**EXPERIMENT 5b**

**NAME: Chirag Rana, UID: 2018130043**

**Prerak Parekh, UID: 2018130035**

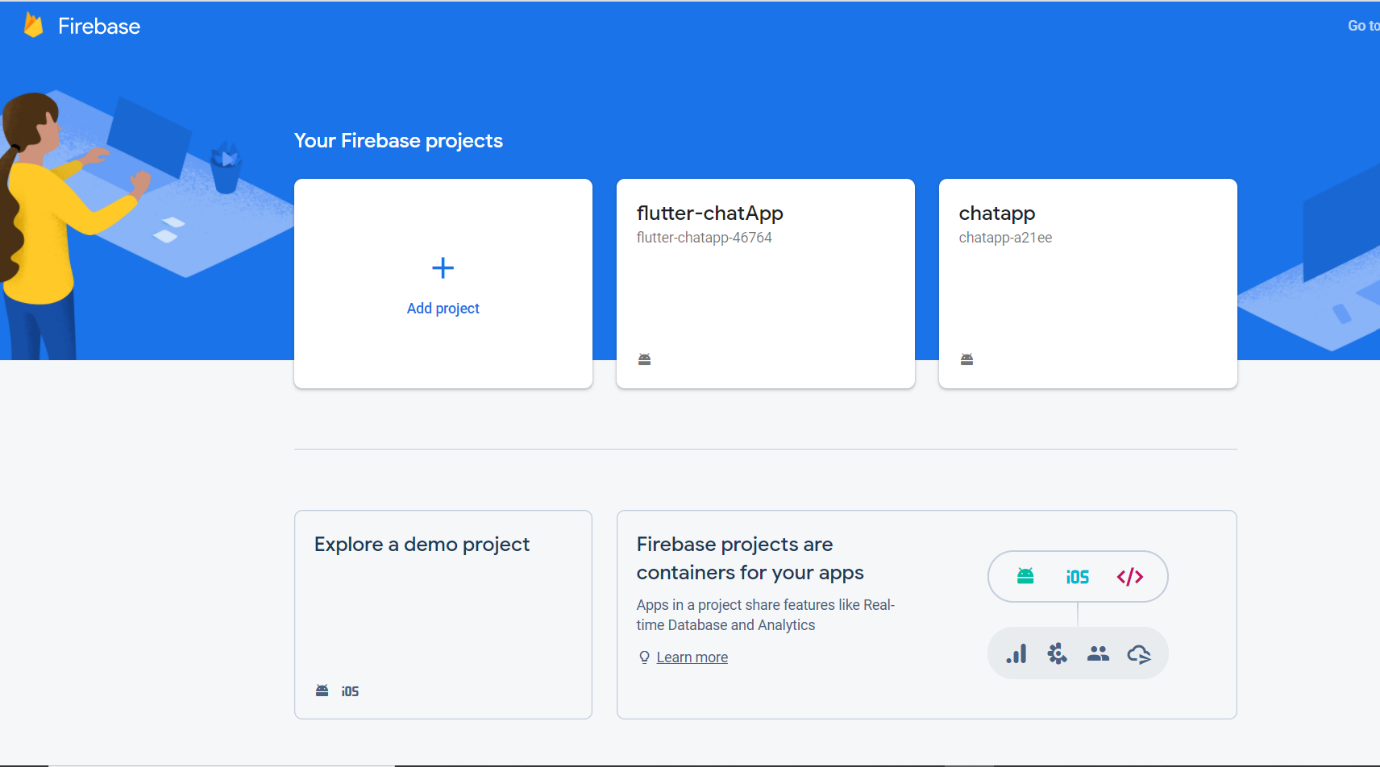
**Kunal Nalawade. UID: 2018130031**

**CLASS: TE COMPS BATCH: C**

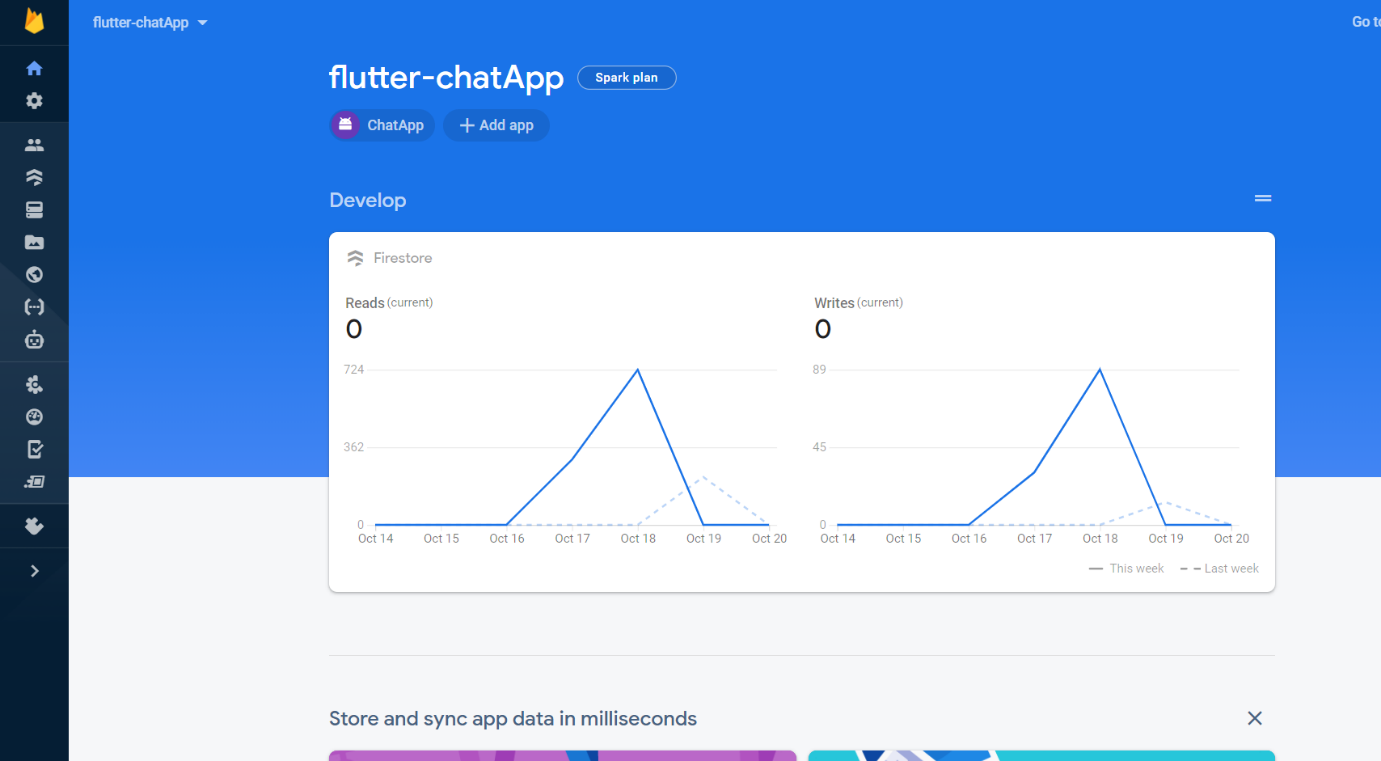
**DATE: 19/10/2020 Group 2**

**Aim:** Develop a mobile application.[Flutter]

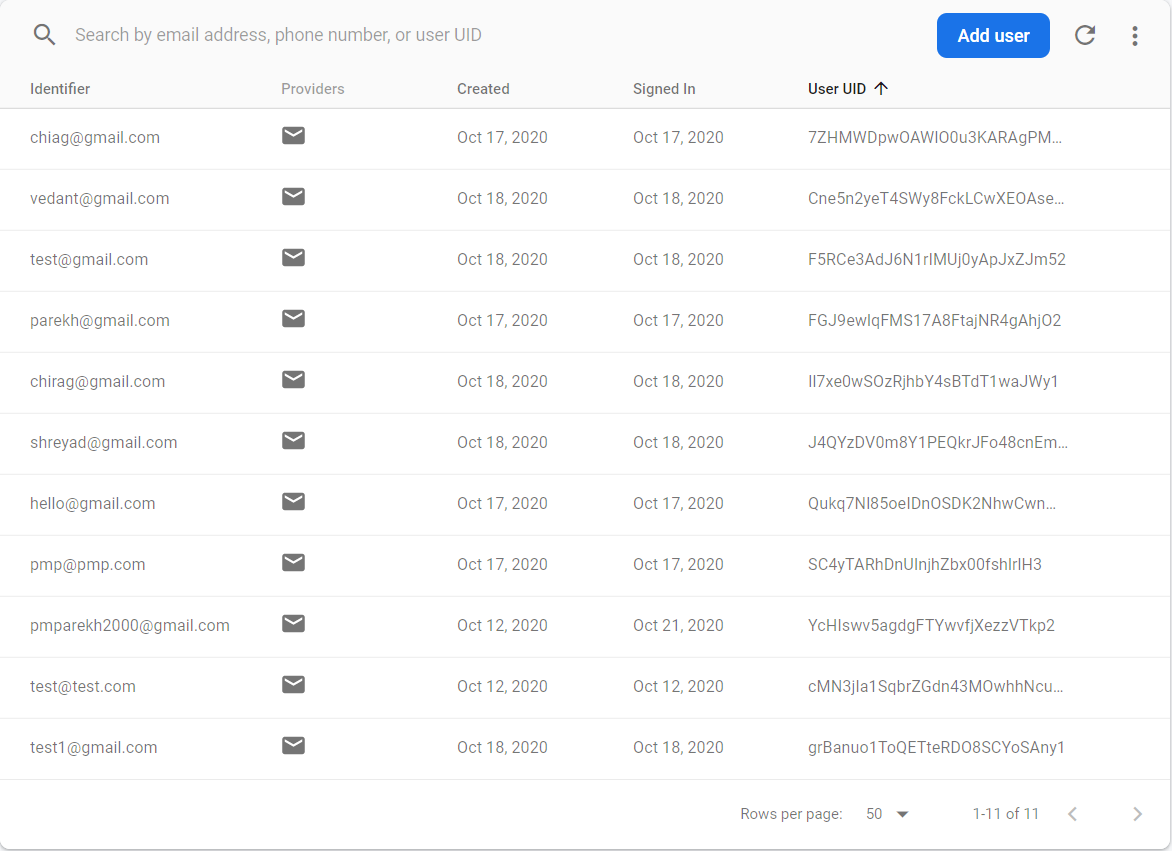
Backend of firebase



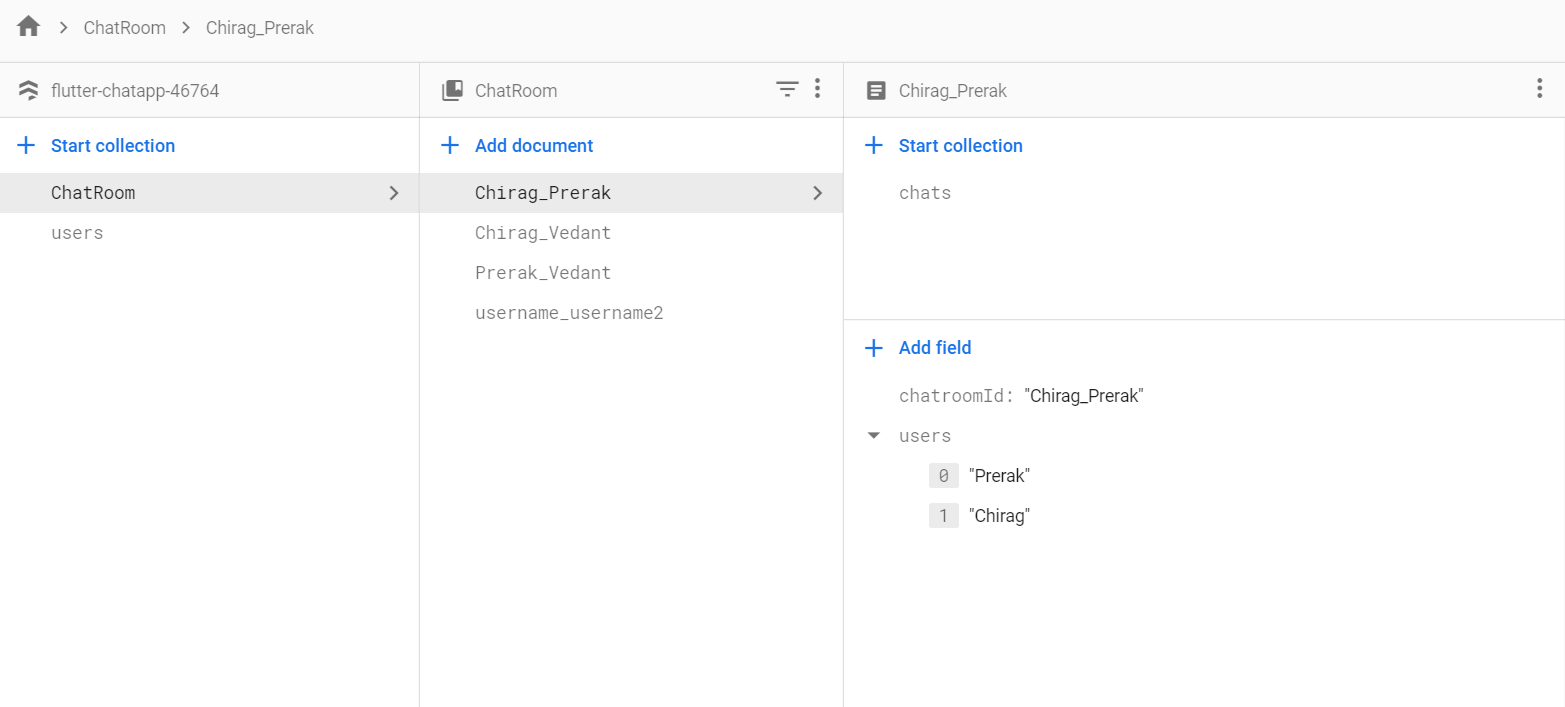
Project Dashboard



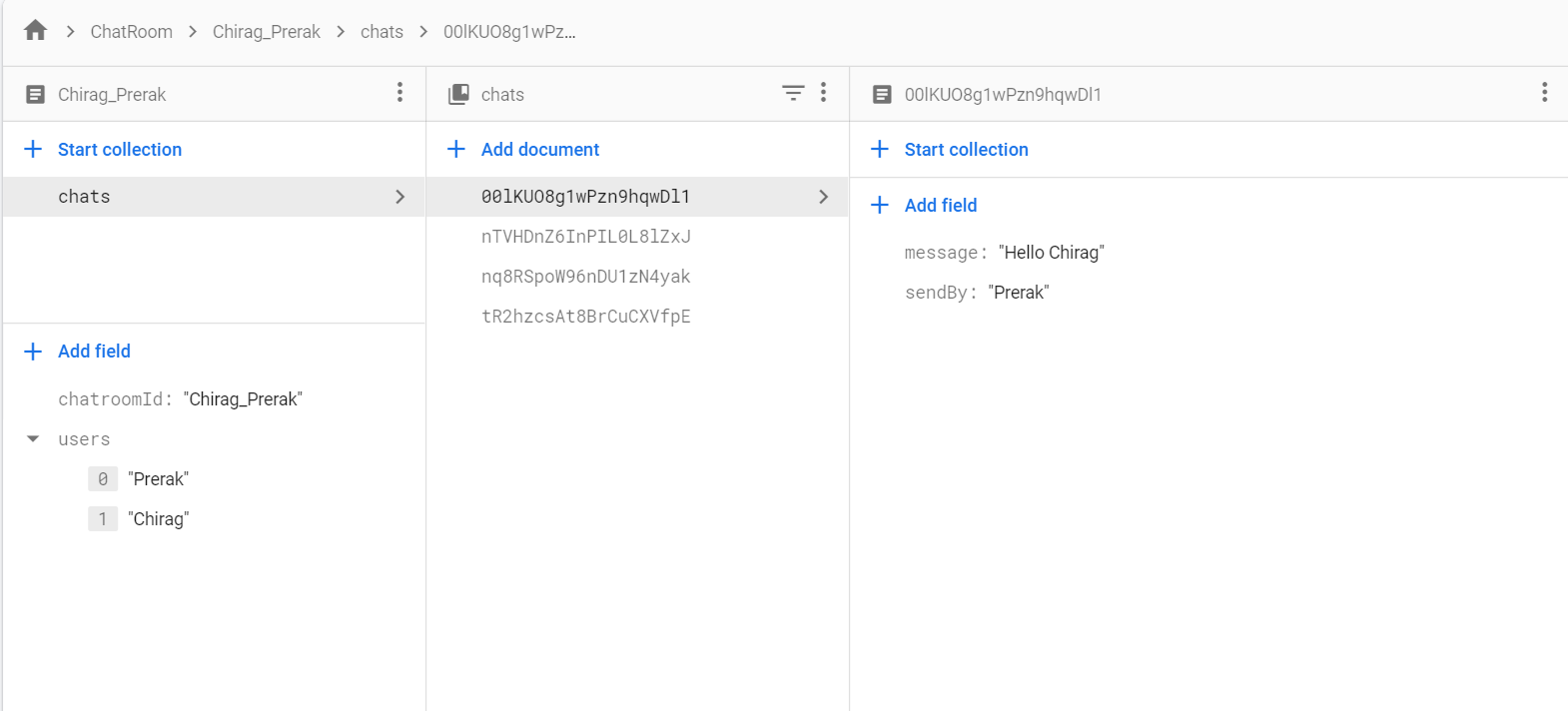
Authentication database



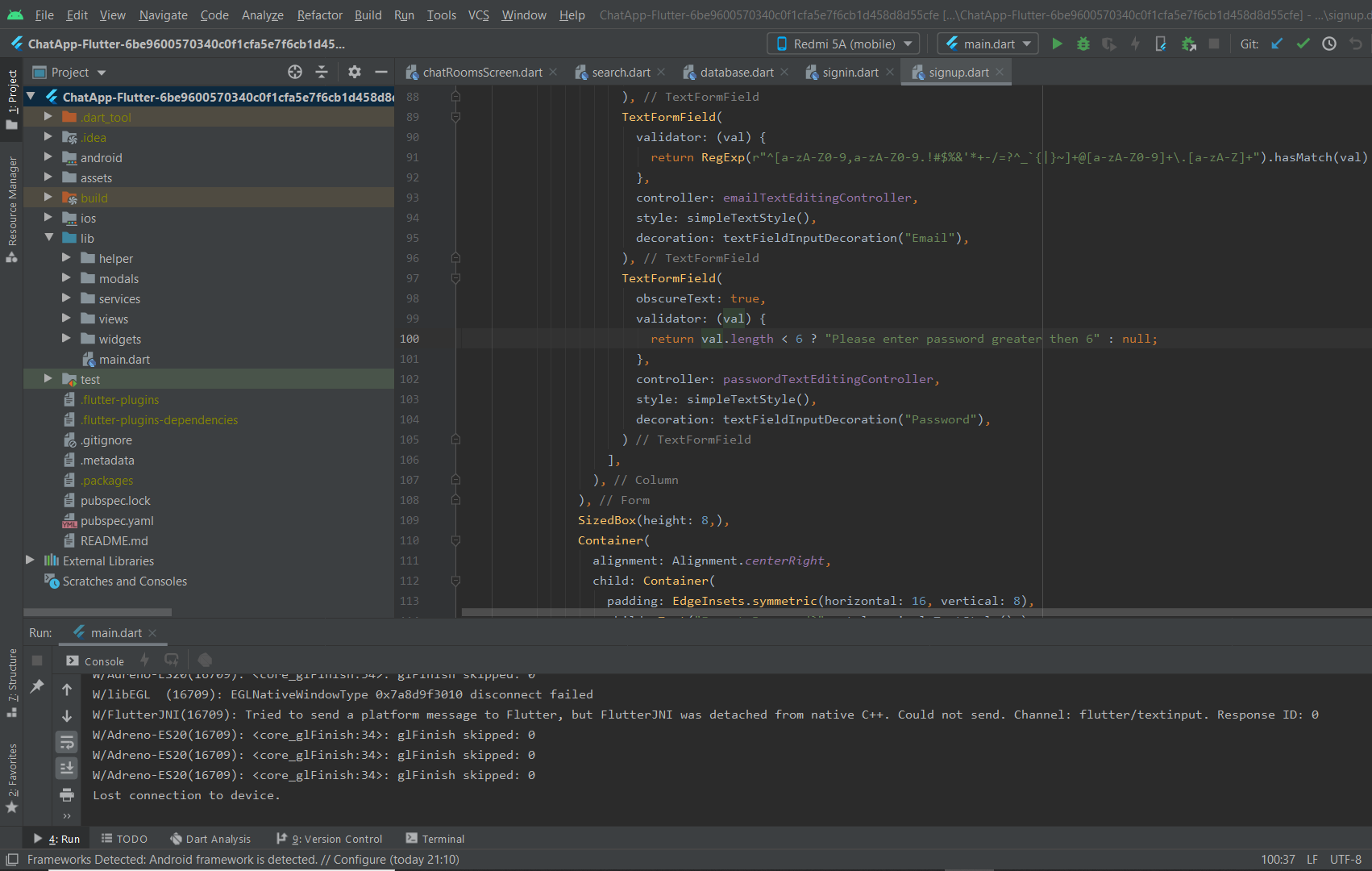
Firestore chatroom



Exploring chat database



Exploring the code



main.dart file from where the entire execution begins

import 'package:chatapp/helper/authenticate.dart';  
import 'package:chatapp/helper/helperfunction.dart';  
import 'package:chatapp/views/chatRoomsScreen.dart';  
import 'package:flutter/material.dart';  
  
void main() {  
 runApp(MyApp());  
}  
  
class MyApp extends StatefulWidget {  
 // This widget is the root of your application.  
 @override  
 \_MyAppState createState() => \_MyAppState();  
}  
  
class \_MyAppState extends State<MyApp> {  
  
 bool userIsLoggedIn = false;  
  
 @override  
 void initState() {  
 // *TODO: implement initState* getLoggedInState();  
 super.initState();  
 }  
  
 getLoggedInState() async {  
 await HelperFunctions.*getUserLoggedInSharedPreference*().then((val) {  
 setState(() {  
 userIsLoggedIn = val;  
 });  
 });  
 }  
  
 @override  
 Widget build(BuildContext context) {  
 return MaterialApp(  
 title: 'Flutter Demo',  
 debugShowCheckedModeBanner: false,  
 theme: ThemeData(  
 primaryColor: Color(0xff145C9E),  
 scaffoldBackgroundColor: Color(0xff1F1F1F),  
 primarySwatch: Colors.*blue*,  
 visualDensity: VisualDensity.*adaptivePlatformDensity*,  
 ),  
 home: userIsLoggedIn ? ChatRoom() : Authenticate(),  
 );  
 }  
}

Sign up screen code

Widget build(BuildContext context) {  
 return Scaffold(  
 appBar: appBarMain(context),  
 body: isLoading ? Container(  
 child: Center(child: CircularProgressIndicator()),  
 ) :SingleChildScrollView(  
 child: Container(  
 alignment: Alignment.*bottomCenter*,  
 child: Container(  
 height: MediaQuery.*of*(context).size.height-50,  
 padding: EdgeInsets.symmetric(horizontal: 24),  
 child: Column(  
 mainAxisAlignment: MainAxisAlignment.end,  
 children: [  
 Form(  
 key: formKey,  
 child: Column(  
 children: [  
 TextFormField(  
 validator: (val) {  
 return (val.isEmpty || val.length < 2) ? "Please provide a valid username" : null;  
 },  
 controller: userNameTextEditingController,  
 style: simpleTextStyle(),  
 decoration: textFieldInputDecoration("Username"),  
 ),  
 TextFormField(  
 validator: (val) {  
 return RegExp(r"^[a-zA-Z0-9,a-zA-Z0-9.!#$%&'\*+-/=?^\_`{|}~]+@[a-zA-Z0-9]+\.[a-zA-Z]+").hasMatch(val) ? null : "Enter a valid email";  
 },  
 controller: emailTextEditingController,  
 style: simpleTextStyle(),  
 decoration: textFieldInputDecoration("Email"),  
 ),  
 TextFormField(  
 obscureText: true,  
 validator: (val) {  
 return val.length < 6 ? "Please enter password greater then 6" : null;  
 },  
 controller: passwordTextEditingController,  
 style: simpleTextStyle(),  
 decoration: textFieldInputDecoration("Password"),  
 )  
 ],  
 ),  
 ),  
 SizedBox(height: 8,),  
 Container(  
 alignment: Alignment.*centerRight*,  
 child: Container(  
 padding: EdgeInsets.symmetric(horizontal: 16, vertical: 8),  
 child: Text("Forgot Password?", style: simpleTextStyle(),),  
 ),  
 ),  
 SizedBox(height: 8,),  
 GestureDetector(  
 onTap: () {  
 // function to be executed when Sign Up button is clicked  
 signMeUp();  
 },  
 child: Container(  
 alignment: Alignment.*center*,  
 width: MediaQuery.*of*(context).size.width,  
 padding: EdgeInsets.symmetric(vertical: 20),  
 decoration: BoxDecoration(  
 gradient: LinearGradient(  
 colors: [  
 const Color(0xff007EF4),  
 const Color(0xff2A75BC)  
 ]  
 ),  
 borderRadius: BorderRadius.circular(30)  
 ),  
 child: Text("Sign Up", style: mediumTextStyle(),),  
 ),  
 ),  
 SizedBox(height: 16,),  
// Container(  
// alignment: Alignment.center,  
// width: MediaQuery.of(context).size.width,  
// padding: EdgeInsets.symmetric(vertical: 20),  
// decoration: BoxDecoration(  
// color: Colors.white,  
// borderRadius: BorderRadius.circular(30)  
// ),  
// child: Text("Sign Up with Google", style: TextStyle(  
// color: Colors.black,  
// fontSize: 17  
// ),  
// ),  
// ),  
 SizedBox(height: 16,),  
 Row(  
 mainAxisAlignment: MainAxisAlignment.center,  
 children: [  
 Text("Have an account?", style: mediumTextStyle(),),  
 GestureDetector(  
 onTap: () {  
 widget.toggle();  
 },  
 child: Container(  
 padding: EdgeInsets.symmetric(vertical: 8),  
 child: Text("Register now", style: TextStyle(  
 color: Colors.*white*,  
 fontSize: 17,  
 decoration: TextDecoration.*underline*,  
 ),  
 ),  
 ),  
 ),  
 ],  
 ),  
 SizedBox(height: 50,),  
 ],  
 ),  
 ),  
 ),  
 ),  
 );  
 }  
}

Sign In Screen code

Widget build(BuildContext context) {  
 return Scaffold(  
 appBar: appBarMain(context),  
 body: SingleChildScrollView(  
 child: Container(  
 alignment: Alignment.*bottomCenter*,  
 child: Container(  
 height: MediaQuery.*of*(context).size.height-50,  
 padding: EdgeInsets.symmetric(horizontal: 24),  
 child: Column(  
 mainAxisAlignment: MainAxisAlignment.end,  
 children: [  
 Form(  
 key: formKey,  
 child: Column(children: [  
 TextFormField(  
 validator: (val) {  
 return RegExp(r"^[a-zA-Z0-9,a-zA-Z0-9.!#$%&'\*+-/=?^\_`{|}~]+@[a-zA-Z0-9]+\.[a-zA-Z]+").hasMatch(val) ? null : "Enter a valid email";  
 },  
 controller: emailTextEditingController,  
 style: simpleTextStyle(),  
 decoration: textFieldInputDecoration("Email"),  
 ),  
 TextFormField(  
 obscureText: true,  
 validator: (val) {  
 return val.length < 6 ? "Please enter password greater then 6" : null;  
 },  
 controller: passwordTextEditingController,  
 style: simpleTextStyle(),  
 decoration: textFieldInputDecoration("Password"),  
 ),  
 ],),  
 ),  
 SizedBox(height: 8,),  
 Container(  
 alignment: Alignment.*centerRight*,  
 child: Container(  
 padding: EdgeInsets.symmetric(horizontal: 16, vertical: 8),  
 child: Text("Forgot Password?", style: simpleTextStyle(),),  
 ),  
 ),  
 SizedBox(height: 8,),  
 GestureDetector(  
 onTap: () {  
 signMeIn();  
 },  
 child: Container(  
 alignment: Alignment.*center*,  
 width: MediaQuery.*of*(context).size.width,  
 padding: EdgeInsets.symmetric(vertical: 20),  
 decoration: BoxDecoration(  
 gradient: LinearGradient(  
 colors: [  
 const Color(0xff007EF4),  
 const Color(0xff2A75BC)  
 ]  
 ),  
 borderRadius: BorderRadius.circular(30)  
 ),  
 child: Text("Sign In", style: mediumTextStyle(),),  
 ),  
 ),  
 SizedBox(height: 16,),  
// Container(  
// alignment: Alignment.center,  
// width: MediaQuery.of(context).size.width,  
// padding: EdgeInsets.symmetric(vertical: 20),  
// decoration: BoxDecoration(  
// color: Colors.white,  
// borderRadius: BorderRadius.circular(30)  
// ),  
// child: Text("Sign In with Google", style: TextStyle(  
// color: Colors.black,  
// fontSize: 17  
// ),  
// ),  
// ),  
 SizedBox(height: 16,),  
 Row(  
 mainAxisAlignment: MainAxisAlignment.center,  
 children: [  
 Text("No account? ", style: mediumTextStyle(),),  
 GestureDetector(  
 onTap: () {  
 widget.toggle();  
 },  
 child: Container(  
 padding: EdgeInsets.symmetric(vertical: 8),  
 child: Text("Register now", style: TextStyle(  
 color: Colors.*white*,  
 fontSize: 17,  
 decoration: TextDecoration.*underline*,  
 ),  
 ),  
 ),  
 ),  
 ],  
 ),  
 SizedBox(height: 50,),  
 ],  
 ),  
 ),  
 ),  
 ),  
 );  
 }  
}

Firebase functions

import 'package:cloud\_firestore/cloud\_firestore.dart';  
  
class DatabaseMethods{  
 getUserByUsername(String username) async {  
 return await Firestore.*instance*.collection("users").where("name", isEqualTo: username).getDocuments();  
 }  
  
 getUserByEmail(String userEmail) async {  
 return await Firestore.*instance*.collection("users").where("name", isEqualTo: userEmail).getDocuments();  
 }  
  
 uploadUserInfo(userMap) {  
 Firestore.*instance*.collection("users").add(userMap);  
 }  
  
 createChatRoom(String chatRoomId, chatRoomMap) {  
 Firestore.*instance*.collection("ChatRoom").document(chatRoomId).setData(chatRoomMap).catchError((e){  
 print(e.toString());  
 });  
 }  
  
 addConversationMessages(String chatRoomId, Map messageMap) {  
 Firestore.*instance*.collection("ChatRoom").document(chatRoomId).collection("chats").add(messageMap).catchError((e){print(e.toString());});  
 }  
  
 getConversationMessages(String chatRoomId) async {  
 print(chatRoomId);  
 return Firestore.*instance*.collection("ChatRoom").document(chatRoomId).collection("chats").orderBy("time").snapshots();  
 }  
  
 getChatRooms(String userName) async {  
 return Firestore.*instance*.collection("ChatRoom").where("users", arrayContains: userName).snapshots();  
 }  
}

Search for another friend code

class \_SearchScreenState extends State<SearchScreen> {  
  
 DatabaseMethods databaseMethods = new DatabaseMethods();  
 TextEditingController searchTextEditingController = new TextEditingController();  
  
 QuerySnapshot searchSnapshot;  
  
 Widget searchList() {  
 return searchSnapshot != null ? ListView.builder(  
 itemCount: searchSnapshot.documents.length,  
 shrinkWrap: true,  
 itemBuilder: (context, index) {  
 return SearchTile(  
 userName: searchSnapshot.documents[index].data["name"],  
 userEmail: searchSnapshot.documents[index].data["email"],  
 );  
 }  
 ):  
 Container();  
 }  
  
 initiateSearch() {  
 databaseMethods.getUserByUsername(searchTextEditingController.text).then((val) {  
 setState(() {  
 searchSnapshot = val;  
 });  
 });  
 }  
  
 */// create chatroom, send user to conversation screen, pushreplacement* createChatroomAndStartConversation({String userName}) {  
 print("${Constants.*myName*}");  
 if(userName != Constants.*myName*) {  
 String chatRoomId = getChatRoomId(userName, Constants.*myName*);  
  
 List<String> users = [userName, Constants.*myName*];  
 Map<String, dynamic> chatRoomMap = {  
 "users" : users,  
 "chatroomId": chatRoomId  
 };  
  
 // Below we are creating a chat Room  
 DatabaseMethods().createChatRoom(chatRoomId, chatRoomMap);  
 Navigator.*push*(context, MaterialPageRoute(  
 builder:(context) => ConversationScreen(  
 chatRoomId  
 )  
 ));  
 } else {  
 print("You cannot send message to yourself");  
 }  
  
  
 }  
  
 Widget SearchTile({String userName, final String userEmail}) {  
 return Container(  
 padding: EdgeInsets.symmetric(horizontal: 5, vertical: 10),  
 child: Row(  
 children: [  
 Column(  
 crossAxisAlignment: CrossAxisAlignment.start,  
 children: [  
 Text(userName, style: simpleTextStyle(),),  
 Text(userEmail, style: simpleTextStyle(),)  
 ]  
 ),  
 Spacer(),  
 GestureDetector(  
 onTap: () {  
 createChatroomAndStartConversation(userName: userName);  
 },  
 child: Container(  
 decoration: BoxDecoration(  
 color: Colors.*blue*,  
 borderRadius: BorderRadius.circular(30)  
 ),  
 padding: EdgeInsets.symmetric(horizontal: 10, vertical: 16),  
 child: Text("Message", style: mediumTextStyle(),),  
 ),  
 )  
 ],  
 ),  
 );  
 }  
  
 @override  
  
 void initState() {  
 super.initState();  
 }  
  
 Widget build(BuildContext context) {  
 return Scaffold(  
 appBar: appBarMain(context),  
 body: Container(  
 child: Column(  
 children: [  
 Container(  
 decoration: BoxDecoration(  
 color: Color(0x54FFFFFF),  
 borderRadius: BorderRadius.circular(20)  
 ),  
 padding: EdgeInsets.symmetric(horizontal: 24, vertical: 10),  
 child: Row(  
 children: [  
 Expanded(  
 child: TextField(  
 controller: searchTextEditingController,  
 style: TextStyle(  
 color: Colors.*white* ),  
 decoration: InputDecoration(  
 hintText: "Search username...",  
 hintStyle: TextStyle(  
 color: Colors.*white54* ),  
 border: InputBorder.*none* ),  
 )  
 ),  
 GestureDetector(  
 onTap: () {  
 initiateSearch();  
 },  
 child: Container(  
 height: 40,  
 width: 40,  
 decoration: BoxDecoration(  
 gradient: LinearGradient(  
 colors: [  
 const Color(0x36FFFFFF),  
 const Color(0x0FFFFFF),  
 ]  
 ),  
 borderRadius: BorderRadius.circular(40)  
 ),  
 padding: EdgeInsets.all(10),  
 child: Image.asset("assets/images/search.png")),  
 )  
 ],  
 ),  
 ),  
 searchList()  
 ],  
 ),  
 ),  
 );  
 }  
}  
  
getChatRoomId(String a, String b) {  
 if(a.substring(0, 1).codeUnitAt(0) > b.substring(0, 1).codeUnitAt(0)) {  
 return "$b\\_$a";  
 } else {  
 return "$a\\_$b";  
 }  
}

Custom widgets used throughout the app

import 'package:flutter/material.dart';  
  
Widget appBarMain(BuildContext context) {  
 return AppBar(  
 title: Image.asset("assets/images/logo.png", height: 50,),  
 );  
}  
  
InputDecoration textFieldInputDecoration(String hintText) {  
 return InputDecoration(  
 hintText: hintText,  
 hintStyle: TextStyle(  
 color: Colors.*white54*,  
 ),  
 focusedBorder: UnderlineInputBorder(  
 borderSide: BorderSide(color: Colors.*white*),  
 ),  
 enabledBorder: UnderlineInputBorder(  
 borderSide: BorderSide(color: Colors.*white*),  
 )  
 );  
}  
  
TextStyle simpleTextStyle() {  
 return TextStyle(  
 color: Colors.*white*,  
 fontSize: 17  
 );  
}  
  
TextStyle mediumTextStyle(){  
 return TextStyle(  
 color: Colors.*white*,  
 fontSize: 17  
 );  
}

Dependencies used

dependencies:  
 flutter:  
 sdk: flutter  
  
  
 *# The following adds the Cupertino Icons font to your application.  
 # Use with the CupertinoIcons class for iOS style icons.* cupertino\_icons: ^0.1.3  
  
 *# Below is the firebase auth added* firebase\_auth: ^0.16.0  
  
 *# The below is the firestore dependency* cloud\_firestore: ^0.13.5  
  
 *#* shared\_preferences: ^0.5.12+2

**Conclusion** – After completing the experiment, I have understood how to connect Flutter app to Firebase backend, how to fetch and store data on the backend server and also understood how to do dart programming.