A Project/Dissertation ETE Report

on

Flash Chat App with Real Time Chat

Submitted in partial fulfillment of the
requirement for the award of the degree of

B.Tech in CSE

Under The Supervision
of Name of Supervisor :
Dr. Shraddha Sagar

Submitted By

Pulkit Agrawal - 18021010114

Kaustub Ratan Pachoury - 18021011364

SCHOOL OF COMPUTING SCIENCE AND ENGINEERING
DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING GALGOTIAS UNIVERSITY, GREATER NOIDA
2022

SCHOOL OF COMPUTING SCIENCE AND ENGINEERING
GALGOTIAS UNIVERSITY, GREATER NOIDA

CANDIDATE'S DECLARATION

I/We hereby certify that the work which is being presented in the thesis/project/dissertation, entitled **"Flash Chat App with Real Time Chat"** in partial fulfillment of the requirements for the award of the -submitted in the School of Computing Science and Engineering of Galgotias University, Greater Noida, is an original work carried out during the period of month, Year to Month and Year, under the supervision of Shraddha Sagar Designation, Department of Computer Science and Engineering/Computer Application and Information and Science, of School of Computing Science and Engineering , Galgotias University, Greater Noida

The matter presented in the thesis/project/dissertation has not been submitted by me/us for the award of any other degree of this or any other places.

This is to certify that the above statement made by the candidates is correct to the best of my knowledge.

Shraddha Sagar

The Final Thesis/Project/ Dissertation Viva-Voce examination of Final Project.

Signature of Examiner(s) **Signature of Supervisor(s)**

Signature of Project Coordinator **Signature of Dean**

Date:

Place: Greater Noida

Abstract

Communication through the internet is becoming vital these days. Online communication allows the users to communicate with other people in a fast and convenient way. Considering this, the online communication application must be able share the texts or images or any other files in a faster way with minimum delay or with no delay. Firebase is one of the platforms which provides a real-time database and cloud services which allows the developer to make these applications with ease. Instant messaging can be considered as a platform to maintain communication. Android provides a better platform to develop various applications for instant messaging compared to other platforms such as iOS. The main objective of this project is to present a software application for the launching of a real time communication between operators/users. The system developed on android will enable the users to communicate with other users through text messages with the help of the internet. The system requires both devices to be connected via the internet.

This application is based on Android with the backend provided by Google Firebase. This App is being built using Flutter as the Frontend Framework, Dart as the language to support it and Firebase as the Backend Platform to use the CRUD operations on all the data. This app will have the option for the user to login to the app using their email id and password after which they can chat with the other users using the app and also share images, videos and other files through it to the other user in the app.

Acronyms

B.Tech.	Bachelor of Technology
M.Tech.	Master of Technology
BCA	Bachelor of Computer Applications
MCA	Master of Computer Applications
B.Sc. (CS)	Bachelor of Science in Computer Science
M.Sc. (CS)	Master of Science in Computer Science
SCSE	School of Computing Science and Engineering

Table of Contents

Kaustub Ratan Pachoury.docx

ORIGINALITY REPORT

4%
SIMILARITY INDEX

PRIMARY SOURCES

1	scopedatabase.com Internet	43 words — 1%
2	www.coursehero.com Internet	35 words — 1%
3	Banking Academy Publications	23 words — 1%
4	Chen, Ziheng. "Measurement of the W Boson Branching Fractions in Proton-Proton Collisions at 13 TeV Center-Of-Mass Energy with the CMS Experiment", Northwestern University ProQuest	11 words — < 1%
5	Springer Proceedings in Mathematics & Statistics, 2014. Crossref	7 words — < 1%

EXCLUDE QUOTES	ON	EXCLUDE SOURCES	OFF
EXCLUDE BIBLIOGRAPHY	ON	EXCLUDE MATCHES	OFF

CHAPTER-1
Introduction

In the real world communication plays a very vital role. People have been communicating with each other through various applications or mediums. In the beginning people communicated with each other using letters or other sources, as these mediums could take much time to deliver the content. Cell phones are another medium of communication but the drawback is for any limited or small message which needs to be passed to another user then phone call is not an ideal way. The developers then looked to implement a text-based communication which would allow an instant communication service. The main objective of this project is to present a software application for the launching of a real time communication between operators/users. The system developed on android will enable the users to communicate with other users through text messages with the help of the internet. The system requires both devices to be connected via the internet. This application is based on Android with the backend provided by Google Firebase. This App is being built using Flutter as the Frontend Framework, Dart as the language to support it and Firebase as the Backend Platform to use the CRUD operations on all the data.. Firebase Authentication is useful to both developers and the users. Developing and maintaining sign-in set-up may be a bit difficult and time taking. Firebase provides an easy API for sign in. It also provides the data backup using real time databases.This proposed chatting messenger instantly connects you to people on the same WiFi network, without ever accessing Internet.It is the fastest tool for immediate file transfer and content sharing,share photos, files, music, videos and even other applications a local network without slowness of the internet. It allows us to Start sharing with friends at campus,college,office and home. WiFi is a technology that uses radio waves to provide network connectivity. A WiFi connection is established using a wireless adapter to create hotspots - areas in the vicinity of a wireless router that are connected to the network and allow users to access internet services. Once configured, WiFi provides wireless connectivity to your devices by emitting frequencies between 2.4GHz - 5GHz, based on the amount of data on the network.

CHAPTER-2

Literature Survey

Technology trends in both hardware and software have driven the hardware industry towards smaller, faster and more capable mobile hand-held devices that can support a wider-range of functionality and open source operating systems. Mobile hand-held devices are popularly called smart gadgets. Adding text messaging functionality to mobile devices began to gain traction in the mobile communication services community in the early 1980s. The first action plan of the Group GSM was approved in December 1982, requesting "The services and facilities offered in the public switched telephone networks and public data networks should be available in the mobile system". This plan included the exchange of text messages either directly between mobile stations, or transmitted via Message Handling Systems widely in use at that time. The first proposal which initiated the development of exchanging information or sending messages to the user was made by a contribution of Germany and France into the GSM group meeting in February 1985 in Oslo. Initial growth was slow, with customers in 1995 sending on average only 0.4 messages per GSM customer per month. In 2013, 6.1 trillion text messages were sent[2].

This translates into 193000 SMS per second. While SMS reached its popularity as a person-to-person messaging, another type of SMS is growing fast: application-to-person (A2P) messaging. A2P is a type of SMS sent from a subscriber to an application or sent from an application to a subscriber. It is commonly used by financial institutions, airlines, hotel booking sites, social networks, and other organizations sending SMS from their systems to their customers. According to research in 2013, A2P traffic is growing faster than P2P messaging traffic. Over the years several approaches and solutions presented considering the secure exchanging of messages through client and web server. The various researches have been done and are going on location based projects and in the same ratio various applications have been developed on location-based and message sharing systems. As the amount of users deal to exchange the information with other people to store the large amount of data the existing system cannot support the centralized database. Sensitive data may also be leaked accidentally due to improper disposal or resale of storage media.Diesburg et al. surveys, summarizes and compares existing methods of providing confidential storage of data but it cannot support the secured communication between the user application and the webserver.

CHAPTER-3
Tools and Technology Used

Our app is built using Flutter which uses Dart as the language to support it and Firebase as the backend to store chats and provide user login functionality.

Flutter

Flutter is an open source framework by Google for building beautiful, natively compiled, multi-platform applications from a single codebase. Flutter code compiles to ARM or Intel machine code as well as JavaScript, for fast performance on any device. Flutter's engine, written primarily in C++, provides low-level rendering support using Google's Skia graphics library. Additionally, it interfaces with platform-specific SDKs such as those provided by Android and iOS. The Flutter Engine is a portable runtime for hosting Flutter applications. It implements Flutter's core libraries, including animation and graphics, file and network I/O, accessibility support, plugin architecture, and a Dart runtime and compile toolchain. Most developers interact with Flutter via the Flutter Framework, which provides a reactive framework and a set of platform, layout, and foundation widgets.The first version of Flutter was known as "Sky" and ran on the Android operating system. It was unveiled at the 2015 Dart developer summit with the stated intent of being able to render consistently at 120 frames per second. During the keynote of Google Developer Days in Shanghai in September 2018, Google announced Flutter Release Preview 2, the last major release before Flutter 1.0. On December 4th of that year, Flutter 1.0 was released at the Flutter Live event, denoting the first stable version of the framework. On December 11, 2019, Flutter 1.12 was released at the Flutter Interactive event.

On May 6, 2020, the Dart software development kit (SDK) version 2.8 and Flutter 1.17.0 were released, adding support for the Metal API which improves performance on iOS devices by approximately 50%, as well as new Material widgets and network tracking development tools.

On March 3, 2021, Google released Flutter 2 during an online Flutter Engage event. This major update brought official support for web-based applications with a new CanvasKit renderer and web specific widgets, early-access desktop application support for Windows, macOS, and Linux and improved Add-to-App APIs. This release also utilized Dart 2.0 that featured sound null-safety, which caused many breaking changes and issues with many external packages; however, the Flutter team included instructions and tools to mitigate these issues.

On September 8th, 2021, Dart 2.14 and Flutter 2.5 were released by Google. The update brought improvements to the Android full-screen mode and the latest version of Google's Material Design called Material You. Dart received two new updates, standardizing lint conditions and marking support for Apple Silicon as stable.

The current stable channel of Flutter is 2.16.1 and the Dart version is 2.16.1

Dart

Dart is a programming language designed for client development, such as for the web and mobile apps. It is developed by Google and can also be used to build server and desktop applications. It is an object-oriented, class-based, garbage-collected language with C-style syntax.Flutter apps are written in the Dart language and make use of many of the language's more advanced features.While writing and debugging an application, Flutter runs in the Dart virtual machine, which features a just-in-time execution engine. This allows for fast compilation times as well as "hot reload", with which modifications to source files can be injected into a running application. Flutter extends this further with support for stateful hot reload, where in most cases changes to source code are reflected immediately in the running app without requiring a restart or any loss of state.For better performance, release versions of Flutter apps on all platforms use ahead-of-time (AOT) compilation.

The purpose of Dart programming is to create a frontend user interfaces for the web and mobile apps. It is under active development, compiled to native machine code for building mobile apps, inspired by other programming languages such as Java, JavaScript, C#, and is Strongly Typed. Since Dart is a compiled language so you cannot execute your code directly; instead, the compiler parses it and transfer it into machine code.

It supports most of the common concepts of programming languages like classes, interfaces, functions, unlike other programming languages. Dart language does not support arrays directly. It supports collection, which is used to replicate the data structure such as arrays, generics, and optional typing.

Firebase

Firebase is a platform developed by Google for creating mobile and web applications. It was originally an independent company founded in 2011. In 2014, Google acquired the platform and it is now their flagship offering for app development.Firebase is a Backend-as-a-Service (Baas). It provides developers with a variety of tools and services to help them develop quality apps, grow their user base, and earn profit. It is built on Google's infrastructure.Firebase is categorized as a NoSQL database program, which stores data in JSON-like documents. The Firebase Realtime Database is a cloud-hosted database in which data is stored as JSON. The data is synchronized in real-time to every connected client. All of our clients share one Realtime Database instances and automatically receive updates with the newest data, when we build cross-platform applications with our iOS, and JavaScript SDKs.

The Firebase Realtime Database is a NoSQL database from which we can store and sync the data between our users in real-time. It is a big JSON object which the developers can manage in real-time. By using a single API, the Firebase database provides the application with the current value of the data and updates to that data. Real-time syncing makes it easy for our users to access their data from any device, be it web or mobile.

The Realtime database helps our users collaborate with one another. It ships with mobile and web SDKs, which allow us to build our app without the need for servers. When our users go offline, the Real-time Database SDKs use local cache on the device for serving and storing changes. The local data is automatically synchronized, when the device comes online.

Android Studio

Android Studio is the official integrated development environment for Google's Android operating system, built on JetBrains' IntelliJ IDEA software and designed specifically for Android development.It is available for download on Windows, macOS and Linux based operating systems or as a subscription-based service in 2020.It is a replacement for the Eclipse Android Development Tools (E-ADT) as the primary IDE for native Android application development.Android Studio was announced on May 16, 2013, at the Google I/O conference. It was in the early access preview stage starting from version 0.1 in May 2013, then entered beta stage starting from version 0.8 which was released in June 2014. The first stable build was released in December 2014, starting from version 1.0.On May 7, 2019, Kotlin replaced Java as Google's preferred language for Android app development. Java is still supported, as is C++.

Features of Android Studio

- It has a flexible Gradle-based build system.
- It has a fast and feature-rich emulator for app testing.
- Android Studio has a consolidated environment where we can develop for all Android devices.
- Apply changes to the resource code of our running app without restarting the app.
- Android Studio provides extensive testing tools and frameworks.
- It supports C++ and NDK.
- It provides build-in supports for Google Cloud Platform. It makes it easy to integrate Google Cloud Messaging and App Engine.

Chapter-4
Functionality/ Working of Project

The app is built using flutter so it's ready to be shipped on both iOS and Android. The app, although it is cross platform, can perform almost close to the performance of a native app on both platforms thanks to flutter's skia engine. Unlike React Native flutter uses its own engine to render every pixel of the application on the user's device display, so the app is much faster to use and smoother too.The user is first greeted with a login page which has the app name running on top with animation and an option to either login or register. After a user registers on the app he is taken immediately to the chat screen where he can group chat with other users on the app.

Algorithm1

The algorithm lets the user login into the application with a valid email id. The algorithm first initializes the variable sign_in to 1. That means it's true. The user then enters the email id which is stored in another variable internally in the database. The email id is then verified and the result is stored in a variable request_code. If the value of the request_code matches with the value of the variable if both the values are same then it is considered as the email is valid and the user signed in to the application. If the values do not match then the signin will not be done and the task ends. Send and receive messages: After a successful signin the user is now able to send and receive the messages.

Algorithm2

The function onclick is a function defined and the variable of type EditText is declared and initialized to the id of input text which is retrieved from the layout xml file. Email id and username of the sender is received from the firebase database instance along with the text which needs to be sent. These two are converted to string and stored in the database reference of the database root node. When this process is done then the input is set to null and the user is allowed to send another message. Secondly, for the existing users if the current user name from the firebase database is not equal to null then the user will get a welcome screen with the previous messages and new messages by calling the message display function.

Project Code

main.dart -

```
import 'package:flutter/material.dart';
import 'package:flash_chat/screens/chat_screen.dart';
import 'package:flash_chat/screens/login_screen.dart';
import
'package:flash_chat/screens/registration_screen.dart';
import 'package:flash_chat/screens/welcome_screen.dart';
import 'package:firebase_core/firebase_core.dart';
import 'firebase_options.dart';

void main() async {
  WidgetsFlutterBinding.ensureInitialized();
  await Firebase.initializeApp(
    options: DefaultFirebaseOptions.currentPlatform,
  );
  runApp(FlashChat());
}

class FlashChat extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      initialRoute:
      WelcomeScreen.id, routes: {
        LoginScreen.id: (context) => LoginScreen(),
        RegistrationScreen.id: (context) => RegistrationScreen(),
        WelcomeScreen.id: (context) => WelcomeScreen(),
        ChatScreen.id: (context) => ChatScreen(),
      },
    );
  }
}
```

login_screen.dart -

```
import 'package:flutter/material.dart';
import 'package:flash_chat/constants.dart';
import 'package:flash_chat/screens/chat_screen.dart';
import 'package:flash_chat/components/rounded_button.dart';
import 'package:firebase_auth/firebase_auth.dart';
import 'package:modal_progress_hud/modal_progress_hud.dart';

class LoginScreen extends StatefulWidget {
  static const String id = 'login_screen';

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

class _LoginScreenState extends State<LoginScreen> {
  final _auth = FirebaseAuth.instance;

  String email;
  String password;
  bool showSpinner = false;
  final emailTextContorller = TextEditingController();
  final passwordTextController = TextEditingController();

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      backgroundColor: Colors.white,
      body: ModalProgressHUD(
      inAsyncCall: showSpinner,
      child: Padding(
          padding: EdgeInsets.symmetric(horizontal: 24.0),
          child: Column(
            mainAxisAlignment: MainAxisAlignment.center,
            crossAxisAlignment: CrossAxisAlignment.stretch,
            children: <Widget>[
              Flexible(
                child: Hero(
                tag: 'logo',
                  child: Container(
```

```
      height: 200.0,
      child: Image.asset('images/logo.png'),
    ),
  ),
),
SizedBox(
  height: 48.0,
),
TextField(
  controller: emailTextContorller,
  keyboardType: TextInputType.emailAddress,
  textAlign: TextAlign.center,
  onChanged: (value) {
    email = value;
  },
  decoration: kTextFieldDecoration.copyWith(
    hintText: 'Enter your email',
  ),
),
SizedBox(
  height: 8.0,
),
TextField(
  controller: passwordTextController,
  textAlign: TextAlign.center,
  obscureText: true,
  onChanged: (value) {
    password = value;
  },
  decoration: kTextFieldDecoration.copyWith(
    hintText: 'Enter your password.',
  ),
),
SizedBox(
  height: 24.0,
),
RoundedButton(
  color: Colors.lightBlueAccent,
  onPressed: () async {
  setState(() {
      showSpinner = true;
    });
    try {
      UserCredential userCredential =
```

```
                  await _auth.signInWithEmailAndPassword(
                    email: email,
                    password: password,
                  );
                  if (userCredential != null) {
                    Navigator.pushNamed(context, ChatScreen.id);
                    emailTextContorller.clear();
                    passwordTextController.clear();
                  }
                  setState(() {
                    showSpinner = false;
                  });
                } catch (e) {
                  print(e);
                }
              },
              text: 'Log In',
            ),
          ],
        ),
      ),
    ),
  );
 }
}
```

registration_screen.dart -

```
import 'package:flutter/material.dart';
import 'package:flash_chat/components/rounded_button.dart';
import 'package:flash_chat/constants.dart';
import 'package:firebase_auth/firebase_auth.dart';
import 'package:flash_chat/screens/chat_screen.dart';
import 'package:modal_progress_hud/modal_progress_hud.dart';

class RegistrationScreen extends StatefulWidget {
  static const String id = 'registration_screen';

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

class _RegistrationScreenState extends State<RegistrationScreen> {
  final _auth = FirebaseAuth.instance;

  String email;
  String password;
  bool showSpinner = false;
  final emailTextContorller = TextEditingController();
  final passwordTextController = TextEditingController();

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      backgroundColor: Colors.white,
      body: ModalProgressHUD(
      inAsyncCall: showSpinner,
      child: Padding(
          padding: EdgeInsets.symmetric(horizontal: 24.0),
          child: Column(
            mainAxisAlignment: MainAxisAlignment.center,
            crossAxisAlignment: CrossAxisAlignment.stretch,
            children: <Widget>[
              Flexible(
                child: Hero(
                tag: 'logo',
                  child: Container(
```

```
      height: 200.0,
      child: Image.asset('images/logo.png'),
    ),
  ),
),
SizedBox(
  height: 48.0,
),
TextField(
  controller: emailTextContorller,
  keyboardType: TextInputType.emailAddress,
  textAlign: TextAlign.center,
  onChanged: (value) {
    email = value;
  },
  decoration: kTextFieldDecoration.copyWith(
    hintText: 'Enter your email',
  ),
),
SizedBox(
  height: 8.0,
),
TextField(
  controller: passwordTextController,
  textAlign: TextAlign.center,
  obscureText: true,
  onChanged: (value) {
    password = value;
  },
  decoration: kTextFieldDecoration.copyWith(
    hintText: 'Enter your password',
  ),
),
SizedBox(
  height: 24.0,
),
RoundedButton(
  color: Colors.blueAccent,
  onPressed: () async {
  setState(() {
      showSpinner = true;
    });
    try {
      UserCredential newUser =
```

```
                  await _auth.createUserWithEmailAndPassword(
                    email: email,
                    password: password,
                  );
                  if (newUser != null) {
                    Navigator.pushNamed(context, ChatScreen.id);
                    emailTextContorller.clear();
                    passwordTextController.clear();
                  }
                  setState(() {
                    showSpinner = false;
                  });
                } catch (e) {
                  print(e);
                }
              },
              text: 'Register',
            ),
          ],
        ),
      ),
    ),
  );
 }
}
```

welcome_screen.dart -

```
import 'package:flutter/material.dart';
import 'package:flash_chat/screens/login_screen.dart';
import
'package:flash_chat/screens/registration_screen.dart';
import 'package:animated_text_kit/animated_text_kit.dart';
import 'package:flash_chat/components/rounded_button.dart';

class WelcomeScreen extends StatefulWidget {
  static const String id = 'welcome_screen';

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

class _WelcomeScreenState extends State<WelcomeScreen>
  with SingleTickerProviderStateMixin {
  AnimationController controller;
  Animation animation;

  @override
  void initState() {
    super.initState();

    controller =
      AnimationController( vsync:
      this,
      duration: Duration(seconds: 1),
    );

    animation = ColorTween(begin: Colors.blueGrey, end: Colors.white)
        .animate(controller);

    controller.forward();

    controller.addListener(() {
      setState(() {});
    });
  }
```

```
@override
void dispose() {
```

```
  controller.dispose();
  super.dispose();
}

@override
Widget build(BuildContext context) {
  return Scaffold(
    backgroundColor: animation.value,
    body: Padding(
      padding: EdgeInsets.symmetric(horizontal: 24.0),
      child: Column(
        mainAxisAlignment: MainAxisAlignment.center,
        crossAxisAlignment: CrossAxisAlignment.stretch,
        children: <Widget>[
          Row(
            children: <Widget>[
              Hero(
                tag: 'logo',
                child: Container(
                  child: Image.asset('images/logo.png'),
                  height: 60.0,
                ),
              ),
              AnimatedTextKit(
                animatedTexts: [
                TypewriterAnimatedText(
                  'Flash Chat',
                  textStyle: TextStyle(
                  fontSize: 45.0,
                    fontWeight: FontWeight.w900,
                  ),
                  speed: const Duration(milliseconds: 350),
                ),
                ],
                repeatForever: true,
              ),
            ],
          ),
          SizedBox(
            height: 48.0,
          ),
          RoundedButton(
            color: Colors.lightBlueAccent,
            onPressed: () {
```

```
              Navigator.pushNamed(context, LoginScreen.id);
            },
            text: 'Log In',
          ),
          RoundedButton(
            color:
            Colors.blueAccent,
            onPressed: () {
              Navigator.pushNamed(context, RegistrationScreen.id);
            },
            text: 'Register',
          ),
        ],
      ),
    ),
  );
 }
}
```

chat_screen.dart -

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

final _firestore = FirebaseFirestore.instance;
User loggedInUser;

class ChatScreen extends StatefulWidget {
  static const String id = 'chat_screen';

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

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

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

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

  @override
```

```
Widget build(BuildContext context) {
 return Scaffold(
  appBar: AppBar(
   automaticallyImplyLeading: false,
   actions: <Widget>[
    IconButton(
        icon: Icon(Icons.logout),
        onPressed: () {
         _auth.signOut();
         Navigator.pop(context);
        }),
   ],
   title: Text('⚡Chat'),

   backgroundColor: Colors.lightBlueAccent,
  ),
  body: SafeArea(
   child: Column(
    mainAxisAlignment: MainAxisAlignment.spaceBetween,
    crossAxisAlignment: CrossAxisAlignment.stretch,
    children: [
     MessagesStream()
     , Container(
      decoration: kMessageContainerDecoration,
      child: Row(
       crossAxisAlignment: CrossAxisAlignment.center,
       children: [
        Expanded(
         child: TextField(
          controller:
          messageTextContorller,
          onChanged: (value) {
          messageText = value;
          },
          decoration: kMessageTextFieldDecoration,
         ),
        ),
        TextButton(
         onPressed: () {
          messageTextContorller.clear();
          _firestore.collection('messages').add({
           'sender': loggedInUser.email,
           'text': messageText,
           'timestamp': FieldValue.serverTimestamp(),
```

```
    });
  },
```

```
                child: Text(
                  'Send',
                  style: kSendButtonTextStyle,
                ),
              ),
            ],
          ),
        ),
      ],
    ),
  ),
 );
 }
}

class MessagesStream extends StatelessWidget {
 @override
 Widget build(BuildContext context) {
  return StreamBuilder<QuerySnapshot>(
  stream: _firestore
      .collection('messages')
      .orderBy('timestamp', descending: true)
      .snapshots(),
   builder: (context, asyncSnapshot) {
    if (!asyncSnapshot.hasData) {
    return Center(
      child: CircularProgressIndicator(
       color: Colors.blueAccent,
      ),
    );
    }
    final messages = asyncSnapshot.data.docs;
    List<MessageBubble> messageBubbles = [];
    for (var message in messages) {
     final messageText = message['text'];
     final messageSender = message['sender'];
     final currentUser = loggedInUser.email;

     final messageBubble = MessageBubble(
      text: messageText,
      sender: messageSender,
      isMe: currentUser == messageSender,
     );
```

```
        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 text;
 final String
 sender; final bool
 isMe;
 MessageBubble({this.text, this.sender, this.isMe});
 @override
 Widget build(BuildContext context) {
  return Padding(
    padding: const EdgeInsets.all(10.0),
    child: Column(
    crossAxisAlignment:
        isMe ? CrossAxisAlignment.end : CrossAxisAlignment.start,
      children: [
        Text(
          sender,
          style: TextStyle(
            color: Colors.black54,
            fontSize: 12.0,
          ),
        ),
        Material(
          elevation:
          5.0,
          borderRadius: isMe
              ? BorderRadius.only(
                  bottomLeft: Radius.circular(30.0),
```

```
                topLeft: Radius.circular(30.0),
                bottomRight: Radius.circular(30.0),
              )
            : BorderRadius.only(
                bottomLeft: Radius.circular(30.0),
                topRight: Radius.circular(30.0),
                bottomRight: Radius.circular(30.0),
              ),
        color: isMe ? Colors.lightBlueAccent : Colors.white,
        child: Padding(
          padding: const EdgeInsets.symmetric(
            horizontal: 20.0,
            vertical: 10.0,
          ),
          child: Text(
            text,
            style: TextStyle(
              fontSize: 15.0,
              color: isMe ? Colors.white : Colors.black54,
            ),
          ),
        ),
      ),
    ],
  ),
 );
 }
}
```

rounded_button.dart -

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

class RoundedButton extends StatelessWidget {
  final Color color;
  final Function onPressed;
  final String text;

  RoundedButton(
      {@required this.color, @required this.onPressed, @required this.text});

  @override
  Widget build(BuildContext context) {
    return Padding(
      padding: EdgeInsets.symmetric(vertical: 16.0),
      child: Material(
        elevation: 5.0,
        color: color,
        borderRadius: BorderRadius.circular(30.0),
        child: MaterialButton(
          onPressed: onPressed,
          minWidth: 200.0,
          height: 42.0,
          child: Text(
          text,
            style: TextStyle(
              color: Colors.white,
            ),
          ),
        ),
      ),
    );
  }
}
```

firebase_options.dart -

```
// File generated by FlutterFire CLI.
// ignore_for_file: lines_longer_than_80_chars
import 'package:firebase_core/firebase_core.dart' show
FirebaseOptions; import 'package:flutter/foundation.dart'
    show defaultTargetPlatform, kIsWeb, TargetPlatform;

/// Default [FirebaseOptions] for use with your Firebase apps.
///
/// Example:
/// ```dart
/// import 'firebase_options.dart';
/// // ...
/// await Firebase.initializeApp(
///   options: DefaultFirebaseOptions.currentPlatform,
/// );
/// ```
class DefaultFirebaseOptions {
  static FirebaseOptions get currentPlatform {
    if (kIsWeb) {
      throw UnsupportedError(
        'DefaultFirebaseOptions have not been configured for web - '
        'you can reconfigure this by running the FlutterFire CLI again.',
      );
    }
    // ignore: missing_enum_constant_in_switch
    switch (defaultTargetPlatform) {
      case TargetPlatform.android:
        return android;
      case TargetPlatform.iOS:
        throw UnsupportedError(
          'DefaultFirebaseOptions have not been configured for ios - '
          'you can reconfigure this by running the FlutterFire CLI again.',
        );
      case TargetPlatform.macOS:
        throw UnsupportedError(
          'DefaultFirebaseOptions have not been configured for macos - '
          'you can reconfigure this by running the FlutterFire CLI again.',
        );
    }
```

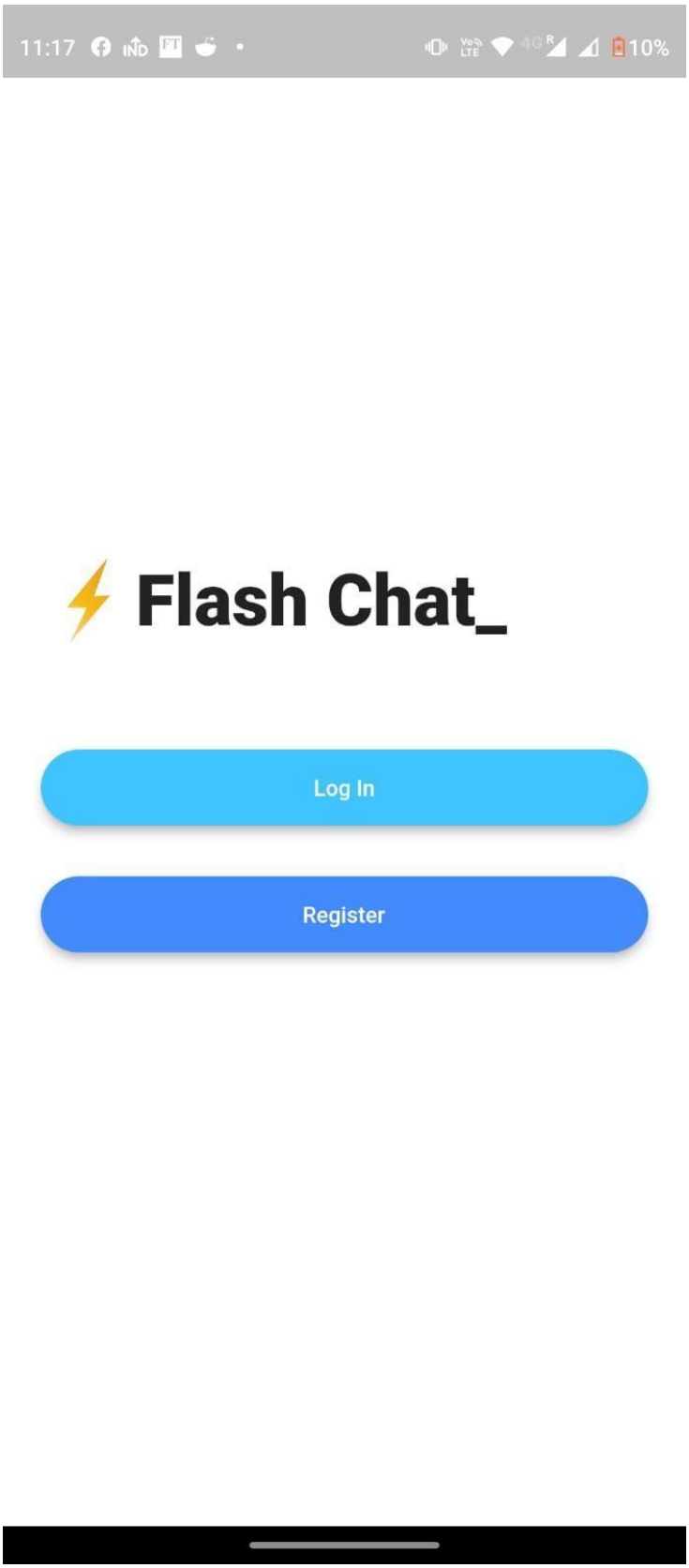
```
    throw UnsupportedError(
      'DefaultFirebaseOptions are not supported for this platform.',
    );
  }

  static const FirebaseOptions android = FirebaseOptions(
    apiKey: 'AIzaSyDBom0DG4Kv2IWSnkUtzQAKnXhFfzn_1L8',
    appId: '1:852185669833:android:3c6b84503a2377750fad18',
    messagingSenderId: '852185669833',
    projectId: 'flash-chat-d5059',
    storageBucket: 'flash-chat-d5059.appspot.com',
  );
}
```

constants.dart -

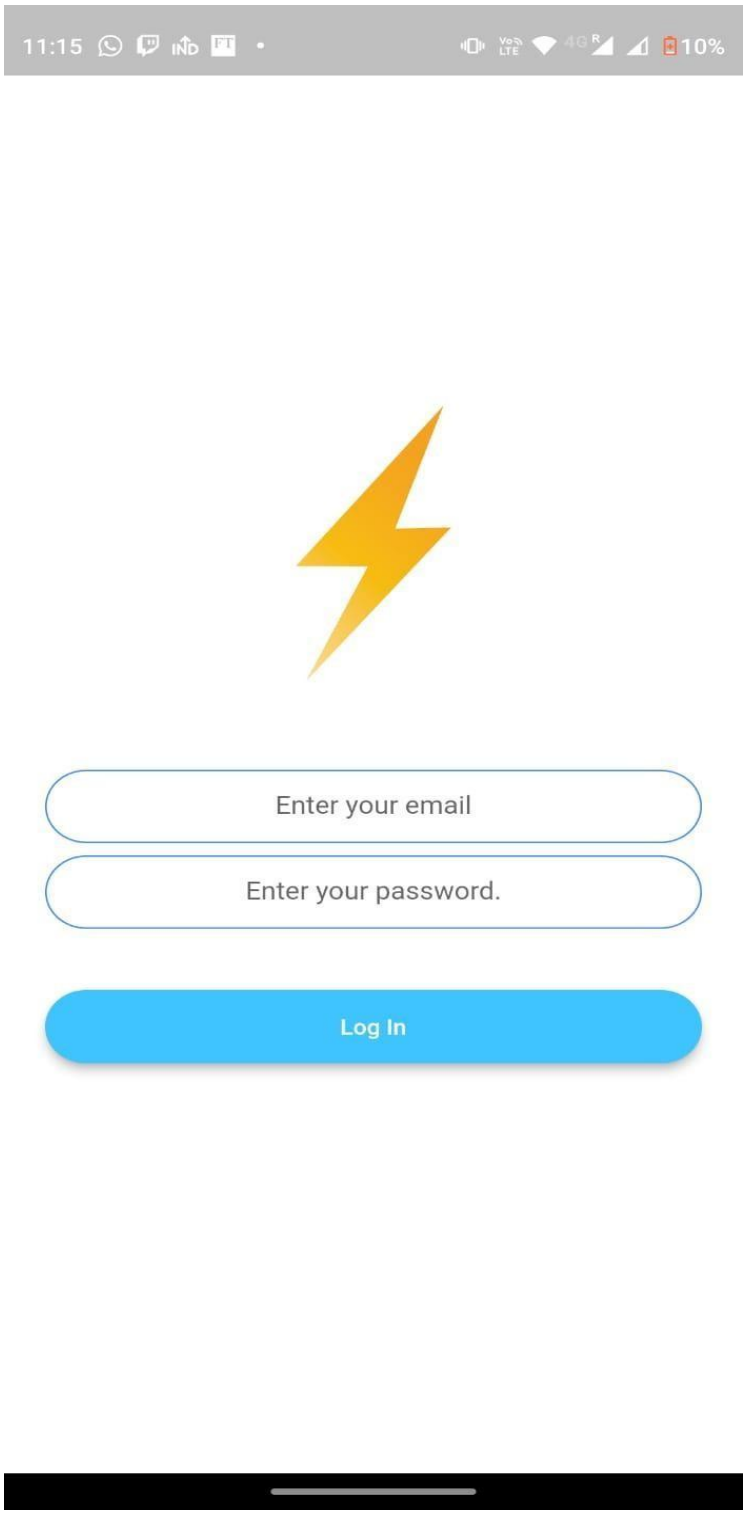
```
import 'package:flutter/material.dart';

const kSendButtonTextStyle = TextStyle(
  color: Colors.lightBlueAccent,
  fontWeight: FontWeight.bold,
  fontSize: 18.0,
);

const kMessageTextFieldDecoration = InputDecoration(
  contentPadding: EdgeInsets.symmetric(vertical: 10.0, horizontal:
  20.0), hintText: 'Type your message here...',
  border: InputBorder.none,
);

const kMessageContainerDecoration = BoxDecoration(
  border: Border(
    top: BorderSide(color: Colors.lightBlueAccent, width: 2.0),
  ),
);

const kTextFieldDecoration = InputDecoration(
  contentPadding: EdgeInsets.symmetric(vertical: 10.0, horizontal:
  20.0), border: OutlineInputBorder(
    borderRadius: BorderRadius.all(Radius.circular(32.0)),
  ),
  enabledBorder: OutlineInputBorder(
    borderSide: BorderSide(color: Colors.blueAccent, width: 1.0),
    borderRadius: BorderRadius.all(Radius.circular(32.0)),
  ),
  focusedBorder: OutlineInputBorder(
    borderSide: BorderSide(color: Colors.blueAccent, width: 2.0),
    borderRadius: BorderRadius.all(Radius.circular(32.0)),
  ),
);
```

Working App Screenshots

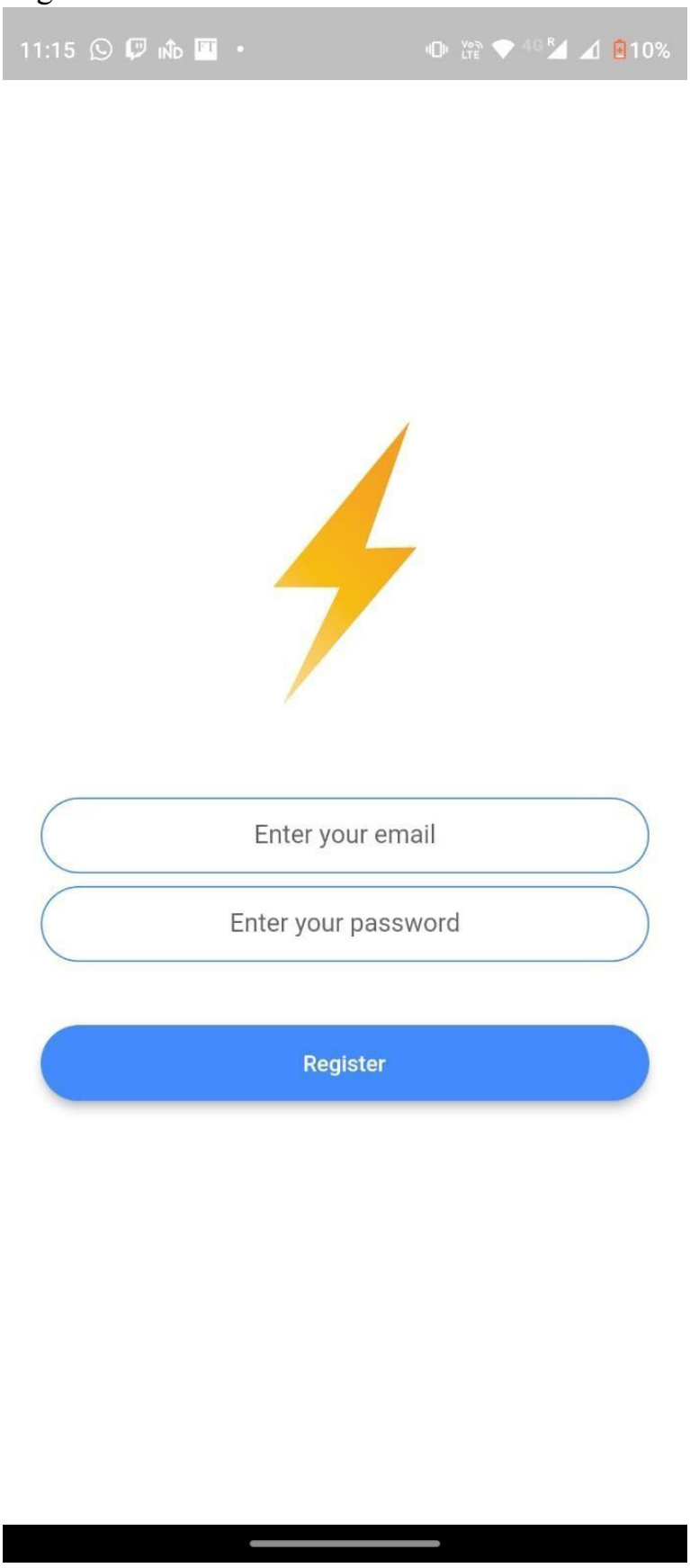
Welcome Screen -


11:17
10%
Flash Chat_
Log In
Register

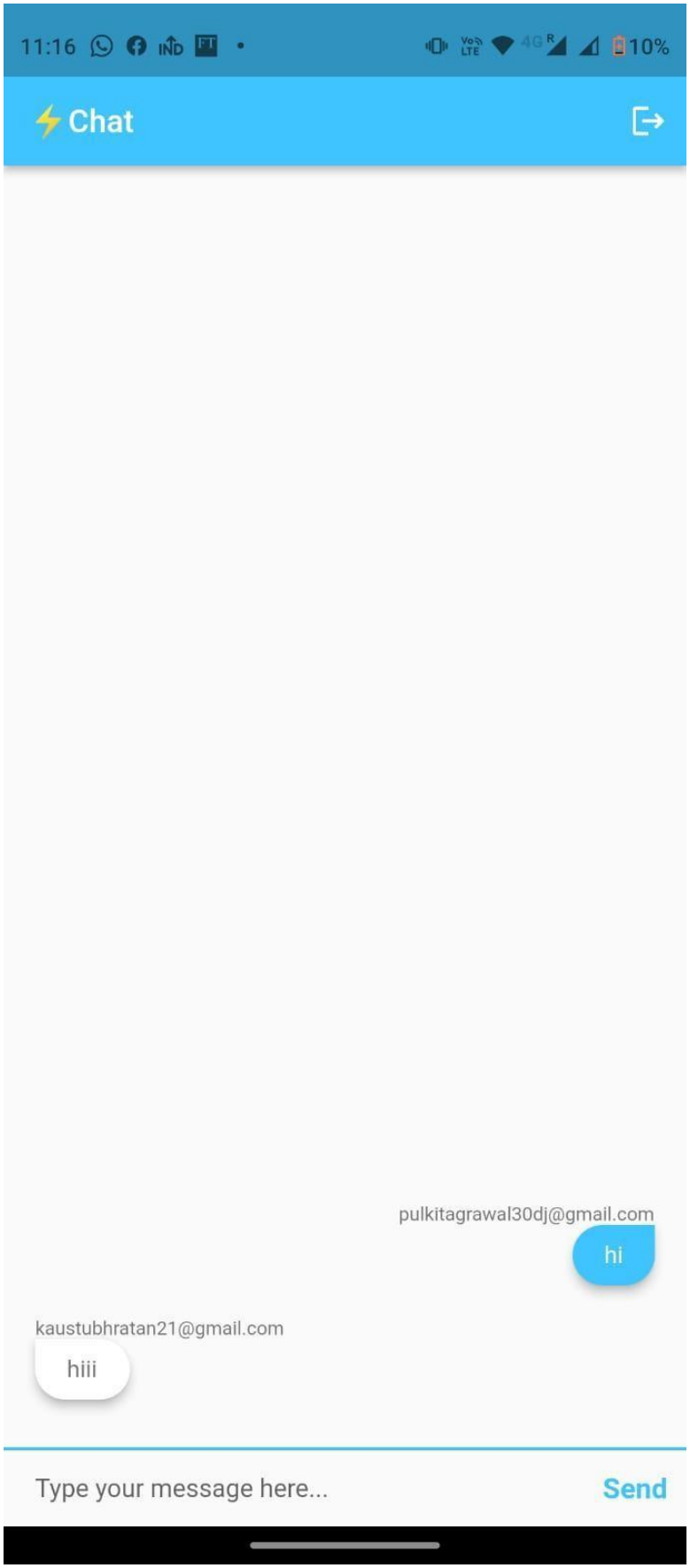
Login Screen -



Registration Screen -



Chat Screen -


11:16
10%
Chat
pulkitagrawal30dj@gmail.com
hi
kaustubhratan21@gmail.com
hiii
Type your message here...
Send

Chapter-5
Conclusion and Future Scope

There is always some place for enhancements in any software application, however good and efficient the application may be. Right now, we are dealing with only the instant messaging between the peers. In future the application may further developed to include some features such as:

1. Voice messaging.
2. Group calling
3. Live streaming
4. Messages auto delete after a given time.
5. Personalized message tunes.

And a messaging application feature which allows the user to create chat room while in conversation with another user by just sending the chatroom name with the hash symbol at the beginning.

References

1. Anon., 2015. Development of a Health Care Assistant App for the Seniors. International Journal of Applied Science and Engineering, pp. 3-5.
2. Jianye Liu; Jiankun Yu, Research on Development of Android Applications, 4th International Conference on Intelligent Networks and Intelligent Systems, 15 December 2011
3. Abhinav Kathuria et al, Challenges in Android Application Development: A Case Study, Vol.4 Issue.5, May- 2015, pg. 294-299
4. Li Ma et al, Research and Development of Mobile Application for Android Platform, International Journal of Multimedia and Ubiquitous Engineering 9(4):187-198 • April 2014

5. Nikhil M. Dongre, Nikhil M. Dongre, Journal of Computer Engineering (IOSR-JCE), Volume 19, Issue 2, Ver. I (Mar.-Apr. 2017), PP 65-77
6. Javed Ahmad Shaheen et al, Android OS with its Architecture and Android Application with Dalvik Virtual Machine Review, International Journal of Multimedia and Ubiquitous Engineering Vol. 12, No. 7 (2017), pp. 19-30
7. Sajid Nabi Khan, Ikhlaq Ul Firdous, Review on Android App Security, International Journal of Advanced Research in Computer Science and Software Engineering, Volume 7, Issue 4, April 2017
8. Lazarela Lazareska, Kire Jakimoski et al, Analysis of the Advantages and Disadvantages of Android and iOS Systems and Converting Applications from Android to iOS Platform and Vice Versa, American Journal of Software Engineering and Applications 2017; 6(5): 116-120
9. Bin Peng et al, The Android Application Development College Challenge, 2012 IEEE 14th International Conference on High Performance Computing and Communication & 2012 IEEE 9th International Conference on Embedded Software and Systems, 18 October 2012
10. Shao Guo-Hong, Application Development Research Based on Android Platform,2014 7th International Conference on

Intelligent Computation Technology and Automation, 08 January 2015
11. S Karthick, Android security issues and solutions, 2017 International Conference on Innovative Mechanisms for Industry Applications (ICIMIA), 13 July 2017
12. Pravin Auti, Sangam Mahale, Vikram Zanjad, Madhuri Dangat, n.d. An Android Based Global Chat Application. 4(1), pp. 1-2.
13. Pravin Auti, Sangam Mahale, Vikram Zanjad, Madhuri Dangat, n.d. An Android Based Global Chat Application. 4(1).
14. S, A. K., n.d. Mastering Firebase for Android Development: Build real-time, scalable, and cloud-enabled Android apps with Firebase. s.l.: s.n