MAD LAB EXPT 6

Name: Harsh Gawali D15A 19

Aim: To connect Flutter UI with Firebase.

Theory:

Step1:
Go to the Firebase Console (https://console.firebase.google.com/) and create a new project.Follow the instructions to add your app to the Firebase project. You'll need to provide your app's package name (Android) or bundle identifier (iOS).

Step 2:
Add the Firebase SDK dependencies to your Flutter app's pubspec.yaml file. These dependencies vary depending on the Firebase services you want to use (e.g., Firebase Authentication, Firestore, Realtime Database, Cloud Storage). Run flutter pub get to install the dependencies.

Step 3:
In your Flutter app, initialize Firebase by calling Firebase.initializeApp() in the main() function or at the entry point of your app.
This initialization step is crucial and should be done before accessing any Firebase services.

Step 4:
Once Firebase is initialized, you can start using Firebase services like Firestore (NoSQL database), Realtime Database (JSON database), Cloud Storage (file storage), Cloud Functions (serverless functions), etc.
You'll typically use Firebase APIs to read and write data, handle user authentication, and perform other tasks.

Step 5:
Use Firebase listeners to listen for real-time updates to your data. For example, in Firestore, you can set up listeners to receive updates whenever the data in a collection or document changes.

Step 6:

Implement error handling logic to handle exceptions and errors that may occur when interacting with Firebase services.

Output:

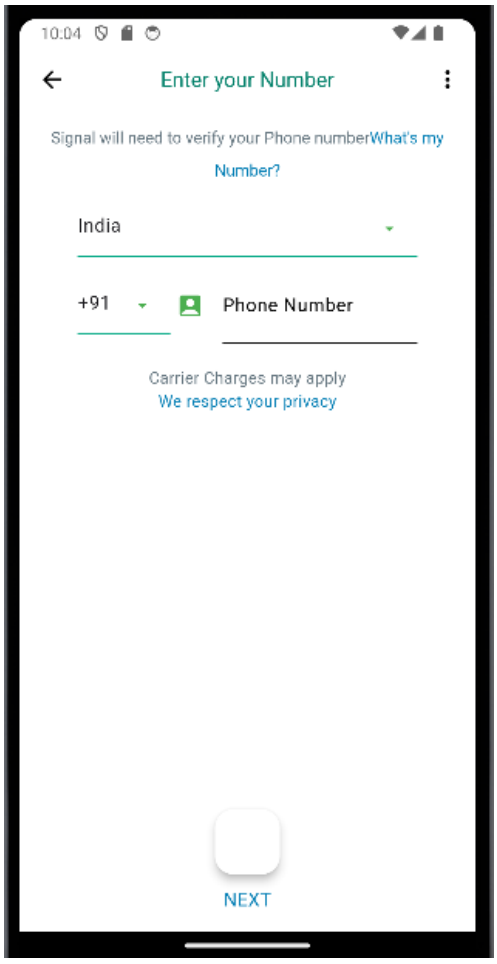
```dart
// File generated by FlutterFire CLI.
// ignore_for_file: lines_longer_than_80_chars,
avoid_classes_with_only_static_members
import 'package:firebase_core/firebase_core.dart' show FirebaseOptions;
import 'package:flutter/foundation.dart'
    show defaultTargetPlatform, kIsWeb, TargetPlatform;

/// Default [FirebaseOptions] for use with your Firebase apps.
///
/// Example:
/// ```dart
/// import 'firebase_options.dart';
/// // ...
/// await Firebase.initializeApp(
///   options: DefaultFirebaseOptions.currentPlatform,
/// );
/// ```
class DefaultFirebaseOptions {
  static FirebaseOptions get currentPlatform {
    if (kIsWeb) {
      throw UnsupportedError(
        'DefaultFirebaseOptions have not been configured for web - '
        'you can reconfigure this by running the FlutterFire CLI again.',
      );
    }
    switch (defaultTargetPlatform) {
      case TargetPlatform.android:
        return android;
      case TargetPlatform.iOS:
        throw UnsupportedError(
          'DefaultFirebaseOptions have not been configured for ios - '
          'you can reconfigure this by running the FlutterFire CLI again.',
        );
      case TargetPlatform.macOS:
        throw UnsupportedError(
          'DefaultFirebaseOptions have not been configured for macos - '
          'you can reconfigure this by running the FlutterFire CLI again.',
        );
      case TargetPlatform.windows:
        throw UnsupportedError(
          'DefaultFirebaseOptions have not been configured for windows - '
          'you can reconfigure this by running the FlutterFire CLI again.',
        );
      case TargetPlatform.linux:
        throw UnsupportedError(
          'DefaultFirebaseOptions have not been configured for linux - '
          'you can reconfigure this by running the FlutterFire CLI again.',
        );
      default:
```

```
        throw UnsupportedError(
          'DefaultFirebaseOptions are not supported for this platform.',
        );
    }
  }

  static const FirebaseOptions android = FirebaseOptions(
    apiKey: 'AIzaSyAQ0Rqa0Ql_a3IKREiYIEocGr1BQSGhi14',
    appId: '1:860315971576:android:7b303c871e55a3bc28700d',
    messagingSenderId: '860315971576',
    projectId: 'whatsappfire-a0ace',
    storageBucket: 'whatsappfire-a0ace.appspot.com',
  );
}
```

Connected with firebase for authentication.



<u>Conclusion: Firebase has been connected to the Flutter application for user authentication purposes.</u>