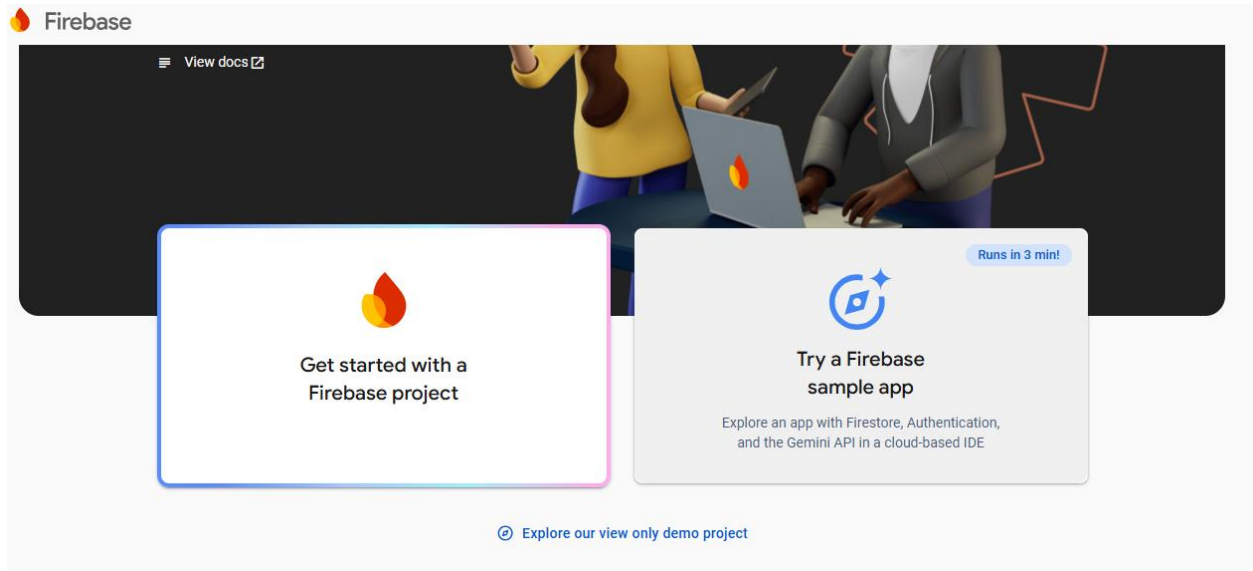


**FLUTTER BOOTCAMP 2025**  
**JAN 23- JAN 26, 2025**  
**by ABDUL SAMI**  
**INTRODUCTION TO FLUTTER**  
**AND DART**  
**FINAL PROJECT**  
**KANWAL NOOR UL AIN**



## × Create a project

Project name

flutterapp

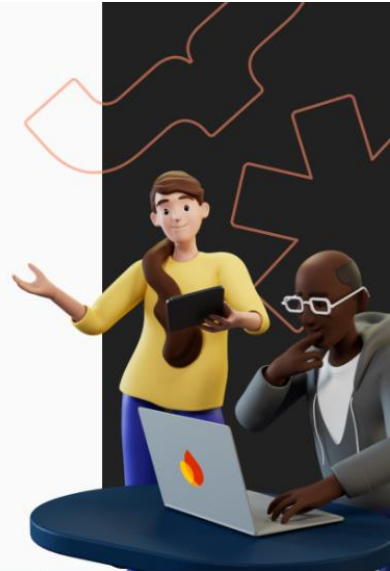
flutterapp-e726d

☒ I accept the [Firebase terms](#).

☒ I confirm that I will use Firebase exclusively for purposes relating to my trade, business, craft or profession.

Already have a Google Cloud project?  
[Add Firebase to Google Cloud project](#)

Continue



## × Create a project

craft or profession.

Data-sharing settings and Google Analytics terms

☒ Use the default settings for sharing Google Analytics data. [Learn more](#)

- ☐ Share your Analytics data with Google to improve Google Products and Services
- ☒ Share your Analytics data with Google to enable Benchmarking
- ☒ Share your Analytics data with Google to enable Technical Support
- ☒ Share your Analytics data with Google Account Specialists

☒ I accept the [Google Analytics Terms](#)

Upon project creation, a new Google Analytics property will be created and linked to your Firebase project. This link will enable data flow between the products. Data exported from your Google Analytics property into Firebase is subject to the [Firebase Terms of Service](#), while Firebase data imported into Google Analytics is subject to the [Google Analytics Terms of Service](#). [Learn more](#)

Previous

Create project

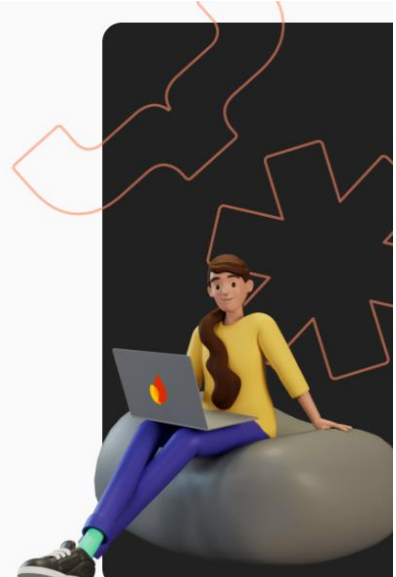




flutterapp

✓ Your Firebase project is ready

Continue



Your apps

There are no apps in your project

Select a platform to get started



 Delete project



## 1

②

```
com.company.appname
```

②

My Android App

②

00:00:00:00:00:00:00:00:00:00:00:00:00:00:00:00:!



Edit SHA-1s in settings.


Register app

Android package name (?)

com.first.flutter\_note\_app

App nickname(optional) 

my first app

Debug signing certificate SHA-1 (optional) 

00:00:00:00:00:00:00:00:00:00:00:00:00:00:00:00:(

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

## 2 Download and then add config file

Instructions for Android Studio below | [Unity](#) [C++](#)

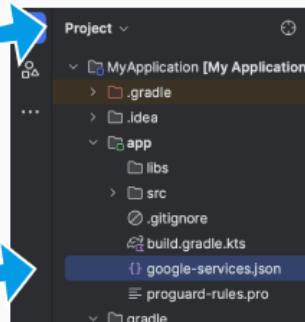
[Download google-services.json](#)

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.



google-services.json



Next

### 3 Add Firebase SDK

Instructions for Gradle | [Unity](#) [C++](#)

★ 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`)

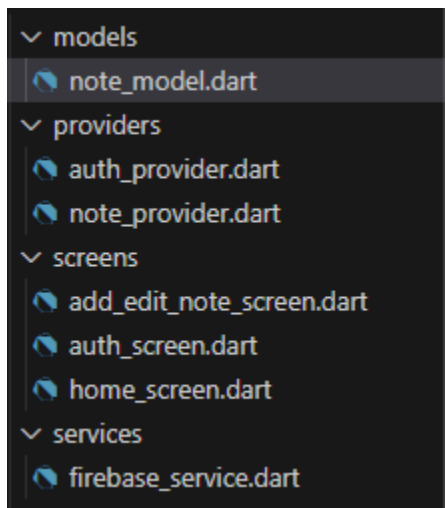
Add the plug-in as a dependency to your **project-level** `build.gradle.kts` file:

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

```
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.kts` 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.kts`):



### **main.dart**

```
import 'package:flutter/material.dart';  
  
import 'package:firebase_core/firebase_core.dart';  
  
import 'package:provider/provider.dart';  
  
import 'firebase_options.dart';  
  
import 'providers/auth_provider.dart';
```

```
import 'providers/note_provider.dart';
import 'screens/auth_screen.dart';
import 'screens/home_screen.dart';

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

class MyApp extends StatelessWidget {
  const MyApp({super.key});

  @override
  Widget build(BuildContext context) {
    return MultiProvider(
      providers: [
        ChangeNotifierProvider(create: (context) => AuthProvider()),
        ChangeNotifierProvider(create: (context) => NoteProvider()),
      ],
      child: MaterialApp(
        debugShowCheckedModeBanner: false,
        title: 'Note App',
        theme: ThemeData(primarySwatch: Colors.blue),
        home: Consumer<AuthProvider>(
          builder: (context, authProvider, _) {
            return authProvider.isAuthenticated ? HomeScreen() : AuthScreen();
          },
        ),
      ),
    );
  }
}
```

```
    ),  
    );  
  }  
}
```

### **Provider.dart**

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

```
class AuthProvider with ChangeNotifier {  
  final FirebaseAuth _auth = FirebaseAuth.instance;  
  User? _user;  
  
  User? get user => _user;  
  bool get isAuthenticated => _user != null;
```

```
  AuthProvider() {  
    _auth.authStateChanges().listen((user) {  
      _user = user;  
      notifyListeners();  
    });  
  }
```

```
  Future<void> signIn(String email, String password) async {  
    await _auth.signInWithEmailAndPassword(email: email, password: password);  
  }
```

```
  Future<void> signUp(String email, String password) async {  
    await _auth.createUserWithEmailAndPassword(email: email, password: password);
```



```
}
```

```
Future<void> signOut() async {
```

```
  await _auth.signOut();
```

```
}
```

```
}
```

### **model.dart**

```
class Note {
```

```
  final String id;
```

```
  final String title;
```

```
  final String content;
```

```
  final DateTime createdAt;
```

```
Note({required this.id, required this.title, required this.content, required this.createdAt});
```

```
Map<String, dynamic> toMap() {
```

```
  return {'id': id, 'title': title, 'content': content, 'createdAt': createdAt.millisecondsSinceEpoch};
```

```
}
```

```
factory Note.fromMap(Map<String, dynamic> map) {
```

```
  return Note(
```

```
    id: map['id'],
```

```
    title: map['title'],
```

```
    content: map['content'],
```

```
    createdAt: DateTime.fromMillisecondsSinceEpoch(map['createdAt']),
```

```
  );
```

```
}
```

```
}
```

### **service.dart**

```
import 'package:cloud_firestore/cloud_firestore.dart';
import '../models/note_model.dart';

class FirebaseService {
  final CollectionReference notesCollection = FirebaseFirestore.instance.collection('notes');

  Future<void> addNote(Note note) async {
    await notesCollection.doc(note.id).set(note.toMap());
  }

  Future<void> updateNote(Note note) async {
    await notesCollection.doc(note.id).update(note.toMap());
  }

  Future<void> deleteNote(String notelId) async {
    await notesCollection.doc(notelId).delete();
  }

  Stream<List<Note>> getNotes() {
    return notesCollection.snapshots().map((snapshot) {
      return snapshot.docs.map((doc) => Note.fromMap(doc.data() as Map<String, dynamic>)).toList();
    });
  }
}
```

### **provider.dart**

```
import 'package:flutter/material.dart';
import '../models/note_model.dart';
```

```
import '../services/firebase_service.dart';

class NoteProvider with ChangeNotifier {
  final FirebaseService _firebaseService = FirebaseService();
  List<Note> _notes = [];

  List<Note> get notes => _notes;

  void fetchNotes() {
    _firebaseService.getNotes().listen((notes) {
      _notes = notes;
      notifyListeners();
    });
  }

  Future<void> addNote(Note note) async {
    await _firebaseService.addNote(note);
  }

  Future<void> updateNote(Note note) async {
    await _firebaseService.updateNote(note);
  }

  Future<void> deleteNote(String noteId) async {
    await _firebaseService.deleteNote(noteId);
  }
}
```

**screen.dart**

```
import 'package:flutter/material.dart';
import 'package:provider/provider.dart';
import '../providers/auth_provider.dart';

class AuthScreen extends StatelessWidget {
  final TextEditingController emailController = TextEditingController();
  final TextEditingController passwordController = TextEditingController();

  @override
  Widget build(BuildContext context) {
    final authProvider = Provider.of<AuthProvider>(context);

    return Scaffold(
      appBar: AppBar(title: Text("Login / Signup")),
      body: Column(
        children: [
          TextField(controller: emailController, decoration: InputDecoration(labelText: "Email")),
          TextField(controller: passwordController, decoration: InputDecoration(labelText: "Password"),
            obscureText: true),
          ElevatedButton(onPressed: () => authProvider.signIn(emailController.text, passwordController.text),
            child: Text("Login")),
          TextButton(onPressed: () => authProvider.signUp(emailController.text, passwordController.text),
            child: Text("Signup")),
        ],
      ),
    );
  }
}
```

## Login / Signup

Email

Password

Login

Signup

## Login / Signup

Email

kanwal@gmail.com

Password

\*\*\*\*\*

Login

Signup

## Home Screen



Load Notes

## Home Screen



Load Notes

Note 1  
This is note 1

Note 2  
This is note 2