modal_progress_hud/modal_progress_hud.dart';
import 'package:path/path.dart';
import 'package:comperio/constants.dart';
import 'package:commons/commons.dart';

class ProfileScreen extends StatefulWidget {
  final String id = 'ProfileScreen';

  @override
  _ProfileScreenState createState() => _ProfileScreenState();
}

class _ProfileScreenState extends State<ProfileScreen> {
  String userName;
  String emailId;
  String picURL =
      'https://firebasestorage.googleapis.com/v0/b/comperio-
1071d.appspot.com/o/default-profile.webp?alt=media&token=52b10457-
a10a-417e-b5af-3d84e5833fae';
  File _image;
  bool showSpinner = false;
  String url = "";
  String role;

  getUserInfo() async {
    String username = await
HelperFunctions.getUserNameSharedPreference();
    String email = await
HelperFunctions.getUserEmailSharedPreference();
    String myrole = await
HelperFunctions.getUserRoleSharedPreference();
    var dp = await FirebaseFirestore.instance
        .collection('users')
        .doc(username)
        .get();
    setState(() {
      userName = username;
      emailId = email;
      role = myrole;
      picURL = dp.data()['profileURL'];
    });
    print(picURL);
  }

  @override
  void initState() {
    super.initState();
    getUserInfo();
  }
  Future getImages() async {
    PickedFile pickedFile =
        await ImagePicker().getImage(source: ImageSource.gallery);
    var image = File(pickedFile.path);

    setState(() {
      _image = image;
      // print('Image path $_image');
    });
  }

  Future uploadPic(BuildContext context) async {
    String fileName = basename(_image.path);
    StorageReference firebaseStorageRef =
        FirebaseStorage.instance.ref().child(fileName);
    StorageUploadTask uploadTask =
firebaseStorageRef.putFile(_image);
    var dowUrl = await (await
uploadTask.onComplete).ref.getDownloadURL();

    url = dowUrl.toString();

    //print(url);
    StorageTaskSnapshot taskSnapshot = await uploadTask.onComplete;
    setState(() {
      print("Profile Picture uploaded");
      // Scaffold.of(context)
      //     .showSnackBar(SnackBar(content: Text('Profile Picture
Uploaded')));
    });
  }

  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      home: Scaffold(
        backgroundColor: Colors.teal,
        body: SafeArea(
          child: ModalProgressHUD(
            inAsyncCall: showSpinner,
            child: Container(
              constraints: BoxConstraints.expand(),
              decoration: BoxDecoration(
                  image: DecorationImage(
                      image: AssetImage('images/app-background-
3.jpg'),
                      fit: BoxFit.cover)),
              child: Column(
                crossAxisAlignment: CrossAxisAlignment.center,
                mainAxisAlignment: MainAxisAlignment.center,
                children: <Widget>[
                  Container(
                    decoration: BoxDecoration(
                      color: Colors.white,
                      shape: BoxShape.circle,
                      boxShadow: [
                        BoxShadow(
                          blurRadius: 7.0,
                          color: Colors.black,
                        )
                      ],
                    ),
                    child: Stack(
                      alignment: Alignment.center,
                      overflow: Overflow.visible,
                      children: [
                        _image != null
                            ? CircleAvatar(
                                radius: 50.0,
                                backgroundColor: Colors.black12,
                                child: ClipOval(
                                  child: SizedBox(
                                    width: 100.0,
                                    height: 100.0,
                                    child: (_image != null)
                                        ? Image.file(
                                            _image,
                                            fit: BoxFit.fill,
                                          )
                                        : Image.asset(
                                            'images/default-
profile.jpg',
                                            fit: BoxFit.fill,
                                          ),
                                  ),
                                ),
                              )
                            : CircleAvatar(
                                radius: 50.0,
                                backgroundImage: (picURL != null)
                                    ? NetworkImage(picURL)
                                    : AssetImage('images/default-
profile.jpg'),
                              ),
                        Positioned(
                          top: -1,
                          right: -45,
                          bottom: -50,
                          left: 40,
                          child: IconButton(
                            icon: Icon(
                              Icons.camera,
                              color: Colors.white,
                              size: 20,
                            ),
                            onPressed: () {
                              getImages();
                            },
                          ),
                        ),
                      ],
                    ),
                  ),
                  SizedBox(
                    height: 15.0,
                  ),
                  Text(
                    userName != null ? userName : 'abc',
                    style: KProfileUsernameTextStyle,
                  ),
                  Text(
                    role != null ? "@" + role : "@role",
                    style: KProfileUserRoleTextStyle,
                  ),

                  SizedBox(
                    height: 10.0,
                  ),

                  Card(
                    color: Colors.white,
                    margin:
                        EdgeInsets.symmetric(vertical: 20.0,
horizontal: 30.0),
                    child: ListTile(
                      visualDensity: VisualDensity(horizontal: 0,
vertical: -4),
                      leading: Icon(Icons.email, color:
Colors.teal),
                      title: Align(
                        child: Text(
                          emailId != null ? emailId :
'abc@gmail.com',
                          style: KProfileEmailTextStyle,
                        ),
                        alignment: Alignment.bottomLeft,
                      ),
                    ),
                  ),

                  // adding rating bar showing the overall average
rating
//                Container(
//                  RatingBar(
//    initialRating: 3,
//    minRating: 1,
//    direction: Axis.horizontal,
//    allowHalfRating: true,
//    itemCount: 5,
//    itemPadding: EdgeInsets.symmetric(horizontal: 4.0),
//    itemBuilder: (context, _) => Icon(
//      Icons.star,
//      color: Colors.amber,
//    ),
//    onRatingUpdate: (rating) {
//      print(rating);
//    },
// );),
                  Container(
                      height: 50.0,
                      width: 170.0,
                      child: Material(
                        borderRadius: BorderRadius.circular(20.0),
                        shadowColor: Colors.blueAccent,
                        color: Colors.blueAccent,
                        elevation: 7.0,
                        child: GestureDetector(
                          onTap: () async {
                            setState(() {
                              showSpinner = true;
                            });
                            if (_image != null) {
                              await uploadPic(context);
                            }
                            // print(url);
                            FirebaseFirestore.instance
                                .collection('users')
                                .doc(userName)
                                .update({
                              'profileURL': url,
                            });
                            setState(() {
                              showSpinner = false;
                            });
                            // showDialog(
                            //     context: context,
                            //     builder: (BuildContext context)
{
                            //       return AlertDialog(
                            //          title: Text("Successful!"),
                            //          content: Text("Profile Pic 
Updated."),
                            //          actions: [
                            //            FlatButton(
                            //              child: Text("OK"),
                            //              onPressed: () {
                            //                
Navigator.of(context).pop();
                            //              },
                            //            )
                            //          ],
                            //        );
                            //      });
                            successDialog(
                              context,
                              "Profile Pic Updated",
                              neutralText: "Okay",
                            );
                          },
                          child: Center(
                            child: Text(
                              'Save',
                              style: KProfileButtonTextStyle,
                            ),
                          ),
                        ),
                      ))
                ],
              ),
            ),
          ),
        ),
      ),
    );
  }
}
```


5:04
61%
DEBUG
testnew2
@Student
test.new_2@gla.ac.in
Save

Search Screen:

This Screen shows the list of people who are registered on app, matching with the string entered by user. Clicking on "Message" navigate to that person chat screen and "You" navigate to his/her profile.

```
import 'package:comperio/app_icons.dart';
import 'package:comperio/search_stream_builder.dart';
import 'package:flutter/material.dart';
import 'package:font_awesome_flutter/font_awesome_flutter.dart';
import 'package:comperio/constants.dart';

class SearchScreen extends StatefulWidget {
  final String id = 'SearchScreen';

  @override
  _SearchScreenState createState() => _SearchScreenState();
}

class _SearchScreenState extends State<SearchScreen> {
  final searchText = TextEditingController();
  String name = "";

  @override
  Widget build(BuildContext context) {
    return SafeArea(
      child: Scaffold(
        body: Container(
          child: Column(
            children: [
              Row(
                children: [
                  IconButton(
                    icon: AppIcons(
                      iconName: Icons.arrow_back,
                      iconSize: 30.0,
                      colour: Colors.indigo,
                    ),
                    onPressed: () {
                      Navigator.pop(context);
                    },
                  ),
                  Expanded(
                    child: TextField(
                      controller: searchText,
                      cursorColor: Colors.indigo,
                      autofocus: true,
                      textInputAction: TextInputAction.search,
                      style: TextStyle(
                        height: 1.5,
                        fontSize: 20.0,
                      ),
                      decoration: InputDecoration(
                        prefixIcon:
                        Hero(
                            tag:'logo1',
                            child: Icon(FontAwesomeIcons.search,
color: Colors.indigo)),
                        suffixIcon: name.length > 0
                            ? IconButton(
                                icon: Icon(
                                  FontAwesomeIcons.times,
                                  color: Colors.blueGrey,
                                ),
                                onPressed: () {
                                  setState(() {
                                    searchText.clear();
                                    name = "";
                                  });
                                },
                              )
                            : null,
                        hintText: 'Search...',
                        hintStyle: KSearchScreenHintTextStyle,
                        border: InputBorder.none,
                      ),
                      onChanged: (value) {
                        // Search by username
                        setState(() {
                          name = value;
                        });
                      },
                    ),
                  ),
                ],
              ),
              Expanded(
                child: Container(
                  child: SearchStreamBuilder(
                    username: name,
                  ),
                ),
              ),
            ],
          ),
        ),
      ),
    );
  }
}
```



test

Test2
Professor

Testnew
Professor

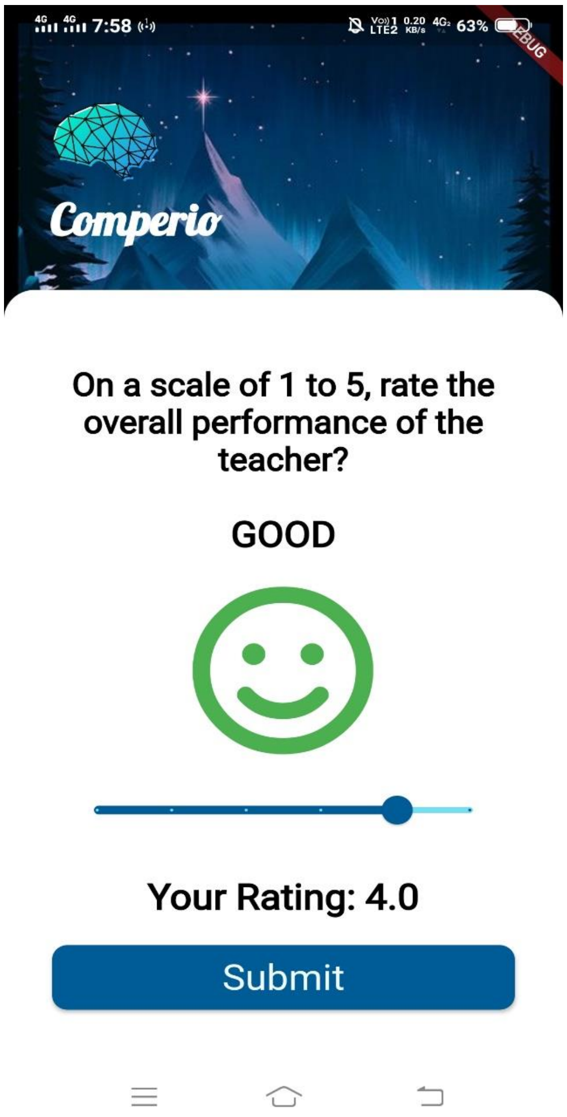
Testnew2
Student

Testnew4
Student

Feedback screen:

Students has option to give feedback to professors which cannot be seen by them. Feedback can be given from scale of 0 to 5.

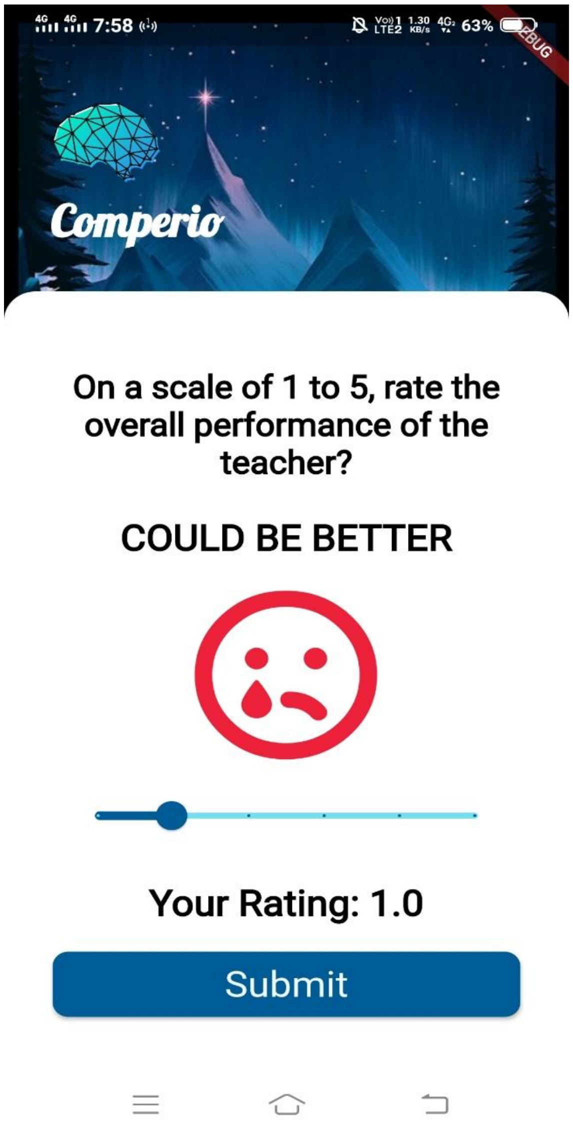
```
import 'package:cloud_firestore/cloud_firestore.dart' show
FirebaseFirestore;
import 'package:cloud_firestore/cloud_firestore.dart';
import 'package:flutter/cupertino.dart';
import 'package:flutter/material.dart';
import 'package:font_awesome_flutter/font_awesome_flutter.dart';
import 'package:modal_progress_hud/modal_progress_hud.dart';
import 'package:rich_alert/rich_alert.dart';
import '../constants.dart';
import '../screen_app_logo.dart';
import 'chat_screen.dart';

class FeedbackScreen extends StatefulWidget {
  final String id = 'FeedbackScreen';

  @override
  _FeedbackScreenState createState() => _FeedbackScreenState();
}

class _FeedbackScreenState extends State<FeedbackScreen> {
  String myFeedbackText = "COULD BE BETTER";
  double sliderValue = 0.0;
  IconData myFeedback = FontAwesomeIcons.sadTear;
  Color myFeedbackColor = Colors.white;
  bool showSpinner = false;

  addFeedbackToDatabase(double sliderValue, String user) async {
    var fd =
        await
FirebaseFirestore.instance.collection('users').doc(user).get();
    double fdValue = fd.data()['feedbackValue'];
    // double fdValueDouble = double.parse(fdValue);
    sliderValue = sliderValue + fdValue;

    double fdRate = fd.data()['feedbackRate'];
    // double fdRateDouble;
    // fdRateDouble = double.parse(fdRate);
    fdRate = fdRate + 1.0;

    // fdValue = sliderValue.toString();
    // fdRate = fdRateDouble.toString();

    double overallFeedbackDouble = sliderValue / fdRate;
    // String overallFeedback =
overallFeedbackDouble.toStringAsFixed(2);

    setState(() {

FirebaseFirestore.instance.collection('users').doc(user).update({
        'feedbackValue': sliderValue,
        'feedbackRate': fdRate,
        'overallFeedback': overallFeedbackDouble,
      });
    });
  }

  @override
  Widget build(BuildContext context) {
    return Container(
      decoration: BoxDecoration(
        image: DecorationImage(
            image: AssetImage("images/app-background.jpg"), fit:
BoxFit.cover),
      ),
      constraints: BoxConstraints.expand(),
      child: Scaffold(
        resizeToAvoidBottomInset: true,
        resizeToAvoidBottomPadding: true,
        backgroundColor: Colors.transparent,
        body: ModalProgressHUD(
          inAsyncCall: showSpinner,
          child: Container(
            child: Column(
              crossAxisAlignment: CrossAxisAlignment.start,
              children: <Widget>[
                Container(
                  padding: EdgeInsets.only(
                      top: 60.0, left: 30.0, right: 30.0, bottom:
30.0),
                  child: Container(
                      child: Column(
                    crossAxisAlignment: CrossAxisAlignment.start,
                    children: <Widget>[
                      ScreenAppLogo(),
                      Text(
                        'Comperio',
                        style: KAppNameTextStyle,
                      ),
                    ],
                  )),
                ),
                Flexible(
                  //fit: FlexFit.tight,
                  child: Container(
                    child: Container(
                        width: double.infinity,
                        padding: EdgeInsets.only(left: 10.0, right:
10.0),

                        decoration: BoxDecoration(
                          color: Colors.white,
                          borderRadius: BorderRadius.only(
                            topLeft: Radius.circular(20.0),
                            topRight: Radius.circular(20.0),
                          ),
                        ),
                        child: Padding(
                          padding: const EdgeInsets.only(
                              left: 16.0, right: 16.0, top: 25,
bottom: 16.0),
                          child: Column(
                            mainAxisAlignment:
MainAxisAlignment.center,
                            crossAxisAlignment:
CrossAxisAlignment.stretch,
                            children: <Widget>[
                              Padding(
                                padding: EdgeInsets.all(10.0),
                                child: Container(
                                  child: Text(
                                    "On a scale of 1 to 5, rate the 
overall performance of the teacher?",
                                    style: KFeedbackScaleHint,
                                    textAlign: TextAlign.center,
                                  ),
                                ),
                              ),
                              Padding(
                                padding: EdgeInsets.only(
                                    left: 14.0,
                                    right: 14.0,
                                    top: 14,
                                    bottom: 5.0),
                                child: Container(
                                    child: Text(
                                  myFeedbackText,
                                  style: KFeedbackTextStyle,
                                  textAlign: TextAlign.center,
                                )),
                              ),
                              Flexible(
                                child: Padding(
                                  padding: EdgeInsets.all(10.0),
                                  child: Container(
                                      child: Icon(
                                    myFeedback,
                                    color: myFeedbackColor,
                                    size: 120.0,
                                  )),
                                ),
                              ),
                              Padding(
                                padding: EdgeInsets.all(8.0),
                                child: Container(
                                  child: Slider(
                                    min: 0.0,
                                    max: 5.0,
                                    divisions: 5,
                                    value: sliderValue,
                                    activeColor: Color(0xFF005C97),
                                    inactiveColor:
Color(0xFF70e1f5),
                                    onChanged: (newValue) {
                                      setState(() {
                                        sliderValue = newValue;
                                        if (sliderValue <= 0.4) {
                                          myFeedback =
FontAwesomeIcons.neos;
                                          myFeedbackColor =
Color(0xeeb6fbff);
                                          myFeedbackText =
"REMARK";
                                        }
                                        if (sliderValue >= 0.5 &&
                                            sliderValue <= 1.0) {
                                          myFeedback =
FontAwesomeIcons.sadTear;
                                          myFeedbackColor =
Color(0xFFED213A);
                                          myFeedbackText = "COULD
BE BETTER";
                                        }
                                        if (sliderValue >= 1.1 &&
                                            sliderValue <= 2.0) {
                                          myFeedback =
FontAwesomeIcons.frown;
                                          myFeedbackColor =
Color(0xFFF27121);
                                          myFeedbackText = "BELOW
AVERAGE";
                                        }
                                        if (sliderValue >= 2.1 &&
                                            sliderValue <= 3.0) {
                                          myFeedback =
FontAwesomeIcons.meh;
                                          myFeedbackColor =
Colors.amber;
                                          myFeedbackText =
"NORMAL";
                                        }
                                        if (sliderValue >= 3.1 &&
                                            sliderValue <= 4.0) {
                                          myFeedback =
FontAwesomeIcons.smile;
                                          myFeedbackColor =
Colors.green;
                                          myFeedbackText = "GOOD";
                                        }
                                        if (sliderValue >= 4.1 &&
                                            sliderValue <= 5.0) {
                                          myFeedback =
FontAwesomeIcons.laugh;
                                          myFeedbackColor =
Color(0xFF00B4DB);
                                          myFeedbackText =
"EXCELLENT";
                                        }
                                      });
                                    },
                                  ),
                                ),
                              ),
                              Padding(
                                padding: EdgeInsets.all(14.0),
                                child: Container(
                                    child: Text(
                                  "Your Rating: $sliderValue",
                                  style: KFeedbackTextStyle,
                                  textAlign: TextAlign.center,
                                )),
                              ),
                              Padding(
                                padding: EdgeInsets.all(5.0),
                                child: Container(
                                    height: 45.0,
                                    width: 80.0,
                                    child: RaisedButton(
                                      child: Text(
                                        'Submit',
                                        style:
KFeedbackSubmitTextStyle,

                                        textAlign:
TextAlign.center,

                                      ),
                                      shape:
RoundedRectangleBorder(

                                        borderRadius:

BorderRadius.circular(10.0),

                                      ),
                                      color: Color(0xFF005C97),
                                      onPressed: () {
                                        setState(() {
                                          addFeedbackToDatabase(
                                              sliderValue,
chatRoomId);

                                        });
                                        showDialog(
                                            context: context,
                                            builder: (BuildContext
context) {

                                              return
RichAlertDialog(

                                                //uses the custom
alert dialog

                                                alertTitle:

richTitle("Success!"),

                                                alertSubtitle:
richSubtitle(

                                                    "Thanks for
your valuable feedback."),

                                                alertType:

RichAlertType.SUCCESS,

                                              );
                                            });
                                      },
                                    )),
                              ),
                            ],
                          ),
                        )),
                  ),
                ),
              ],
            ),
          ),
        ),
      ),
    );
  }
}
```



On a scale of 1 to 5, rate the overall performance of the teacher?

GOOD

Your Rating: 4.0

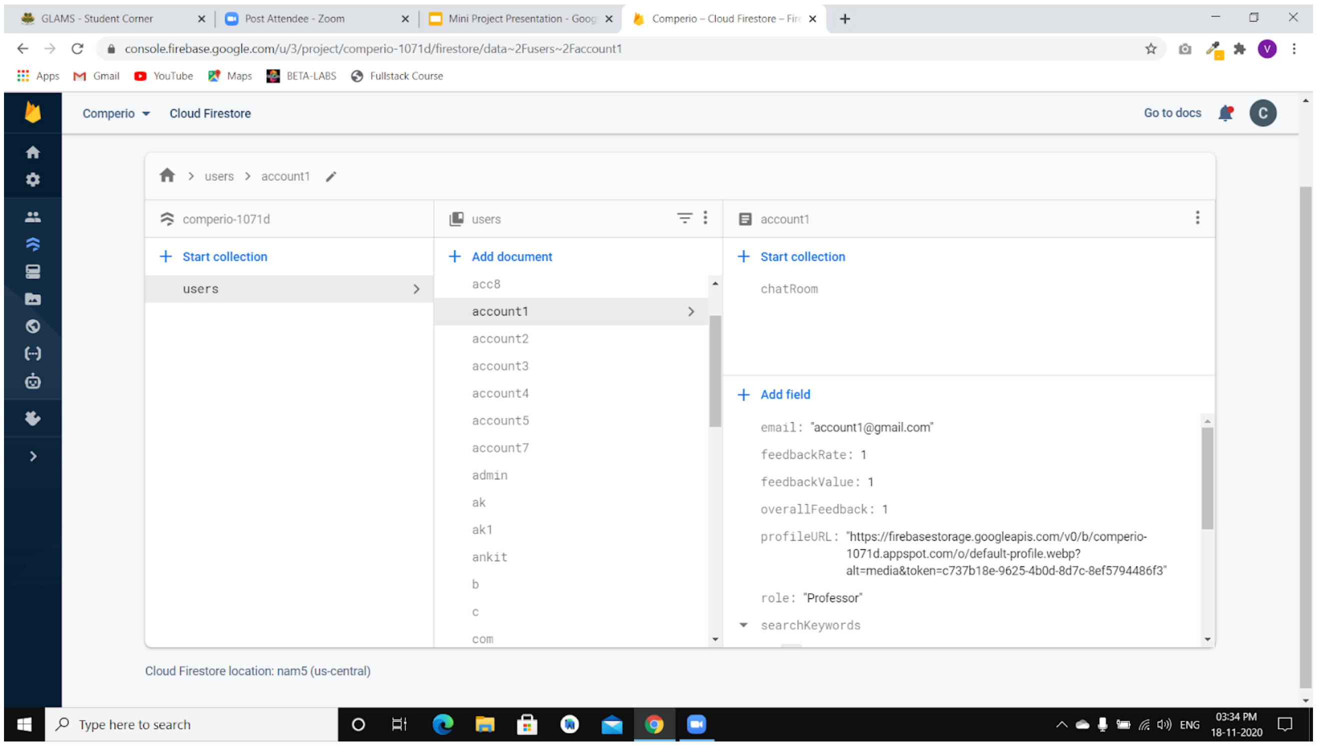


On a scale of 1 to 5, rate the overall performance of the teacher?

COULD BE BETTER

Your Rating: 1.0

User Collection with their Details:-



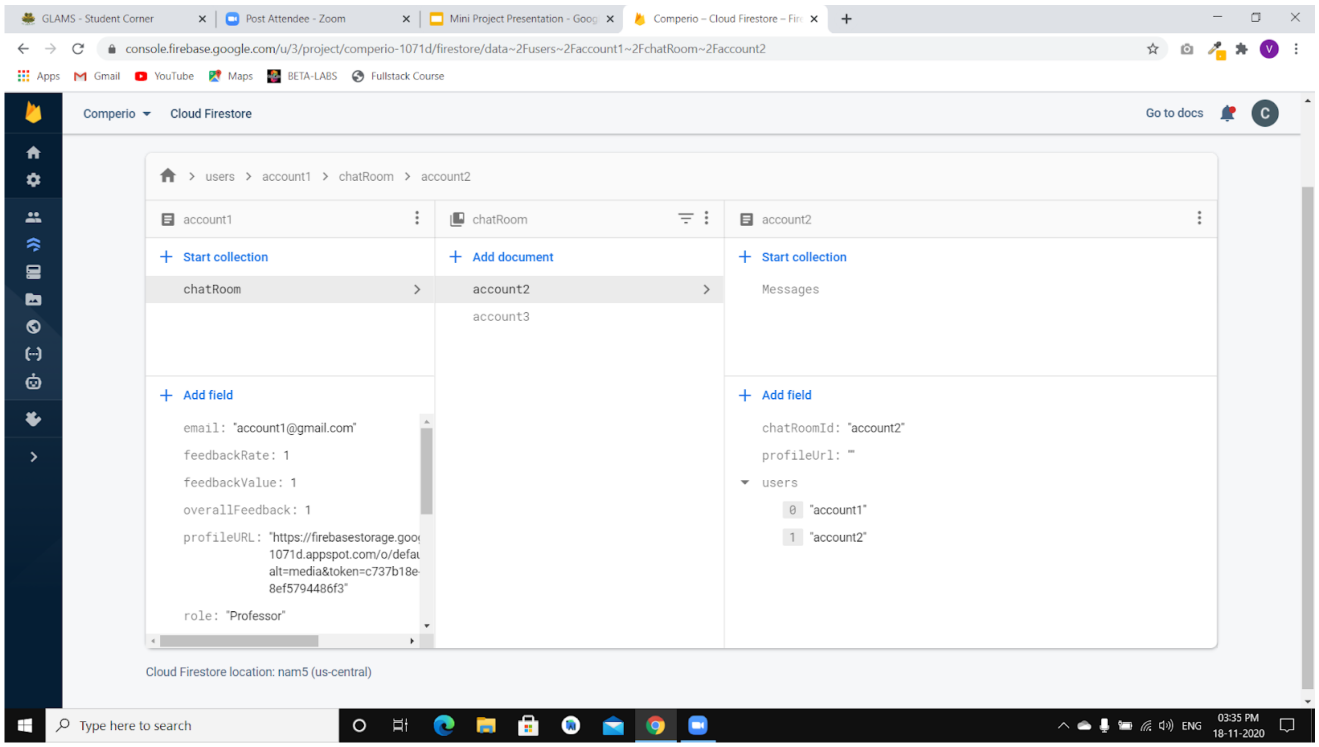
console.firebase.google.com/u/3/project/comperio-1071d/firestore/data~2Fusers~2Faccount1

users > account1

comperio-1071d	users	account1
users	acc8	chatRoom
	account1	
	account2	
	account3	
	account4	
	account5	
	account7	
	admin	
	ak	
	ak1	
	ankit	
	b	
	c	
	com	

```
email: "account1@gmail.com"
feedbackRate: 1
feedbackValue: 1
overallFeedback: 1
profileURL: "https://firebasestorage.googleapis.com/v0/b/comperio-
            1071d.appspot.com/o/default-profile.webp?
            alt=media&token=c737b18e-9625-4b0d-8d7c-8ef5794486f3"
role: "Professor"
searchKeywords
```

Chatroom Collection within user Collection:-



console.firebase.google.com/u/3/project/comperio-1071d/firestore/data~2Fusers~2Faccount1~2FchatRoom~2Faccount2

users > account1 > chatRoom > account2

account1	chatRoom	account2
chatRoom	account2	Messages
	account3	

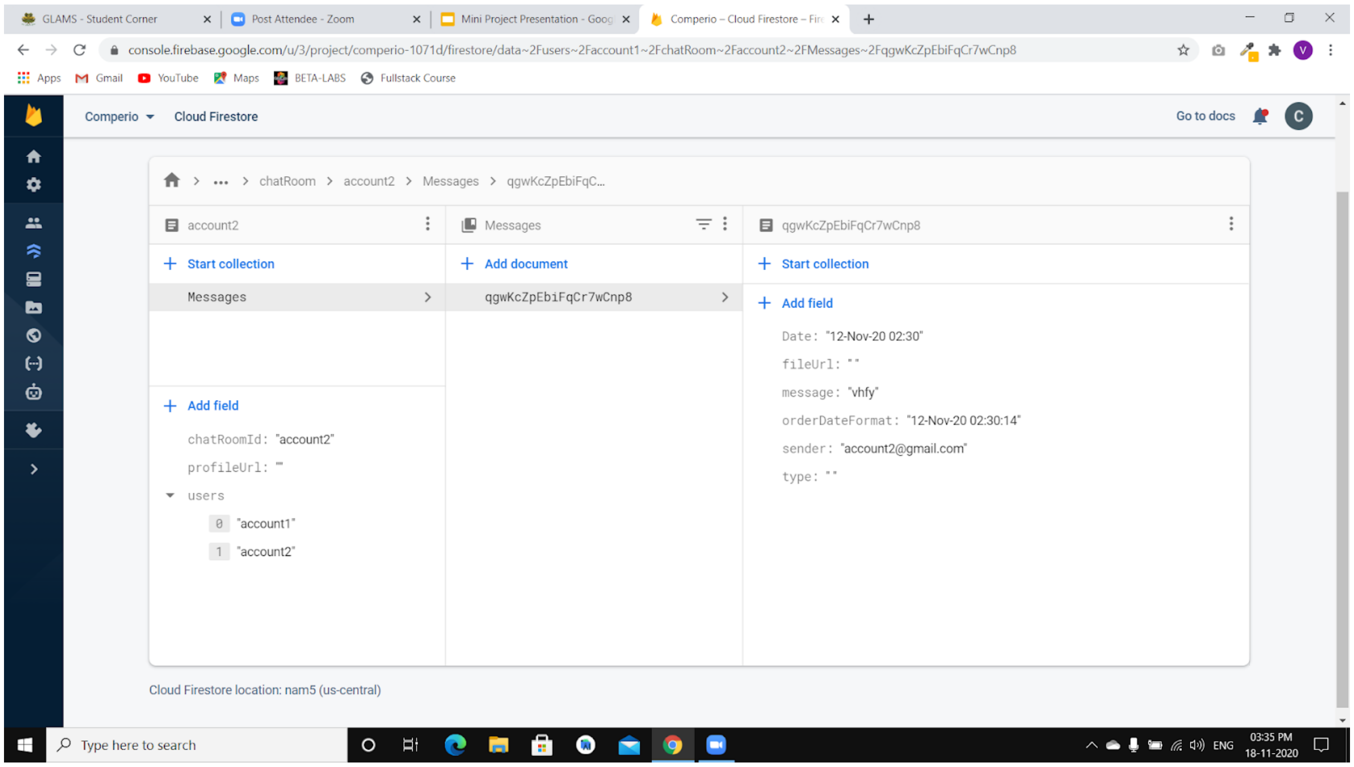
account1:

```
email: "account1@gmail.com"
feedbackRate: 1
feedbackValue: 1
overallFeedback: 1
profileURL: "https://firebasestorage.goo
            1071d.appspot.com/o/defau
            alt=media&token=c737b18e-
            8ef5794486f3"
role: "Professor"
```

account2:

```
chatRoomId: "account2"
profileUrl: ""
users
    0 "account1"
    1 "account2"
```

Messages Collection within chatroom Collection:-



console.firebase.google.com/u/3/project/comperio-1071d/firestore/data~2Fusers~2Faccount1~2FchatRoom~2Faccount2~2FMessages~2FqgwKcZpEbiFqCr7wCnp8

Comperio · Cloud Firestore

chatRoom > account2 > Messages > qgwKcZpEbiFqC...

account2: Messages

Messages: qgwKcZpEbiFqCr7wCnp8

qgwKcZpEbiFqCr7wCnp8

```
Date: "12-Nov-20 02:30"
fileUrl: ""
message: "vhfy"
orderDateFormat: "12-Nov-20 02:30:14"
sender: "account2@gmail.com"
type: ""
```

```
chatRoomId: "account2"
profileUrl: ""
users
  0 "account1"
  1 "account2"
```

Cloud Firestore location: nam5 (us-central)

Technical Support:

For technical support Contact with following correspondent of our technical team:

Comperio181500@gmail.com

Conclusion:

From above discussion, it is evident how chat applications are purposed to play various roles in addition to acting as a chat communication medium.

The main objective of the project is to develop a Secure Chat Application. We had taken a wide range of literature review in order to achieve all the tasks, where we came to know about some of the products that are existing in the market. We made detailed research on that path to cover the loopholes that existing systems are facing and to eradicate them in our application. In the process of research, we came to know about the latest technologies and different algorithms.

References:

1. Reference for flutter documentation https://flutter.dev/.
2. About Android Studio https://developer.android.com/studio.
3. About Dart Language https://dart.dev/.
4. Some of the contents are from Internet.
5. https://medium.com/
6. https://s3.amazonaws.com/ppt-download/finaldoconchatapp-170418182505.pdf?response-content-disposition=attachment&Signature=s1prvZf9uTjI%2F0VbbGpdrPiBHFA%3D&Expires=1606226633&AWSAccessKeyId=AKIAIA5TS2BVP74IAVEQ