Chapter-1

INTRODUCTION

1.1 Project Summary

The previous work of this already exists. The similar application can be found on the Android market. This project will focus on providing chatting ability with user friendly user interface developed using Flutter. Chat App is a live chat platform that allows users to collaborate and connect with each other.

The app doesn't have functionality yet to send images and videos which can be implemented in future. Once the user logs in he/she can search for other users using Email id and enter the chat room for chatting. The project also makes use of Firebase authentication and firestore for database at the backend.

1.2 Project Purpose

Communication is a means for people to exchange messages. It has started since the beginning of human creation. Distant communication began as early as 1800 century with the introduction of televisions, telegraphs and then telephony. Interestingly enough, telephone communication stands out as the fastest growing technology, from fixed lines to mobile wireless, from voice call to data transfer. The emergence of computer network and telecommunication technologies bears the same objective that is to allow people to communicate. All this while, much efforts has been drawn towards consolidating the device into one and therefore indiscriminate the service. Chatting is a method of using technology to bring people and ideas together despite of the geographical barriers. Chat app is a social-networking tool that leverages on technology advancement thereby allowing its users communicate. It offers a wonderful one stop shop experience for keeping in touch with people you know.

1.3 Project Scope

The purpose of the chat application is to allow users be able to the chat with each other, like a normal chat application. The users will able to communicate with each other through texts, group chatting option, sending images and videos option aren't developed yet but can be developed in the future. The application is written using the programming language called dart which is something new and help us to gain new skills on Flutter based applications, which is better than java/kotlin based applications. The application will be run and tested at each phase as it is developed and will be tested completely towards the end of the project to ensure there are no bugs that make the project to operate slowly.

1.4 Technology and Literature Review

Flutter – a simple and high performance framework based on Dart language, provides high performance by rendering the UI directly in the operating system's canvas rather than through native framework. Flutter also offers many ready to use widgets (UI) to create a modern application. These widgets are optimized for mobile environment and designing the application using widgets is as simple as designing HTML. To be specific, Flutter application is itself a widget. Flutter widgets also supports animations and gestures. The application logic is based on reactive programming. Widget may optionally have a state. By changing the state of the widget, Flutter will automatically (reactive programming) compare the widget's state (old and new) and render the widget with only the necessary changes instead of re-rendering the whole widget.

Flutter framework offers the following features to developers −

- Modern and reactive framework.
- Uses Dart programming language and it is very easy to learn.
- Beautiful and fluid user interfaces.
- Runs same UI for multiple platforms
- High performance applications

1.5 Problem Statement

This project is used to create a chat application which involves a server maintained by Firebase and enables the user to communicate/chat with each other through texts. To develop an application to provide instant messaging and provide a novice friendly user interface.

1.6 Objective of the project

1. To implement an application that provides users to chat with each other.
2. To provide authentication for the users before entering the chat.
3. To design a user interface which is novice friendly.

Chapter-2

SYSTEM REQUIREMENTS

2.1 User Characteristics

- Start the application.
- After the application is started the first screen contains login user interface where the user has to enter the email id and password.
- The user has to click on the login button if he has already created an account else the user has to create an account by clicking on “Create Account”.
- In account creation screen the user has to enter his/her name ,email id and password and click on create account button.
- After registering the account the user will be redirected to login page where the user has to enter his/her credentials and login.
- After logging in the home screen opens up with a search bar at the top where the user has to enter the other user’s email id and find the account.
- If the other user has a account created his/her profile will be displayed
- The user can click on the other user’s profile and enter the chat screen where he can enter the text in the text box and send the message.
- The user can then log out of the application at the end.

2.2 Hardware and Software Requirements

Hardware requirements:

1. Android phone
2. A minimum of 2gb ram, 8gb recommended.
3. A minimum of 64gb of storage, 128gb recommended.
4. Internet connectivity required.

Software requirements:

1. Android 4.1 (API level 16) or higher.
2. iOS 10 (or above).

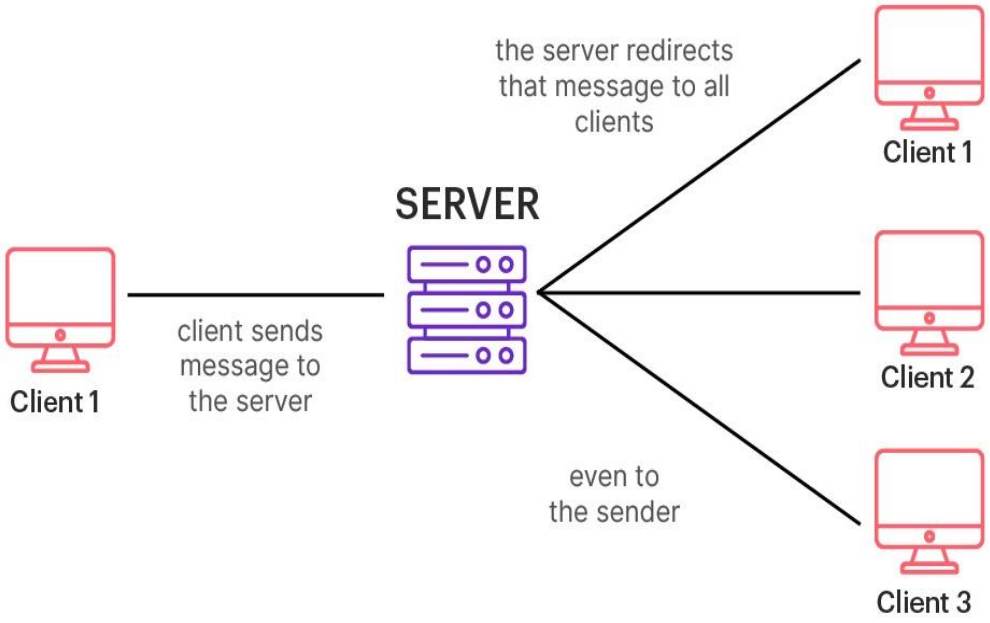
2.3 Project Constraints

1. The mobile phone must be connected with internet.
2. The application doesn’t have group chatting, audio and video sharing, calling options.

Chapter – 3

DESIGN

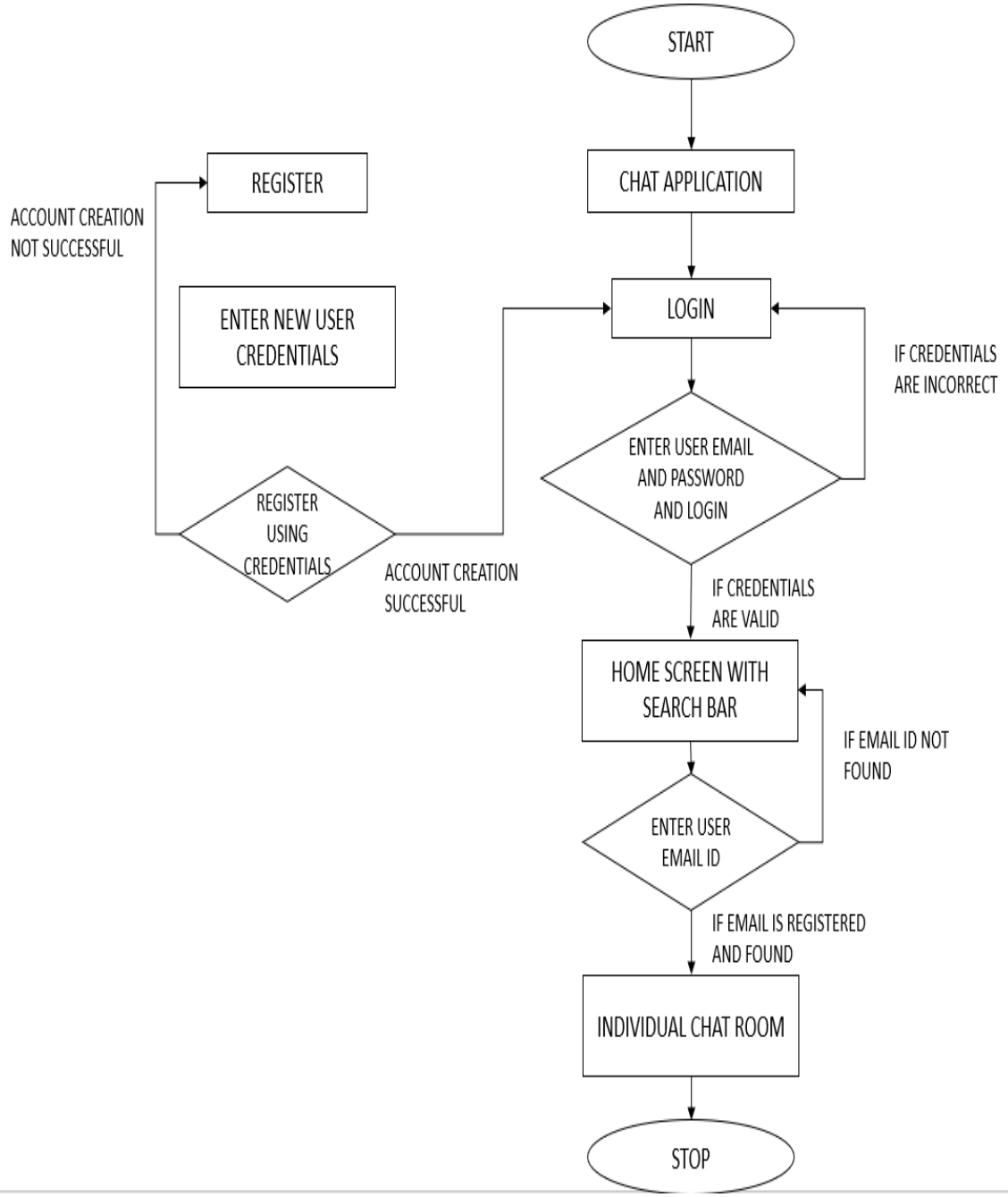
3.1 System architecture and design:



The user first creates an account on the Firebase server through the application. Then the user searches for the other user's email id. After the other user's email id is found and sends the message, the message will be sent to the server.

The server stores all these messages in a collection and redirects the messages to the other user's account.

3.2 Control Flow Graph:



3.3 Module Description:

Login:

The users have to login with proper credentials to enter the home screen of the application.

Register:

The user if he/she is using the application for the first time has to register with the Firebase server.

Main menu:

When the user logs in he will be directed to the main menu screen of the app where they can type/search an email to whom they want to chat. If the email is valid then the account with the desired email id is displayed.

Chat room:

If a user wishes to chat he can click on the email id displayed then he will be re directed to the chat room screen here he will be able to chat with the person. The user can click on the text area and type the desired text they want to send. Once the message is typed the user can click on the send button to send the message.

Logout:

After the chatting is completed the user logs out.

Chapter – 4

IMPLEMENTATION

4.1 Built in Functions

Flutter widgets are built using a modern framework that takes inspiration from React. The central idea is that you build your UI out of widgets. Widgets describe what their view should look like given their current configuration and state. When a widget's state changes, the widget rebuilds its description, which the framework diffs against the previous description in order to determine the minimal changes needed in the underlying render tree to transition from one state to the next.

1. runApp()

void runApp(Widget app)

The runApp() function takes the given Widget and makes it the root of the Widget tree. The framework forces the root widget to cover the entire screen.

2. Container()

The Container widget lets you create a rectangular visual element. A container can be decorated with a BoxDecoration, such as a background, a border, or a shadow. A Container can also have margins, padding, and constraints applied to its size. In addition, a Container can be transformed in three dimensional space using a matrix.

3. MaterialApp()

Flutter provides a number of widgets that help you build apps that follow Material Design. A Material app starts with the MaterialApp widget, which builds a number of useful widgets at the root of your app, including a Navigator, which manages a stack of widgets identified by strings, also known as "routes".

4. **Text()**

   The Text widget lets you create a run of styled text within your application.

5. **Stateful ()**

   A stateful widget is dynamic: for example, it can change its appearance in response to events triggered by user interactions or when it receives data. Checkbox, Radio, Slider, InkWell, Form, and TextField are examples of stateful widgets. Stateful widgets subclass StatefulWidget.

6. **Stateless()**

   A stateless widget never changes. Icon, IconButton, and Text are examples of stateless widgets. Stateless widgets subclass StatelessWidget.

7. **Scaffold()**

   Scaffold is a class in flutter which provides many widgets or we can say APIs like Drawer, SnackBar, BottomNavigationBar, FloatingActionButton, AppBar, etc. Scaffold will expand or occupy the whole device screen. It will occupy the available space. Scaffold will provide a framework to implement the basic material design layout of the application.

8. **GestureDetector()**

   A widget that detects gestures. Attempts to recognize gestures that correspond to its non-null callbacks.

9. **AppBar()**

   A stateless widget never changes. Icon, IconButton, and Text are examples of stateless widgets. Stateless widgets subclass StatelessWidget.

10. **Icon()**

    A graphical icon widget drawn with a glyph from a font described in an IconData such as material's predefined IconDatas in Icons.

Dart code:

1. main.dart:

```
import 'package:chats/Authenticate.dart';
import 'package:firebase_core/firebase_core.dart';
import 'package:flutter/material.dart';

Future main() async {
  WidgetsFlutterBinding.ensureInitialized();
  await Firebase.initializeApp();
  runApp(MyApp());
}

class MyApp extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      debugShowCheckedModeBanner: false,
      home: Authenticate(),
    );
  }
}
```

2. LoginScreen.dart:

```
import 'package:chats/CreateAccount.dart';
import 'package:chats/HomeScreen.dart';
import 'package:chats/Methods.dart';
import 'package:flutter/material.dart';

import 'HomeScreen.dart';

class LoginScreen extends StatefulWidget {
  @override
  _LoginScreenState createState() => _LoginScreenState();
}

class _LoginScreenState extends State<LoginScreen> {
  final TextEditingController _email = TextEditingController();
  final TextEditingController _password = TextEditingController();
  bool isLoading = false;

  @override
  Widget build(BuildContext context) {
```

```
final size = MediaQuery.of(context).size;

return Scaffold(
  body: isLoading
      ? Center(
          child: Container(
            height: size.height / 20,
            width: size.height / 20,
            child: CircularProgressIndicator(),
          ),
        )
      : SingleChildScrollView(
          child: Column(
            children: [
              SizedBox(
                height: size.height / 20,
              ),
              Container(
                alignment: Alignment.centerLeft,
                width: size.width / 0.5,
                child: IconButton(
                    icon: Icon(Icons.arrow_back_ios), onPressed: () {}),
              ),
              SizedBox(
                height: size.height / 50,
              ),
              Container(
                width: size.width / 1.1,
                child: Text(
                  "Welcome",
                  style: TextStyle(
                    fontSize: 34,
                    fontWeight: FontWeight.bold,
                  ),
                ),
              ),
              Container(
                width: size.width / 1.1,
                child: Text(
                  "Sign In to Contiue!",
                  style: TextStyle(
                    color: Colors.grey[700],
                    fontSize: 25,
                    fontWeight: FontWeight.w500,
```

```
            ),
          ),
        ),
        SizedBox(
          height: size.height / 10,
        ),
        Container(
          width: size.width,
          alignment: Alignment.center,
          child: field(size, "email", Icons.account_box, _email),
        ),
        Padding(
          padding: const EdgeInsets.symmetric(vertical: 18.0),
          child: Container(
            width: size.width,
            alignment: Alignment.center,
            child: field(size, "password", Icons.lock, _password),
          ),
        ),
        SizedBox(
          height: size.height / 10,
        ),
        customButton(size),
        SizedBox(
          height: size.height / 40,
        ),
        GestureDetector(
          onTap: () => Navigator.of(context).push(
              MaterialPageRoute(builder: (_) => CreateAccount())),
          child: Text(
            "Create Account",
            style: TextStyle(
              color: Colors.blue,
              fontSize: 16,
              fontWeight: FontWeight.w500,
            ),
          ),
        )
      ],
    ),
  ),
 );
}
```

```
Widget customButton(Size size) {
  return GestureDetector(
    onTap: () {
      if (_email.text.isNotEmpty && _password.text.isNotEmpty) {
        setState(() {
          isLoading = true;
        });

        logIn(_email.text, _password.text).then((user) {
          if (user != null) {
            print("Login Sucessfull");
            setState(() {
              isLoading = false;
            });
            Navigator.push(
                context, MaterialPageRoute(builder: (_) => HomeScreen()));
          } else {
            print("Login Failed");
            setState(() {
              isLoading = false;
            });
          }
        });
      } else {
        print("Please fill form correctly");
      }
    },
    child: Container(
        height: size.height / 14,
        width: size.width / 1.2,
        decoration: BoxDecoration(
          borderRadius: BorderRadius.circular(5),
          color: Colors.blue,
        ),
        alignment: Alignment.center,
        child: Text(
          "Login",
          style: TextStyle(
            color: Colors.white,
            fontSize: 18,
            fontWeight: FontWeight.bold,
          ),
        )),
  );
```

```
  }

  Widget field(
      Size size, String hintText, IconData icon, TextEditingController cont) {
    return Container(
      height: size.height / 14,
      width: size.width / 1.1,
      child: TextField(
        controller: cont,
        decoration: InputDecoration(
          prefixIcon: Icon(icon),
          hintText: hintText,
          hintStyle: TextStyle(color: Colors.grey),
          border: OutlineInputBorder(
            borderRadius: BorderRadius.circular(10),
          ),
        ),
      ),
    );
  }
}
```

3. CreateAccount.dart:

```
import 'package:chats/Methods.dart';
import 'package:flutter/material.dart';

import 'package:chats/HomeScreen.dart';

class CreateAccount extends StatefulWidget {
  @override
  _CreateAccountState createState() => _CreateAccountState();
}

class _CreateAccountState extends State<CreateAccount> {
  final TextEditingController _name = TextEditingController();
  final TextEditingController _email = TextEditingController();
  final TextEditingController _password = TextEditingController();
  bool isLoading = false;

  @override
  Widget build(BuildContext context) {
```

```
final size = MediaQuery.of(context).size;

return Scaffold(
  body: isLoading
      ? Center(
          child: Container(
            height: size.height / 20,
            width: size.height / 20,
            child: CircularProgressIndicator(),
          ),
        )
      : SingleChildScrollView(
          child: Column(
            children: [
              SizedBox(
                height: size.height / 20,
              ),
              Container(
                alignment: Alignment.centerLeft,
                width: size.width / 0.5,
                child: IconButton(
                    icon: Icon(Icons.arrow_back_ios), onPressed: () {}),
              ),
              SizedBox(
                height: size.height / 50,
              ),
              Container(
                width: size.width / 1.1,
                child: Text(
                  "Welcome",
                  style: TextStyle(
                    fontSize: 34,
                    fontWeight: FontWeight.bold,
                  ),
                ),
              ),
              Container(
                width: size.width / 1.1,
                child: Text(
                  "Create Account to Contiue!",
                  style: TextStyle(
                    color: Colors.grey[700],
                    fontSize: 20,
                    fontWeight: FontWeight.w500,
```

```
    ),
  ),
),
SizedBox(
  height: size.height / 20,
),
Padding(
  padding: const EdgeInsets.symmetric(vertical: 18.0),
  child: Container(
    width: size.width,
    alignment: Alignment.center,
    child: field(size, "Name", Icons.account_box, _name),
  ),
),
Container(
  width: size.width,
  alignment: Alignment.center,
  child: field(size, "email", Icons.account_box, _email),
),
Padding(
  padding: const EdgeInsets.symmetric(vertical: 18.0),
  child: Container(
    width: size.width,
    alignment: Alignment.center,
    child: field(size, "password", Icons.lock, _password),
  ),
),
SizedBox(
  height: size.height / 20,
),
customButton(size),
Padding(
  padding: const EdgeInsets.all(8.0),
  child: GestureDetector(
    onTap: () => Navigator.pop(context),
    child: Text(
      "Login",
      style: TextStyle(
        color: Colors.blue,
        fontSize: 16,
        fontWeight: FontWeight.w500,
      ),
    ),
  ),
```

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

Widget customButton(Size size) {
  return GestureDetector(
    onTap: () {
      if (_name.text.isNotEmpty &&
          _email.text.isNotEmpty &&
          _password.text.isNotEmpty) {
        setState(() {
          isLoading = true;
        });

        createAccount(_name.text, _email.text, _password.text).then((user) {
          if (user != null) {
            setState(() {
              isLoading = false;
            });
            Navigator.push(
                context, MaterialPageRoute(builder: (_) => HomeScreen()));
            print("Account Created Sucessfully");
          } else {
            print("Login Failed");
            setState(() {
              isLoading = false;
            });
          }
        });
      } else {
        print("Please enter Fields");
      }
    },
    child: Container(
        height: size.height / 14,
        width: size.width / 1.2,
        decoration: BoxDecoration(
          borderRadius: BorderRadius.circular(5),
          color: Colors.blue,
        ),
        alignment: Alignment.center,
```

```
        child: Text(
          "Create Account",
          style: TextStyle(
            color: Colors.white,
            fontSize: 18,
            fontWeight: FontWeight.bold,
          ),
        )),
    );
  }


 Widget field(
     Size size, String hintText, IconData icon, TextEditingController cont) {
   return Container(
     height: size.height / 14,
     width: size.width / 1.1,
     child: TextField(
       controller: cont,
       decoration: InputDecoration(
         prefixIcon: Icon(icon),
         hintText: hintText,
         hintStyle: TextStyle(color: Colors.grey),
         border: OutlineInputBorder(
           borderRadius: BorderRadius.circular(10),
         ),
       ),
     ),
   );
  }
}
```

4. Authenticate.dart

```
import 'package:chats/HomeScreen.dart';
import 'package:chats/LoginScreen.dart';
import 'package:firebase_auth/firebase_auth.dart';
import 'package:flutter/material.dart';

class Authenticate extends StatelessWidget {
  final FirebaseAuth _auth = FirebaseAuth.instance;
```

```
  @override
  Widget build(BuildContext context) {
    if (_auth.currentUser != null) {
      return HomeScreen();
    } else {
      return LoginScreen();
    }
  }
}
```

5. HomeScreen.dart

```
import 'package:chats/Methods.dart';
import 'package:cloud_firestore/cloud_firestore.dart';
import 'package:firebase_auth/firebase_auth.dart';
import 'package:flutter/material.dart';
import 'ChatRoom.dart';

class HomeScreen extends StatefulWidget {
  @override
  State<HomeScreen> createState() => _HomeScreenState();
}

class _HomeScreenState extends State<HomeScreen> with
WidgetsBindingObserver {
  Map<String, dynamic>? userMap;
  bool isLoading = false;
  final TextEditingController _search = TextEditingController();
  final FirebaseAuth _auth = FirebaseAuth.instance;
  final FirebaseFirestore _firestore = FirebaseFirestore.instance;

  @override
  void initState() {
    super.initState();
    WidgetsBinding.instance.addObserver(this);
    setStatus("Online");
  }

  void setStatus(String status) async {
    await _firestore.collection('users').doc(_auth.currentUser!.uid).update({
```

```
    "status": status,
  });
}

@override
void didChangeAppLifecycleState(AppLifecycleState state) {
  if (state == AppLifecycleState.resumed) {
    // online
    setStatus("Online");
  } else {
    // offline
    setStatus("Offline");
  }
}

String chatRoomId(String user1, String user2) {
  if (user1[0].toLowerCase().codeUnits[0] >
      user2.toLowerCase().codeUnits[0]) {
    return "$user1$user2";
  } else {
    return "$user2$user1";
  }
}

void onSearch() async {
  FirebaseFirestore _firestore = FirebaseFirestore.instance;

  setState(() {
    isLoading = true;
  });

  await _firestore
      .collection('users')
      .where("email", isEqualTo: _search.text)
      .get()
      .then((value) {
    setState(() {
      userMap = value.docs[0].data();
      isLoading = false;
    });
    print(userMap);
  });
}
```

```
@override
Widget build(BuildContext context) {
 final size = MediaQuery.of(context).size;

 return Scaffold(
  appBar: AppBar(
   title: Text("Home Screen"),
   actions: [
    IconButton(icon: Icon(Icons.logout), onPressed: () => logOut(context))
   ],
  ),
  body: isLoading
     ? Center(
        child: Container(
         height: size.height / 20,
         width: size.height / 20,
         child: CircularProgressIndicator(),
        ),
       )
     : Column(
        children: [
         SizedBox(
          height: size.height / 20,
         ),
         Container(
          height: size.height / 14,
          width: size.width,
          alignment: Alignment.center,
          child: Container(
           height: size.height / 14,
           width: size.width / 1.15,
           child: TextField(
            controller: _search,
            decoration: InputDecoration(
             hintText: "Search",
             border: OutlineInputBorder(
              borderRadius: BorderRadius.circular(10),
             ),
            ),
           ),
          ),
         ),
         SizedBox(
          height: size.height / 50,
```

```
            ),
            ElevatedButton(
              onPressed: onSearch,
              child: Text("Search"),
            ),
            SizedBox(
              height: size.height / 30,
            ),
            userMap != null
                ? ListTile(
                    onTap: () {
                      String roomId = chatRoomId(
                          _auth.currentUser!.displayName!,
                          userMap!['name']);

                      Navigator.of(context).push(
                        MaterialPageRoute(
                          builder: (_) => ChatRoom(
                            chatRoomId: roomId,
                            userMap: userMap!,
                          ),
                        ),
                      );
                    },
                    leading: Icon(Icons.account_box, color: Colors.black),
                    title: Text(
                      userMap!['name'],
                      style: TextStyle(
                        color: Colors.black,
                        fontSize: 17,
                        fontWeight: FontWeight.w500,
                      ),
                    ),
                    subtitle: Text(userMap!['email']),
                    trailing: Icon(Icons.chat, color: Colors.black),
                  )
                : Container(),
          ],
        ),
      );
    }}
```

6. ChatRoom.dart

```
import 'dart:io';
import 'package:cloud_firestore/cloud_firestore.dart';
import 'package:firebase_auth/firebase_auth.dart';
import 'package:flutter/material.dart';

class ChatRoom extends StatelessWidget {
  final Map<String, dynamic> userMap;
  final String chatRoomId;

  ChatRoom({required this.chatRoomId, required this.userMap});

  final TextEditingController _message = TextEditingController();
  final FirebaseFirestore _firestore = FirebaseFirestore.instance;
  final FirebaseAuth _auth = FirebaseAuth.instance;

  void onSendMessage() async {
    if (_message.text.isNotEmpty) {
      Map<String, dynamic> messages = {
        "sendby": _auth.currentUser!.displayName,
        "message": _message.text,
        "type": "text",
        "time": FieldValue.serverTimestamp(),
      };

      _message.clear();
      await _firestore
          .collection('chatroom')
          .doc(chatRoomId)
          .collection('chats')
          .add(messages);
    } else {
      print("Enter Some Text");
    }
  }

  @override
  Widget build(BuildContext context) {
    final size = MediaQuery.of(context).size;

    return Scaffold(
      appBar: AppBar(
        title: StreamBuilder<DocumentSnapshot>(
          stream:
              _firestore.collection("users").doc(userMap['uid']).snapshots(),
          builder: (context, snapshot) {
            if (snapshot.data != null) {
              return Container(
                child: Column(
                  children: [
                    Text(userMap['name']),
```

```
              Text(
                snapshot.data!['status'],
                style: TextStyle(fontSize: 14),
              ),
            ],
          ),
        );
      } else {
        return Container();
      }
    },
  ),
),

//3rd video
body: SingleChildScrollView(
  child: Column(
    children: [
      Container(
        height: size.height / 1.25,
        width: size.width,
        child: StreamBuilder<QuerySnapshot>(
          stream: _firestore
              .collection('chatroom')
              .doc(chatRoomId)
              .collection('chats')
              .orderBy("time", descending: false)
              .snapshots(),
          builder: (BuildContext context,
              AsyncSnapshot<QuerySnapshot> snapshot) {
            if (snapshot.data != null) {
              return ListView.builder(
                itemCount: snapshot.data!.docs.length,
                itemBuilder: (context, index) {
                  Map<String, dynamic> map = snapshot.data!.docs[index]
                      .data() as Map<String, dynamic>;
                  return messages(size, map, context);
                },
              );
            } else {
              return Container();
            }
          },
        ),
      ),
      Container(
        height: size.height / 10,
        width: size.width,
        alignment: Alignment.center,
        child: Container(
```

```
            height: size.height / 12,
            width: size.width / 1.1,
            child: Row(
              mainAxisAlignment: MainAxisAlignment.center,
              children: [
                Container(
                  height: size.height / 17,
                  width: size.width / 1.3,
                  child: TextField(
                    controller: _message,
                    decoration: InputDecoration(
                        hintText: "Send Message",
                        border: OutlineInputBorder(
                          borderRadius: BorderRadius.circular(8),
                        )),
                  ),
                ),
                IconButton(
                    icon: Icon(Icons.send), onPressed: onSendMessage),
              ],
            ),
          ),
        ),
      ],
    ),
  ),
 );
}

Widget messages(Size size, Map<String, dynamic> map, BuildContext context) {
  return map['type'] == "text"
      ? Container(
          width: size.width,
          alignment: map['sendby'] == _auth.currentUser!.displayName
              ? Alignment.centerRight
              : Alignment.centerLeft,
          child: Container(
            padding: EdgeInsets.symmetric(vertical: 10, horizontal: 14),
            margin: EdgeInsets.symmetric(vertical: 5, horizontal: 8),
            decoration: BoxDecoration(
              borderRadius: BorderRadius.circular(15),
              color: Colors.blue,
            ),
            child: Text(
              map['message'],
              style: TextStyle(
                fontSize: 16,
                fontWeight: FontWeight.w500,
                color: Colors.white,
              ),
```

```
                ),
              ),
            )
          : Container(
              height: size.height / 2.5,
              width: size.width,
              padding: EdgeInsets.symmetric(vertical: 5, horizontal: 5),
              alignment: map['sendby'] == _auth.currentUser!.displayName
                  ? Alignment.centerRight
                  : Alignment.centerLeft,
              child: InkWell(
                onTap: () => Navigator.of(context).push(
                  MaterialPageRoute(
                    builder: (_) => ShowImage(
                      imageUrl: map['message'],
                    ),
                  ),
                ),
                child: Container(
                  height: size.height / 2.5,
                  width: size.width / 2,
                  decoration: BoxDecoration(border: Border.all()),
                  alignment: map['message'] != "" ? null : Alignment.center,
                  child: map['message'] != ""
                      ? Image.network(
                          map['message'],
                          fit: BoxFit.cover,
                        )
                      : CircularProgressIndicator(),
                ),
              ),
            );
    }
  }
}
```

7. Methods.dart

```
import 'package:chats/LoginScreen.dart';
import 'package:cloud_firestore/cloud_firestore.dart';
import 'package:firebase_auth/firebase_auth.dart';
import 'package:flutter/material.dart';

Future<User?> createAccount(String name, String email, String password) async {
  FirebaseAuth _auth = FirebaseAuth.instance;

  FirebaseFirestore _firestore = FirebaseFirestore.instance;

  try {
```

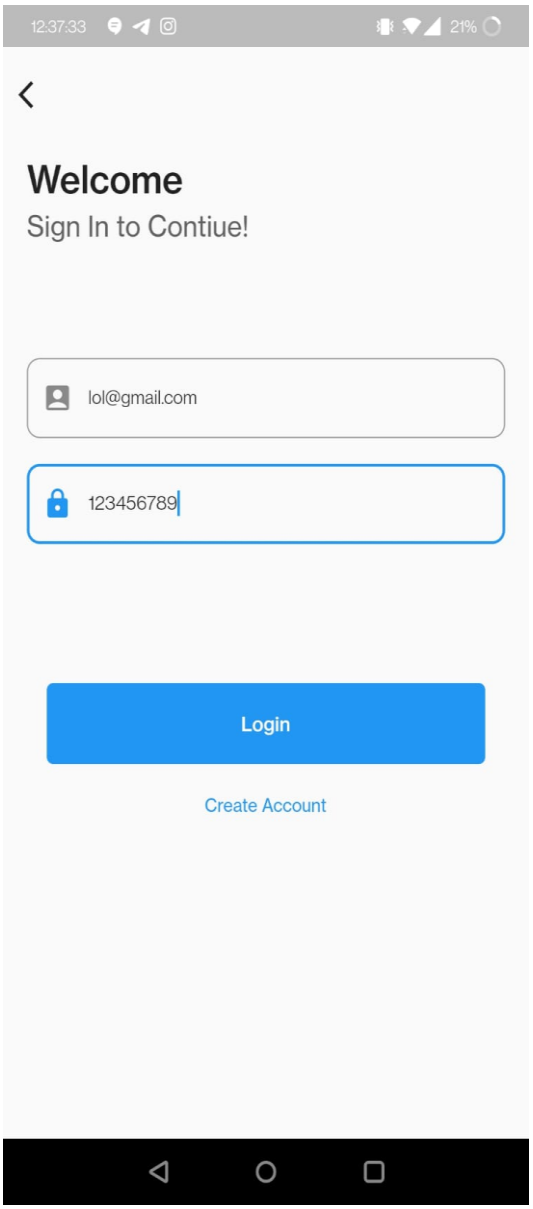
```
    UserCredential userCrendetial = await _auth.createUserWithEmailAndPassword(
        email: email, password: password);

    print("Account created Succesfull");

    userCrendetial.user!.updateDisplayName(name);

    await _firestore.collection('users').doc(_auth.currentUser?.uid).set({
      "name": name,
      "email": email,
      "status": "Unavalible",
      "uid": _auth.currentUser!.uid,
    });

    return userCrendetial.user;
  } catch (e) {
    print(e);
    return null;
  }
}

Future<User?> logIn(String email, String password) async {
  FirebaseAuth _auth = FirebaseAuth.instance;
  FirebaseFirestore _firestore = FirebaseFirestore.instance;

  try {
    UserCredential userCredential = await _auth.signInWithEmailAndPassword(
        email: email, password: password);

    print("Login Sucessfull");
    _firestore
        .collection('users')
        .doc(_auth.currentUser!.uid)
        .get()
        .then((value) => userCredential.user!.updateDisplayName(value['name']));

    return userCredential.user;
  } catch (e) {
    print(e);
    return null;
  }
}
```

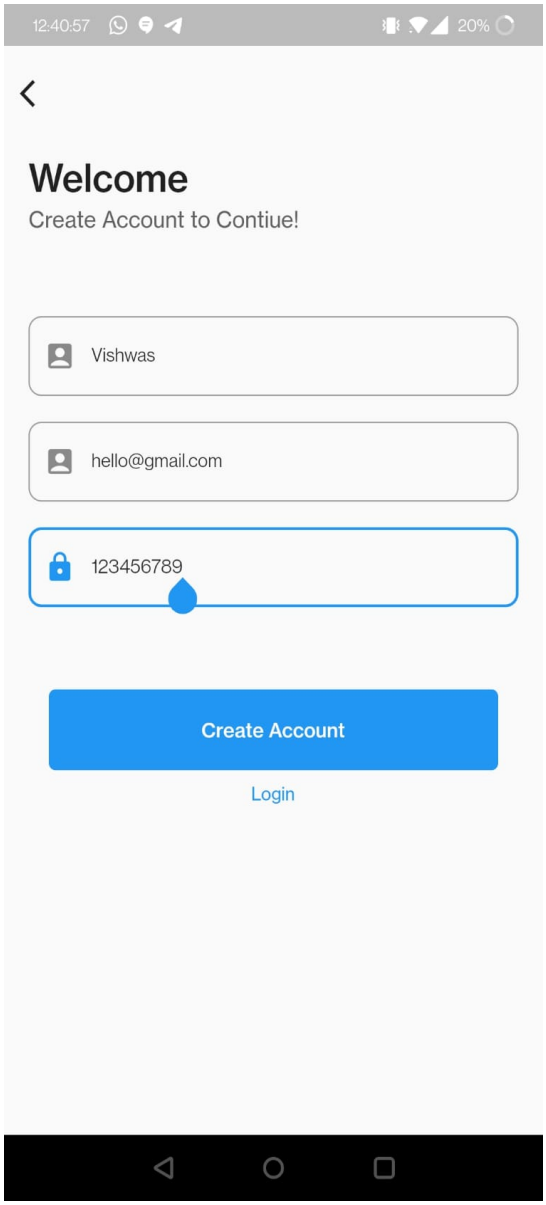
```
        Future logOut(BuildContext context) async {
          FirebaseAuth _auth = FirebaseAuth.instance;

          try {
            await _auth.signOut().then((value) {
              Navigator.push(context, MaterialPageRoute(builder: (_) => LoginScreen()));
            });
          } catch (e) {
            print("error");
          }
}
```

Chapter – 5

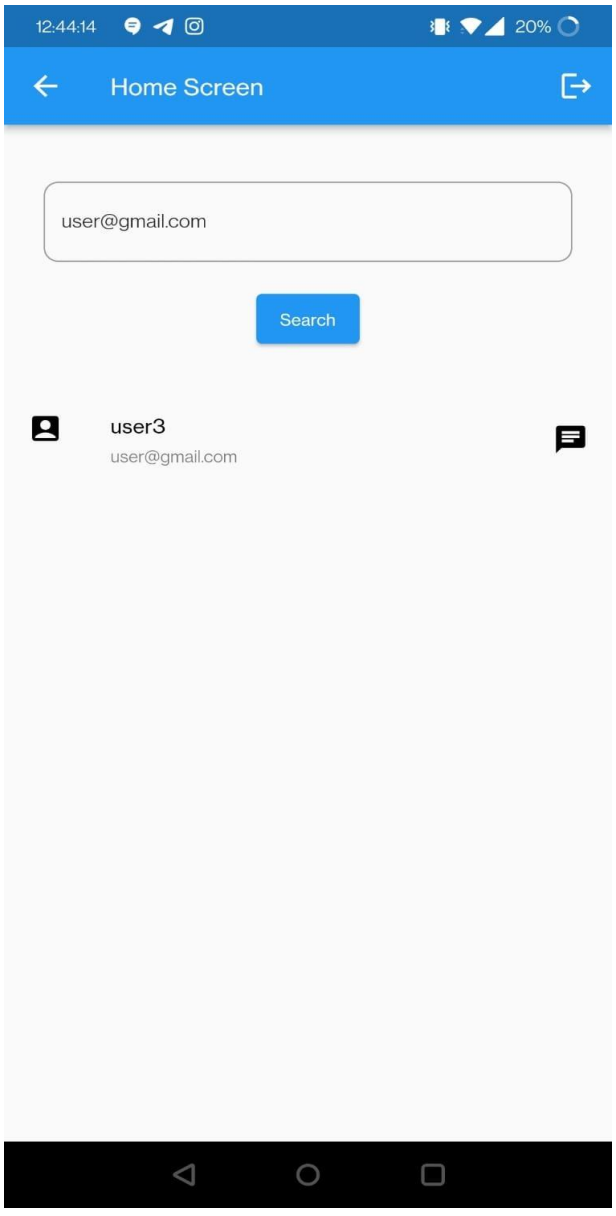
SNAPSHOTS

1. Login Screen and Register Screen:





2. Home Screen and logout:



3. ChatRoom screen:

Chapter-6

CONCLUSION

The project aims in developing a chat application with a simple yet user friendly user interface to help the users to chat with each other with ease. For this project we have used Flutter framework and its supporting language Dart to implement the user interface and functionalities and the project also makes use of the Firebase server for authentication and firestore database to store the user credentials and messages.

FUTURE ENHANCEMENTS

Any apps which we use currently will have improvements to be made, so does our application. In future improvements we will target to provide group chat functionality, image and video sharing. We can also improve the user interface to help the user so that he/she doesn’t face any problems while using the application.

BIBLIOGRAPHY

Websites:

1. https://docs.flutter.dev/development/ui/widgets
2. https://firebase.flutter.dev/docs/overview/
3. https://www.geeksforgeeks.org/flutter-tutorial/