

EXPERIMENT NO: - 06

Name : Atharva Prabhu

Class : D15A Roll:No : 42

AIM: To connect flutter UI with firebase database.

Introduction to Firebase and Flutter Integration

Firebase is a comprehensive platform developed by Google, designed to help developers build high-quality applications for both mobile and web. It provides essential services such as real-time databases, authentication, cloud storage, hosting, and much more. One of the most widely used Firebase services is the Firebase Realtime Database, which is a NoSQL cloud database that allows data to be stored and synced in real-time across all connected devices. Flutter, on the other hand, is an open-source UI software development kit created by Google, which allows developers to build natively compiled applications for mobile, web, and desktop from a single codebase. Its rich set of pre-designed widgets and powerful tools makes Flutter an attractive option for developing visually appealing and performant applications. Integrating Firebase with Flutter allows developers to leverage the full potential of Firebase services in their applications. By using Firebase's Realtime Database, Flutter apps can achieve features such as real-time data synchronization, secure authentication, and cloud-based storage. This combination enables developers to create powerful, scalable, and feature-rich mobile and web applications.

Firebase Realtime Database Overview

Firebase Realtime Database is a cloud-hosted NoSQL database that stores data in a JSON-like format. The key characteristic of this database is its real-time synchronization feature, meaning that any changes made to the database are instantly reflected on all clients (i.e., devices) connected to it. This makes it an ideal solution for applications that require frequent updates and need to maintain synchronized data across multiple users or devices, such as messaging apps, social media platforms, or collaborative tools.

The Firebase Realtime Database is structured as a tree of data, where each node in the tree can contain key value pairs. This structure allows for easy data retrieval and modification. Firebase's real-time capabilities enable apps to immediately receive updates to the data whenever it changes, without the need to refresh or reload the page. Additionally, the database supports offline data persistence, meaning that even if the user's device loses its internet connection, the app can still function by using the locally cached data.

Setting Up Firebase in Flutter :

To connect a Flutter app with Firebase, the following steps are typically followed:

1. **Creating a Firebase Project:** To start using Firebase with Flutter, the first step is to create a Firebase project in the Firebase Console. Once the project is created, developers can associate their Flutter app with the Firebase project by following the platform-specific instructions for Android or iOS. This usually involves configuring API keys, downloading configuration files, and adding them to the Flutter project.
2. **Integrating Firebase SDK in Flutter:** After the Firebase project is set up, developers need to integrate Firebase's SDK into the Flutter app. This involves adding the necessary dependencies to the Flutter project's `pubspec.yaml` file. For Firebase's Realtime Database, the package `firebase_database` is used. Additionally, Firebase's core SDK (`firebase_core`) must also be included to initialize Firebase services.

3. Initializing Firebase: Before any Firebase functionality can be used, it is essential to initialize Firebase in the Flutter app. This is done by calling `Firebase.initializeApp()` in the main entry point of the app (usually in the `main.dart` file). Firebase needs to be initialized before interacting with any Firebase services, such as the Realtime Database, Cloud Firestore, or Authentication.

Code:

Signup.dart

```
import 'package:flutter/material.dart';
import '../widgets/custom_text_field.dart';
import '../widgets/custom_button.dart';
import '../widgets/gender_selection.dart';
import '../widgets/profile_avatar.dart';
```

```
class SignupPage extends StatefulWidget {
  const SignupPage({super.key});
```

```
  @override
  _SignupPageState createState() => _SignupPageState();
}
```

```
class _SignupPageState extends State<SignupPage> {
  final TextEditingController _emailController = TextEditingController();
  final TextEditingController _passwordController = TextEditingController();
  final TextEditingController _firstNameController = TextEditingController();
  final TextEditingController _lastNameController = TextEditingController();
  final TextEditingController _birthdayController = TextEditingController();
  String gender = "Male";
```

```
  void _signup() {
    // Handle signup logic here
    Navigator.pushNamed(context, '/home');
  }
```

```
  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        leading: IconButton(
          icon: const Icon(Icons.arrow_back),
          onPressed: () {
            Navigator.pop(context);
          },
        ),
      ),
      body: Padding(
        padding: const EdgeInsets.all(20.0),
        child: Column(
          children: [
            ProfileAvatar(onTap: () {
              print("Profile Avatar Clicked!");
            })
          ]
        )
      )
    );
  }
```

```

    }),
    const SizedBox(height: 20),
    Row(
      children: [
        Expanded(
          child: CustomTextField(
            controller: _firstNameController, label: "First name")),
        const SizedBox(width: 10),
        Expanded(
          child: CustomTextField(
            controller: _lastNameController, label: "Last name")),
      ],
    ),
    const SizedBox(height: 10),
    CustomTextField(controller: _birthdayController, label: "Birthday"),
    const SizedBox(height: 10),
    GenderSelection(onGenderSelected: (selectedGender) {
      setState(() {
        gender = selectedGender;
      });
    }),
    CustomTextField(controller: _emailController, label: "Email"),
    const SizedBox(height: 10),
    CustomTextField(
      controller: _passwordController,
      label: "Password",
      isPassword: true),
    const SizedBox(height: 20),
    GestureDetector(
      onTap: _signup,
      child: CustomButton(
        text: "Next",
        onPressed: () {},
      ),
    ),
  ],
),
);
}
}

```

Login.dart

```

import 'package:flutter/material.dart';
import 'signup_page.dart';
import '../widgets/custom_text_field.dart';
import '../widgets/custom_button.dart';
import 'home_page.dart'; // Import the HomePage

```

```

class LoginPage extends StatelessWidget {
  const LoginPage({super.key});

```

```

@override
Widget build(BuildContext context) {
  final TextEditingController _emailController = TextEditingController();
  final TextEditingController _passwordController = TextEditingController();

  void _login() {
    // Handle login logic here
    Navigator.push(
      context,
      MaterialPageRoute(builder: (context) => const HomePage()),
    );
  }

  return Scaffold(
    body: Padding(
      padding: const EdgeInsets.all(20.0),
      child: Column(
        mainAxisAlignment: MainAxisAlignment.center,
        children: [
          const Icon(Icons.facebook, size: 80, color: Colors.blue),
          const SizedBox(height: 20),
          CustomTextField(
            controller: _emailController,
            label: "Email",
          ),
          const SizedBox(height: 10),
          CustomTextField(
            controller: _passwordController,
            label: "Password",
            isPassword: true,
          ),
          const SizedBox(height: 20),
          GestureDetector(
            onTap: _login,
            child: CustomButton(
              text: "Login",
              onPressed: () {
                Navigator.pushNamed(context, '/home');
              },
            ),
          ),
          TextButton(
            onPressed: () {},
            child: const Text("Forgot Password?"),
          ),
          const SizedBox(height: 20),
          GestureDetector(
            onTap: () {
              Navigator.push(
                context,

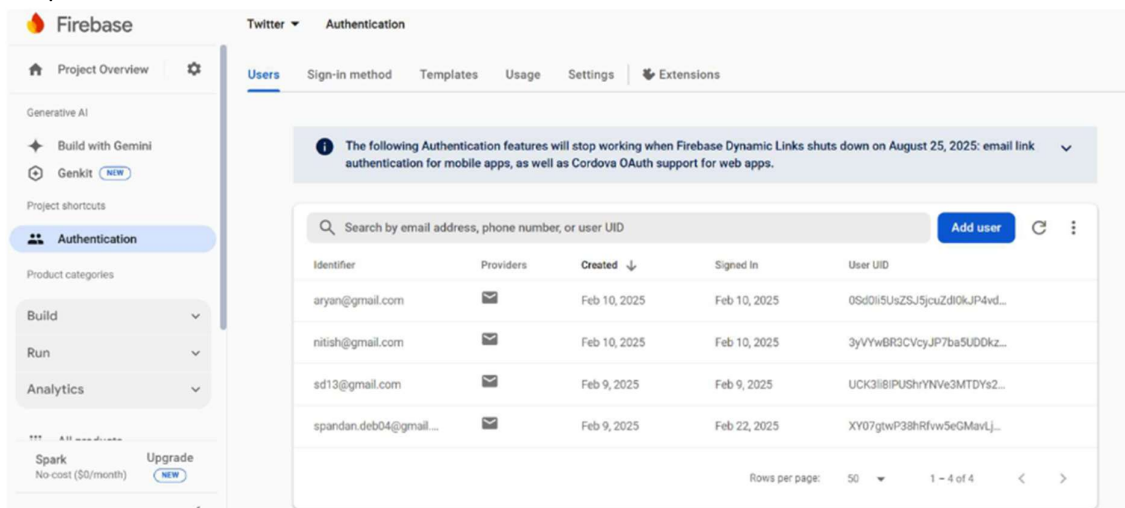
```

```

        MaterialPageRoute(builder: (context) => const SignupPage()),
      );
    },
    child: CustomButton(
      text: "Create new account",
      onPressed: () {
        Navigator.pushNamed(context, '/signup');
      },
    ),
  ),
  const SizedBox(height: 10),
  const Text("Meta", style: TextStyle(color: Colors.grey)),
],
),
),
);
}
}

```

Output



The screenshot shows the Firebase Authentication console. The left sidebar contains navigation options: Project Overview, Generative AI (Build with Gemini, Genkit), Project shortcuts (Authentication), and Product categories (Build, Run, Analytics). The main content area is titled 'Authentication' and includes a warning banner about Dynamic Links shutting down on August 25, 2025. Below the banner is a search bar and a table of users.

Identifier	Providers	Created ↓	Signed In	User UID
aryan@gmail.com	📧	Feb 10, 2025	Feb 10, 2025	0Sd0i5UsZSj5jcuZd0kJP4vd...
nitish@gmail.com	📧	Feb 10, 2025	Feb 10, 2025	3yVYwBR3CVcyJP7ba5UDDkz...
sd13@gmail.com	📧	Feb 9, 2025	Feb 9, 2025	UCK3li8iPUSHrYNNVe3MTDys2...
spandan.deb04@gmail...	📧	Feb 9, 2025	Feb 22, 2025	XY07ghwP38hRfVw5eGMavLj...

At the bottom right of the table, it says 'Rows per page: 50' and '1 - 4 of 4'.