EXPERIMENT NO. 6
MAD and PWA Lab

Aim: To Connect Flutter UI with firebase database

Theory:

Google **Firebase** is a Google-backed application development software that enables developers to develop iOS, Android and Web apps. Firebase provides tools for tracking analytics, reporting and fixing app crashes, creating marketing and product experiments.

Firebase offers a number of services, including:

1. **Analytics** – Google Analytics for Firebase offers free, unlimited reporting on as many as 500 separate events. Analytics presents data about user behavior in iOS and Android apps, enabling better decision-making about improving performance and app marketing.

2. **Authentication** – Firebase Authentication makes it easy for developers to build secure authentication systems and enhances the sign-in and onboarding experience for users. This feature offers a complete identity solution, supporting email and password accounts, phone auth, as well as Google, Facebook, GitHub, Twitter login and more.

3. **Cloud messaging** – Firebase Cloud Messaging (FCM) is a cross-platform messaging tool that lets companies reliably receive and deliver messages on iOS, Android and the web at no cost.

4. **Realtime database** – the Firebase Realtime Database is a cloud-hosted NoSQL database that enables data to be stored and synced between users in real time. The data is synced across all clients in real time and is still available when an app goes offline.

5. **Crashlytics** – Firebase Crashlytics is a real-time crash reporter that helps developers track, prioritize and fix stability issues that reduce the quality of their apps. With crashlytics, developers spend less time organizing and troubleshooting crashes and more time building features for their apps.

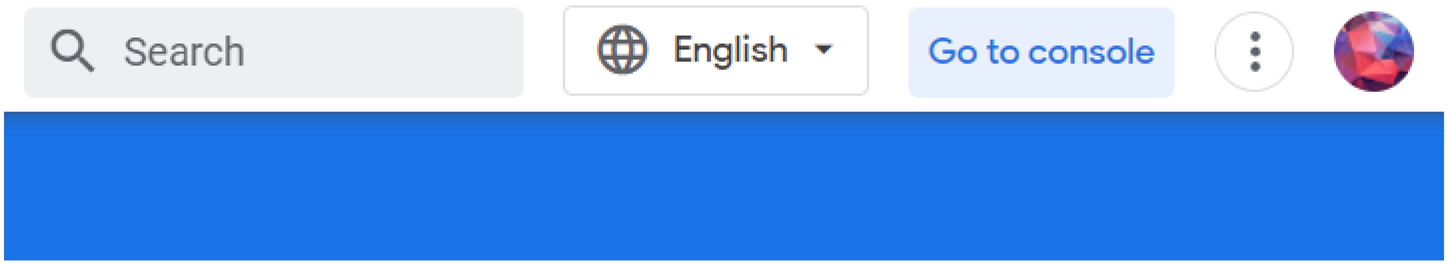
6. **Performance** – Firebase Performance Monitoring service gives developers insight into the performance characteristics of their iOS and Android apps to help them determine where and when the performance of their apps can be improved.

7. **Test lab** – Firebase Test Lab is a cloud-based app-testing infrastructure. With one operation, developers can test their iOS or Android apps across a variety of devices and device configurations. They can see the results, including videos, screenshots and logs, in the Firebase console.

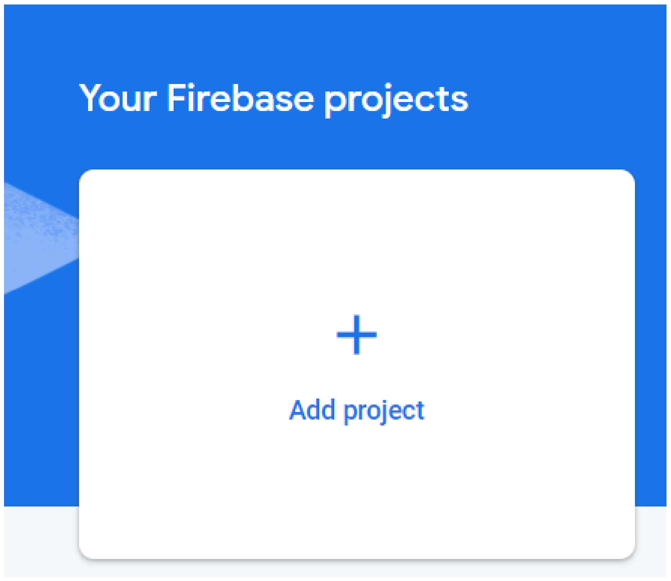
Creating a Firebase Project

Add Firebase to your existing Google Cloud project:

1. Log in to the Firebase console, then click Add project.
2. Select your existing Google Cloud project from the dropdown menu, then click Continue.
3. (Optional) Enable Google Analytics for your project, then follow the prompts to select or create a Google Analytics account.
4. Click Add Firebase.

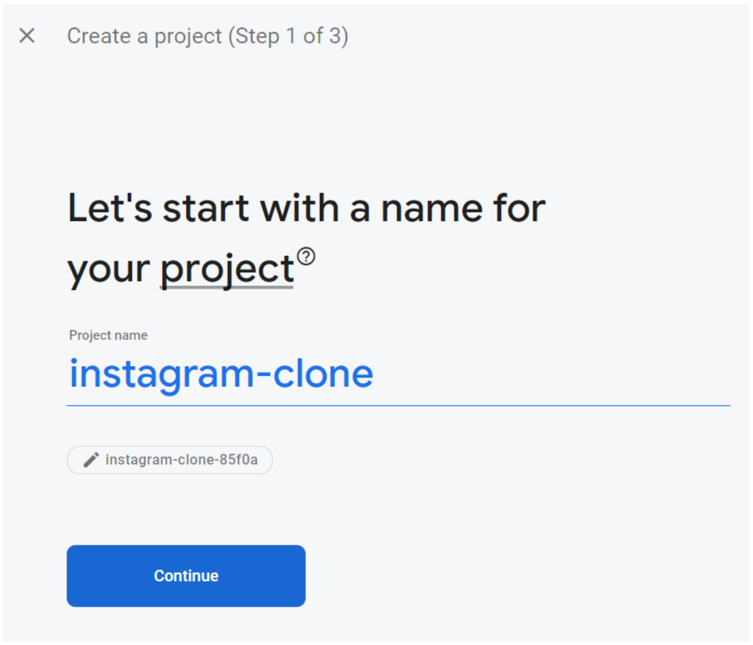


Now open the Firebase console and click Create New Project.



Enter the following information and click on Create Project:

1. Project name



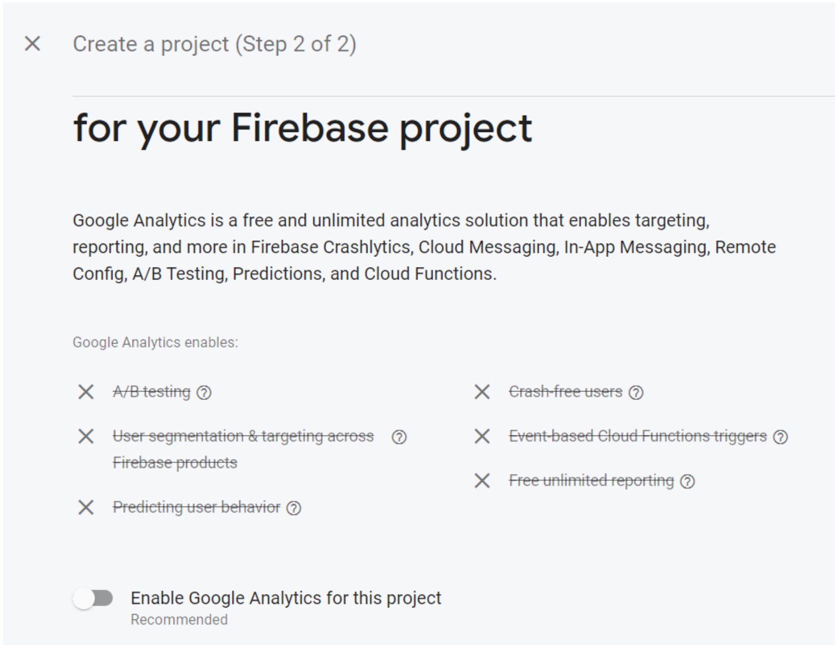
Create a project (Step 1 of 3)

Let's start with a name for your project

Project name: instagram-clone

instagram-clone-85f0a

2. Disable the Google Analytics for the project, we do not need this now unless we deploy the app.



Create a project (Step 2 of 2)

for your Firebase project

Google Analytics is a free and unlimited analytics solution that enables targeting, reporting, and more in Firebase Crashlytics, Cloud Messaging, In-App Messaging, Remote Config, A/B Testing, Predictions, and Cloud Functions.

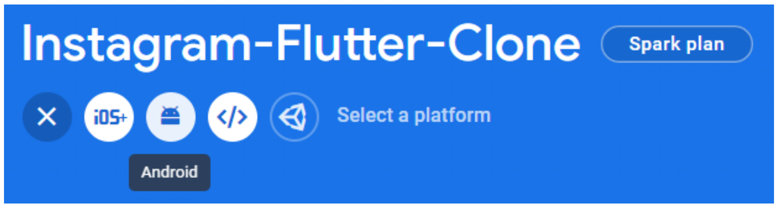
Google Analytics enables:

- ~~A/B testing~~
- ~~User segmentation & targeting across Firebase products~~
- ~~Predicting user behavior~~
- ~~Crash-free users~~
- ~~Event-based Cloud Functions triggers~~
- ~~Free unlimited reporting~~

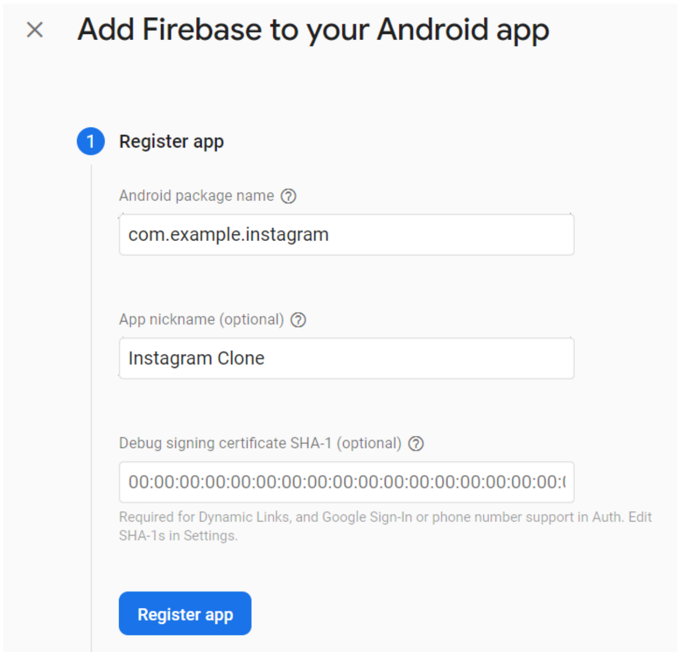
Enable Google Analytics for this project
Recommended

Setting up the Android App/IOS App

Now add an Android app if you are running your app on Android or else add an IOS app. I will be adding the Android app.


Instagram-Flutter-Clone

1. Register your Android App using the package name in the app-level build.gradle file and a random app nickname.


Add Firebase to your Android app

1 Register app

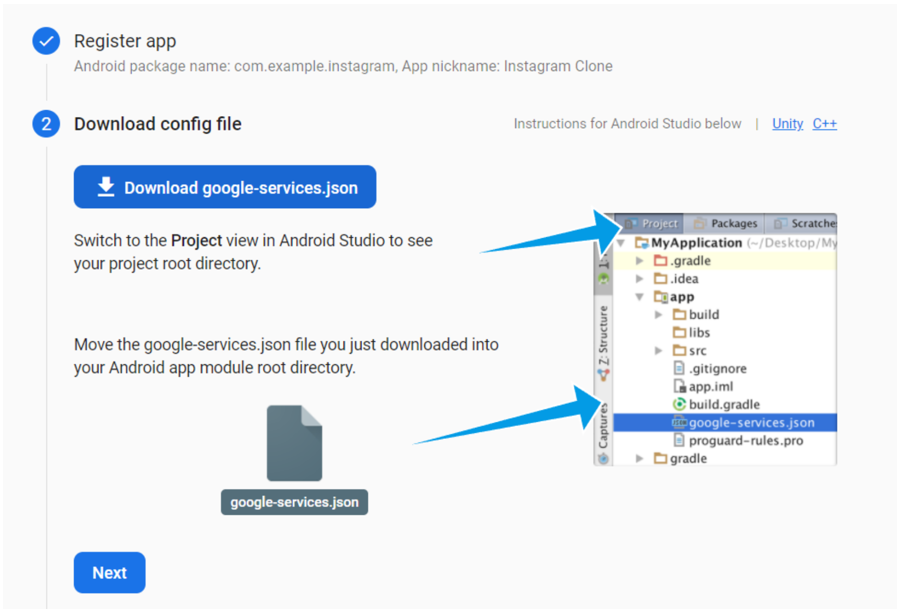
Android package name: com.example.instagram

App nickname (optional): Instagram Clone

Debug signing certificate SHA-1 (optional): 00:00:00:00:00:00:00:00:00:00:00:00:00:00:00:00:00:00:0

Required for Dynamic Links, and Google Sign-In or phone number support in Auth. Edit SHA-1s in Settings.

2. Download the google-services.json file and store it in the android/app directory.


Register app

Android package name: com.example.instagram, App nickname: Instagram Clone

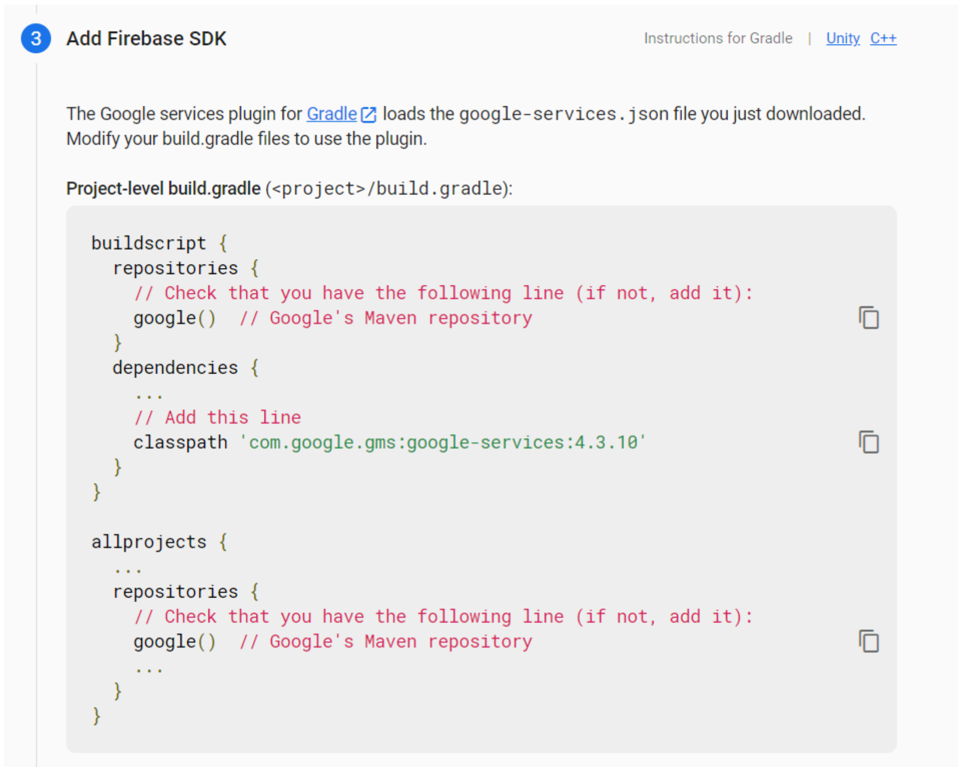
2 Download config file

Instructions for Android Studio below | Unity C++

Switch to the **Project** view in Android Studio to see your project root directory.

Move the google-services.json file you just downloaded into your Android app module root directory.

3. Add Firebase SDK to the project.



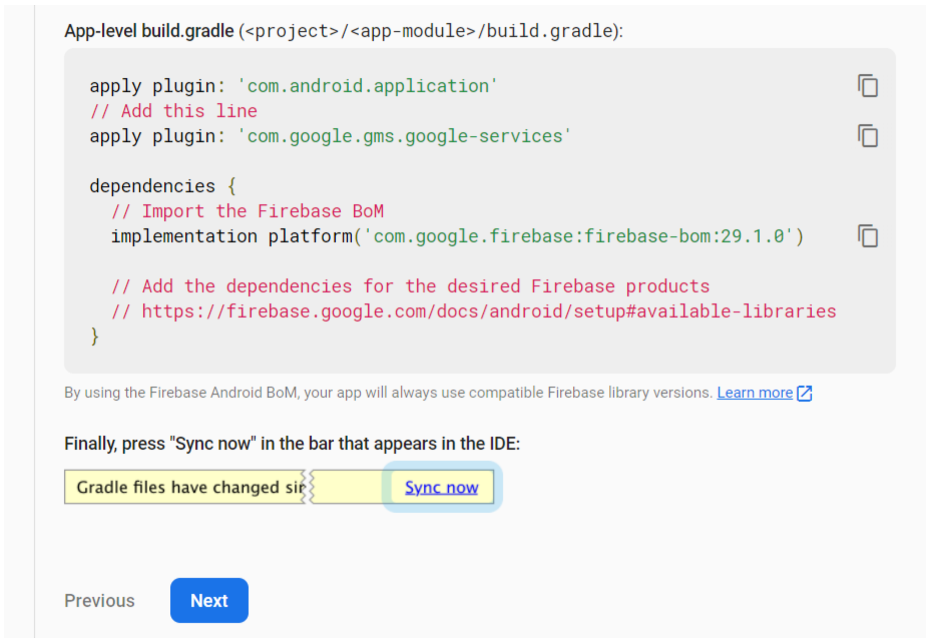
3 Add Firebase SDK

Instructions for Gradle | Unity C++

The Google services plugin for Gradle loads the `google-services.json` file you just downloaded. Modify your build.gradle files to use the plugin.

Project-level build.gradle (`<project>/build.gradle`):

```
buildscript {
  repositories {
    // Check that you have the following line (if not, add it):
    google()  // Google's Maven repository
  }
  dependencies {
    ...
    // Add this line
    classpath 'com.google.gms:google-services:4.3.10'
  }
}

allprojects {
  ...
  repositories {
    // Check that you have the following line (if not, add it):
    google()  // Google's Maven repository
    ...
  }
}
```



App-level build.gradle (`<project>/<app-module>/build.gradle`):

```
apply plugin: 'com.android.application'
// Add this line
apply plugin: 'com.google.gms.google-services'

dependencies {
  // Import the Firebase BoM
  implementation platform('com.google.firebase:firebase-bom:29.1.0')

  // Add the dependencies for the desired Firebase products
  // https://firebase.google.com/docs/android/setup#available-libraries
}
```

By using the Firebase Android BoM, your app will always use compatible Firebase library versions. Learn more

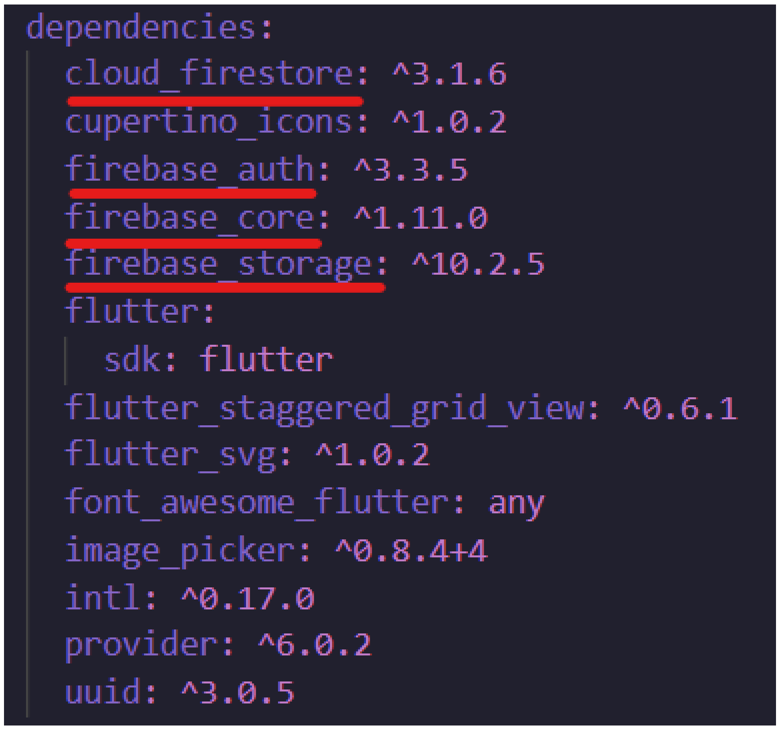
Finally, press "Sync now" in the bar that appears in the IDE:

Gradle files have changed si Sync now

Previous Next

Installing the FlutterFire libraries

Add the following packages from pub.dev to your pubspec.yaml file



```
dependencies:
  cloud_firestore: ^3.1.6
  cupertino_icons: ^1.0.2
  firebase_auth: ^3.3.5
  firebase_core: ^1.11.0
  firebase_storage: ^10.2.5
  flutter:
    sdk: flutter
  flutter_staggered_grid_view: ^0.6.1
  flutter_svg: ^1.0.2
  font_awesome_flutter: any
  image_picker: ^0.8.4+4
  intl: ^0.17.0
  provider: ^6.0.2
  uuid: ^3.0.5
```

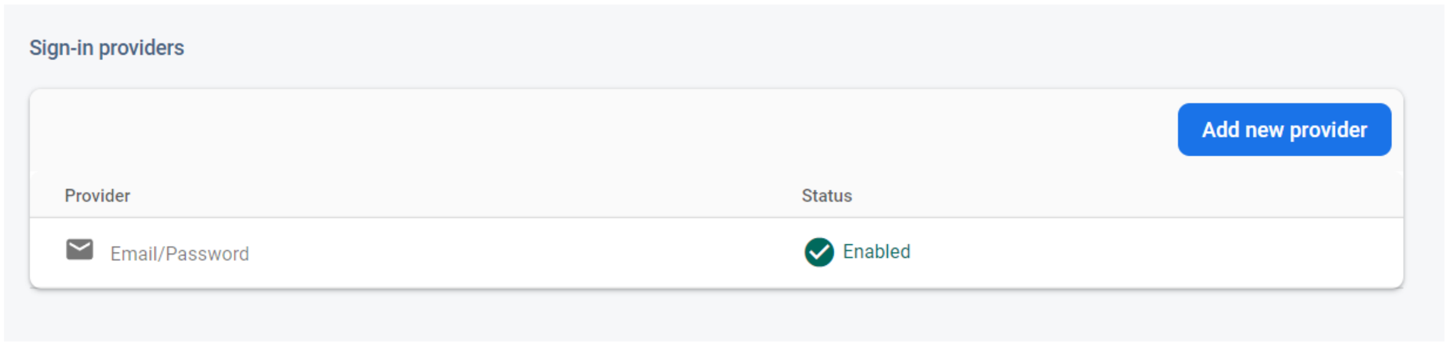
Initializing Firebase in the code

Once done installing these packages, update the main.dart to initialize firebase when the app starts.

```
void main() async {
  WidgetsFlutterBinding.ensureInitialized();
  await Firebase.initializeApp();
  runApp(const MyApp());
}
```

Making a simple Realtime Database call

For Login and Sign Up, I have used Email/Password provider in Authentication of the Firebase project.



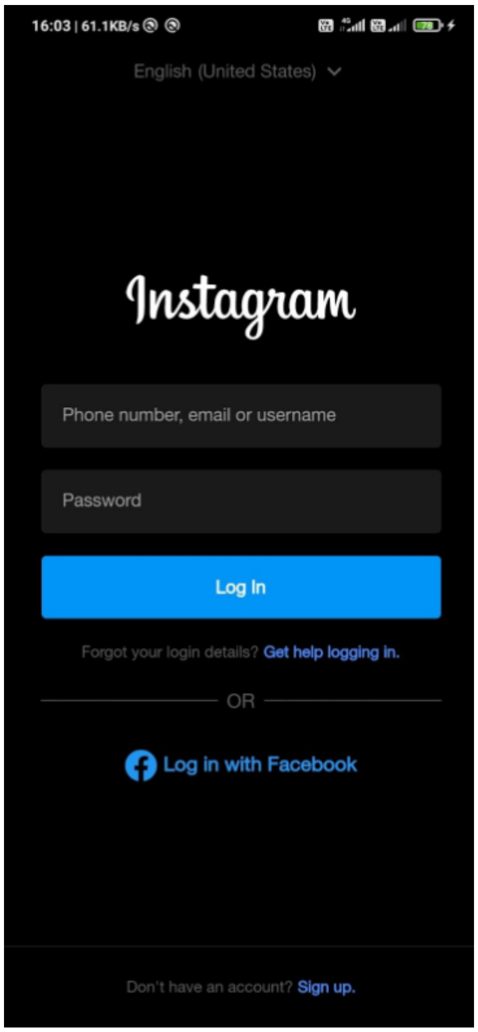
Sign-in providers

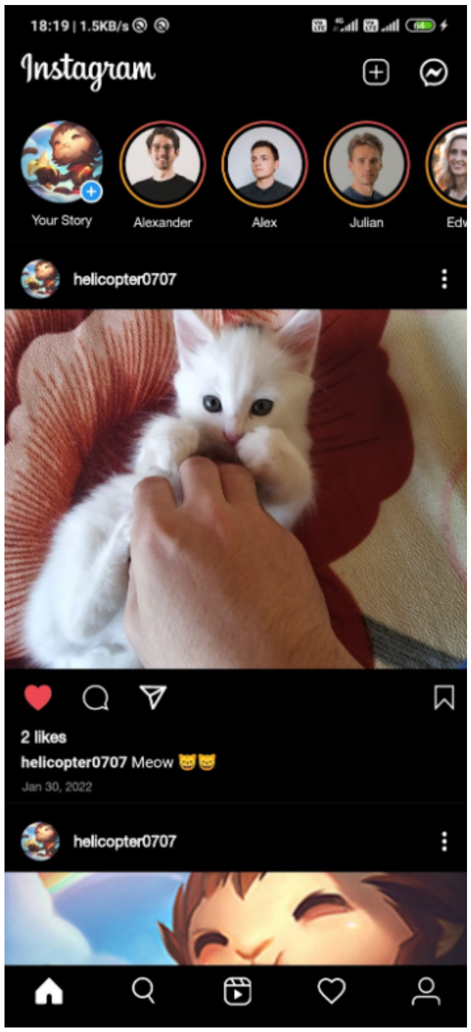
Provider	Status
Email/Password	✓ Enabled

For user authentication, I have made a separate **auth_methods.dart** file where I have added the functions to get user details, login and register.

```dart
import 'dart:typed_data';
import 'package:cloud_firestore/cloud_firestore.dart';
import 'package:firebase_auth/firebase_auth.dart';
import 'package:instagram_clone/models/user.dart' as model;
import 'package:instagram_clone/resources/storage_methods.dart';

class AuthMethods {
  final FirebaseAuth _auth = FirebaseAuth.instance;
  final FirebaseFirestore _firestore = FirebaseFirestore.instance;

  Future<model.User> getUserDetails() async {
    User currentUser = _auth.currentUser!;
    DocumentSnapshot snap =
        await _firestore.collection('users').doc(currentUser.uid).get();
    return model.User.fromSnap(snap);
  }

  // Sign up the user
  Future<String> signUpUser({
    required String email,
    required String fullName,
    required String password,
    required String username,
    required String bio,
    // required int phoneNo,
    required Uint8List file,
  }) async {
    String res = "Some error occurred";
    try {
      if (fullName.isNotEmpty ||
          email.isNotEmpty ||
          password.isNotEmpty ||
          username.isNotEmpty ||
          bio.isNotEmpty) {
        // Register user
        UserCredential cred = await _auth.createUserWithEmailAndPassword(
          email: email,
          password: password,
        );

        String photoUrl = await StorageMethods().uploadImage(
          'profilePics',
          file,
          false,
        );

        model.User user = model.User(
          username: username,
          uid: cred.user!.uid,
          fullName: fullName,
          email: email,
          bio: bio,
          followers: [],
          following: [],
          photoUrl: photoUrl,
        );

        // Add user to our database
        await _firestore.collection('users').doc(cred.user!.uid).set(
              user.toJson(),
            );
        res = "Success";
      }
    } catch (err) {
      res = err.toString();
    }
    return res;
  }

  // Login user
  Future<String> loginUser({
    required String email,
    required String password,
  }) async {
    String res = "Some error occurred";
    try {
      if (email.isNotEmpty || password.isNotEmpty) {
        await _auth.signInWithEmailAndPassword(
          email: email,
          password: password,
        );
        res = "Success";
      }
    } catch (err) {
      res = err.toString();
    }
    return res;
  }
}
```

Now when we login to the app, the screen goes to Home Screen from Login Screen.


16:03 | 61.1KB/s
English (United States)
Instagram
Phone number, email or username
Password
Log In
Forgot your login details? Get help logging in.
OR
Log in with Facebook
Don't have an account? Sign up.



Conclusion: Hence, we understood how to connect Flutter UI with firebase database.