

Aim: To Connect Flutter UI with fireBase database

Theory:

What is Firebase?

Firebase is a platform developed by Google that offers Backend-as-a-Service (BaaS) to support the development of web and mobile applications. It provides a wide range of tools and services such as Firestore (a cloud-hosted NoSQL database), Realtime Database, Firebase Authentication, Cloud Storage, Cloud Functions, Analytics, Crash Reporting, and Push Notifications. These services allow developers to focus more on frontend development while Firebase handles backend tasks like data storage, user authentication, and serverless computing.

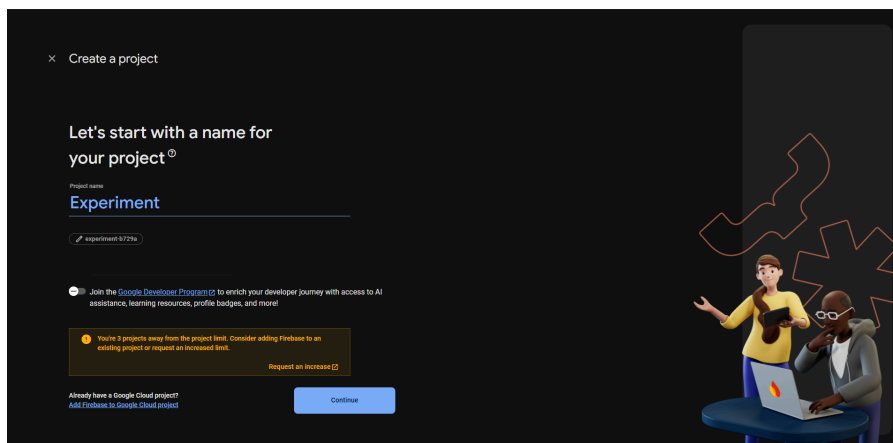
Why is Firebase Used?

Firebase is widely used in application development due to its real-time capabilities, scalability, and ease of integration. It allows developers to synchronize data across clients in real time without needing to manage their own servers. Firebase provides simple APIs and SDKs that reduce development time and complexity, especially for real-time applications like chat apps, collaborative tools, and live dashboards. Additionally, Firebase supports user authentication, secure data access, and automatic scaling, making it suitable for both small and large-scale applications.

Steps to Connect Firebase with Flutter UI

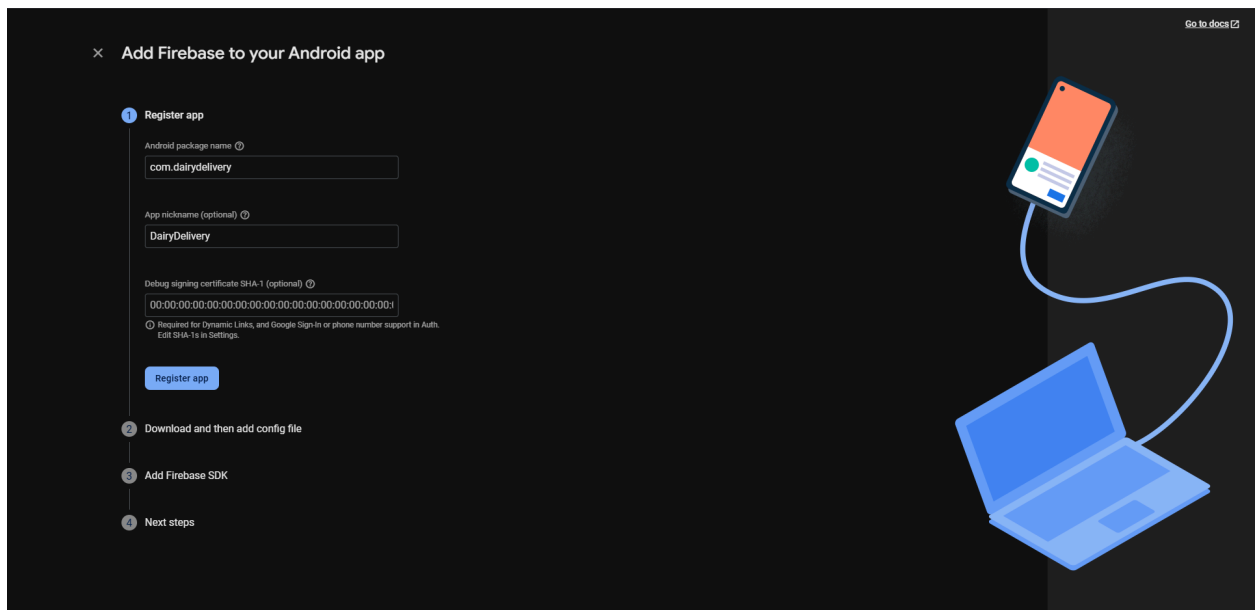
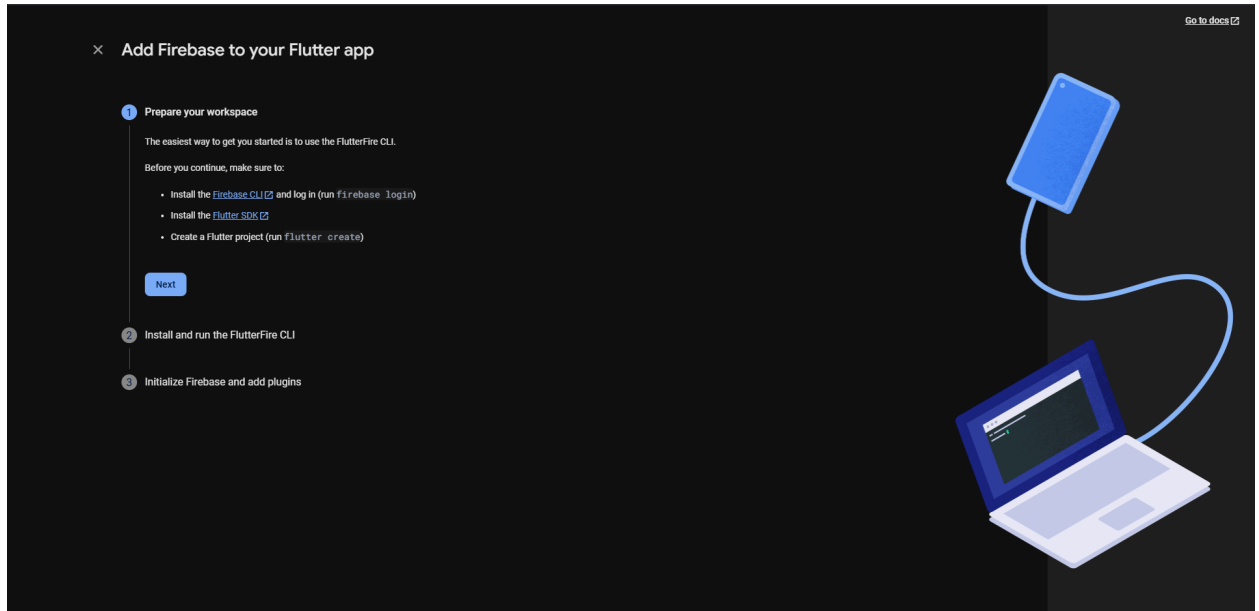
1. Create a Firebase Project

Go to the Firebase Console (<https://console.firebase.google.com/>) and create a new project by providing a project name and enabling necessary services like Google Analytics if required.



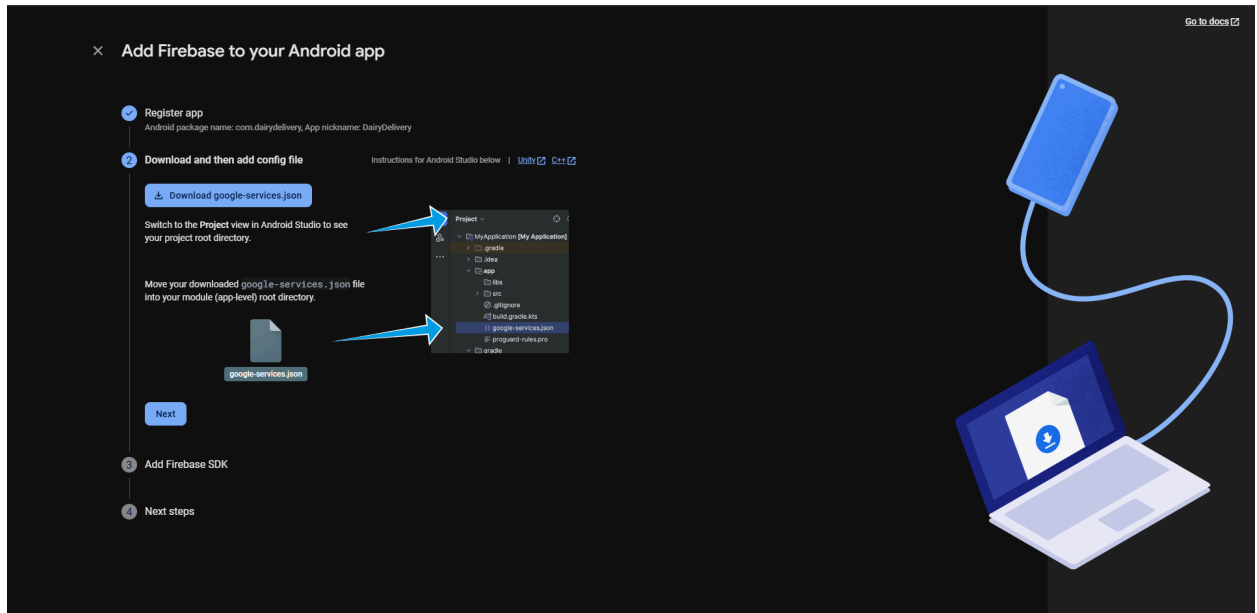
2. Register Your App with Firebase

In the Firebase console, add a new application by selecting the Android or iOS platform. For Android, provide the package name, nickname, and SHA-1 key (if needed).



3. Download and Add Configuration File

For Android, download the google-services.json file and place it in the android/app directory of your Flutter project. For iOS, download the GoogleService-Info.plist file and add it to the iOS runner project using Xcode.



4. Add Firebase Dependencies

Open your pubspec.yaml file and add necessary Firebase dependencies, such as:

dependencies:

firebase_core: latest_version

cloud_firestore: latest_version

firebase_auth: latest_version

Then run **flutter pub get** to fetch the packages.

5. Configure Firebase in Your App

Import and initialize Firebase in your Flutter app. In the main.dart file, ensure Firebase is initialized before the app runs:

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

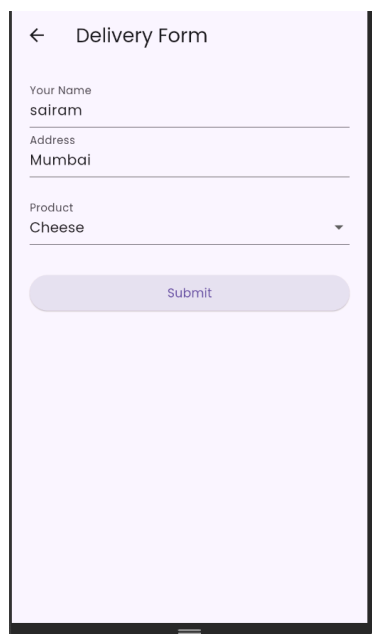
6. Use Firebase Services in the UI

After initializing Firebase, you can use its services within your UI. For example, you can use FirebaseAuth for authentication and Firestore to fetch or write data in the app interface.

7. Update Android and iOS Configuration

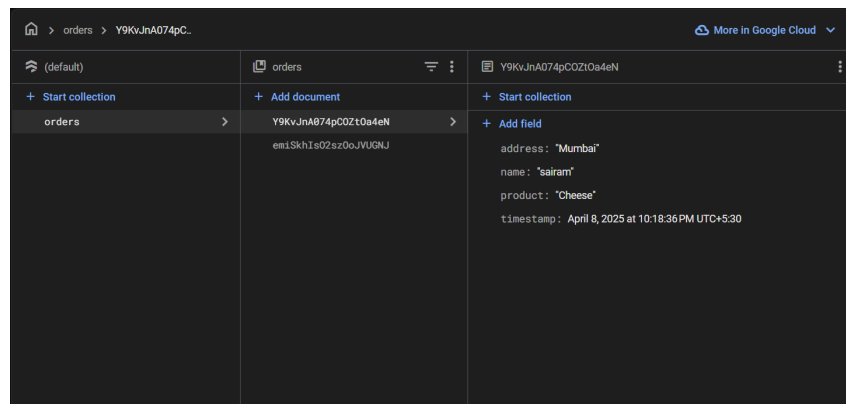
For Android, make sure to apply the Google Services plugin in android/build.gradle and android/app/build.gradle. For iOS, ensure the platform version is at least 10.0 and required permissions are configured in Info.plist.

Output:



A screenshot of a mobile application interface titled "Delivery Form". The form contains three input fields: "Your Name" with the value "sairam", "Address" with the value "Mumbai", and "Product" with a dropdown menu showing "Cheese". Below the fields is a rounded rectangular "Submit" button. The background is a light purple color.

Form filled



Data updated in firestore database

Conclusion: Integrating Firebase with a Flutter application provides a powerful and efficient way to manage backend services such as data storage, user authentication, and real-time synchronization. Firebase simplifies many of the challenges developers face when building scalable and secure applications by offering ready-to-use tools and services. With straightforward integration steps and official Flutter support, developers can quickly connect their UI to Firebase and focus on delivering rich user experiences without worrying about backend infrastructure. This makes Firebase an ideal choice for both small prototypes and full-scale production applications.