Yogeshchandra Paranjpe
D15B 52
MPL Lab Exp 6

AIM : To connect Flutter app UI with Firebase database

THEORY :
To integrate Flutter with Firebase, begin by setting up a Firebase project, linking your Flutter app, and configuring dependencies in the `pubspec.yaml` file. Initialize Firebase in your app's entry point and consider adding the optional `firebase_auth` package for authentication.

For database operations, utilize the `cloud_firestore` package to interact with Firestore, performing CRUD operations and enabling real-time data updates. If your app involves file storage, integrate Firebase Storage using the `firebase_storage` package. Implement robust error-handling mechanisms for asynchronous Firebase operations. Thoroughly test and debug your app, ensuring seamless functionality on both iOS and Android platforms. Once satisfied, deploy your Flutter app, now connected to Firebase, to make the most of the platform's features for authentication, real-time databases, and storage.

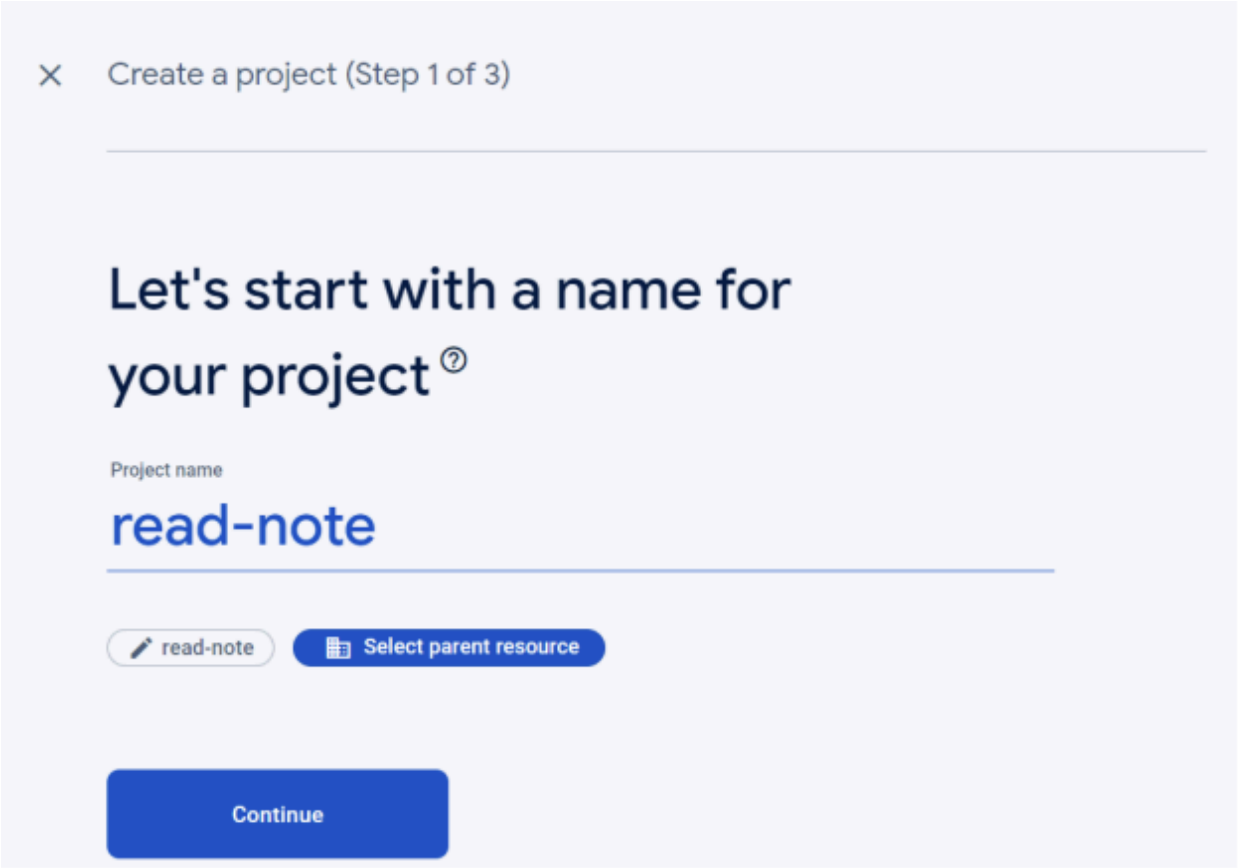
Connecting a Flutter app UI with a Firebase database involves several steps. Firebase is a mobile and web application development platform that provides various services, including a real-time NoSQL database.

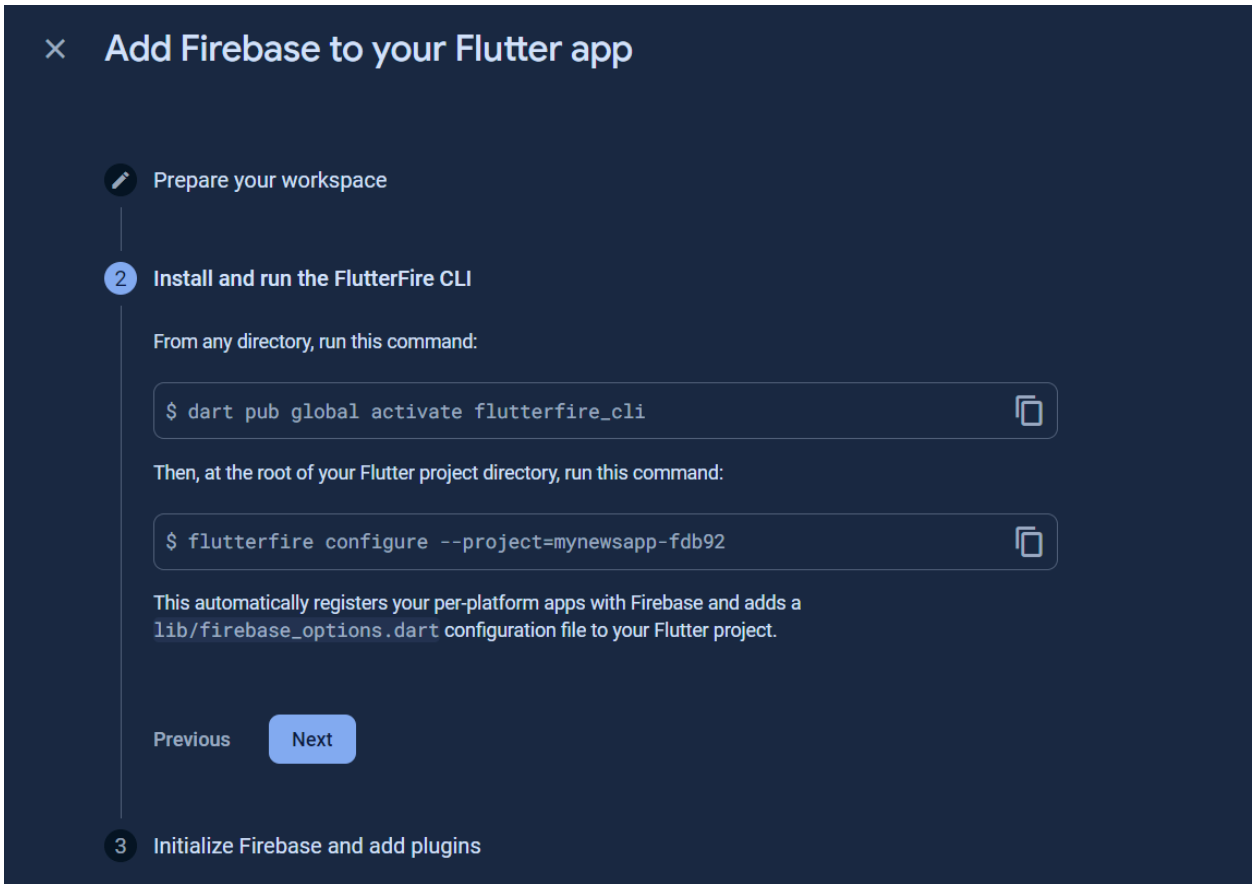
1. **Create a Firebase Project:**

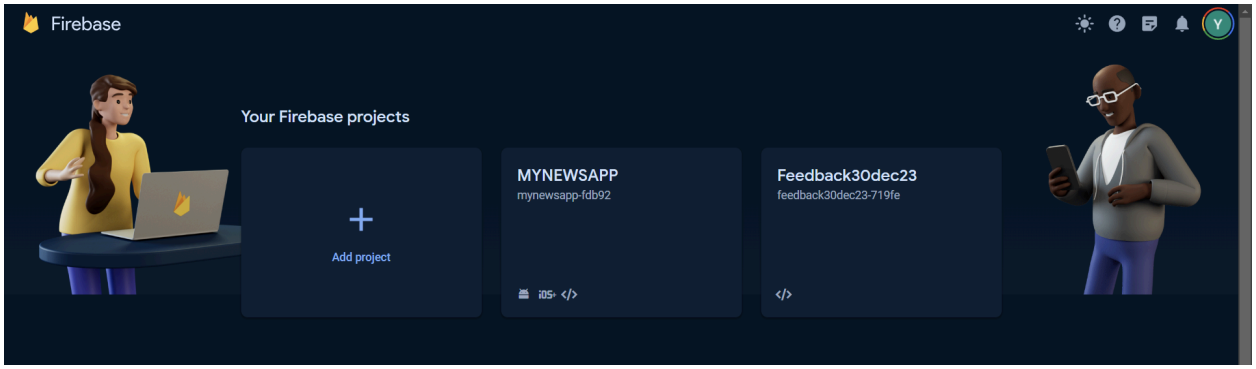
```
- Go to the [Firebase Console](https://console.firebase.google.com/).
- Click on "Add Project" and follow the setup instructions.
```

```
~ This may take a moment
+ Alright, your CLI is set up!

~ Looks like you're not authenticated. Let's log in!
i  Firebase optionally collects CLI and Emulator Suite usage and error reporting information to help improve our product
s. Data is collected in accordance with Google's privacy policy (https://policies.google.com/privacy) and is not used to
 identify you.

? Allow Firebase to collect CLI and Emulator Suite usage and error reporting information? Yes
i  To change your data collection preference at any time, run `firebase logout` and log in again.

Visit this URL on this device to log in:
https://accounts.google.com/o/oauth2/auth?client_id=563584335869-fgrhgmd47bqnekij5i8b5pr03ho849e6.apps.googleusercontent
.com&scope=email%20openid%20https%3A%2F%2Fwww.googleapis.com%2Fauth%2Fcloudplatformprojects.readonly%20https%3A%2F%2Fwww
.googleapis.com%2Fauth%2Ffirebase%20https%3A%2F%2Fwww.googleapis.com%2Fauth%2Fcloud-platform&response_type=code&state=65
3964705&redirect_uri=http%3A%2F%2Flocalhost%3A9005

Waiting for authentication...

+  Success! Logged in as yogeshchandraparanjpe@gmail.com

+ You can now use the 'firebase' or 'npm' commands!
~ For more help see https://firebase.google.com/docs/cli/

-----------------------------------------------------------------------------------------------------------------------

> firebase login
Already logged in as yogeshchandraparanjpe@gmail.com
```

Yogeshchandra Paranjpe
D15B 52
MPL Lab Exp 6





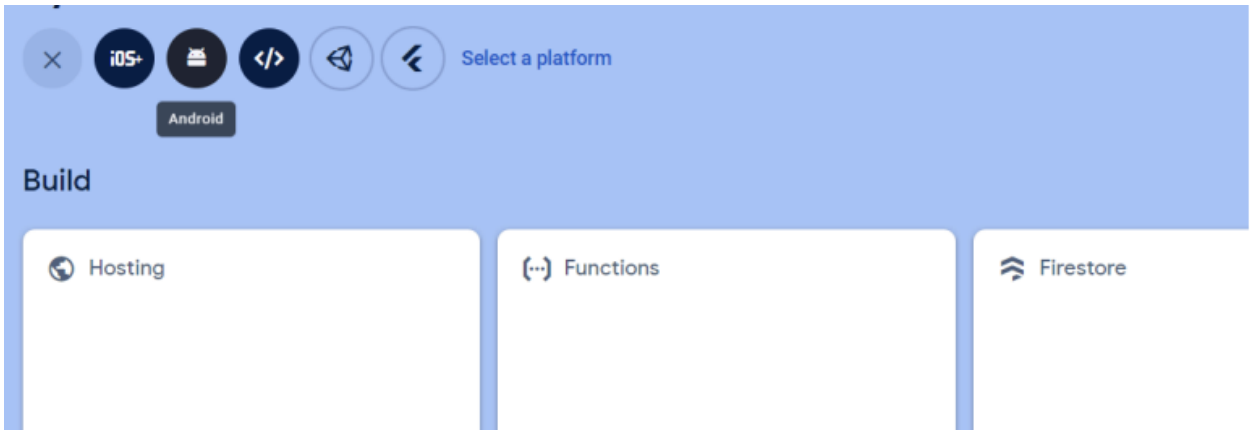


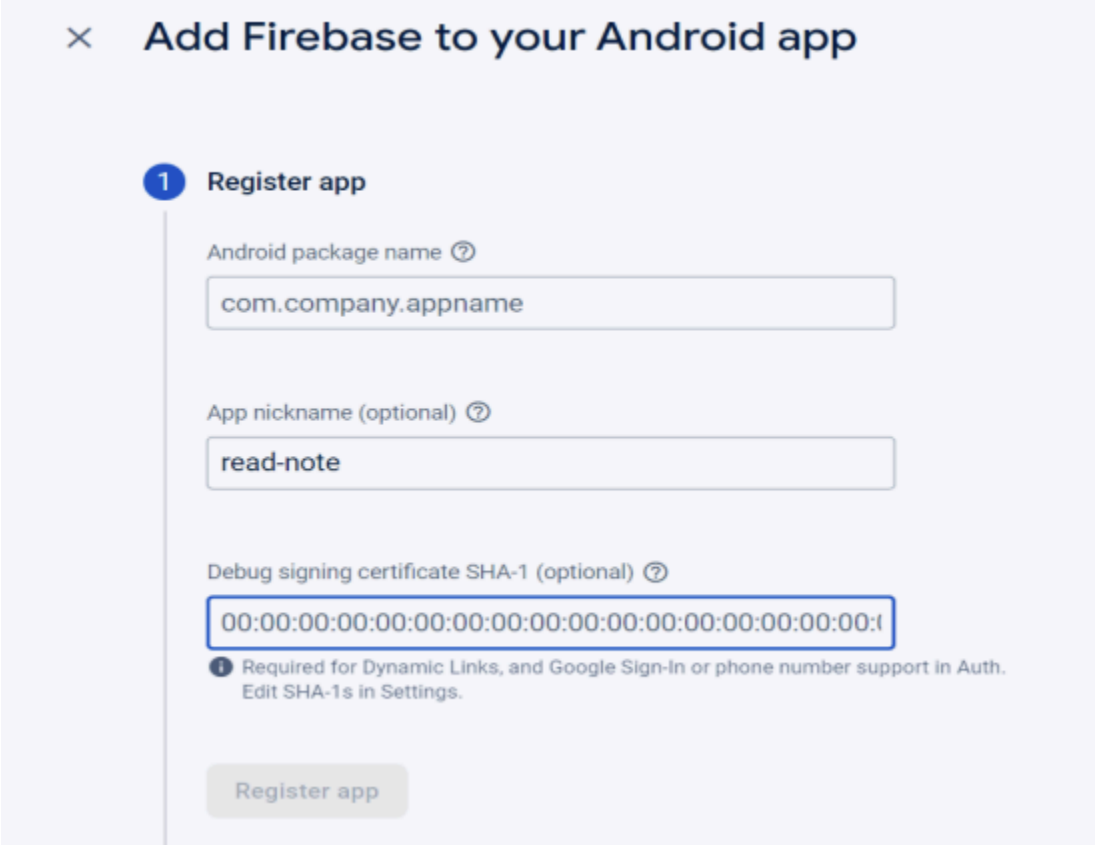
Here we have added an project MYNEWSAPP to the firebase console.

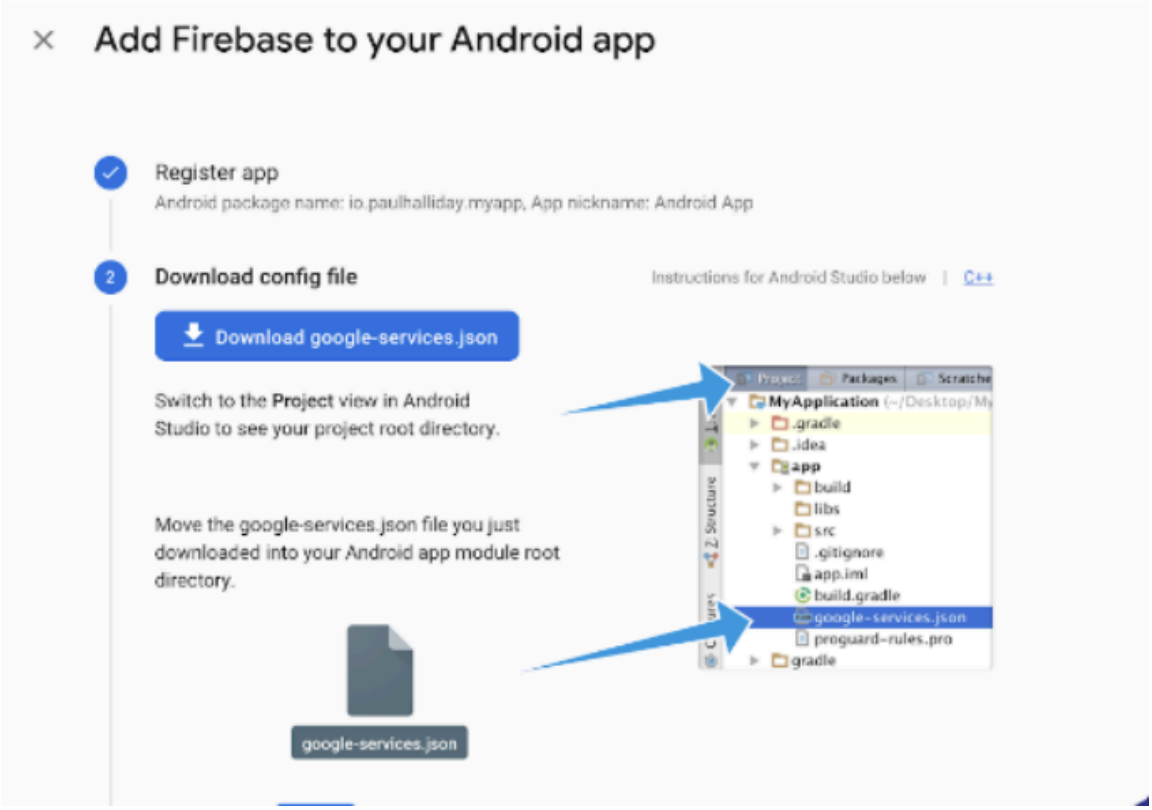
2. **Add a Flutter App to Firebase Project:**

- In the Firebase Console, select your project.
- Click on "Add app" and choose the Flutter icon.
- Follow the setup instructions to register your app

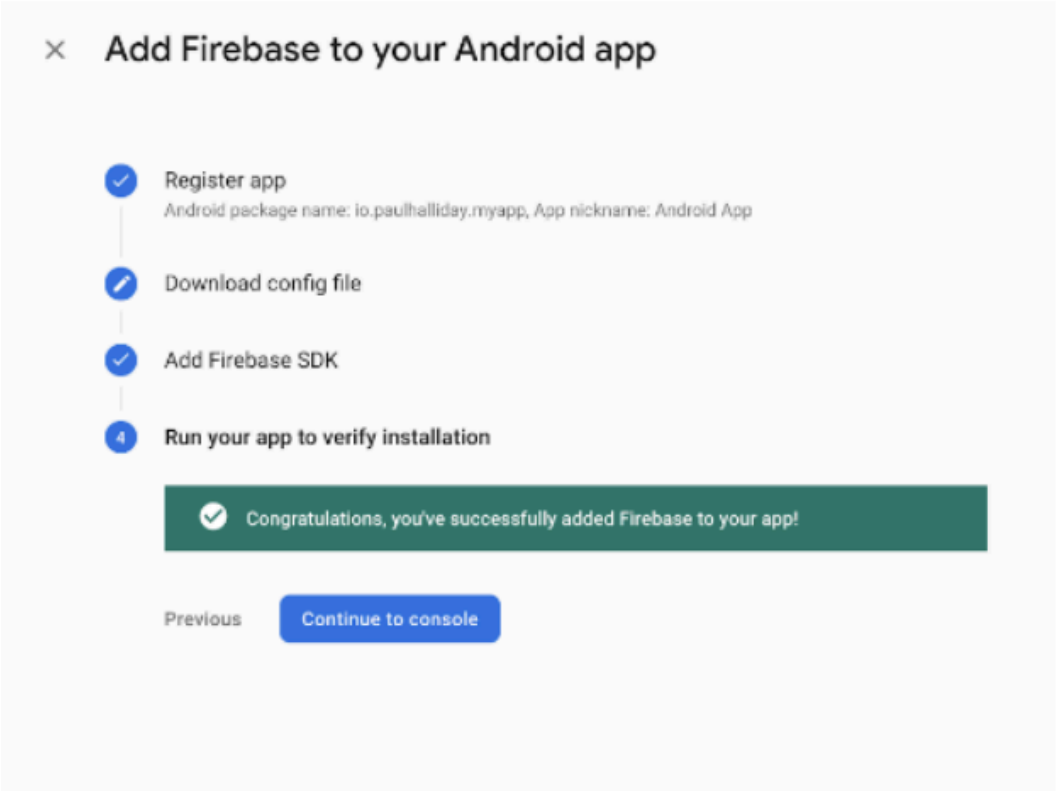
Yogeshchandra Paranjpe
D15B 52
MPL Lab Exp 6







3. **Configure Firebase in Flutter**



- Add the necessary dependencies to your pubspec.yaml file:

```
 yaml dependencies:
flutter:
sdk: flutter
firebase_core: ^latest_version
firebase_database: ^latest_version
```

- Run flutter pub get to install the dependencies

pubspec.yaml

news > pubspec.yaml

```yaml

environment:
  sdk: '>=3.2.1 <4.0.0'

dependencies:
  flutter:
    sdk: flutter
  flutter_localizations:
    sdk: flutter
  cupertino_icons: ^1.0.2
  flutter_bloc: ^8.1.3
  equatable: ^2.0.5
  get_it: ^7.6.4
  intl: any
  floor: ^1.4.2
  retrofit: ^4.0.3
  cached_network_image: ^3.3.0
  ionicons: ^0.2.2
  url_launcher: ^6.2.2
  firebase_core: ^2.24.2
  firebase_auth: ^4.15.3
  flutter_svg: ^2.0.9
  google_sign_in: ^6.2.1
  flutter_facebook_auth: ^6.0.3
  uni_links: ^0.5.1
  firebase_storage: ^11.5.6
  image_picker: ^1.0.5

```

PROBLEMS 46 OUTPUT DEBUG CONSOLE TERMINAL PORTS

4. Initialize Firebase in Flutter:

- Import the Firebase packages in your main.dart file:

dart

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

- Initialize Firebase in the main function:

dart

```
void main() async {
WidgetsFlutterBinding.ensureInitialized();
```

```
await Firebase.initializeApp();
runApp(MyApp());
}
```

5. Set Up Firebase Realtime Database:

- In the Firebase Console, go to "Database" and click on "Create Database."
- Choose "Start in test mode" and click "Next."
- Set up your database rules.

6. Create Firebase Database Reference:

```
- Import the necessary package in your Dart file: dart
import 'package:firebase_database/firebase_database.dart';
- Create a reference to your Firebase database:
dart
final databaseReference = FirebaseDatabase.instance.reference();
```

7. Read Data from Firebase:

```
- To read data from Firebase, you can use methods like once() or
onValue() :
Dart
DatabaseReference reference = FirebaseDatabase.instance.reference();
// Read data once
DataSnapshot snapshot = await reference.once();
// Access the data
print('Data: ${snapshot.value}');
```

8. Write Data to Firebase:

```
- To write data to Firebase, you can use methods like set() or push()
:
Dart
DatabaseReference reference = FirebaseDatabase.instance.reference();
// Set data
reference.child('users').set({'username': 'JohnDoe', 'email':
'john@example.com'});
```

9. Update or Delete Data:

```
- Use the update() or remove() methods to update or delete data:
Dart
DatabaseReference reference = FirebaseDatabase.instance.reference();
// Update data
reference.child('users').child('userId').update({'username':
'NewUsername'});
```

```
// Delete data
reference.child('users').child('userId').remove();
```

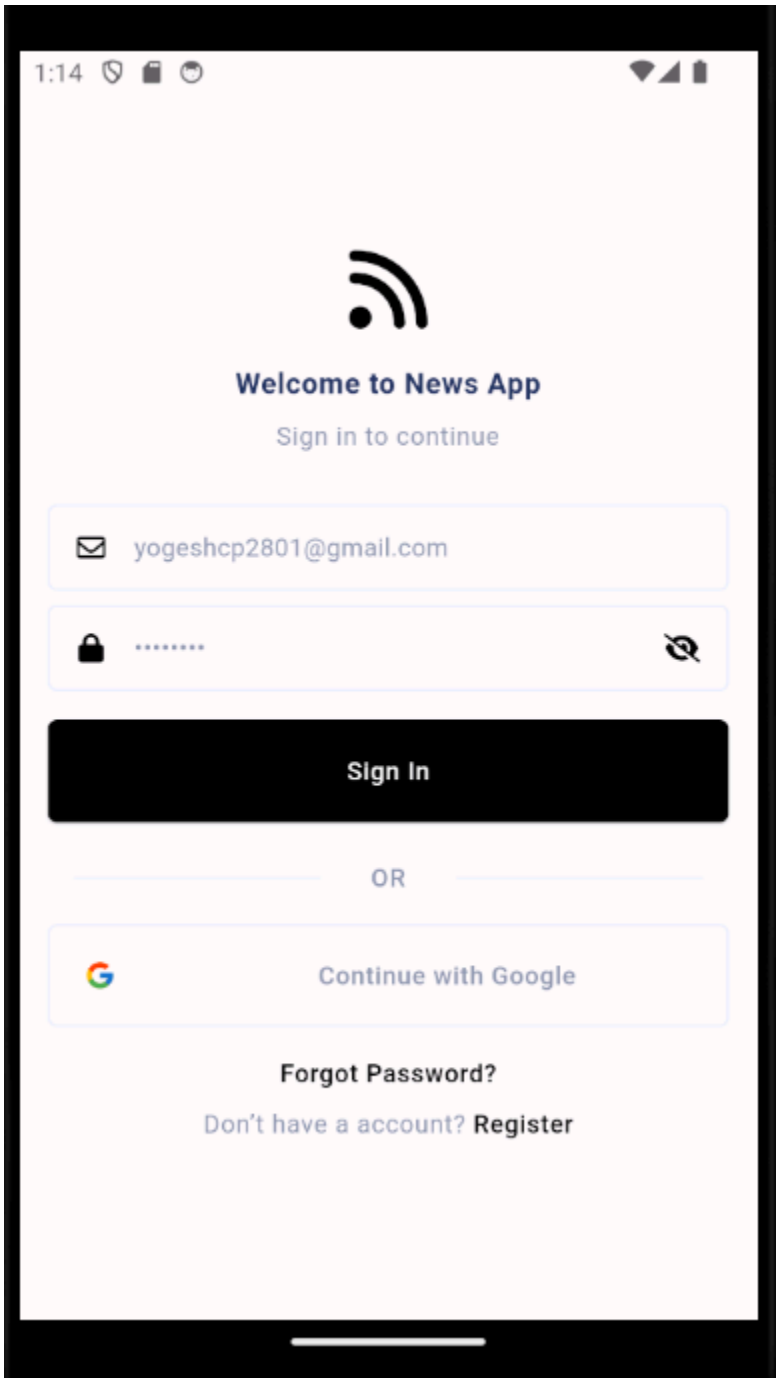
10. Handle Asynchronous Operations:

- Since Firebase operations are asynchronous, make sure to handle them using async/await
or .then() .

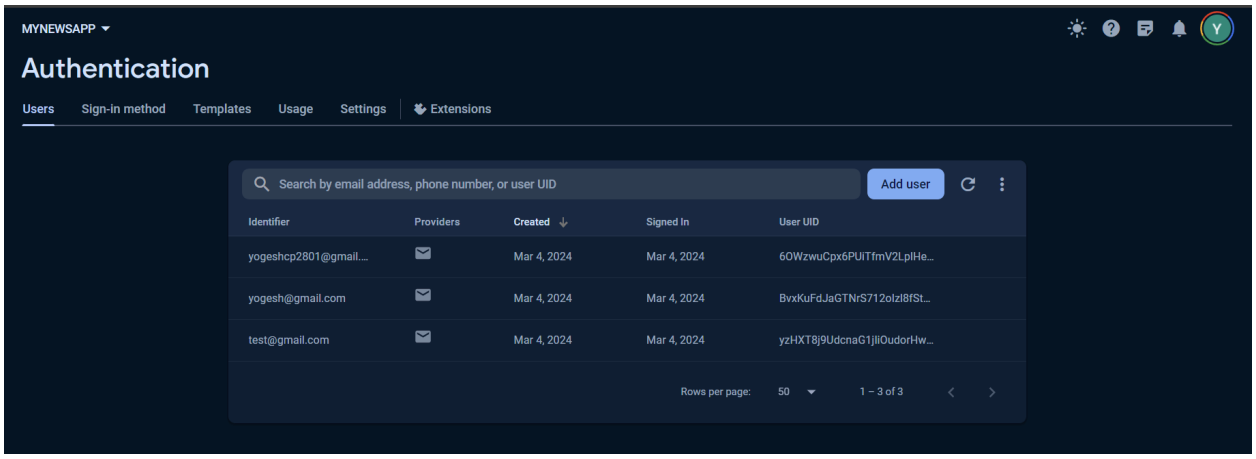
11. Display Data in Flutter UI:

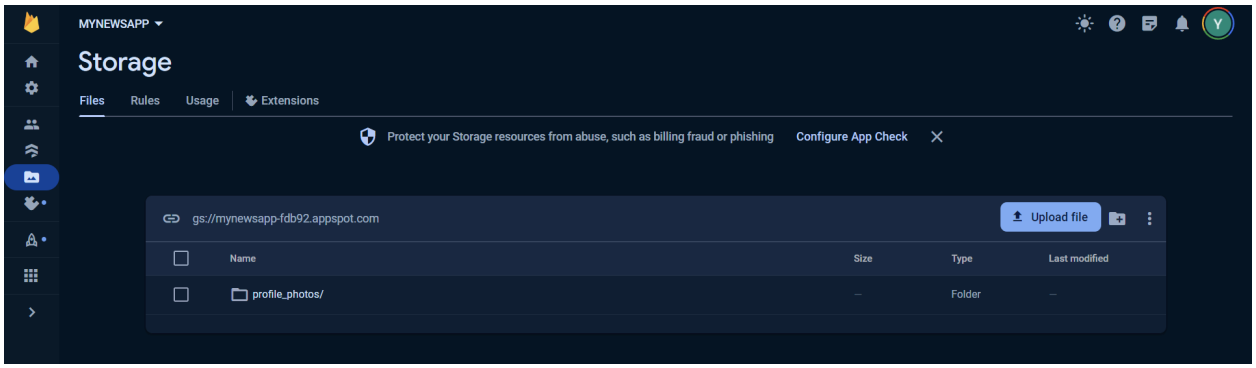
- Use Flutter widgets to display the data retrieved from Firebase in your app's UI.

Yogeshchandra Paranjpe
D15B 52
MPL Lab Exp 6



Yogeshchandra Paranjpe
D15B 52
MPL Lab Exp 6





Conclusion:
Therefore, we've seamlessly incorporated Flutter with Firebase, providing a robust solution for constructing potent cross-platform applications. Through the amalgamation of Flutter's UI capabilities and Firebase's authentication, real-time database, and storage features, developers can craft scalable and feature-rich apps. This efficient process encompasses project setup, dependency configuration, initialization, and rigorous testing, ultimately resulting in a smooth deployment across both iOS and Android platforms.