Flutter Mobile App Assignment

SE 4010 – Current Trends in Software Engineering

Student ID	Name
IT 17 7064 38	N.Shriram

BSc. Special (Hons) Degree in Information Technology

Department of Software Engineering

Sri Lanka Institute of Information Technology

Sri Lanka

May 2020

Table of Contents

1. Introduction

We have developed a "Lunch Ping Partner App", which has a basic object of Pinging the list of contacts in a particular group with the Ping Message at the Time we have set the Message to be set.

Basic flow of the Application is as follows, User will be able to Login to the application using Google authentication, and once login is successful user will be directed to the Menu page of the Application. Here user can create a new Group by providing mandatory fields.
Later user can update the Group with list of contacts that are fetched from the Smart phone and also for every group user can provide Days on which Message needs to be sent.
Additionally User can give priority value for groups based on which Groups will be ordered in the List view. User also can mark a group as favorite so that selected group will be filtered out and shown in the Favorite group tab control.

Technologies used to implement Application

- Flutter
- Material UI
- Firebase
- Google Authentication

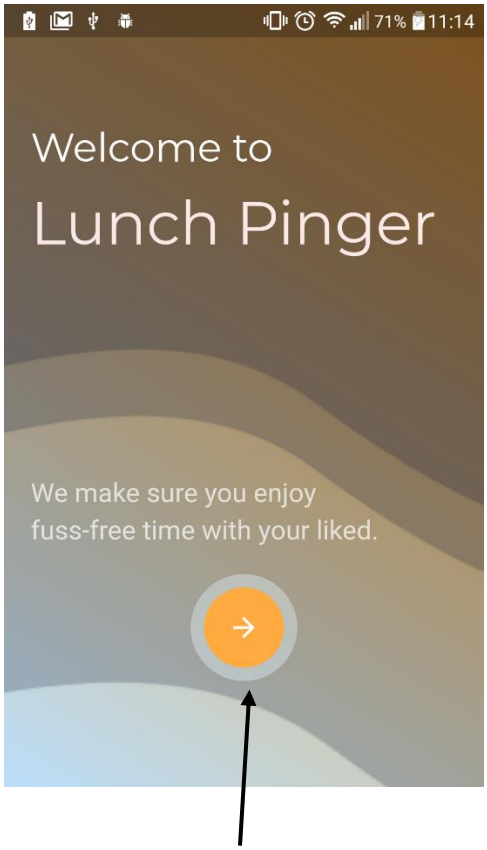
2. Individual Components

This Individual implementation consist of following functionalities:

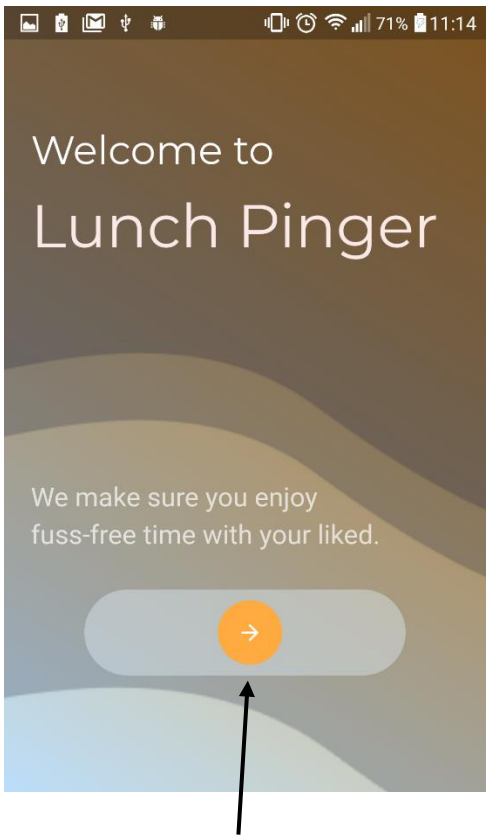
- Splash Landing Screen Implementation for the Application
- Login Screen Implementation for the Application
- Google Configuration and Firebase mapping implementation for Authentication
- Application Home Menu page with Logged in User info display
- All Groups listing and Favorite group listing with Flutter Tab control
- Ordering Groups based on the priority value assigned for the Group
- Filter out Favorite Groups and List them Separately
- Create new Group with basic mandatory fields linked with Firestore
- Update Group with appropriate values that linked with Firestore
- Delete particular group from list that is stored in Firestore
- Display appropriate message to User after completion
- Prompt user to confirm processing highly critical operation with Flutter Dialog

3. Individual Functionality Descriptions

3.1 Splash Screen



1. Click on Arrow button

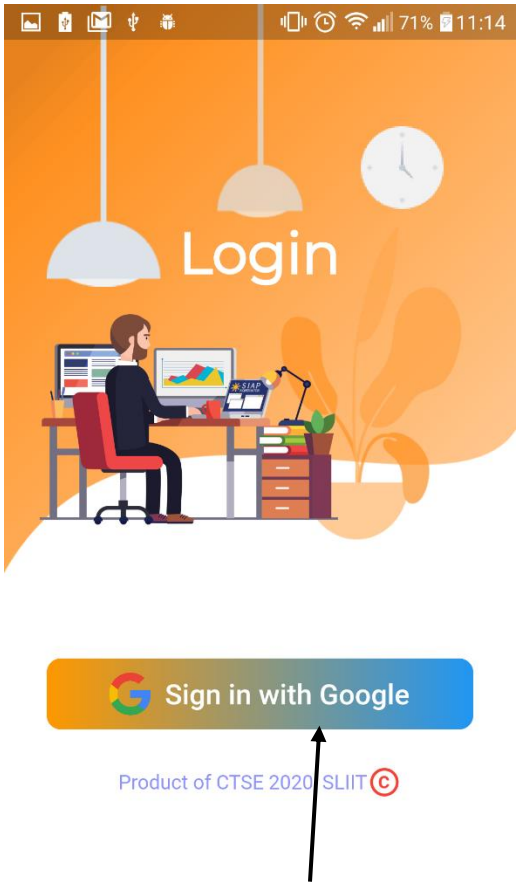


2. Animation swipes the button

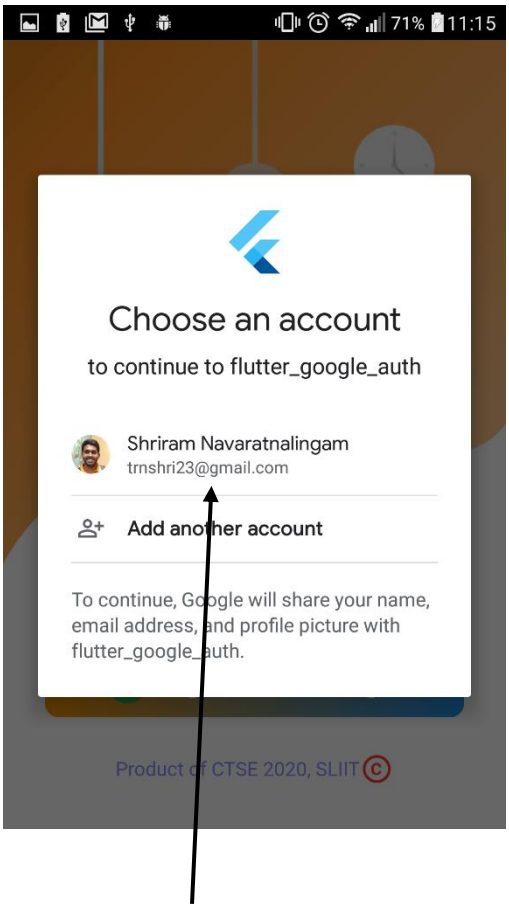
This is the Landing page of the Application, where we have implemented the Animation which is one of the Flutter feature.

This UI contains the Name of the Application with the Welcome note and a small moto for the application that illustrate the primary purpose of this application.

3.2 Google Authentication



1. Click on the Google Sign In button



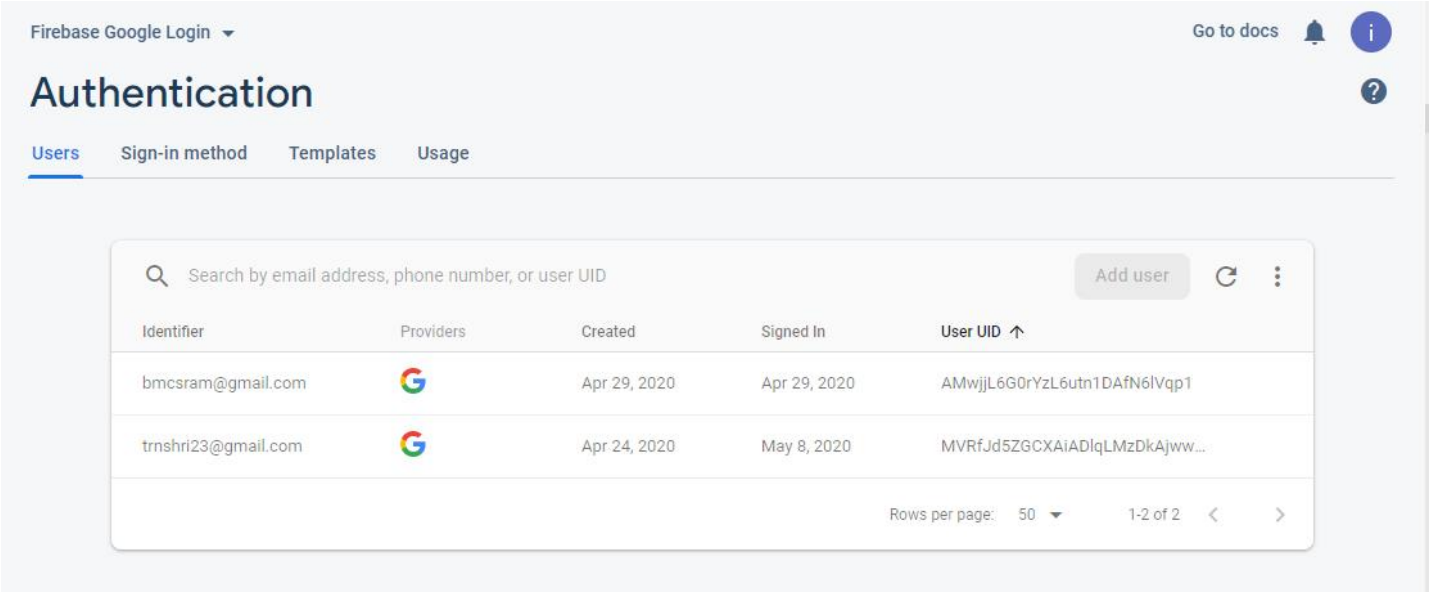
2. Select the Google Account that to be used

Once user clicks the Landing page User will be redirected to the Landing page where we have one main user interaction button to enable user authentication using the Google Authentication.

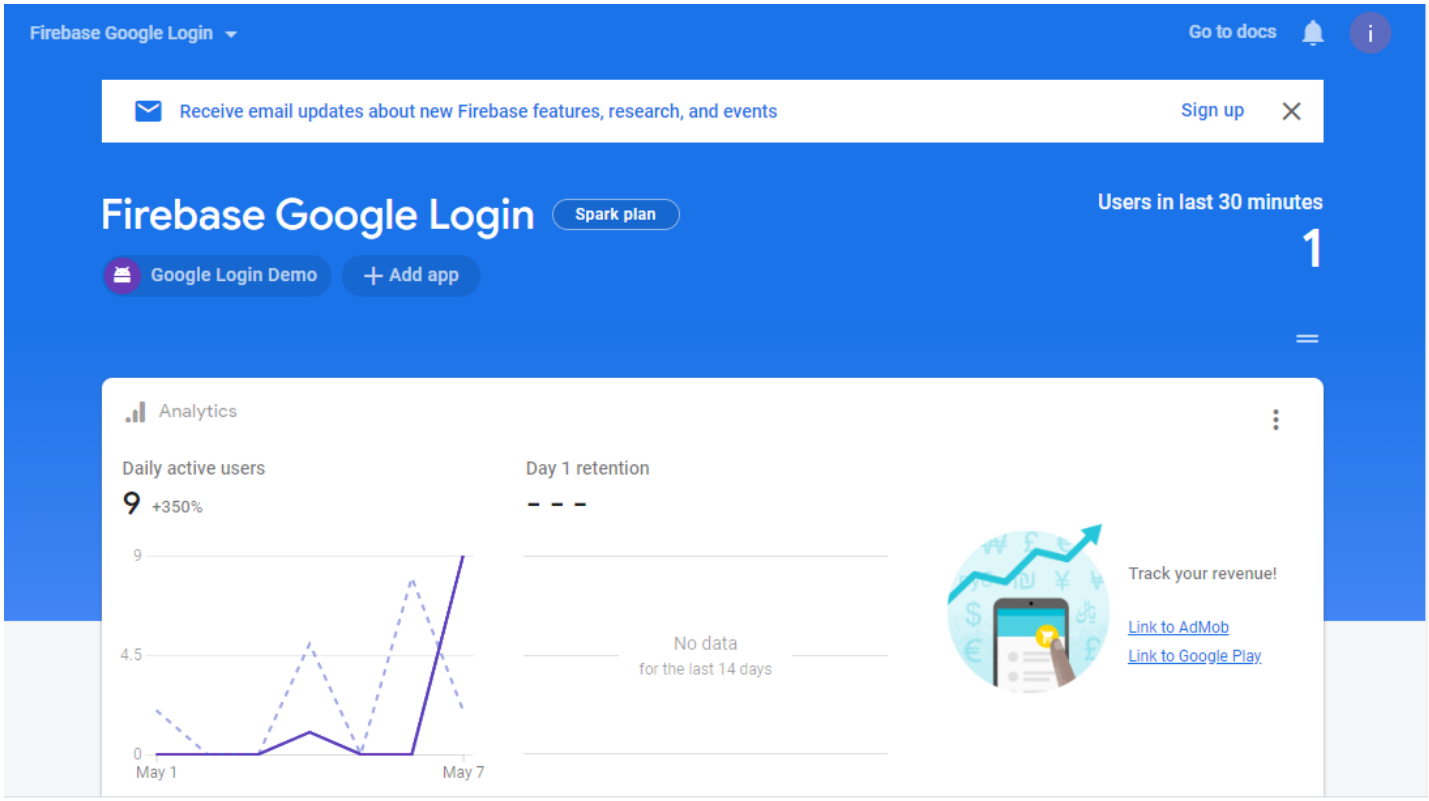
3.2.1 Enhance User Experience via Google Authentication

Main purpose of using Google authentication instead of regular sign in is to enhance the user experience. This enable user to start using the application as quick as possible without delaying by filling the form in order to get registered separately and later login with the credentials.

This effectively reduce the overhead end user having while they start using and make the application more user friendly.

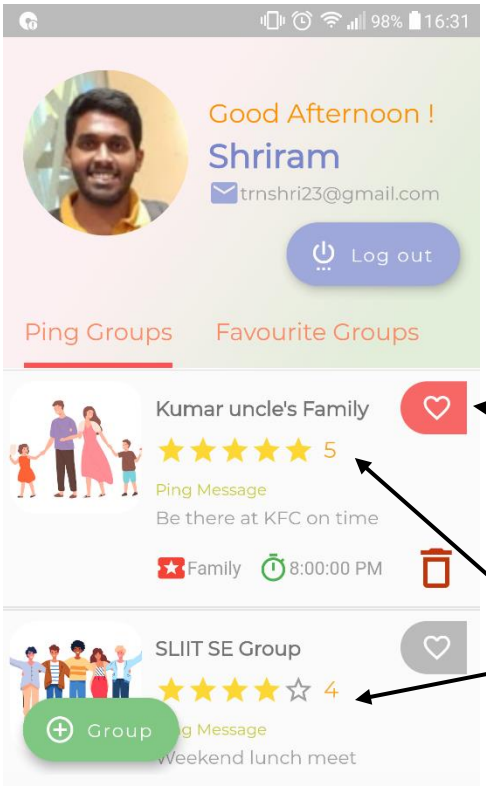


1. Soon after new user registered into Application the user record will be included firebase backed User Authentication



2. Reference of User interaction with the Application based on Authentication

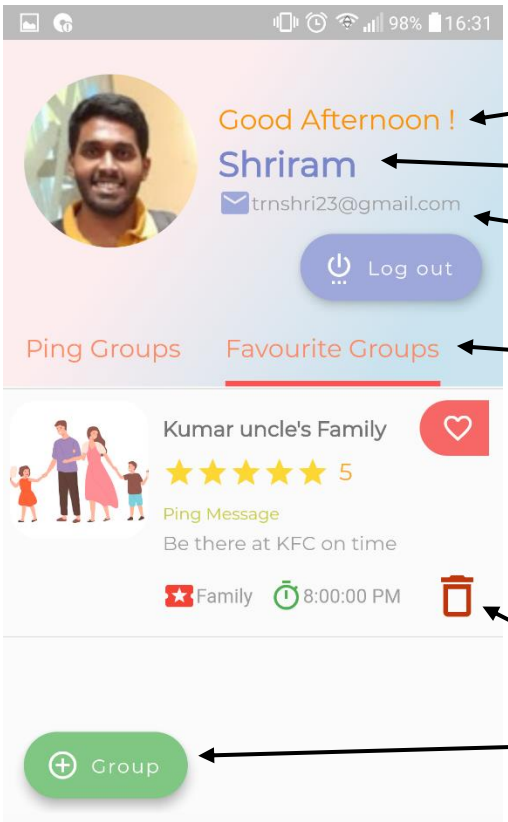
3.3 Menu Page (**Logged In User Display and Group Listing implementation**)



Once User is successfully logged in User will be redirected to the Menu page of the Application. Which displays the logged in user information and listing of the existing groups list view.

Displays red colored favorite icon if group is marked as favorite

Ordering group done dynamically based on the priority given to the group



Greeting for User based on current time

User Name fetched from the Google Auth

Email fetched from the Google Auth

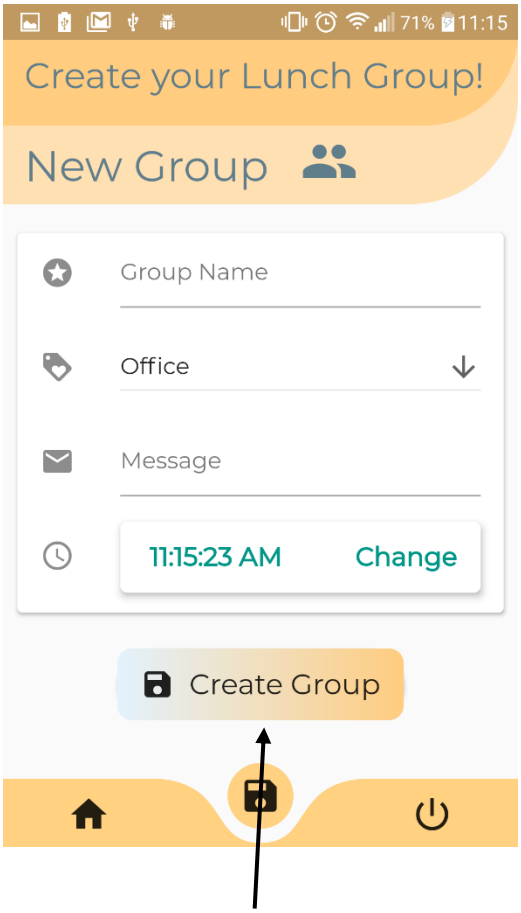
Customized listing for Favorite Groups only

Flutter tab feature is used to swipe between the listing, where one displays all the groups and other filter out the favorite groups and list them separately

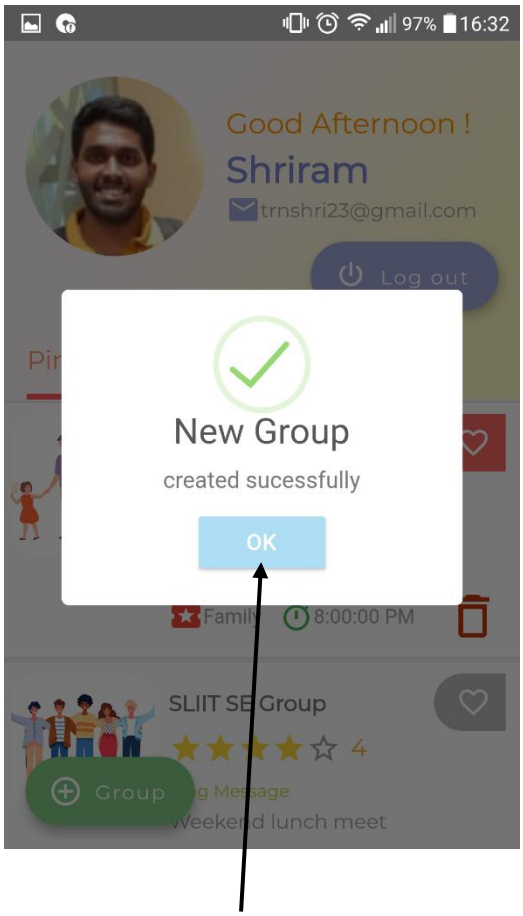
Delete Group Button

New Group Create button

3.4 Create New Group



1. After filling above mandatory fields click create button

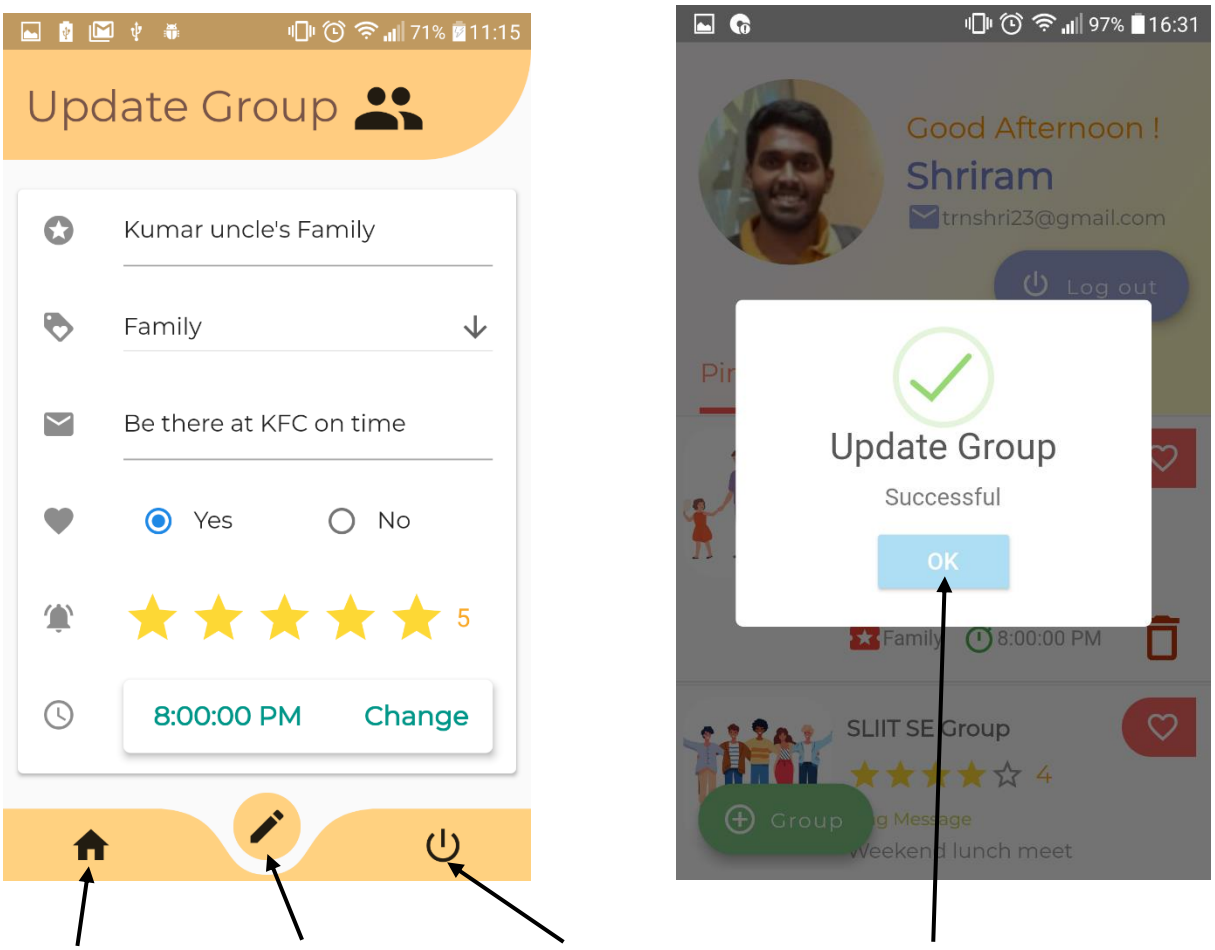


2. Success message after saving

Once user clicked the Create group button on the menu page, User will be redirected to the Create Group page. Where user will be requested to fill only main 4 mandatory fields to create the group.

Once User successfully created the group a Success message will be displayed to acknowledge the user group has been created. This process flow is used to enhance the user experience.

3.5 Update Group

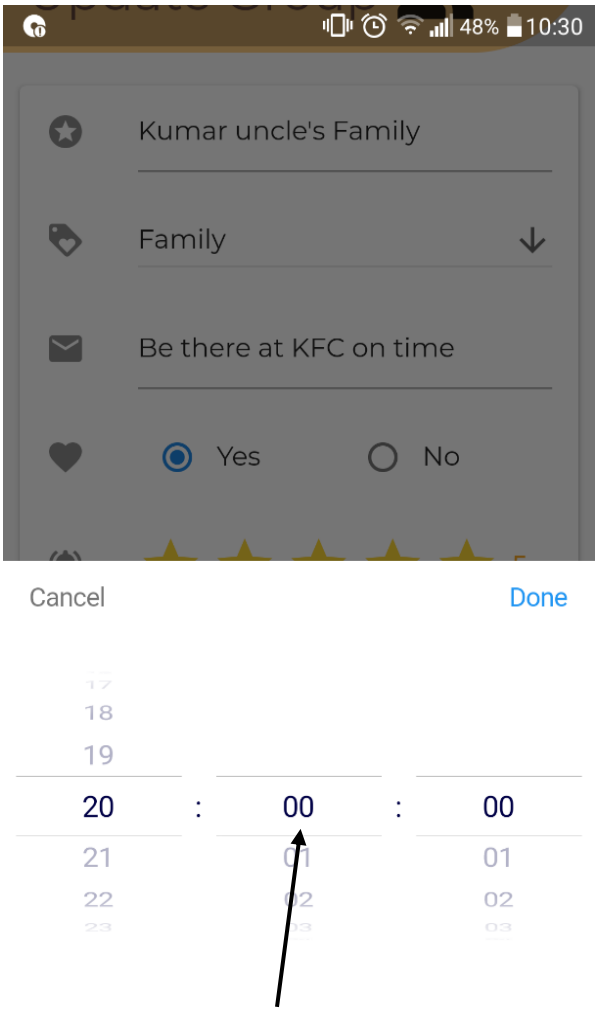


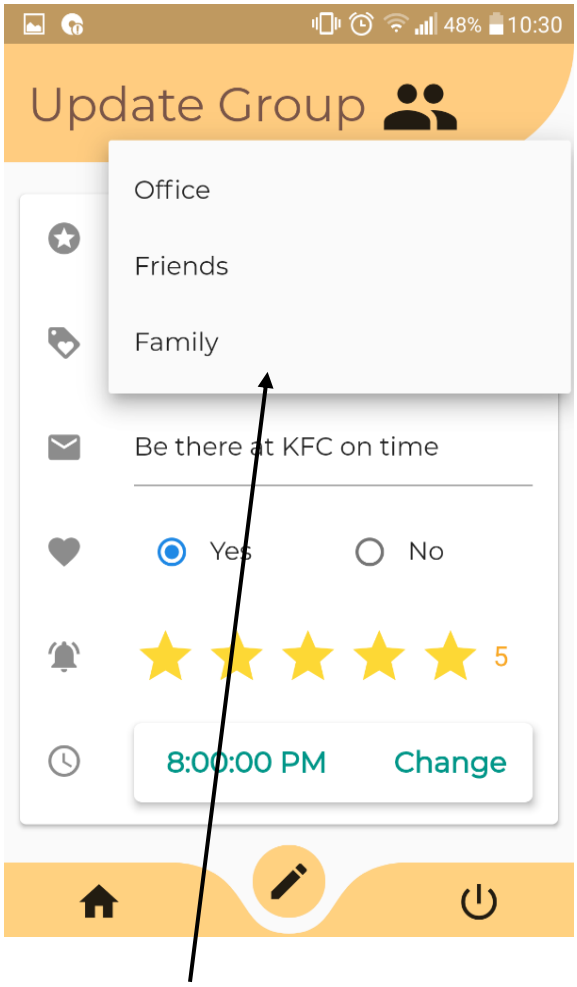
1. Home button 2.Update button 3. Log off button 4. Update success message

Once user clicked any one of the Groups that are listed on the Menu page, user will be directed to the Group update page along with the Group data relevant to particular selection.

User is allowed to Change Group Name, Ping message, type of the Group, priority to the group and allow to mark the group as favorite or not.

Once User successfully updated the group user will be acknowledged with the Success message for better user experience.





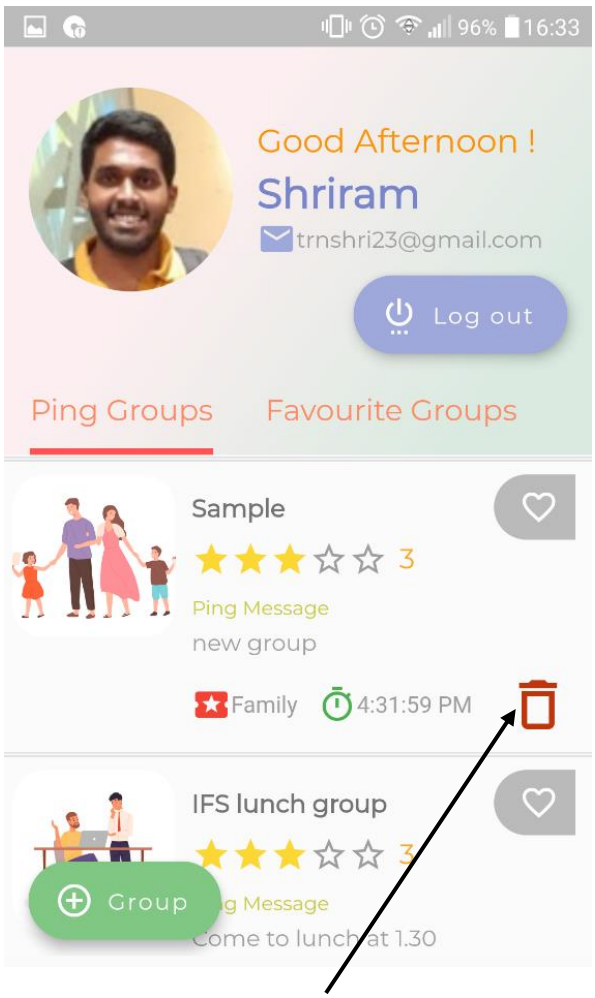
1. Time picker flutter feature

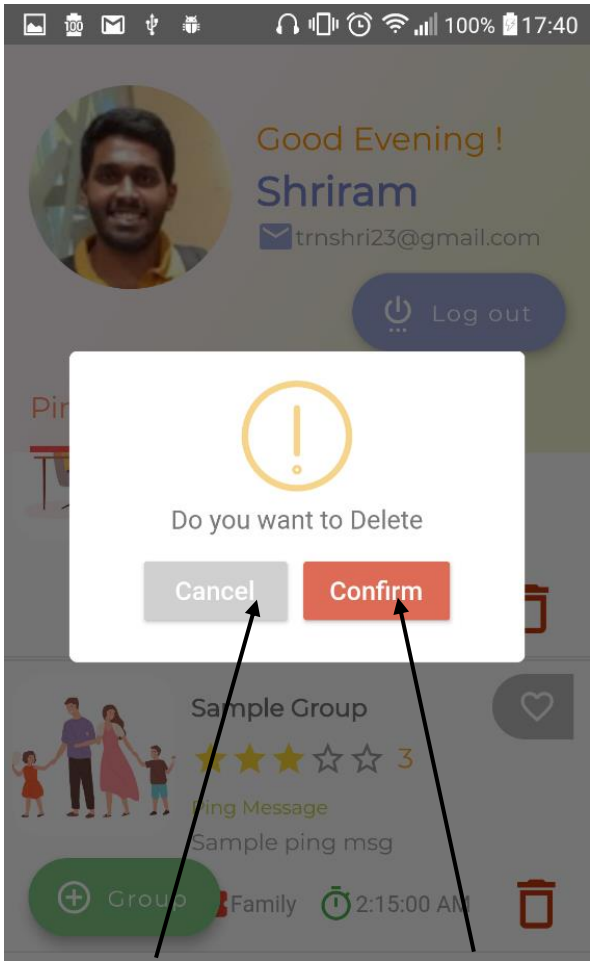
2. Drop down list to select type

In both the Group creation scenario and the Group update scenario, we have included Flutter drop down feature and Time picking feature.

We have used drop down in order to select the type of the Group we are creating based on which image will be displayed in the Group listing. Time Picker is used in selecting the message pinging time for that particular group. Through this inclusion we allow user to efficiently input information rather than manually typing the type and time which may also result in errors.

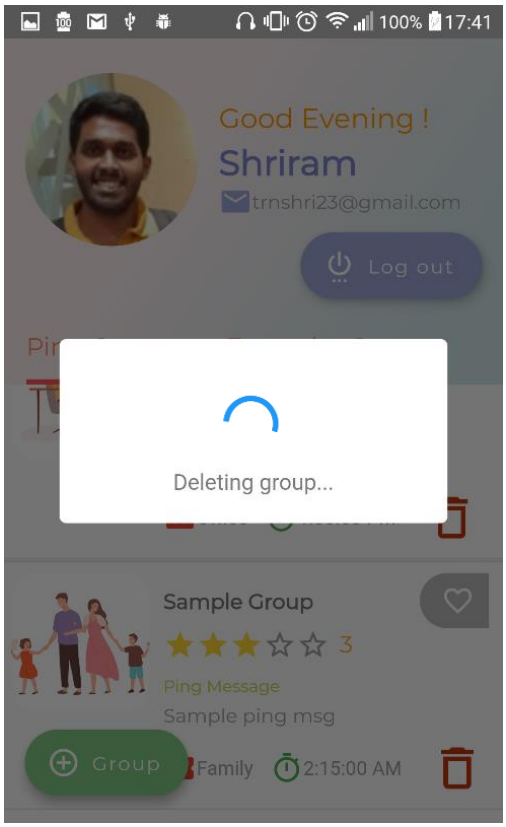
3.6 Delete Group



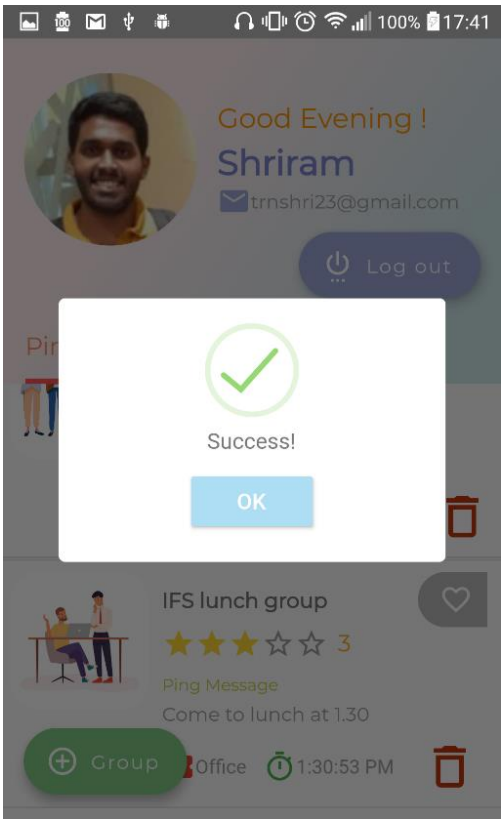


1. Delete button for particular group 2. Cancel the Delete 3. Confirm Delete

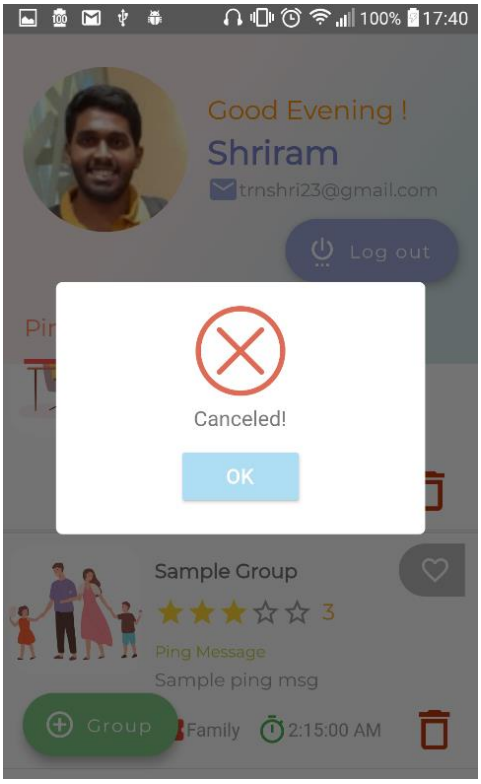
User can delete a particular group that is listed in the Menu page by clicking the Bin icon. Since Delete operation is preserved operation, user will be prompted a message to confirm the delete operation. Based on the User response to the prompt user will be acknowledged with the appropriate message for enhanced user experience.



1. After confirm operation



2. Delete Success Message



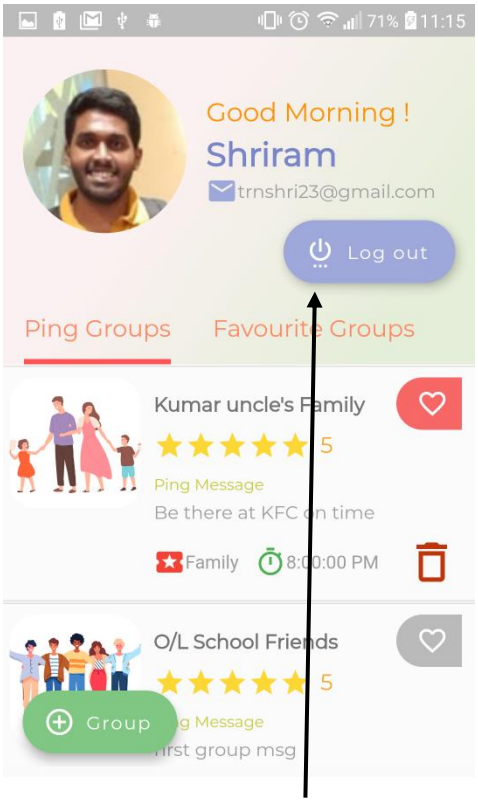
3. If Cancelled

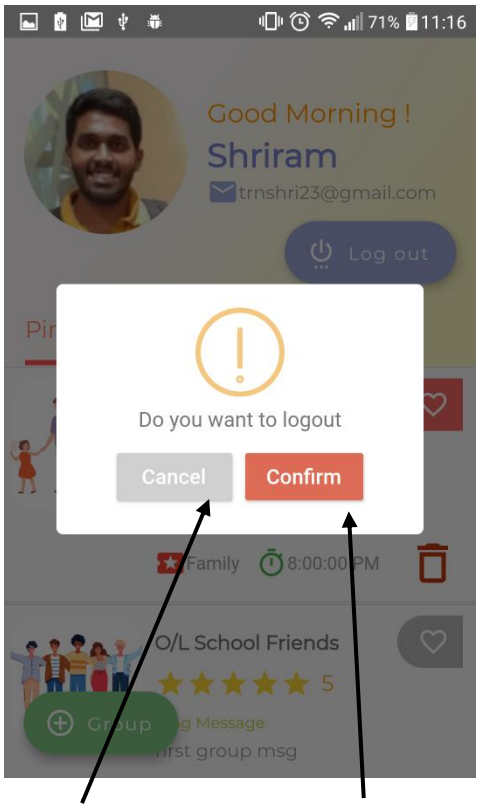
In order to enhance the experience of the User we have included Firebase Dialog box notification to acknowledge the result of the operation they have requested.

In this delete Group scenario, we have enabled an implementation to confirm the delete operation after user clicks delete icon.

By this feature we have improved the User experience of the Application.

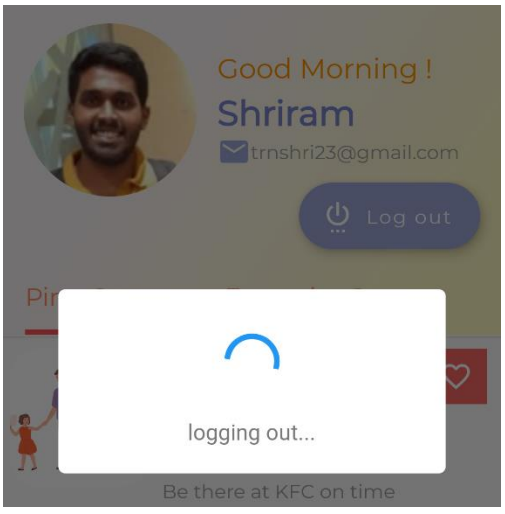
3.7 Logout from Application

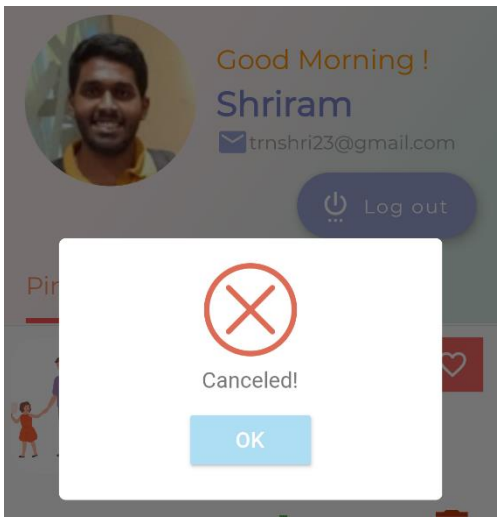




1. User Logout Button 2. Cancel Logout 3. Confirm Logout

User can Logout from the Application by Clicking the Log out button, once user select this option user will be requested to confirm the operation since logout is one of the preserved button. If user confirms user will be logged out from the application and redirected to Application Landing page else if cancelled user will be acknowledged with cancelled message.





1. If User confirms Logout 2. If User cancelled Logout

4. Appendix

```
//Group.dart
//This is the Core Model class that is used as Data Transfer object.
//In Order to manipulate and carry data along the Pages and Widgts

class Group{

  String groupName;
  String type;
  String message;
  DateTime pingTime;
  bool isFavourite;
  int priority;
  DocumentReference reference;

  //Constructor used to instantiate object when creating

Group.setValue({this.groupName,this.type,this.message,this.pingTime,this.isFavourite,
this.priority});

  //Constructor that used to validate fields and assign to instance properties
  //Inorder to instantiate Group instance

  Group.fromMap(Map<String,dynamic>map,{this.reference}){
    assert(map['groupName']!=null);
    assert(map['type']!=null);
    this.groupName = map['groupName'];
    this.type = map['type'];
    this.message = map['message'];
    this.pingTime = (map['pingTime'] as Timestamp).toDate();
    this.isFavourite = (map['isFavourite']==null)?false:map['isFavourite'];
    this.priority = (map['priority']==null)?0: map['priority'];
  }
 //Constructor called from GroupListItem by passing document snapshot to create an
instance of Group
  Group.fromSnapshot(DocumentSnapshot
snapshot):this.fromMap(snapshot.data,reference:snapshot.reference);

//Method used to get the Map<String,dynamic> type from the object properties
  toJson(){
    return{
      "groupName":groupName,
      "type":type,
      "message":message,
      "pingTime":pingTime,
      "isFavourite":isFavourite,
```

```
      "priority":priority
    };
  }
}

// FirestoreApi.dart
// Repository class that act as the connection point to the Firebase

class FirestoreApi{

  //Create method that is used when a Group need to be created.
   createGroup(Group group){
    try{
      Firestore.instance.runTransaction((Transaction transaction) async {
        await Firestore.instance.collection("groups").document()
            .setData(group.toJson());
      });
    }
    catch(e){
      print(e.toString());
    }
  }
  //Read Stream that takes boolean parameter to determine,
   // whether to return All the groups or favourite groups
  Stream<QuerySnapshot> getStream(bool isFavouriteList){

     if(isFavouriteList)
       return Firestore.instance.collection("groups").where('isFavourite',isEqualTo:
true).orderBy('priority',descending: true).snapshots();
     else
       return Firestore.instance.collection("groups").orderBy('priority',descending:
true).snapshots();
  }

  //Method used update particular method with the new Map data
  updateGroup(Group group, Map<String,dynamic> newData){

     try{
       Firestore.instance.runTransaction((Transaction transaction) async{
         await transaction.update(group.reference, newData);
       });
     }
     catch(e){
       print(e.toString());
     }
  }
```

```
  //Method used to delete the Group pointed by the reference number
  deleteGroup(Group group){
     Firestore.instance.runTransaction(
       (Transaction transaction) async{
         await transaction.delete(group.reference);
       }
     );
  }
}

//GroupMenu.dart
//Main menu page of the Application
//Which displays the User logged in information with list of Groups

import '../main.dart';
import '../GoogleSignIn.dart';
import 'Group/GroupListPage.dart';

//GroupMenu StatefulWidgt definition
class GroupMenu extends StatefulWidgt {

  //Firebase Auth user instance that is passed from Google Firebase
Authentication
  final FirebaseUser user;
  GroupMenu({Key key, @required this.user}) : super(key: key);

  //Define State for the GroupMenu Stateful Widgt
  @override
  _MyHomePageState createState() => _MyHomePageState();
}

//Define State class for the roupMenu Stateful Widgt
class _MyHomePageState extends State<GroupMenu>
    with SingleTickerProviderStateMixin {

  //Tab controller instance used to swipe between all group list and
favourite group list
  TabController tabController;


  //define init lifecycle method to instantiate tab controller with
particular length
  @override
  void initState() {
    super.initState();
```

```
    tabController = TabController(vsync: this, length: 2);
  }

  //Override build method to return ListView
  //Where 1st part used to display Logged in users
  //2nd Part used to List the Groups with Tab controller
  @override
  Widgt build(BuildContext context) {
    return Scaffold(
      body: ListView(
        children: <Widgt>[
          Stack(
            children: <Widgt>[
              Container(
                height: 250.0,
                width: double.infinity,
                child: ControlledAnimation(
                  startPosition: 0.2,
                  playback: Playback.MIRROR,
                  tween: tween,
                  duration: tween.duration,
                  builder: (context, animation) {
                    return Container(
                      decoration: BoxDecoration(
                          gradient: LinearGradient(
                              begin: Alignment.topCenter,
                              colors: [animation["color1"],
animation["color2"]])),
                    );
                  },
                ),
              ),
              Positioned(
                top: 140.0,
                right: 15.0,
                child: FloatingActionButton.extended(
                  heroTag: "btn2",
                  label: Text('Log out',style: TextStyle(fontFamily:
"Montserrat",),),),
                  backgroundColor: Colors.indigo.shade200,
                  icon: Icon(Icons.settings_power),
                  onPressed: () {
                    SweatAlert.show(context,
                        subtitle: "Do you want to logout",
```

```
                        style: SweatAlertStyle.confirm,
                        showCancelButton: true, onPress: (bool isConfirm) {
                          if(isConfirm){
                            SweatAlert.show(context,subtitle: "logging
out...", style: SweatAlertStyle.loading);
                            new Future.delayed(new Duration(seconds: 2),(){
                              signOutGoogle();

Navigator.of(context).pushAndRemoveUntil(MaterialPageRoute(builder: (context)
{return HomePage();}), ModalRoute.withName('/'));
                              SweatAlert.show(context,subtitle: "Success!",
style: SweatAlertStyle.success);
                            });
                          }else{
                            SweatAlert.show(context,subtitle: "Canceled!",
style: SweatAlertStyle.error);
                          }
                          // return false to keep dialog
                          return false;
                        });
                    },
                  ),
                ),
                Positioned(
                  top: 25.0,
                  left: 15.0,
                  child: Row(
                    children: <Widgt>[
                      Container(
                        height: 125.0,
                        width: 125.0,
                        decoration: BoxDecoration(
                            borderRadius: BorderRadius.circular(62.5),
                            image: DecorationImage(
                                image: NetworkImage(Widgt.user.photoUrl,),
                                fit: BoxFit.cover
                            )
                        ),
                      ),
                      SizedBox(width: 15.0),
                      Column(
                        crossAxisAlignment: CrossAxisAlignment.start,
                        children: <Widgt>[
                          greetUser(),
```

```
                        SizedBox(height: 5.0),
                        Text(
                          Widgt.user.displayName.split(" ")[0],
                          style: TextStyle(
                              fontFamily: "Montserrat",
                              fontSize: 25.0,
                              fontWeight: FontWeight.bold,
                              color: Colors.indigo.shade300),
                        ),
                        SizedBox(height: 1.0),
                        Row(
                          children: <Widgt>[
                            Icon(Icons.email, color: Colors.indigo.shade200),
                            Container(
                              width: MediaQuery.of(context).size.width -
175.0,
                              child: Text(
                                Widgt.user.email,
                                style: TextStyle(
                                    fontFamily: "Montserrat",
                                    fontSize: 14.0, color: Colors.grey),
                              ),
                            )
                          ],
                        )
                      ],
                    )
                  ],
                ),
              ),
              Padding(
                padding: EdgeInsets.only(top:200.0),
                child: TabBar(
                  controller: tabController,
                  indicatorColor: Colors.redAccent,
                  indicatorSize: TabBarIndicatorSize.label,
                  indicatorWeight: 4.0,
                  labelColor: Color(0xFFFE6E22),
                  unselectedLabelColor: Colors.grey,
                  isScrollable: true,
                  tabs: <Widgt>[
                    Tab(
                      child: Text(
                        'Ping Groups',
```

```
                    style: TextStyle(
                      fontFamily: "Montserrat",
                      fontSize: 18.0,
                      //fontWeight: FontWeight.bold,
                      color: Colors.deepOrange.shade300
                    ),
                  ),
                ),
                Tab(
                  child: Text(
                    'Favourite Groups',
                    style: TextStyle(
                        fontFamily: "Montserrat",
                        fontSize: 18.0,
                        //fontWeight: FontWeight.bold,
                        color: Colors.deepOrange.shade300
                    ),
                  ),
                )
              ],
            ),
          )
        ],
      ),
      Container(
        height: MediaQuery.of(context).size.height - 285.0,
        color: Colors.white24,
        child: TabBarView(
          controller: tabController,
          children: <Widgt>[
            new GroupListPage(isFavouriteList: false,),
            new GroupListPage(isFavouriteList: true,)
          ],
        ),
      )
    ],
  ),
 );
}
```

```dart
// GroupListPage.dart
// Class used for listing of Groups
// Here we pass true/false to constructor of the Widgt in order to determine whether
to list all the groups or favourite groups

import 'CreateGroup.dart';
import 'GroupListItemPage.dart';

//GuessYouLikePage StatefulWidgt instantiation
class GroupListPage extends StatefulWidgt {

  //Property passed from parent inorder to determine whether favourite group listing
or not
  final bool isFavouriteList;
  GroupListPage({Key key,this.isFavouriteList}):super(key:key);

  //Override createstate to define the state for the GroupListPage
  @override
  _GroupListPageState createState() => _GroupListPageState();
}

//State class definition for GroupListPage
class _GroupListPageState extends State<GroupListPage> {

  //Instantiate the instance of FirestoreApi which act as the Repository class to
connect with Firebase
  FirestoreApi api = new FirestoreApi();

  //Override build which returns StreamBuilder
  @override
  Widgt build(BuildContext context) {

    //Set the Stream to Stream<QuerySnapshot> to retrive the list of groups
    return StreamBuilder(
      stream: api.getStream(Widgt.isFavouriteList),
      builder: (context,snapshot){

        //Check whether there is any error in the Snapshot
        if(snapshot.hasError)
          return Text('Error ${snapshot.error}');

        //Check if snapshot has data, if not return Circular progress indicator
        if(!snapshot.hasData)
          return SizedBox(
              height: 150.0,
              width: 150.0,
              child:
              Container(
```

```
                height: 50,
                width: 50,
                child: CircularProgressIndicator(
                    valueColor: AlwaysStoppedAnimation(Colors.blue),
                    strokeWidth: 5.0),
              )
          );
        //Return a List if data exist
        else
          return buildList(context,snapshot.data.documents);

      },
    );
  }
}

// Define a Widgt that returns Stack that includes a ListView and Floating button.
Widgt buildList(BuildContext context,List<DocumentSnapshot> data){

  //Return a Stack view
  return Stack(
    children: <Widgt>[
      //List view that return List of Groups
      ListView(
        children: data.map((doc){
          return Column(
            children: <Widgt>[
              new Container(
                  height: 2,
                  decoration: BoxDecoration(
                    border: Border(
                        bottom: BorderSide( color: Colors.grey.shade300)
                    ),
                  )
              ),
              GroupListItemPage(key: Key(doc.documentID) ,documentSnapshot: doc,),
              new Container(
                  height: 2,
                  decoration: BoxDecoration(
                    border: Border(
                        bottom: BorderSide( color: Colors.grey.shade300)
                    ),
                  )
              )
            ],
          );
        }).toList(),
      ),
```

```
      //Return a Floating button to add a new group
      Container(
        child: Positioned(
          bottom: 10.0,
          left: 15.0,
          child: FloatingActionButton.extended(
            heroTag: "btn1",
            label: Text('Group',style: TextStyle(fontFamily: "Montserrat",),),),
            backgroundColor: Colors.green.shade300,
            icon: Icon(Icons.add_circle_outline),
            onPressed: () {
              Navigator.of(context).push(MaterialPageRoute(
                  builder: (BuildContext context){
                    return CreateGroup();
                  }
              ));
            },
          ),
        ),
      )
    ],
  );
}


//GroupListItemPage.dart
//Class that is used to Define individual Group item, based on the Group object
passed from the parent

import 'UpdateGroup.dart';

//Class definition of GroupListItemPage stateful Widgt
class GroupListItemPage extends StatefulWidgt{

  Group group;
  //Constructor that is used to instatiate using the Group object passed from the
parent
  GroupListItemPage({Key key,documentSnapshot}):super(key:key){
    group = Group.fromSnapshot(documentSnapshot);
  }

  //override createState method that is used to define the State of the Widgt
  @override
  State<StatefulWidgt> createState() => new _GroupListItemPageState();
}

//Define state class of the GroupListItemPage Widgt
class _GroupListItemPageState extends State<GroupListItemPage>{
```

```
  String imgPath;

  //Widgt lifecycle method that is used to define the path of the image based on the
Type of the group
  @mustCallSuper
  void initState() {
    imgPath = (Widgt.group.type=='Office') ?
    'assets/images/officeIcon.png' : ((Widgt.group.type=='Friends')?
'assets/images/friendsIcon.png' : 'assets/images/familyIcon.png');

  }

  //override build method that returns the list item structure, with the Group object
properties
  @override
  Widgt build(BuildContext context) {
    return Padding(
      key: Widgt.key,
      padding: EdgeInsets.symmetric(vertical:8,horizontal: 5),
        child: InkWell(
          onTap: (){

            Navigator.of(context).push(MaterialPageRoute(
                builder: (context){
                  return UpdateGroup(group: Widgt.group,);
                }
            ));
          },
          child: Row(
            crossAxisAlignment: CrossAxisAlignment.start,
            children: <Widgt>[
              Container(
                height: 100.0,
                width: 100.0,
                decoration: BoxDecoration(
                    borderRadius: BorderRadius.circular(20.0),
                    image: DecorationImage(
                        image: AssetImage(imgPath),
                        fit: BoxFit.cover)),
              ),
              SizedBox(width: 10.0),
              Column(
                crossAxisAlignment: CrossAxisAlignment.start,
                children: <Widgt>[
                  Container(
                      width: MediaQuery.of(context).size.width - 125.0,
                      child: Row(
```

```
                        mainAxisAlignment: MainAxisAlignment.spaceBetween,
                        children: <Widgt>[
                          Text(
                            Widgt.group.groupName,
                            style: TextStyle(
                                fontFamily: "Montserrat",fontSize: 15.0, fontWeight:
FontWeight.bold,color: Colors.black54),
                          ),
                          Container(
                            padding: EdgeInsets.only(left: 5.0),
                            height: 40.0,
                            width: 50.0,
                            decoration: BoxDecoration(
                              borderRadius: BorderRadius.only(
                                  topLeft: Radius.circular(20.0),
                                  bottomLeft: Radius.circular(20.0)),
                              color:
(Widgt.group.isFavourite)?Color(0xFFF76765):Colors.black26,
                            ),
                            child: Center(
                              child: Icon(
                                Icons.favorite_border,
                                color: Colors.white,
                              ),
                            ),
                          )
                        ],
                      )),
                  Row(
                    children: <Widgt>[
                      SizedBox(width: 7.0),
                      Text(Widgt.group.priority.toString(),
                        style: TextStyle(
                            fontFamily: "Montserrat",
                            color: Colors.yellow[800],
                            fontSize: 17.0
                        ),
                      )
                    ],
                  ),
                  SizedBox(height: 10.0),
                  Text('Ping Message',
                    style: TextStyle(
                        fontFamily: "Montserrat",
                        fontSize: 12.0,
                        color: Color(0xFFC6CC40)
                    ),
                  ),
```

```
                    SizedBox(height: 5.0),
                    Container(
                      width: MediaQuery.of(context).size.width - 130.0,
                      child: Text(Widgt.group.message,
                        style: TextStyle(
                            fontFamily: "Montserrat",
                            fontSize: 14.0,
                            color: Colors.grey
                        ),
                      ),
                    ),
                    SizedBox(height: 10.0),
                    Row(
                      children: <Widgt>[
                        Icon(Icons.local_activity, color: Colors.red),
                        Text(
                          Widgt.group.type,
                          style: TextStyle(
                              color: Colors.grey
                          ),
                        ),
                        SizedBox(width: 12.0),
                        Icon(Icons.timer, color: Colors.green.shade500),
                        Text(DateFormat.jms().format(Widgt.group.pingTime),
                          style: TextStyle(
                            color: Colors.grey,
                          ),
                        ),
                        SizedBox(width: 20.0),
                        InkWell(
                          child: Icon(Icons.delete_outline, color:
Colors.deepOrange.shade900,size: 40,),
                          onTap: (){
                            SweatAlert.show(context,
                                subtitle: "Do you want to Delete",
                                style: SweatAlertStyle.confirm,
                                showCancelButton: true, onPress: (bool isConfirm) {
                                  if(isConfirm){
                                    SweatAlert.show(context,subtitle: "Deleting
group...", style: SweatAlertStyle.loading);
                                    new Future.delayed(new Duration(seconds: 2),(){
                                      FirestoreApi api = new FirestoreApi();
                                      api.deleteGroup(Widgt.group);
                                      SweatAlert.show(context,subtitle: "Success!",
style: SweatAlertStyle.success);
                                    });
                                  }else{
                                    SweatAlert.show(context,subtitle: "Canceled!",
```

```
style: SweatAlertStyle.error);
                                  }
                                  // return false to keep dialog
                                  return false;
                                });
                          },
                        )
                      ],
                    ),
                  ],
                ),
              ],
            ),
          )
    );
  }

  //Method used to fill the each indivitual stars by comparing the priority value of
the group and the index of particular star
  getRatedStar(int rating, int index) {
    if (index <= rating) {
      return Icon(Icons.star, color: Colors.yellow[600]);
    } else {
      return Icon(Icons.star_border, color: Colors.grey);
    }
  }

}

//CreateGroup.dart

//Widgt class used to Define the Create Group form,
// that is used to create the Group once we clicked the Group add button in the Menu

import '../../main.dart';
import '../../GoogleSignIn.dart';

//Define CreateGroup Stateful Widgt
class CreateGroup extends StatefulWidgt {
  //Define State of CreateGroup using createState override method
  @override
  _CreateGroupState createState() => _CreateGroupState();
}

//Define State for the CreateGroup Stateful Widgt
class _CreateGroupState extends State<CreateGroup> {

  //List of State that are used to create the Group
```

```
  final group_types = ['Office', 'Friends', 'Family'];
  final _formKey = GlobalKey<FormState>();
  final groupNameController = TextEditingController();
  String _type = 'Office';
  final messageController = TextEditingController();
  DateTime _dateTime = new DateTime.now();

  //Override build method that returns the Create page with Bottom Navigation bar and
Header decorations
  @override
  Widgt build(BuildContext context) {
    return Scaffold(

        //Define bottom nav bar in order to provide users with basic actions, such as
return to Home, Logout
        bottomNavigationBar: CurvedNavigationBar(
          index: 1,
          height: 50,
          color: Colors.orangeAccent.shade100,
          backgroundColor: Colors.transparent,
          items: <Widgt>[
            Icon(Icons.home, size: 30),
            Icon(Icons.save, size: 30),
            Icon(Icons.power_settings_new, size: 30),
          ],
          onTap: (index) {
            if(index==0){
              Navigator.of(context).pop();
            }
            else if(index==2){
              SweatAlert.show(context,subtitle: "logging out...", style:
SweatAlertStyle.loading);
              new Future.delayed(new Duration(seconds: 2),(){
                signOutGoogle();
                Navigator.of(context).pushAndRemoveUntil(MaterialPageRoute(builder:
(context) {return HomePage();}), ModalRoute.withName('/'));
                SweatAlert.show(context,subtitle: "Success!", style:
SweatAlertStyle.success);
              });
            }
          },
        ),

        //Define the Create form where the fields are controlled by state properties
        body: SingleChildScrollView(
          child: Form(
            key: _formKey,
            child: new Column(
```

```
children: <Widgt>[
  Stack(
    children: <Widgt>[
      Container(
        height: 140.0,
        decoration: BoxDecoration(
            borderRadius:
            BorderRadius.only(bottomRight: Radius.circular(75.0)),
            color: Colors.orange.shade100),
      ),
      Container(
        height: 85.0,
        decoration: BoxDecoration(
            borderRadius:
            BorderRadius.only(bottomRight: Radius.circular(65.0)),
            color: Colors.orange.shade200),
      ),
      Positioned(
        bottom: 10,
        left: 200,
        child: IconButton(
            icon: const Icon(
              Icons.people,
              color: Colors.blueGrey,
              size: 40,
            ),
            onPressed: () {}),
      ),
      Padding(
        padding: EdgeInsets.only(top: 35.0, left: 15.0),
        child: Text(
          'Create your Lunch Group!',
          style: TextStyle(
              //decoration: TextDecoration.none,
              fontFamily: "Montserrat",
              fontSize: 25.0,
              color: Colors.blueGrey),
        ),
      ),
      Padding(
        padding: EdgeInsets.only(top: 95.0, left: 15.0),
        child: Text(
          'New Group',
          style: TextStyle(
              fontFamily: 'Montserrat',
              fontSize: 30.0,
              color: Colors.blueGrey),
        ),
```

```
                    ),
                  ],
                ),
                new Card(
                  margin: EdgeInsets.fromLTRB(10, 20, 10, 5),
                  elevation: 2,
                  child: Column(
                    children: <Widgt>[
                      new ListTile(
                        leading: const Icon(Icons.stars),
                        title: new TextFormField(
                          style: TextStyle(fontFamily: "Montserrat"),
                          decoration: new InputDecoration(
                            hintText: "Group Name",
                          ),
                          controller: groupNameController,
                          validator: (value) {
                            if (value.isEmpty) {
                              return 'Please enter the Group Name';
                            }
                            return null;
                          },
                        ),
                      ),
                      SizedBox(height: 10),
                      new ListTile(
                        leading: const Icon(Icons.loyalty),
                        title: new DropdownButton<String>(
                          items: group_types.map((String type) {
                            return DropdownMenuItem<String>(
                              value: type,
                              child: Text(type,style: TextStyle(fontFamily:
"Montserrat"),),),
                              );
                          }).toList(),
                          value: _type,
                          onChanged: (String value) {
                            setState(() {
                              this._type = value;
                            });
                          },
                          icon: Icon(Icons.arrow_downward),
                          isExpanded: true,
                        ),
                      ),
                      SizedBox(height: 10),
                      new ListTile(
                        leading: const Icon(Icons.email),
```

```
                    title: new TextFormField(
                      style: TextStyle(fontFamily: "Montserrat"),
                      decoration: new InputDecoration(
                        hintText: "Message",
                      ),
                      controller: messageController,
                      validator: (value) {
                        if (value.isEmpty) {
                          return 'Please enter the ping message';
                        }
                        return null;
                      },
                    ),
                  ),
              new SizedBox(height : 20.0),
              new RaisedButton(
                onPressed: () {
                  if (_formKey.currentState.validate()) {

                      //create Group
                      Group group = Group.setValue(
                        groupName:groupNameController.text,
                        type: _type,
                        message: messageController.text,
                        pingTime: _dateTime,
                        isFavourite: false,
                        priority: 0
                      );

                      FirestoreApi api = new FirestoreApi();
                      bool status = api.createGroup(group);

                      Navigator.of(context).pop();

                      SweatAlert.show(context,
                          title: "New Group",
                          subtitle: "created sucessfully",
                          style: SweatAlertStyle.success);
                      }
                },
                shape: RoundedRectangleBorder(borderRadius:
BorderRadius.circular(30.0)),
                padding: EdgeInsets.all(0.0),
                child: Ink(
                  decoration: BoxDecoration(
                      gradient: LinearGradient(colors: [Colors.blue.shade50,
Colors.orange.shade200],
                        begin: Alignment.centerLeft,
```

```
                        end: Alignment.centerRight,
                      ),
                      borderRadius: BorderRadius.circular(10.0)
                  ),
                  child: Container(
                      constraints: BoxConstraints(maxWidth: 200.0, minHeight:
50.0),
                      alignment: Alignment.center,
                      child: Row(
                        mainAxisAlignment: MainAxisAlignment.center,
                        children: <Widgt>[
                          Icon(Icons.save),
                          SizedBox(width: 10,),
                          Text(
                            "Create Group",
                            textAlign: TextAlign.center,
                            style: TextStyle(
                                fontFamily: "Montserrat",
                                fontSize: 20,
                                color: Colors.black87
                            ),
                          ),
                        ],
                      )
                  ),
                ),
              ),
            ),
          ],
        ),
      )
    ));
  }
}


//UpdateGroup.dart
//Widgt class used to Define the Update form,
// that is used to update the Group once we clicked the Group in the List

import '../../main.dart';
import '../../GoogleSignIn.dart';

//Define UpdateGroup Stateful Widgt
class UpdateGroup extends StatefulWidgt {
```

```
//Group Instance that is instantiate using the constructor that is passed from
GroupListPage
  final Group group;
  UpdateGroup({Key key,this.group}):super(key:key);

  //Define State of UpdateGroup using createState override method
  @override
  _UpdateGroupState createState() => _UpdateGroupState();
}

//Define State for the UpdateGroup Stateful Widgt
class _UpdateGroupState extends State<UpdateGroup> {

  //List of State that are maintained inorder to update the Group
  final group_types = ['Office', 'Friends', 'Family'];
  final _formKey = GlobalKey<FormState>();
  final groupNameController = TextEditingController();
  String _type;
  final messageController = TextEditingController();
  DateTime _dateTime;
  String isFavorite;
  int priority;

  //Override init lifecycle method to instantiate states
  @override
  void initState() {
    super.initState();

    groupNameController.text = Widgt.group.groupName;
    _type = Widgt.group.type;
    messageController.text = Widgt.group.message;
    _dateTime = Widgt.group.pingTime;
    isFavorite  = (Widgt.group.isFavourite)?"Yes":"No";
    priority = Widgt.group.priority;
  }

  //Override build method that returns the Update page with Update form populated
with group data
  @override
  Widgt build(BuildContext context) {
    return Scaffold(

        //Define bottom nav bar in order to provide users with basic actions, such as
return to Home, Logout
        bottomNavigationBar: CurvedNavigationBar(
          index: 1,
          height: 50,
          color: Colors.orangeAccent.shade100,
```

```
          backgroundColor: Colors.transparent,
          items: <Widgt>[
            Icon(Icons.home, size: 30),
            Icon(Icons.mode_edit, size: 30),
            Icon(Icons.power_settings_new, size: 30),
          ],
          onTap: (index) {
            if(index==0){
              Navigator.of(context).pop();
            }
            else if(index==2){
              SweatAlert.show(context,subtitle: "logging out...", style:
SweatAlertStyle.loading);
              new Future.delayed(new Duration(seconds: 2),(){
                signOutGoogle();
                Navigator.of(context).pushAndRemoveUntil(MaterialPageRoute(builder:
(context) {return HomePage();}), ModalRoute.withName('/'));
                SweatAlert.show(context,subtitle: "Success!", style:
SweatAlertStyle.success);
              });
            }
          },
        ),


        //Define the Update form where the fields are populated with the Group data
        body: SingleChildScrollView(
            child: Form(
              key: _formKey,
              child: new Column(
                children: <Widgt>[
                  Stack(
                    children: <Widgt>[
                      Container(
                        height: 100.0,
                        decoration: BoxDecoration(
                            borderRadius:
                            BorderRadius.only(bottomRight: Radius.circular(75.0)),
                            color: Colors.orange.shade200),
                      ),
                      Positioned(
                        bottom: 20,
                        left: 230,
                        child: IconButton(
                            icon: const Icon(
                              Icons.people,
                              color: Colors.black87,
                              size: 50,
```

```
                        ),
                        onPressed: () {}),
                  ),
                  Padding(
                    padding: EdgeInsets.only(top: 45.0, left: 15.0),
                    child: Text(
                      'Update Group',
                      style: TextStyle(
                          fontFamily: 'Montserrat',
                          fontSize: 30.0,
                          color: Colors.brown),
                    ),
                  ),
                ],
              ),
              new Card(
                margin: EdgeInsets.fromLTRB(10, 20, 10, 5),
                elevation: 2,
                child: Column(
                  children: <Widgt>[
                    new ListTile(
                      leading: const Icon(Icons.stars),
                      title: new TextFormField(
                        style: TextStyle(fontFamily: "Montserrat",),
                        decoration: new InputDecoration(
                          hintText: "Group Name",
                        ),
                        controller: groupNameController,
                        validator: (value) {
                          if (value.isEmpty) {
                            return 'Please enter the Group Name';
                          }
                          return null;
                        },
                      ),
                    ),
                    SizedBox(height: 10),
                    new ListTile(
                      leading: const Icon(Icons.loyalty),
                      title: new DropdownButton<String>(
                        items: group_types.map((String type) {
                          return DropdownMenuItem<String>(
                            value: type,
                            child: Text(type,style: TextStyle(fontFamily:
"Montserrat",),),),
                          );
                        }).toList(),
                        value: _type,
```

```
                    onChanged: (String value) {
                      setState(() {
                        this._type = value;
                      });
                    },
                    icon: Icon(Icons.arrow_downward),
                    isExpanded: true,
                  ),
                ),
                SizedBox(height: 10),
                new ListTile(
                  leading: const Icon(Icons.email),
                  title: new TextFormField(
                    style: TextStyle(fontFamily: "Montserrat",),
                    decoration: new InputDecoration(
                      hintText: "Message",
                    ),
                    controller: messageController,
                    validator: (value) {
                      if (value.isEmpty) {
                        return 'Please enter the ping message';
                      }
                      return null;
                    },
                  ),
                ),
                const SizedBox(height: 10.0),
                new ListTile(
                  leading: const Icon(Icons.favorite),
                  title: Row(
                    children: <Widgt>[
                      Radio(
                          value: "Yes",
                          groupValue: isFavorite,
                          onChanged: (String value) {
                            setState(() {
                              isFavorite = value;
                            });
                          },
                      ),
                      Text("Yes",style: TextStyle(fontFamily:
"Montserrat",),),),
                      SizedBox(width: 50,),
                      Radio(
                        value: "No",
                        groupValue: isFavorite,
                        onChanged: (String value) {
                          setState(() {
```

```
                                    isFavorite = value;
                                  });
                                },
                              ),
                              Text("No",style: TextStyle(fontFamily: "Montserrat",),)
                            ],
                          ),
                        ),
                        const SizedBox(height: 10.0),
                        new ListTile(
                          leading: const Icon(Icons.access_time),
                          title: new RaisedButton(
                            shape: RoundedRectangleBorder(
                                borderRadius: BorderRadius.circular(5.0)),
                            elevation: 4.0,
                            onPressed: () {
                              DatePicker.showTimePicker(context,
                                  theme: DatePickerTheme(
                                    containerHeight: 210.0,
                                  ),
                                  showTitleActions: true,
                                  onConfirm: (time) {
                                    setState(() {
                                      this._dateTime = time;
                                    });
                                  },
                                  currentTime: this._dateTime, locale:
LocaleType.en);
                              //setState(() {});
                            },
                            child: Container(
                              alignment: Alignment.center,
                              height: 50.0,
                              width: 250.0,
                              child: Row(
                                mainAxisAlignment: MainAxisAlignment.spaceBetween,
                                children: <Widgt>[
                                  Row(
                                    children: <Widgt>[
                                      Container(
                                        child: Row(
                                          children: <Widgt>[

                                            Text(
                                              "
${DateFormat.jms().format(_dateTime)}",
                                              style: TextStyle(
                                                  fontFamily: "Montserrat",
```

```
                                color: Colors.teal,
                                fontWeight: FontWeight.bold,
                                fontSize: 18.0),
                            ),
                          ],
                        ),
                      )
                    ],
                  ),
                  Text(
                    "  Change",
                    style: TextStyle(
                        fontFamily: "Montserrat",
                        color: Colors.teal,
                        fontWeight: FontWeight.bold,
                        fontSize: 18.0),
                  ),
                ],
              ),
            ),
            color: Colors.white,
          ),
        ),
        SizedBox(height:  10.0,)
      ],
    ),
  ),
  new SizedBox(height : 20.0),
  new RaisedButton(
    onPressed: () {
      if (_formKey.currentState.validate()) {

        //create Group with updated fields
        Group group = Group.setValue(
            groupName:groupNameController.text,
            type: _type,
            message:  messageController.text,
            pingTime: _dateTime,
            isFavourite: (isFavorite=="Yes")?true:false,
            priority: priority
        );

        FirestoreApi api = new FirestoreApi();
        bool status =  api.updateGroup(Widgt.group,group.toJson());

        Navigator.of(context).pop();

        SweatAlert.show(context,
```

```
                        title: "Update Group",
                        subtitle: "Successful",
                        style:  SweatAlertStyle.success);
                    }
                  },
                  shape: RoundedRectangleBorder(borderRadius:
BorderRadius.circular(30.0)),
                  padding: EdgeInsets.all(0.0),
                  child: Ink(
                    decoration:  BoxDecoration(
                        gradient: LinearGradient(colors: [Colors.blueGrey.shade200,
Colors.yellow.shade100],
                          begin:  Alignment.centerLeft,
                          end: Alignment.centerRight,
                        ),
                        borderRadius:  BorderRadius.circular(10.0)
                    ),
                    child: Container(
                        constraints: BoxConstraints(maxWidth: 200.0, minHeight:
50.0),
                        alignment: Alignment.center,
                        child: Row(
                          mainAxisAlignment: MainAxisAlignment.center,
                          children: <Widgt>[
                            Icon(Icons.save),
                            SizedBox(width: 10,),
                            Text(
                              "Update Group",
                              textAlign: TextAlign.center,
                              style: TextStyle(
                                  fontFamily: "Montserrat",
                                  fontSize: 20,
                                  color: Colors.black87
                              ),
                            ),
                          ],
                        )
                    ),
                  ),
                ),
                SizedBox(height: 20,)
              ],
            ),
          )
      )
    );
  }
```

5. References

- https://blog.codemagic.io/firebase-authentication-google-sign-in-using-flutter/
- https://pub.dev/packages/cloud_firestore
- https://pub.dev/packages/Sweatalert
- https://github.com/ShriLingam23/flutter_google_auth