1. Install the following dependencies
   cloud_firestore
   firebase_auth
   firebase_core
   firebase_storage
2. Close and restart the IDE
3. Creating a firebase project and connecting it our project
    a. Go to https://firebase.google.com
    b. Click get started in console
    c. Create a firebase project
    d. Enter a project name eg: user-registration
    e. Click continue(if asked select your domain. youremail@mgits.ac.in/youremail@gmail.com)
    f. Click again continue
    g. Configure Google Analytics=> Default account for firebase
    h. Click create project
    i. After the project is created, click on Build side menu and choose
        i. Authentication

1. Click get started
2. Choose Sign-in providers=> Email and Password
3. Enable Email/Password and click save
        ii. Click on Build=>Firebase Database

1. Create Database
2. Click next without altering the asked values then click create
3. Now you can see a Cloud Firestore
4. Click on Rules
5. Change the rule allow read, write: if false; => allow read, write: if true;
6. Click publish
    j. Click on Project Overview side menu
    k. Click on the flutter Icon
    l. Install the Firebase CLI click on the link provided. You can install it either by windows exe or by installing node
    m. After installation run
       dart pub global activate flutterfire_cli
    n. Then run the command provided by firebase to link this project to the flutter project
    o. When you run this command
        i. Select Android [Deselect other Options by pressing spacebar] then press enter

In case of errors

Install node

npm install -g firebase-tools

rerun the two commands provided by firebase projects

4. If all steps are successfully completed, you can see a file named **firebase_options.dart** in the lib

Using Firebase in the project

1. Edit the main function as follows

```
void main() async {
  WidgetsFlutterBinding.ensureInitialized();
  await Firebase.initializeApp(options:
DefaultFirebaseOptions.currentPlatform);
  runApp(const MainApp());
}
```

```
Future<bool> resgiterUser(UserModel u) async {
  try {
    final UserCredential userCredential = await FirebaseAuth.instance
        .createUserWithEmailAndPassword(
          email: u.userEmail,
          password: u.userPassword,
        );
    if (userCredential != null) {
      final firebaseInstance = await FirebaseFirestore.instance
          .collection('users')
          .doc(userCredential.user!.uid)
          .set({
            'user_name': u.userName,
            'user_email': u.userEmail,
            'user_gender': u.userGender,
            'user_address': u.userAddress,
          });
      return Future.value(true);
    } else {
      return Future.value(false);
    }
  } catch (e) {
    return Future.value(false);
  }
}
```

```
Future<bool> checkLogin(UserModel u) async {
  try {
    final UserCredential userCredential = await FirebaseAuth.instance
        .signInWithEmailAndPassword(
          email: u.userEmail,
          password: u.userPassword,
        );
    if (userCredential != null) {
      currentUserId = userCredential.user!.uid;
      return Future.value(true);
    } else {
```

```
      return Future.value(false);
    }
  } catch (e) {
    return Future.value(false);
  }
}
```

```
Future<UserModel> loadUser(String userID) async {
  final firebaseInstance =
      await FirebaseFirestore.instance.collection('users').doc(userID).get();
  final userData = firebaseInstance.data();
  UserModel u = UserModel(
    userData!['user_name'],
    userData['user_email'],
    '',
    userData['user_gender'],
    userData['user_address'],
  );
  return Future.value(u);
}
```

```
Future<bool> editUser(UserModel u) async {
  try {
    final firebaseInstance = await FirebaseFirestore.instance
        .collection('users')
        .doc(currentUserId)
        .set({
          'user_name': u.userName,
          'user_email': u.userEmail,
          'user_gender': u.userGender,
          'user_address': u.userAddress,
        });
    return Future.value(true);
  } catch (e) {
    return Future.value(false);
  }
}
```

```
Future<bool> deleteUser(String userID) async {
  try {
    final FirebaseInstance =
        await FirebaseFirestore.instance
            .collection('users')
            .doc(userID)
```

```
            .delete();
    return Future.value(true);
  } catch (e) {
    return Future.value(true);
  }
}
```