

Experiment No. 06

Aim : To Connect Flutter UI with fireBase database

Theory :

Firebase and Flutter are two powerful tools for developing modern mobile and web applications. Firebase is a platform developed by Google that provides a suite of tools to develop and grow apps, while Flutter is a UI toolkit developed by Google for building natively compiled applications for mobile, web, and desktop from a single codebase. Let's delve into key concepts, features, and steps for both Firebase and Flutter:

Firebase

Key Concepts:

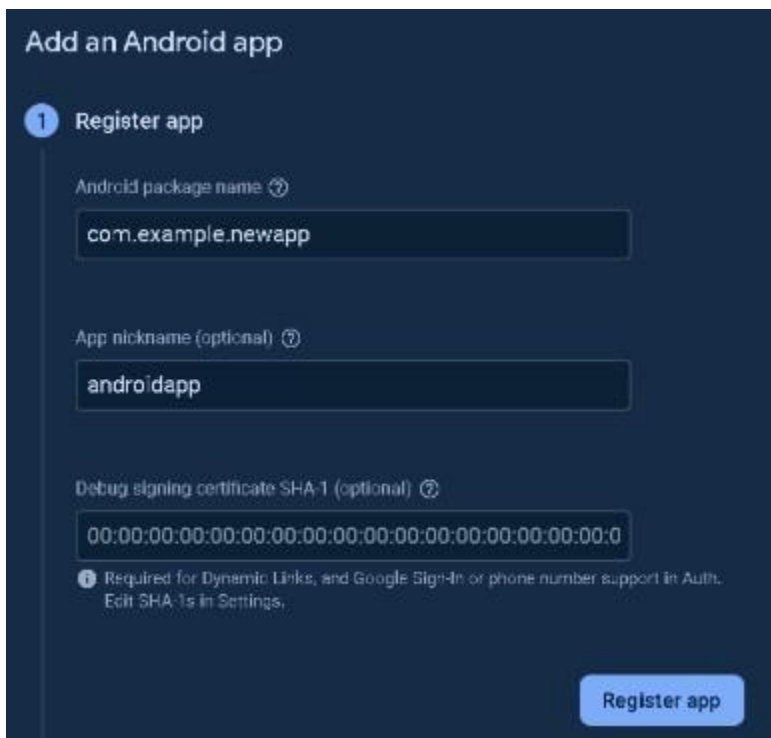
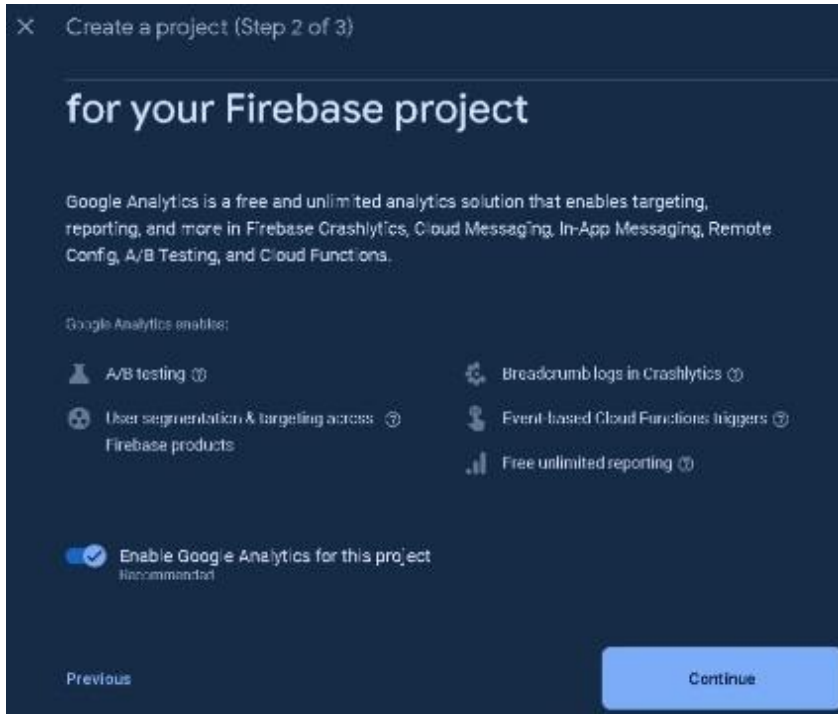
- **Realtime Database:** A NoSQL cloud database that supports data syncing in real time.
- **Authentication:** Provides ready-to-use authentication services like email/password, social logins, and more.
- **Cloud Firestore:** A flexible, scalable database for mobile, web, and server development.
- **Cloud Functions:** Allows you to run backend code in response to events triggered by Firebase features and HTTPS requests.
- **Cloud Storage:** For storing user-generated content like photos and videos.
- **Firebase Hosting:** Fast and secure web hosting for your web app's static and dynamic content.
- **Analytics:** Provides insights into user behavior and app usage.
- **Performance Monitoring:** Allows you to gain insights into your app's performance and stability.
- **Remote Config:** Change the behavior and appearance of your app without publishing an app update.
- **Crashlytics:** A tool to track, prioritize, and fix stability issues that erode app quality.

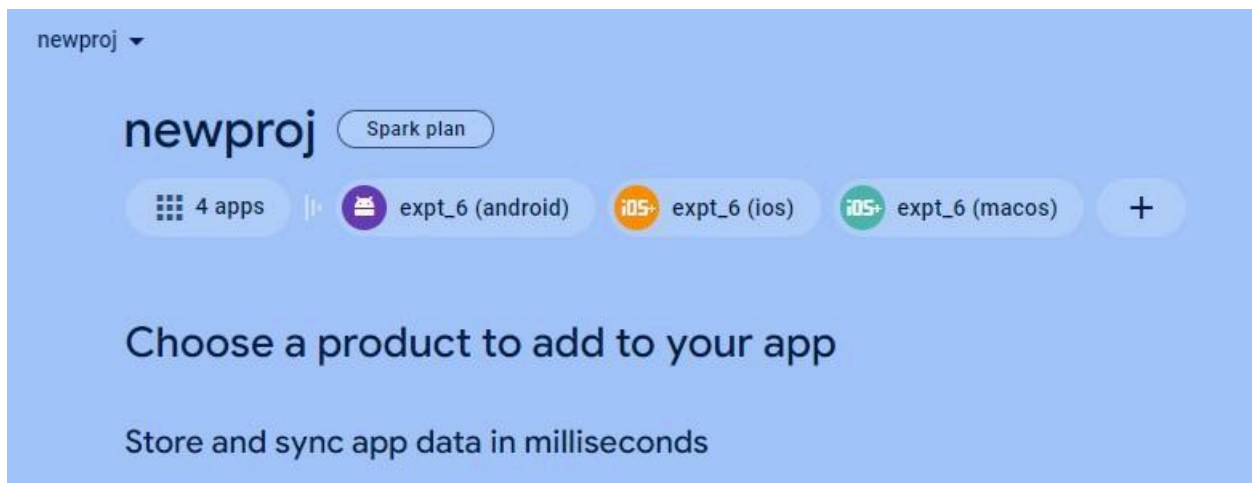
Features:

- Easy Integration: Firebase can be easily integrated into both iOS and Android applications.
- Real-time Updates: Real-time database and Firestore provide seamless updates to connected clients.
- Authentication: Simplifies user authentication with ready-to-use services.
- Scalability: Firebase automatically scales with your usage, handling hundreds to millions of users.
- Analytics and A/B Testing: Understand your users better with analytics and improve user engagement with A/B testing.
- Hosting and Functions: Allows you to host your web app and write serverless functions easily.
- Cost Effective: Firebase has a generous free tier, making it cost-effective for small to medium-sized apps.

Step 1 - Create a Firebase Project

Go to the Firebase Console, create a new project, and follow the setup instructions.





Add Firebase to Your App:

- For Flutter, you'll add the Firebase SDK to your `pubspec.yaml` file.
- Follow the setup instructions provided by Firebase for each service you want to use.

3 Add Firebase SDK

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

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

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

☐ Kotlin DSL (`build.gradle.kts`) ☒ Groovy (`build.gradle`)

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

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

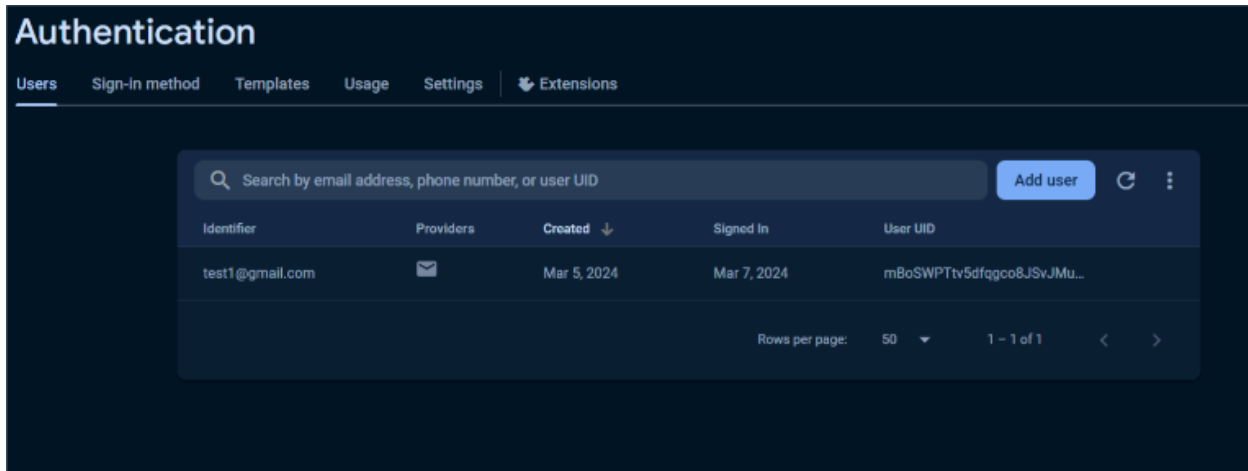
```
plugins {  
    // ...  
  
    // Add the dependency for the Google services Gradle plugin  
    id 'com.google.gms.google-services' version '4.4.1' apply false  
}
```

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

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

```
plugins {  
    id 'com.android.application'  
    // Add the Google services Gradle plugin  
    id 'com.google.gms.google-services'  
    ...  
}  
  
dependencies {  
    // Import the Firebase BoM  
    implementation platform('com.google.firebase:firebase-bom:32.7.2')  
  
    // TODO: Add the dependencies for Firebase products you want to use  
    // When using the BoM, don't specify versions in Firebase dependencies  
    implementation 'com.google.firebase:firebase-analytics'  
  
    // Add the dependencies for any other 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](#)



Initialize Firebase:

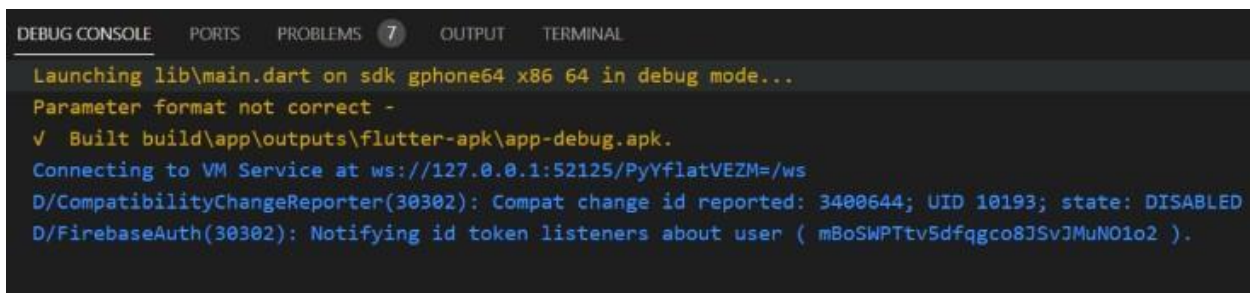
- In your Flutter app, initialize Firebase in the `main.dart` file or wherever appropriate.
- This step typically involves configuring Firebase services with your app's credentials.

Use Firebase Services:

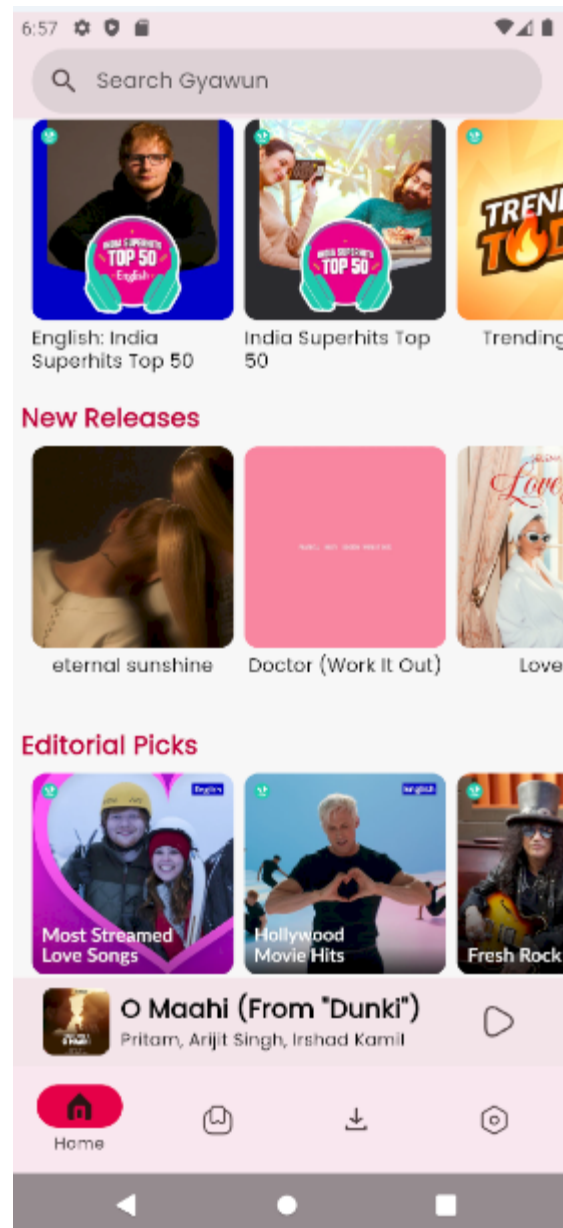
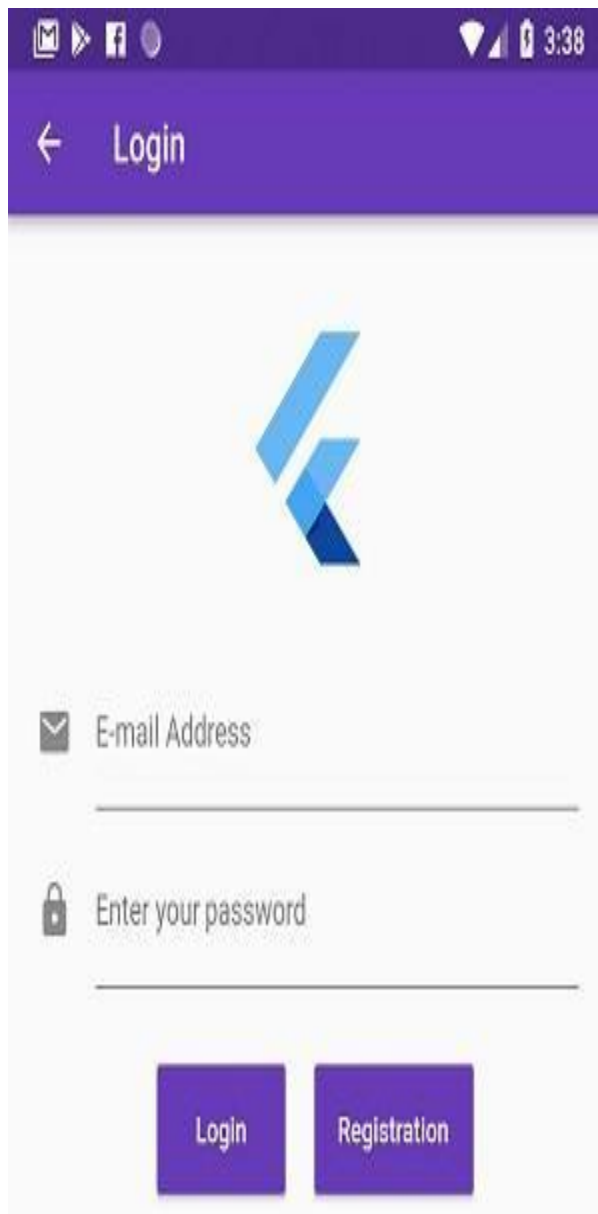
- Use Firebase services like Realtime Database, Firestore, Authentication, etc., in your Flutter app as needed.
- Firebase provides detailed documentation and code examples for each service.

```
dependencies:  
  flutter:  
    sdk: flutter  
  
  firebase_core: ^2.26.0  
  firebase_auth: ^4.17.7
```

After Running the Firebase Code:



```
DEBUG CONSOLE  PORTS  PROBLEMS  7  OUTPUT  TERMINAL
Launching lib\main.dart on sdk gphone64 x86 64 in debug mode...
Parameter format not correct -
√ Built build\app\outputs\flutter-apk\app-debug.apk.
Connecting to VM Service at ws://127.0.0.1:52125/PyYflatVEZM=/ws
D/CompatibilityChangeReporter(30302): Compat change id reported: 3400644; UID 10193; state: DISABLED
D/FirebaseAuth(30302): Notifying id token listeners about user ( mBoSWPTtv5dfqgco8JSvJMuNO1o2 ).
```



Conclusion : From this experiment, first of all we studied how to set up a firebase. Then we created a firebase project, added multiple dependencies and packages to our flutter project so as to integrate our flutter project with firebase. Next step we authenticate the input fields from our app like email and password using Firebase and store it.