EXPERIMENT NO: - 06

Name:- Tejas Gunjal **Class:-** D15A **Roll:No: -** 18

AIM: - To connect Flutter UI with Firebase database.

Theory: -

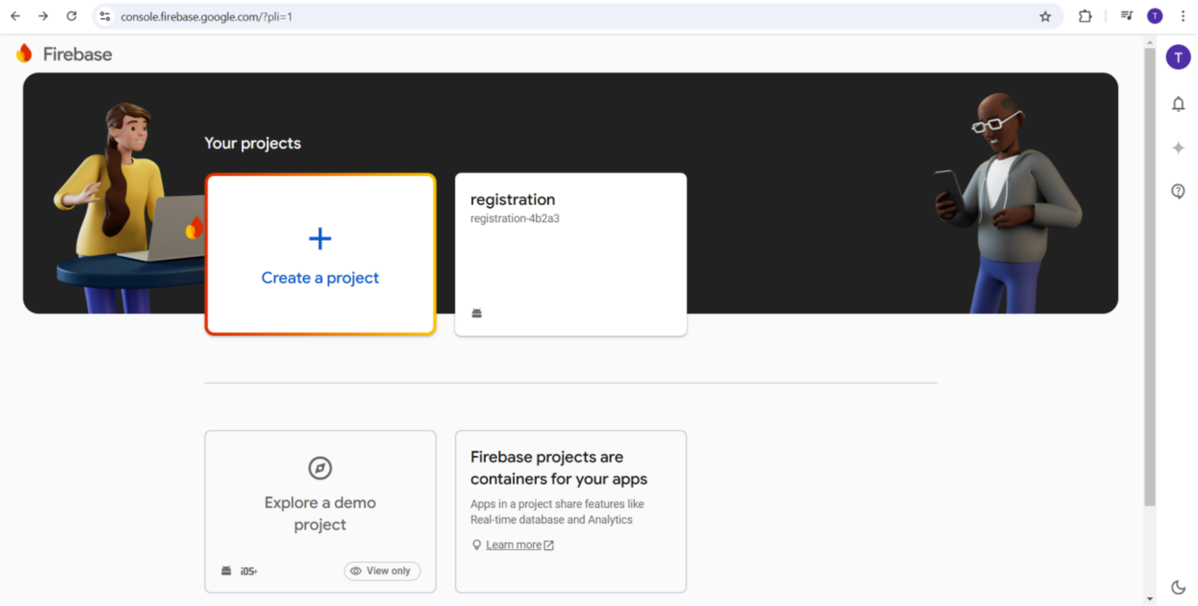
Flutter is an open-source UI toolkit developed by Google for building natively compiled applications for mobile, web, and desktop from a single codebase. Firebase, a Backend-as-a-Service (BaaS) platform, provides real-time database, authentication, and cloud storage services, making it a powerful backend solution for Flutter applications.

By integrating Firebase with Flutter, developers can store and retrieve data in real time, authenticate users, and manage cloud-based data efficiently. This is particularly useful for applications requiring dynamic content updates and user interactions.

➢ Steps to Connect Flutter UI with Firebase Database

Step 1:

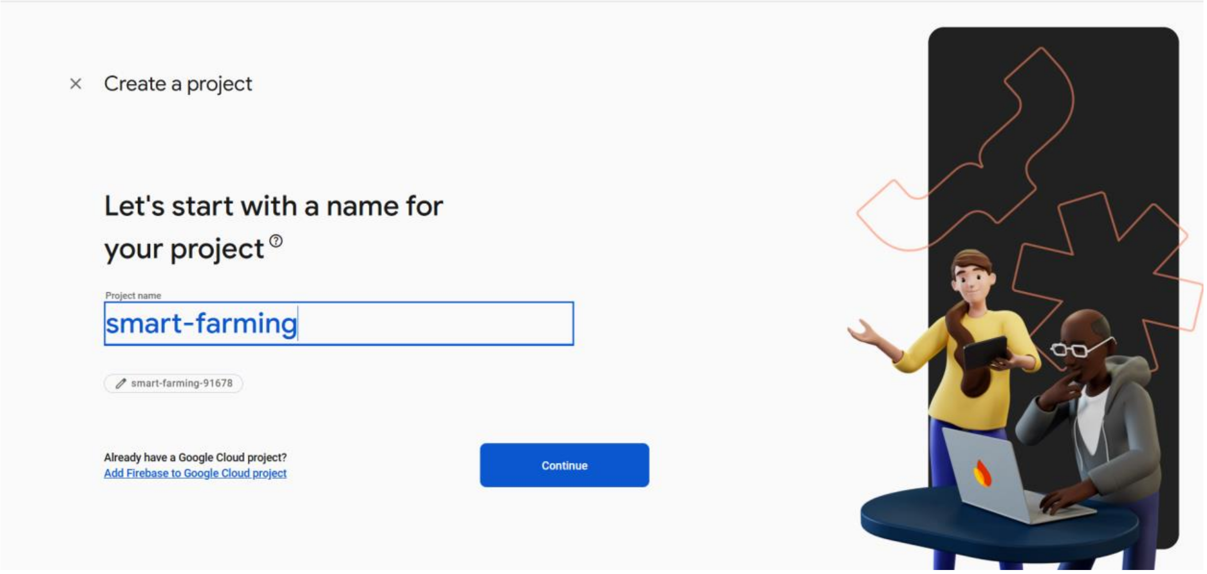
1.1) Go to Firebase Console and Create a Firebase Project



Firebase projects are containers for your apps

Apps in a project share features like Real-time database and Analytics

1.2) Click on Create a Project and give it a suitable name.



× Create a project

Let's start with a name for your project

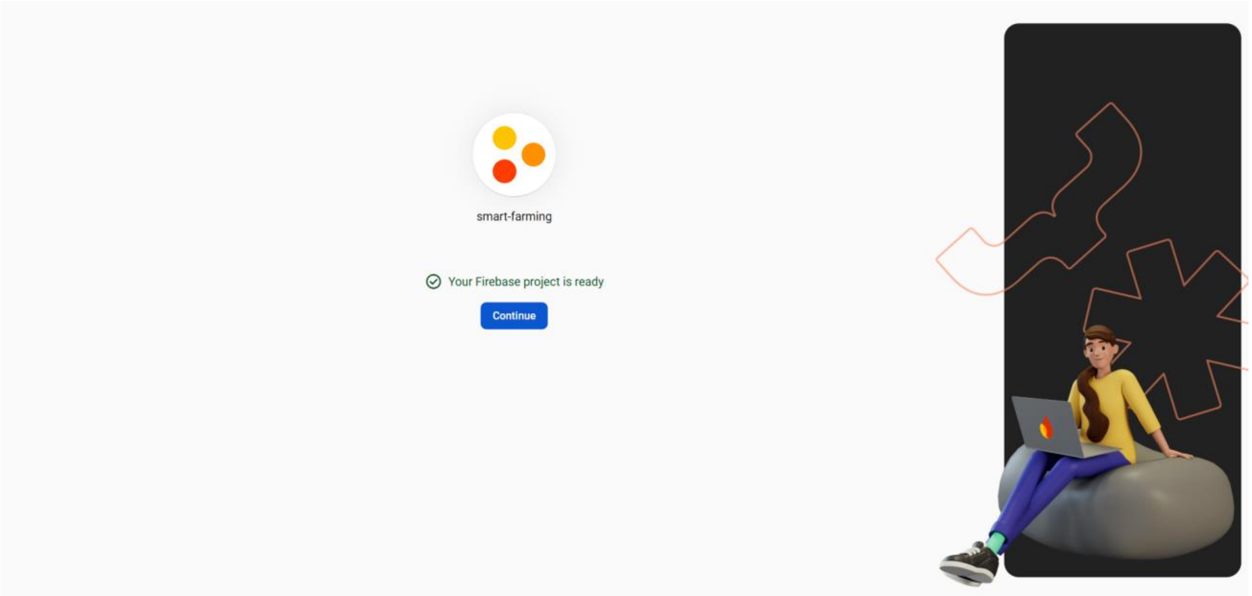
Project name

smart-farming

smart-farming-91678

Already have a Google Cloud project?
Add Firebase to Google Cloud project

1.3) Enable Google Analytics (optional) & Click continue and complete the setup

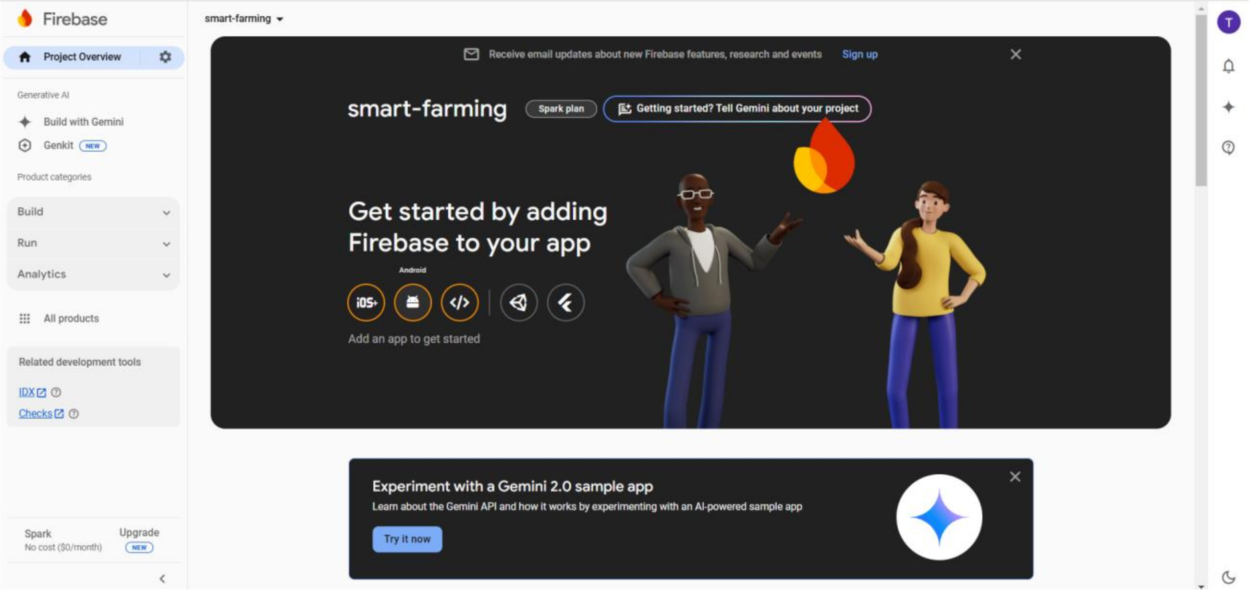


smart-farming

Your Firebase project is ready

Step 2:- Add Firebase to Your Flutter App

2.1) Click on Android/iOS/Web based on your Flutter application



Receive email updates about new Firebase features, research and events

smart-farming

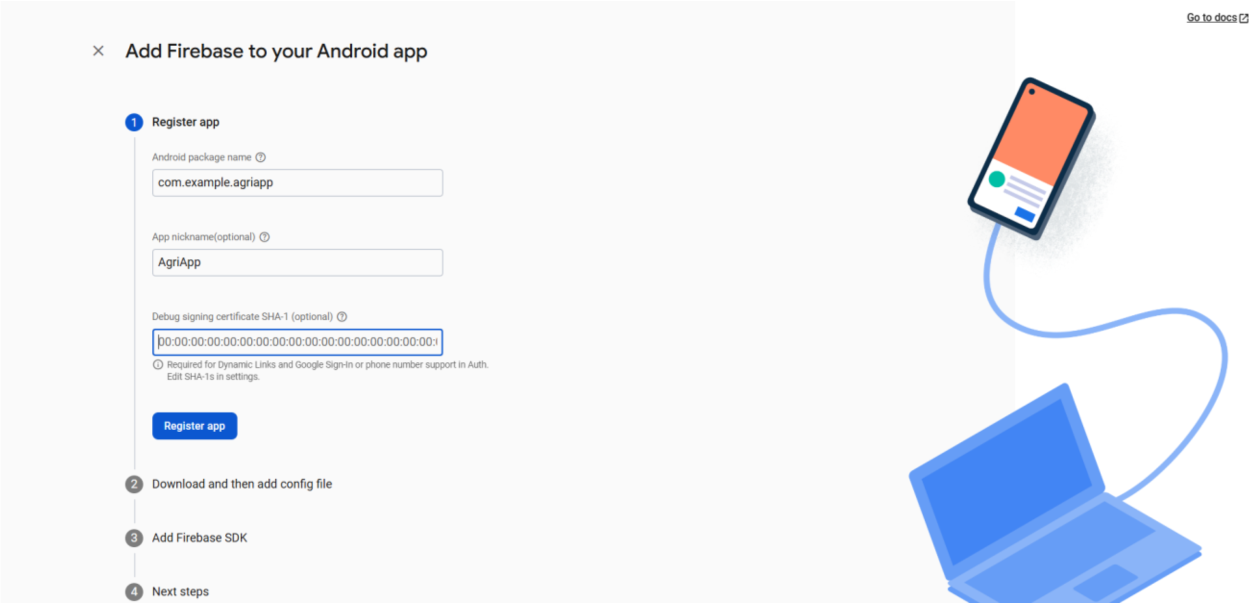
Get started by adding Firebase to your app

Add an app to get started

Experiment with a Gemini 2.0 sample app

Learn about the Gemini API and how it works by experimenting with an AI-powered sample app

2.2) Register your app with a unique package name (found in android/app/build.gradle for Android).



Add Firebase to your Android app

1 Register app

Android package name: com.example.agriapp

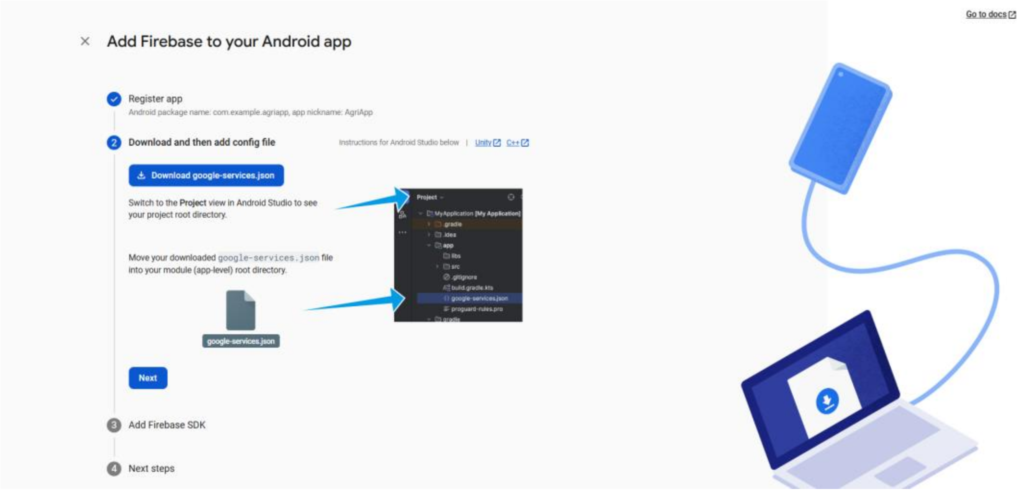
App nickname(optional): AgriApp

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

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

Register app

2.3) Download the google-services.json (for Android) & place the JSON file inside android/app/ directory.



× Add Firebase to your Android app

Go to docs

✓ Register app
Android package name: com.example.agriapp, app nickname: AgriApp

2 Download and then add config file

Instructions for Android Studio below | Unity C++

Download google-services.json

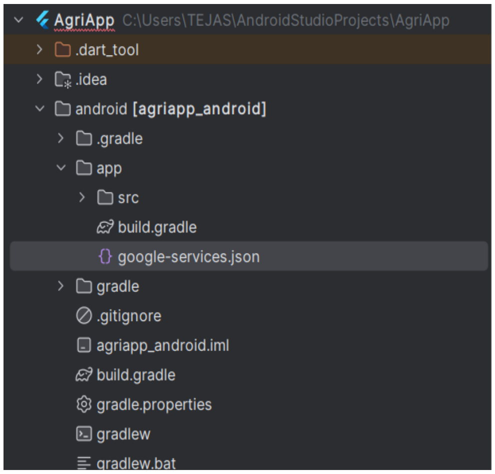
Switch to the **Project** view in Android Studio to see your project root directory.

Move your downloaded `google-services.json` file into your module (app-level) root directory.

Next

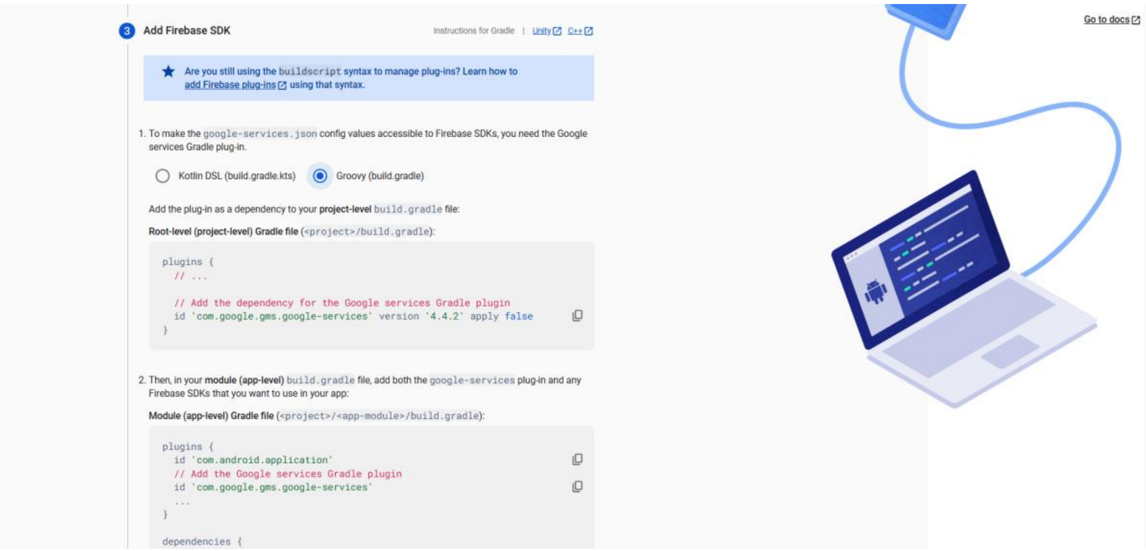
3 Add Firebase SDK

4 Next steps



```
AgriApp C:\Users\TEJAS\AndroidStudioProjects\AgriApp
  > .dart_tool
  > .idea
  v android [agriapp_android]
    > .gradle
    v app
      > src
        build.gradle
        google-services.json
    > gradle
      .gitignore
      agriapp_android.iml
      build.gradle
      gradle.properties
      gradlew
      gradlew.bat
```

2.4) Add Firebase SDK dependencies to android/build.gradle



3 Add Firebase SDK

Instructions for Gradle | Unity C++

Go to docs

★ Are you still using the `buildscript` syntax to manage plug-ins? Learn how to add Firebase plug-ins using that syntax.

1. To make the `google-services.json` config values accessible to Firebase SDKs, you need the Google services Gradle plug-in.

○ Kotlin DSL (build.gradle.kts) ● Groovy (build.gradle)

Add the plug-in as a dependency to your **project-level** `build.gradle` file:

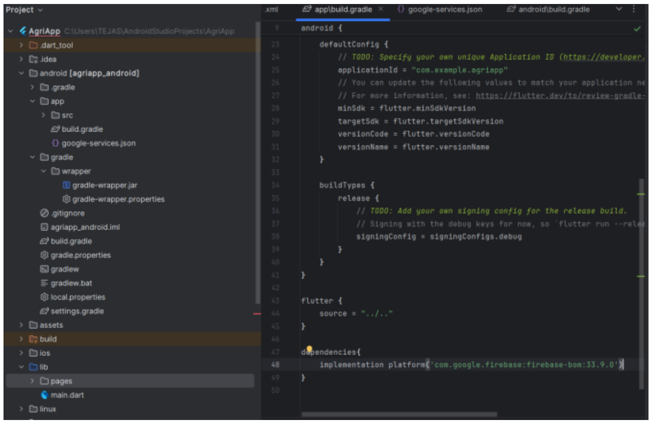
Root-level (project-level) Gradle file (`<project>/build.gradle`):

```
plugins {
  // ...

  // Add the dependency for the Google services Gradle plugin
  id 'com.google.gms.google-services' version '4.4.2' apply false
}
```

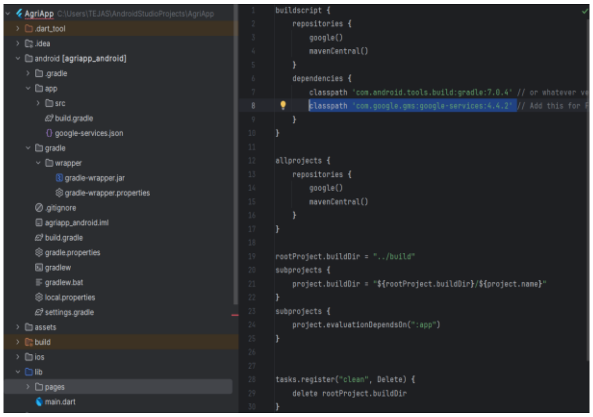
2. Then, in your **module (app-level)** `build.gradle` file, add both the `google-services` plug-in and any Firebase SDKs that you want to use in your app:

Module (app-level) Gradle file (`<project>/<app-module>/build.gradle`):

```
plugins {
  id 'com.android.application'
  // Add the Google services Gradle plugin
  id 'com.google.gms.google-services'
  ...
}

dependencies {
```



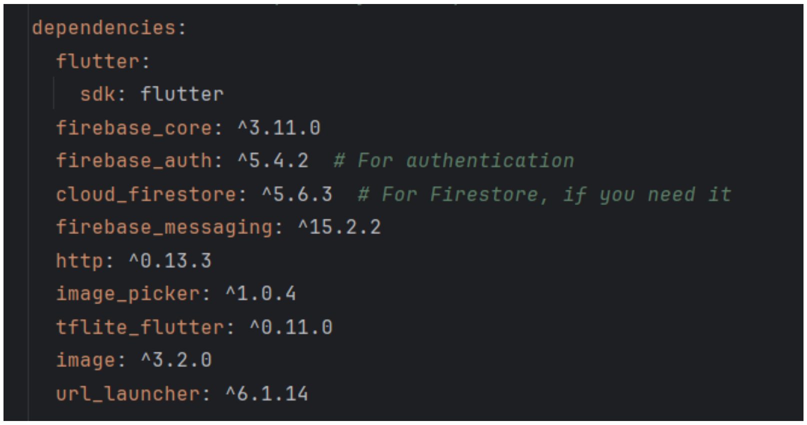
```
android {
    defaultConfig {
        // TODO: Specify your own unique Application ID (https://developer.
        applicationId = "com.example.agriapp"
        // You can update the following values to match your application ne
        // For more information, see: https://flutter.dev/to/review-gradle-
        minSdk = flutter.minSdkVersion
        targetSdk = flutter.targetSdkVersion
        versionCode = flutter.versionCode
        versionName = flutter.versionName
    }

    buildTypes {
        release {
            // TODO: Add your own signing config for the release build.
            // Signing with the debug keys for now, so `flutter run --relea
            signingConfig = signingConfigs.debug
        }
    }
}

flutter {
    source = "../.."
}

dependencies{
    implementation platform('com.google.firebase:firebase-bom:33.9.0')
}
```



```
buildscript {
    repositories {
        google()
        mavenCentral()
    }
    dependencies {
        classpath 'com.android.tools.build:gradle:7.0.4' // or whatever ve
        classpath 'com.google.gms:google-services:4.4.2' // Add this for F
    }
}

allprojects {
    repositories {
        google()
        mavenCentral()
    }
}

rootProject.buildDir = "../build"
subprojects {
    project.buildDir = "${rootProject.buildDir}/${project.name}"
}
subprojects {
    project.evaluationDependsOn(":app")
}


tasks.register("clean", Delete) {
    delete rootProject.buildDir
}
```

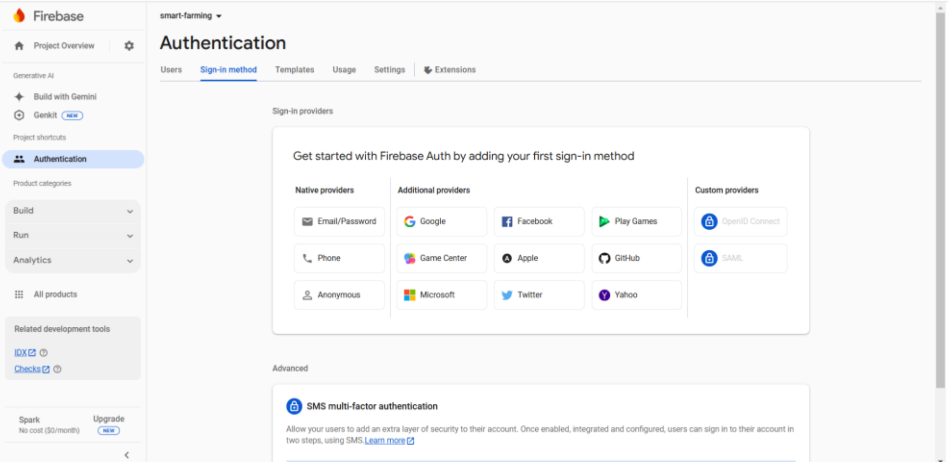
Step 3: - Add Firebase Authentication to Your App

3.1) Add Firebase Authentication Dependencies



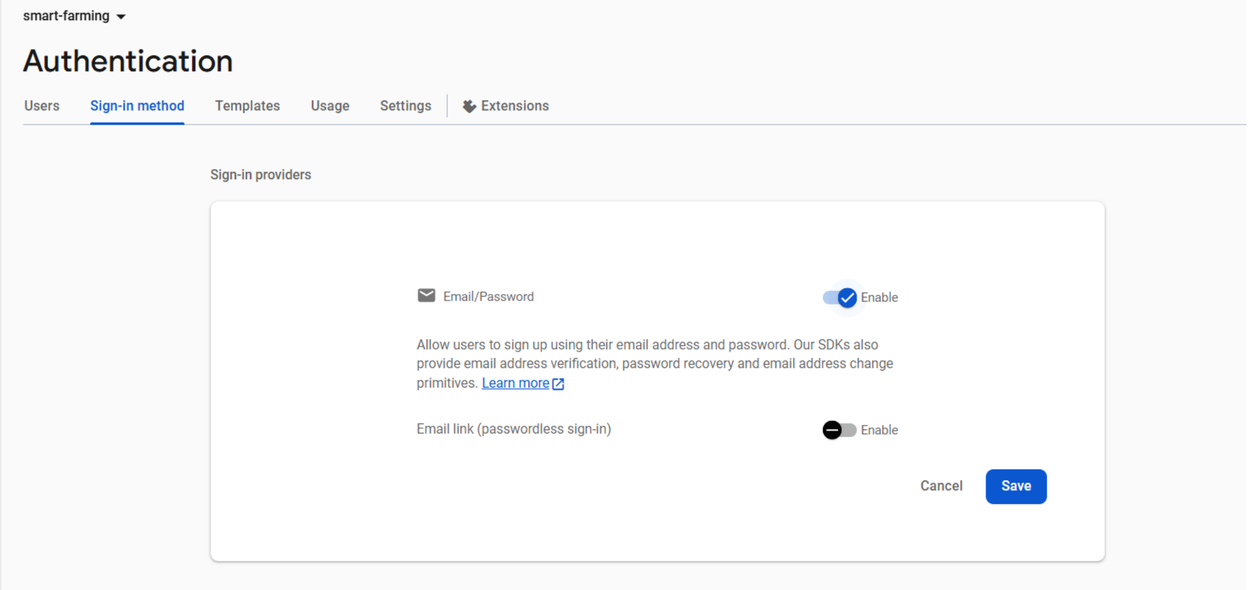
```
dependencies:
  flutter:
    sdk: flutter
  firebase_core: ^3.11.0
  firebase_auth: ^5.4.2  # For authentication
  cloud_firestore: ^5.6.3  # For Firestore, if you need it
  firebase_messaging: ^15.2.2
  http: ^0.13.3
  image_picker: ^1.0.4
  tflite_flutter: ^0.11.0
  image: ^3.2.0
  url_launcher: ^6.1.14
```

3.2) Enable Authentication in Firebase Console
Go to **Firebase Console → Authentication**.
Click on **Sign-in method** and enable **Email/Password** (or any other method like Google).
Click Save



Get started with Firebase Auth by adding your first sign-in method

Allow your users to add an extra layer of security to their account. Once enabled, integrated and configured, users can sign in to their account in two steps, using SMS. Learn more



Sign-in providers

Email/Password ☑ Enable

Allow users to sign up using their email address and password. Our SDKs also provide email address verification, password recovery and email address change primitives. Learn more

Email link (passwordless sign-in) ○ Enable

3.3) Implement Authentication in Flutter
Modify main.dart

```
import 'package:firebase_core/firebase_core.dart';
import 'package:firebase_auth/firebase_auth.dart';

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

Step 4: -Configure Firebase Realtime Database

4.1) Go to Firebase Console → Realtime Database.
4.2) Click **Create Database** → Choose location → Set rules (for development, set read/write to true).
4.3) Click **Publish**.

Code:-

Register_page.dart

```
class _RegistrationPageState extends
State<RegistrationPage> {
  final _formKey =
GlobalKey<FormState>();
  bool _isLoading = false;
  String _name = '';
  String _email = '';
  String _state = '';
  String _district = '';
  String _password = '';
  String _phone = '';

  final FirebaseAuth _auth =
FirebaseAuth.instance;

  // Firebase Registration
  Future<void> _handleRegistration() async
{
    if (_formKey.currentState!.validate()) {
      setState(() {
        _isLoading = true;
      });

      try {
        // Register user with Firebase
Authentication (Email and Password)
        UserCredential userCredential = await
_auth.createUserWithEmailAndPassword(
          email: _email,
          password: _password,
        );

        // Get the user ID from Firebase
Authentication user
        String userId =
userCredential.user!.uid;

        // Store extra user information in
Firestore
        await
FirebaseFirestore.instance.collection('users')
.doc(userId).set({
          'name': _name,
          'email': _email,
          'phone': _phone,
          'state': _state,
          'district': _district,
          // You can add other fields here
        });

        setState(() {
          _isLoading = false;
        });

        // Navigate to OTP verification page
after successful registration
        Navigator.pushReplacementNamed(
          context,
          '/otp',
          arguments: _phone, // Pass the phone
number here
        );
      } catch (e) {
        setState(() {
          _isLoading = false;
        });

ScaffoldMessenger.of(context).showSnackB
ar(
          SnackBar(content: Text("Registration
failed: $e")),
        );
      }
    }
  }
```

login_page.dart

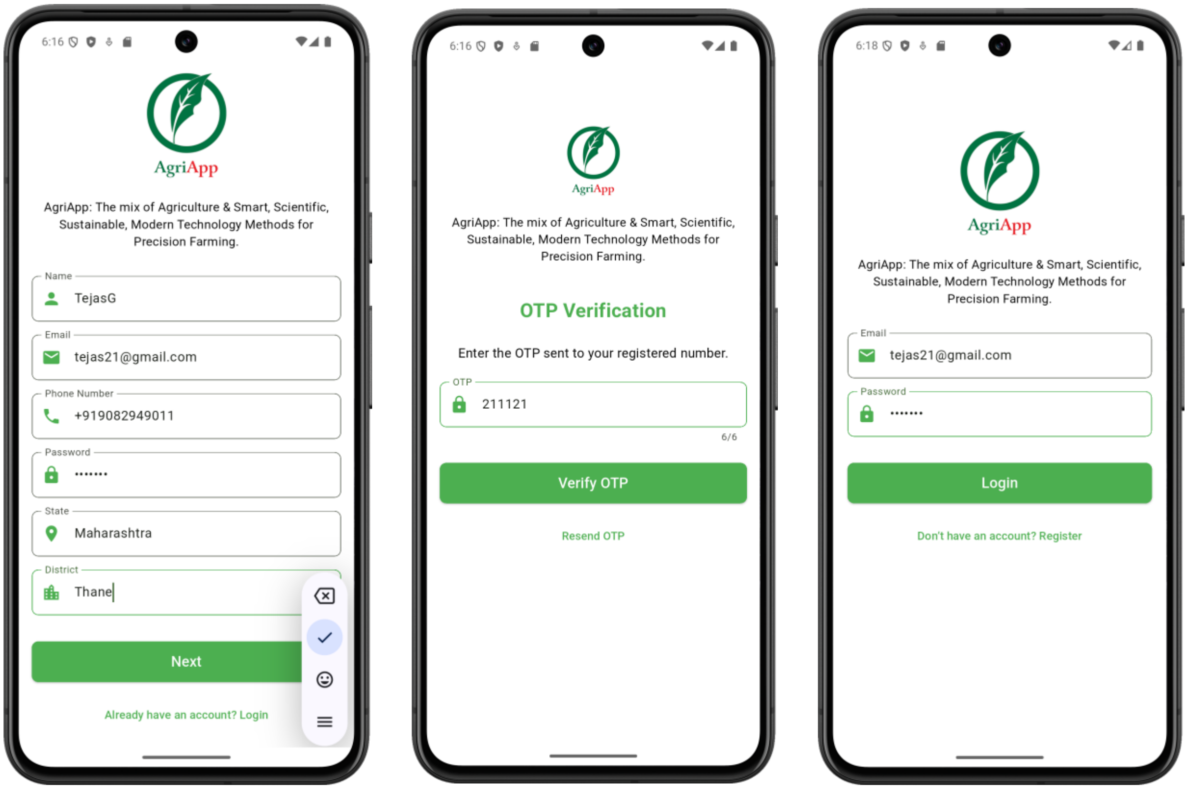
```
import 'package:flutter/material.dart';
import 'package:firebase_auth/firebase_auth.dart';

class LoginPage extends StatefulWidget {
  @override
  _LoginPageState createState() => _LoginPageState();
}

class _LoginPageState extends State<LoginPage> {
  final _formKey = GlobalKey<FormState>();
  final FirebaseAuth _auth = FirebaseAuth.instance;

  TextEditingController _emailController = TextEditingController();
  TextEditingController _passwordController = TextEditingController();

  bool _isLoading = false;
  String _errorMessage = ";

  Future<void> _loginUser() async {
    if (!_formKey.currentState!.validate()) return;

    setState(() {
      _isLoading = true;
      _errorMessage = ";
    });

    try {
      UserCredential userCredential = await _auth.signInWithEmailAndPassword(
        email: _emailController.text.trim(),
        password: _passwordController.text.trim(),
      );
      User? user = userCredential.user;

      if (user != null) {
        Navigator.pushReplacementNamed(context, '/home');
      } else {
        setState(() {
          _errorMessage = "Something went wrong. Please try again.";
        });
      }
    } on FirebaseAuthException catch (e) {
      setState(() {
        _errorMessage = e.message ?? "An error occurred. Please try again.";
      });
    }

    setState(() {
      _isLoading = false;
    });
  }
```

Myaccountpage.dart

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

class MyAccountPage extends
StatefulWidget {
  @override
  _MyAccountPageState createState() =>
_MyAccountPageState();
}

class _MyAccountPageState extends
State<MyAccountPage> {
  final _formKey =
GlobalKey<FormState>();
  String name = '';
  String phone = '';
  String email = '';
  String state = '';
  String language = 'English';
  List<String> myCrops = [];
  bool isLoading = true;

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

  // Fetch data from Firestore
  _fetchUserData() async {
    User? user =
FirebaseAuth.instance.currentUser;

    if (user != null) {
      try {
        DocumentSnapshot doc = await
FirebaseFirestore.instance
            .collection('users')
            .doc(user.uid)
            .get();

        if (doc.exists) {
          setState(() {
            name = doc['name'];
            phone = doc['phone'];
            email = doc['email'];
            state = doc['state'];
            language = doc['language'];
            myCrops =
List<String>.from(doc['myCrops']);
            isLoading = false;
          });
        }
      } catch (e) {
        print("Error fetching user data: $e");
        setState(() {
          isLoading = false;
        });
      }
    }
  }

  // Save updated data to Firestore
  _saveUserData() async {
    User? user =
FirebaseAuth.instance.currentUser;

    if (user != null) {
      try {
        await
FirebaseFirestore.instance.collection('users')
.doc(user.uid).update({
          'name': name,
          'phone': phone,
          'email': email,
          'state': state,
          'language': language,
          'myCrops': myCrops,
        });
        // Show success message
        ScaffoldMessenger.of(context).showSnackBar(SnackBar(content: Text('Changes saved!')));
      } catch (e) {
        print("Error saving user data: $e");
        ScaffoldMessenger.of(context).showSnackBar(SnackBar(content: Text('Error saving changes')));
      }
    }
  }
```

Main.dart

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

class MyApp extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      debugShowCheckedModeBanner: false,
      title: 'AgriApp',
      theme: ThemeData(
        primarySwatch: Colors.green,
        colorScheme: ColorScheme.fromSeed(seedColor: Color(0xFF6A9A5B)),
      ),
      home: FirebaseAuth.instance.currentUser == null ? LoginPage() : MainScreen(),
      routes: {
        '/login': (context) => LoginPage(),
        '/register': (context) => RegistrationPage(),
        '/otp': (context) => OtpVerificationPage(
          phoneNumber: ModalRoute.of(context)!.settings.arguments as String,
        ),
        '/home': (context) => MainScreen(),
      },
    );
  }
}

class MainScreen extends StatefulWidget {
  @override
  _MainScreenState createState() => _MainScreenState();
}

class _MainScreenState extends State<MainScreen> {
  int _currentIndex = 0;

  final List<Widget> _pages = [
    HomePage(),
    WeatherPage(),
    DiseaseDetectionPage(),
    CropListPage(),
    MyAccountPage(),
  ];
```



AgriApp

AgriApp: The mix of Agriculture & Smart, Scientific, Sustainable, Modern Technology Methods for Precision Farming.

Name: TejasG

Email: tejas21@gmail.com

Phone Number: +919082949011

Password: •••••••

State: Maharashtra

District: Thane

AgriApp

AgriApp: The mix of Agriculture & Smart, Scientific, Sustainable, Modern Technology Methods for Precision Farming.

OTP Verification

Enter the OTP sent to your registered number.

OTP: 211121

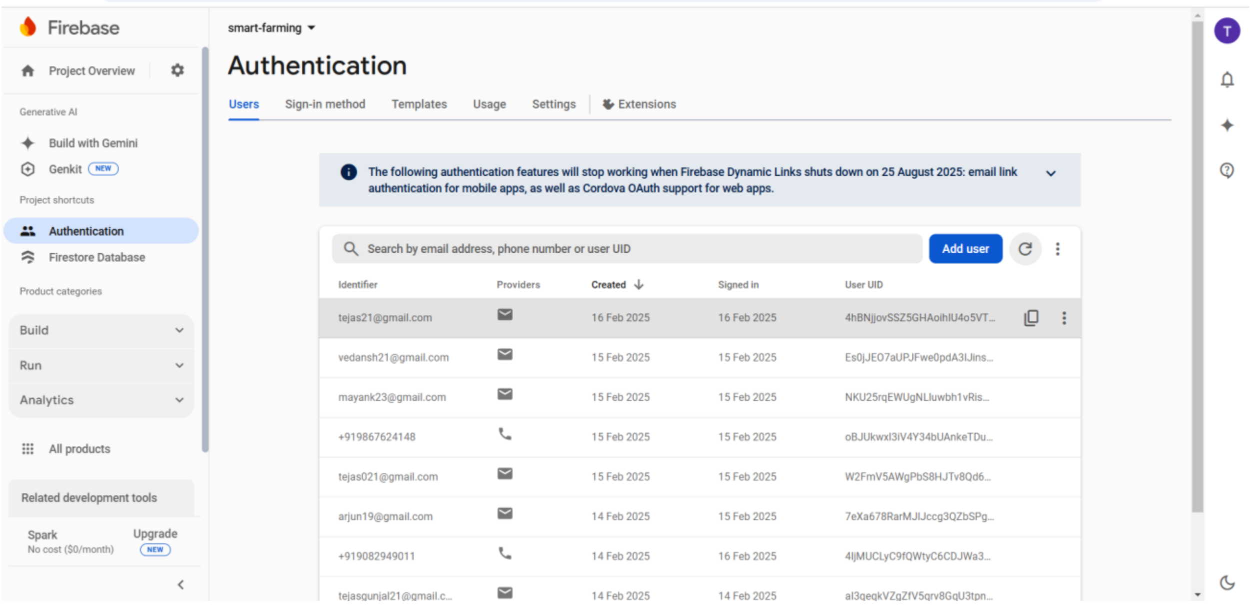
6/6

AgriApp

AgriApp: The mix of Agriculture & Smart, Scientific, Sustainable, Modern Technology Methods for Precision Farming.

Email: tejas21@gmail.com

Password: •••••••



Authentication

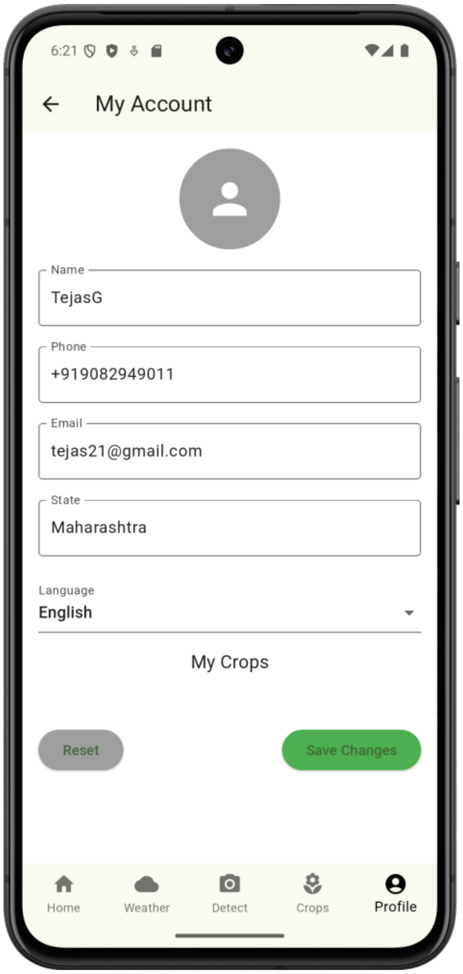
The following authentication features will stop working when Firebase Dynamic Links shuts down on 25 August 2025: email link authentication for mobile apps, as well as Cordova OAuth support for web apps.

Search by email address, phone number or user UID

Identifier	Providers	Created ↓	Signed in	User UID
tejas21@gmail.com	✉	16 Feb 2025	16 Feb 2025	4hBNjjovSSZ5GHAoihlU4o5VT...
vedansh21@gmail.com	✉	15 Feb 2025	15 Feb 2025	Es0jJEO7aUPJFwe0pdA3lJins...
mayank23@gmail.com	✉	15 Feb 2025	15 Feb 2025	NKU25rqEWUgNLluwbh1vRis...
+919867624148	☎	15 Feb 2025	15 Feb 2025	oBJUkwxl3iV4Y34bUAnkeTDu...
tejas021@gmail.com	✉	15 Feb 2025	15 Feb 2025	W2FmV5AWgPbS8HJTv8Qd6...
arjun19@gmail.com	✉	14 Feb 2025	15 Feb 2025	7eXa678RarMJlJccg3QZbSPg...
+919082949011	☎	14 Feb 2025	16 Feb 2025	4ljMUCLyC9fQWtyC6CDJWa3...
tejasgunjal21@gmail.c...	✉	14 Feb 2025	14 Feb 2025	al3qeqkVZgZfV5qrv8GqU3tpn...

After Registering, the user details is saved in the database.

User details get fetched from the database in the profile page.



6:21

My Account

Name: TejasG

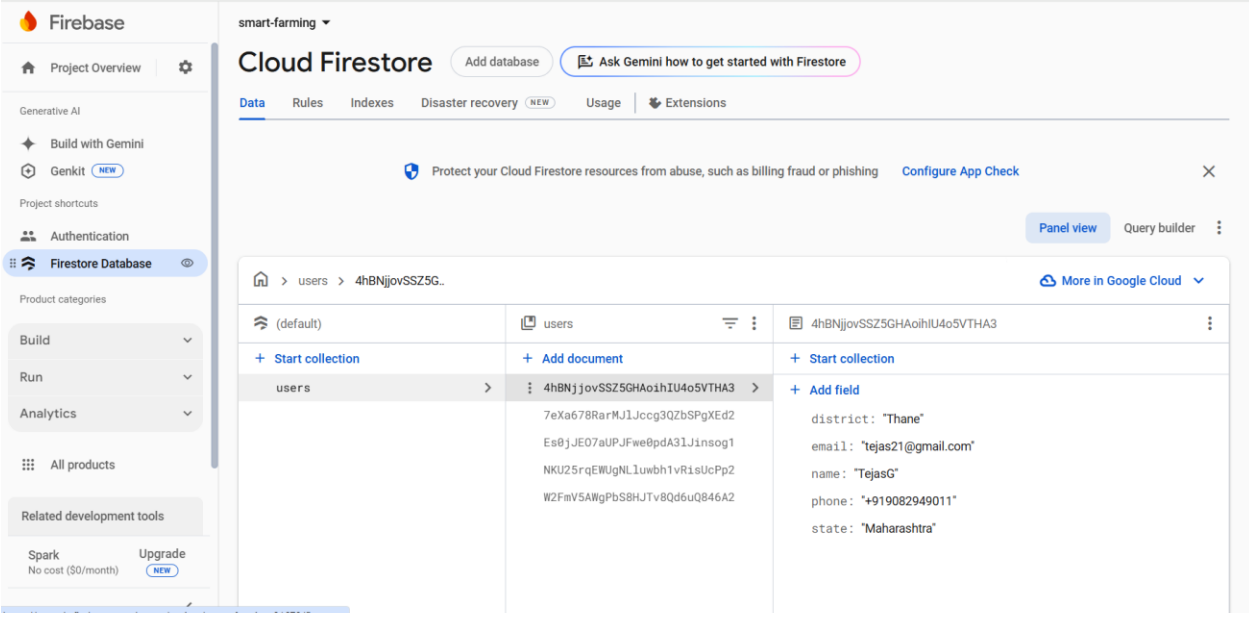
Phone: +919082949011

Email: tejas21@gmail.com

State: Maharashtra

Language: English

My Crops



Firebase

smart-farming

Cloud Firestore

Protect your Cloud Firestore resources from abuse, such as billing fraud or phishing   Configure App Check

users > 4hBNjjovSSZ5G..

(default) | users | 4hBNjjovSSZ5GHAoihIU4o5VTHA3

users

4hBNjjovSSZ5GHAoihIU4o5VTHA3
7eXa678RarMJlJccg3QZbSPgXEd2
Es0jJEO7aUPJFwe0pdA3lJinsog1
NKU25rqEWUgNLluwbh1vRisUcPp2
W2FmV5AWgPbS8HJTv8Qd6uQ846A2

district: "Thane"
email: "tejas21@gmail.com"
name: "TejasG"
phone: "+919082949011"
state: "Maharashtra"