

Aim: To connect Flutter app with Firebase.

Theory:

Firebase is a mobile and web application development platform that provides a variety of services to help developers build high-quality apps quickly. It offers features like real-time database, authentication, cloud messaging, storage, hosting, machine learning, and more, all integrated into a single platform. Firebase is developed by Google and is widely used by developers to build scalable and feature-rich applications.

Key components of Firebase:

1. **Realtime Database:** Firebase Realtime Database is a cloud-hosted NoSQL database that allows developers to store and sync data between users in real-time. It uses data synchronization and operates on JSON data, making it ideal for applications that require real-time updates.
2. **Authentication:** Firebase Authentication provides easy-to-use authentication services, including email/password authentication, social authentication (Google, Facebook, Twitter, etc.), and anonymous authentication. It allows developers to authenticate users quickly and securely.
3. **Cloud Firestore:** Firestore is a flexible, scalable database for mobile, web, and server development. It offers powerful querying capabilities, offline support, real-time synchronization, and automatic scaling, making it suitable for building complex applications.
4. **Cloud Storage:** Firebase Cloud Storage allows developers to store and serve user-generated content, such as images, videos, and files, directly from Google's infrastructure. It offers secure uploads and downloads, robust access controls, and integration with Firebase Authentication.
5. **Cloud Messaging:** Firebase Cloud Messaging (FCM) enables developers to send notifications and messages to users across platforms (iOS, Android, and web). It supports both notification messages and data messages, allowing developers to engage with users effectively.
6. **Hosting:** Firebase Hosting provides fast and secure web hosting for static and dynamic content. It allows developers to deploy web apps quickly, serve content over HTTPS, and integrate with other Firebase services seamlessly.

Connecting Firebase to Flutter App:

To connect Firebase to a Flutter app, follow these steps:

1. **Create a Firebase Project:** Go to the Firebase Console (<https://console.firebase.google.com/>) and create a new project. Follow the setup instructions to configure your project.
2. **Add Firebase to Your App:** In your Flutter project, add the Firebase SDK dependencies by including the necessary plugins in your pubspec.yaml file. For example, to use Firebase Authentication, you would add the `firebase_auth` plugin. Run `flutter pub get` to install the dependencies.
3. **Initialize Firebase:** In your Flutter app, initialize Firebase by calling `Firebase.initializeApp()` in the `main()` function:
4. **Use Firebase Services:** Once Firebase is initialized, you can use Firebase services in your Flutter app. For example, to authenticate users with Firebase Authentication, you can use the `FirebaseAuth` class:
5. **Configure Firebase Services:** Depending on the Firebase services you're using, you may need to configure them in the Firebase Console. For example, for Firebase Authentication, you need to set up sign-in methods like email/password, Google sign-in, etc.
6. **Test Your App:** Test your Flutter app to ensure that Firebase integration is working correctly. You can use Firebase Emulators for local development and testing.

Setup firebase on google console:

X

Create a project (Step 1 of 3)

Let's start with a name for your project[?]

Project name

flutter firebase app

Generating...

☒ I accept the [Firebase terms](#)

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

[illegible]

2 Download and then add config file

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

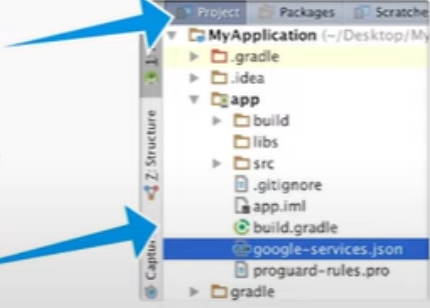
[Download google-services.json](#)

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



Add plugins to build.gradle file:

```
apply plugin: 'com.android.application'
apply plugin: 'com.google.gms.google-services'
apply plugin: 'kotlin-android'
apply from: "$flutterRoot/packages/flutter_tools/gradle/flutter.gradle"
```

Add classpath in dependencies in build.gradle file:

```
android > build.gradle
1  buildscript {
2      ext.kotlin_version = '1.7.10'
3      repositories {
4          google()
5          mavenCentral()
6      }
7
8      dependencies {
9          classpath 'com.android.tools.build:gradle:7.3.0'
10         classpath 'com.google.gms:google-services:4.3.8'
11         classpath "org.jetbrains.kotlin:kotlin-gradle-plugin:$kotlin_version"
12     }
13 }
```

Add following code in main.dart file:

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

Run | Debug | Profile
Future<void> main() async {
  WidgetsFlutterBinding.ensureInitialized();
  await Firebase.initializeApp();
  runApp(const MyApp());
}
```

After running the app firebase will get connected to our flutter application.

Conclusion:

Overall, the experiment demonstrated that integrating a Flutter app with Firebase offers a robust solution for building modern, feature-rich, and scalable applications with ease. The combination of Flutter's UI framework and Firebase's backend services provides developers with a powerful toolkit for creating high-quality cross-platform applications.