

MPL Experiment 6

Name: Prathamesh Palve

Class: D15A

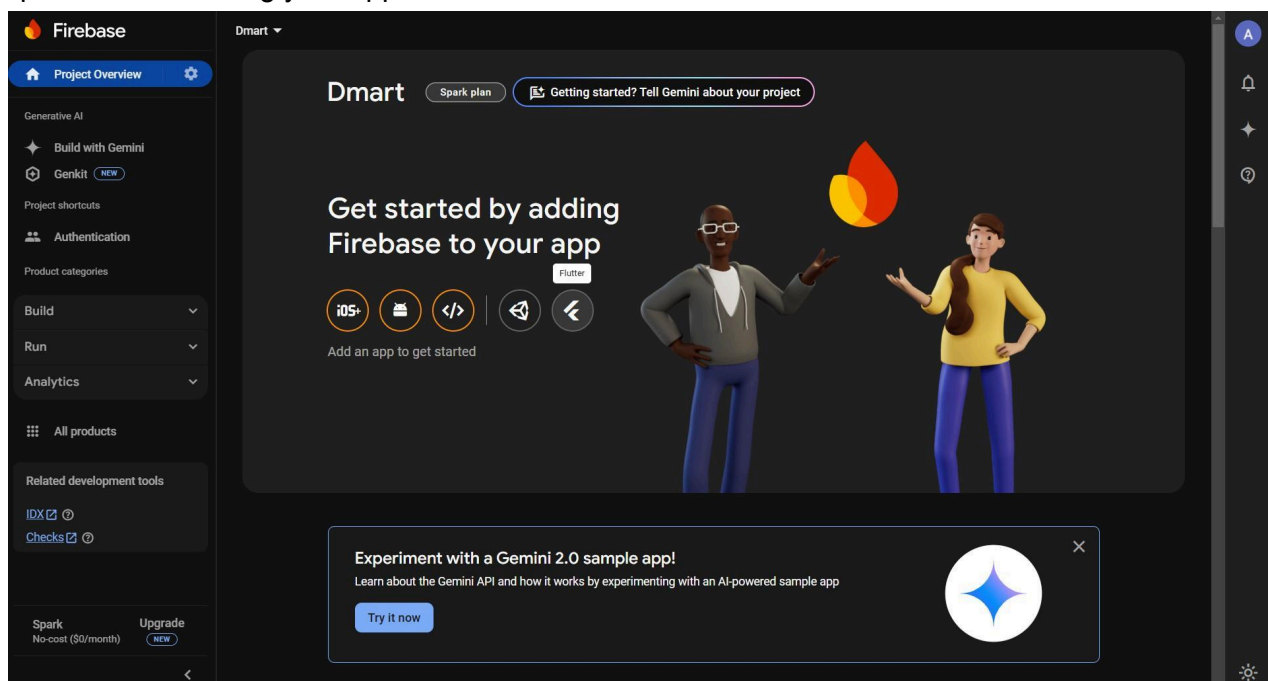
Roll no:31

Aim: How To Set Up Firebase with Flutter for iOS and Android Apps

Steps to Set Up Firebase with Flutter:

Step 1:

Go to the Firebase Console (<https://console.firebase.google.com/>). Click on "Add Project" and follow the steps to create your Firebase project. Once the project is created, select the Flutter option for connecting your app with Firebase.



Step 2:

Open **Windows PowerShell** (or any terminal you prefer).

Run the following commands to install Firebase CLI and verify the installation:

```
npm install -g firebase-tools
```

```
firebase --version
```

```
firebase login
```

This will install the **Firebase CLI**, check the version, and log you into your Firebase account.

```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS C:\Users\User> npm install -g firebase-tools

changed 635 packages in 33s

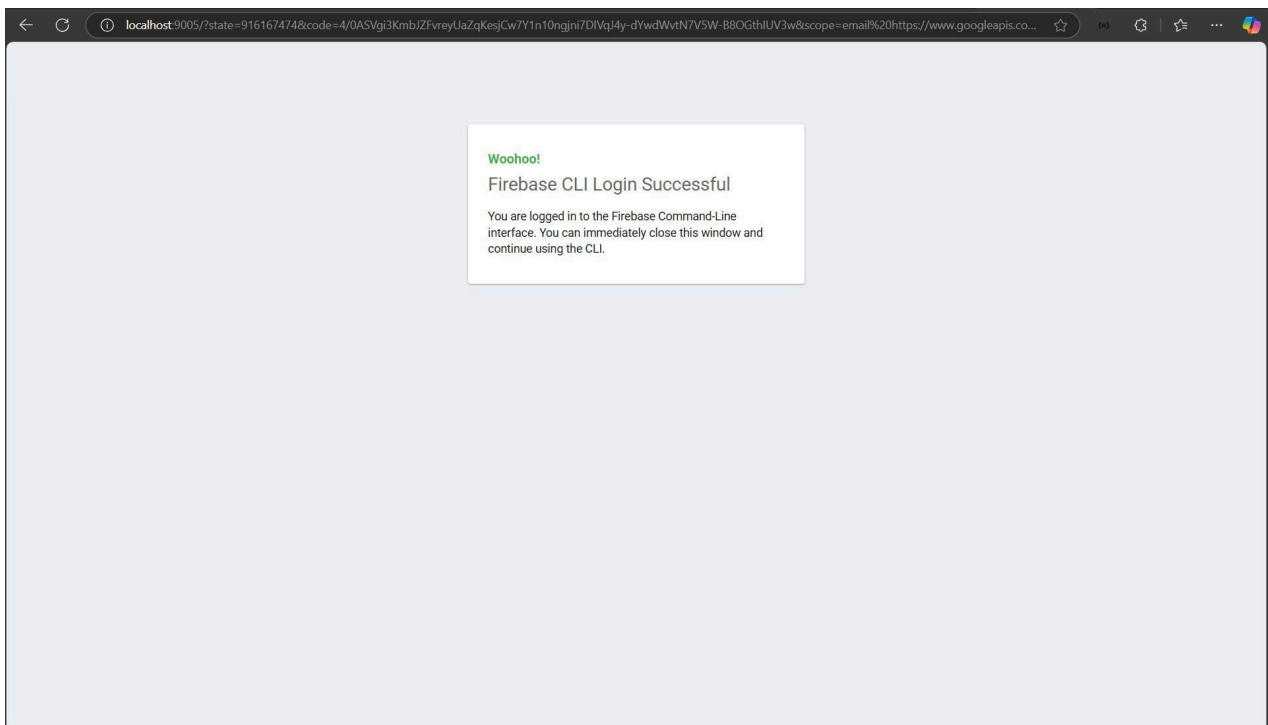
70 packages are looking for funding
  run 'npm fund' for details
PS C:\Users\User> firebase --version
13.29.3
PS C:\Users\User> firebase login
i  Firebase optionally collects CLI and Emulator Suite usage and error reporting information to help improve our product
s. Data is collected in accordance with 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=91
6167474&redirect_uri=http%3A%2F%2Flocalhost%3A9005

Waiting for authentication...

+ Success! Logged in as 2022.anuprita.mhapankar@ves.ac.in
PS C:\Users\User>
```



Step 3:

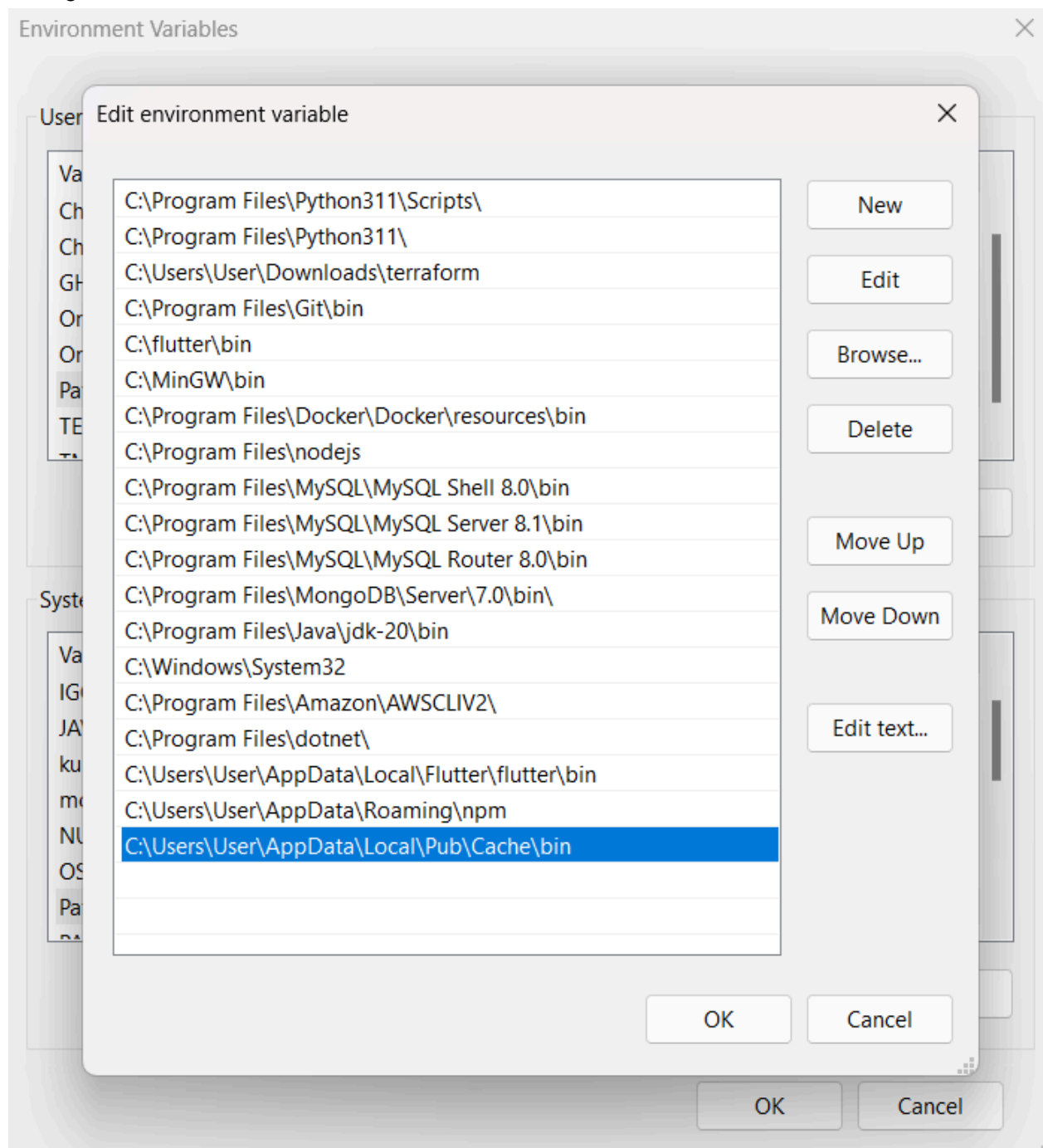
Open your Flutter app in **Android Studio**.

In the terminal of Android Studio, run the following command to activate flutterfire_cli:

```
dart pub global activate flutterfire_cli
```

Add flutterfire to your Environment Variables. You may need to restart Android Studio after

adding it.



Step 4:

Run the following command to configure Firebase with your

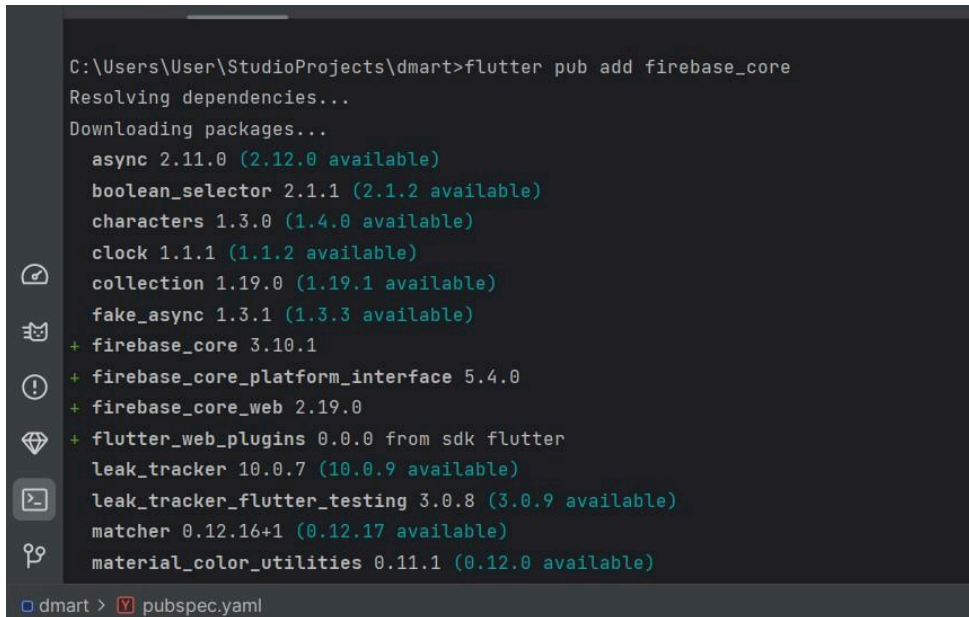
project: `flutterfire configure --project=chatGPT_08`

Replace `chatGPT_08` with your Firebase project ID

Step 5:

Run this command to add Firebase Core dependency to your app:

```
flutter pub add firebase_core
```



```
C:\Users\User\StudioProjects\dmart>flutter pub add firebase_core
Resolving dependencies...
Downloading packages...
  async 2.11.0 (2.12.0 available)
  boolean_selector 2.1.1 (2.1.2 available)
  characters 1.3.0 (1.4.0 available)
  clock 1.1.1 (1.1.2 available)
  collection 1.19.0 (1.19.1 available)
  fake_async 1.3.1 (1.3.3 available)
+ firebase_core 3.10.1
+ firebase_core_platform_interface 5.4.0
+ firebase_core_web 2.19.0
+ flutter_web_plugins 0.0.0 from sdk flutter
  leak_tracker 10.0.7 (10.0.9 available)
  leak_tracker_flutter_testing 3.0.8 (3.0.9 available)
  matcher 0.12.16+1 (0.12.17 available)
  material_color_utilities 0.11.1 (0.12.0 available)
```

dmart > pubspec.yaml

Firebase Authentication SetUp

Step 1:

The screenshot displays the Firebase Authentication console. The left sidebar shows the 'Authentication' section selected. The main area is divided into two tabs: 'Sign-in method' and 'Users'. The 'Sign-in method' tab is active, showing a table of providers with 'Email/Password' listed as 'Enabled'. The 'Users' tab is also visible, showing a list of users. An 'Add user' modal is open, allowing the creation of a new user with the email 'prathameshgaming08@gmail.com' and password 'Prathamesh32'.

Sign-in providers

Provider	Status
Email/Password	Enabled

Advanced

SMS Multi-factor Authentication

Allow your users to add an extra layer of security to their account. Once enabled, integrated and configured, users can sign in to their account in two steps, using SMS. [Learn more](#)

Users

Search by email address, phone number or user UID

Add an Email/Password user

Email: prathameshgaming08@gmail.com Password: Prathamesh32

Users List:

Identifier	Providers	Created	Signed in	User UID
prathmzz2@gmail.com		9 Feb 2025	9 Feb 2025	Xf3C4uy0yc0Mn6t1Y5GR4e2...
prathmzz2@gmail.com		9 Feb 2025	9 Feb 2025	NDHP27ZM6W4123Rj2wq...
prathameshgaming08@gmail.com		9 Feb 2025	1 Mar 2025	rhvYT2g8e0P5Ap0ghwv...
prathamesh08@gmail.com		9 Feb 2025	9 Feb 2025	ITEwVw0nUP0X5wCTEFAG0...

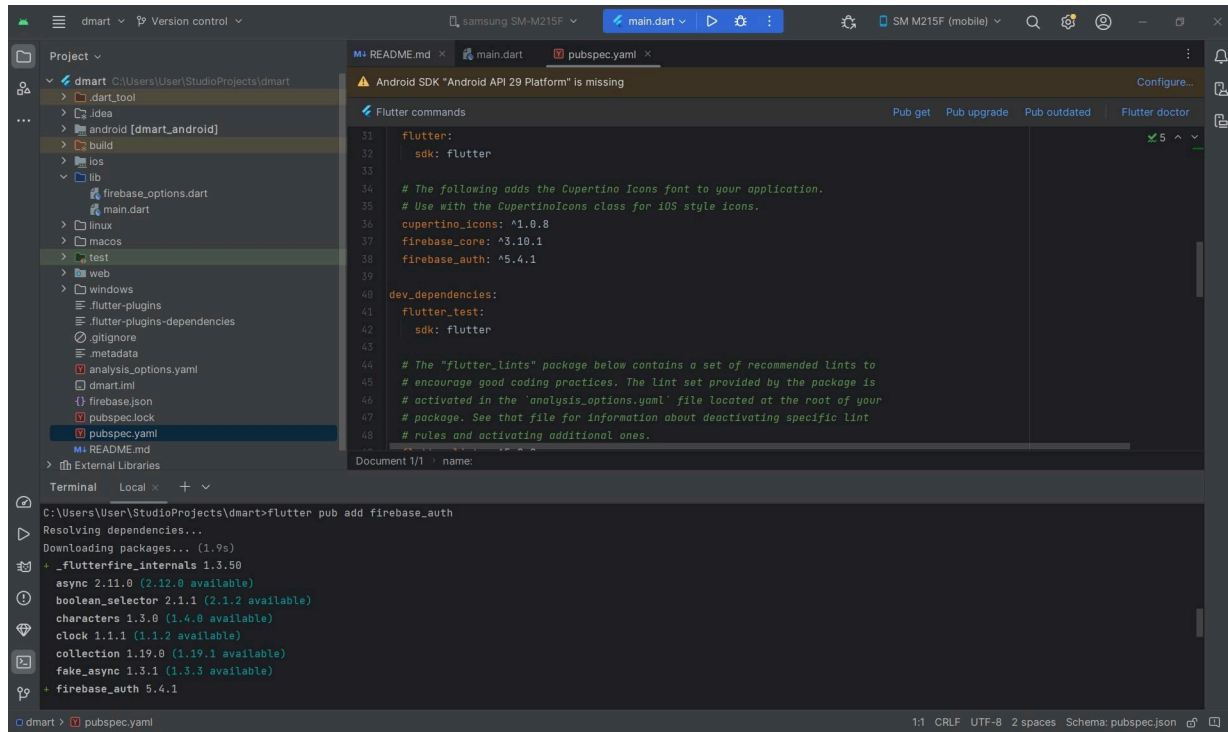
Go to the Firebase Console (<https://console.firebase.google.com/>). In your Firebase project, navigate to **Authentication**. Under the **Sign-in method** tab, enable **Email/Password** sign-in. Once enabled, go to the **Users** section and click on **Add User**. Enter a **username** (email) and a **password** for the new user.

Step 2:

Open your app in **Android Studio**.

In the terminal, run the following command to add the Firebase Authentication dependency:

```
flutter pub add firebase_auth
```

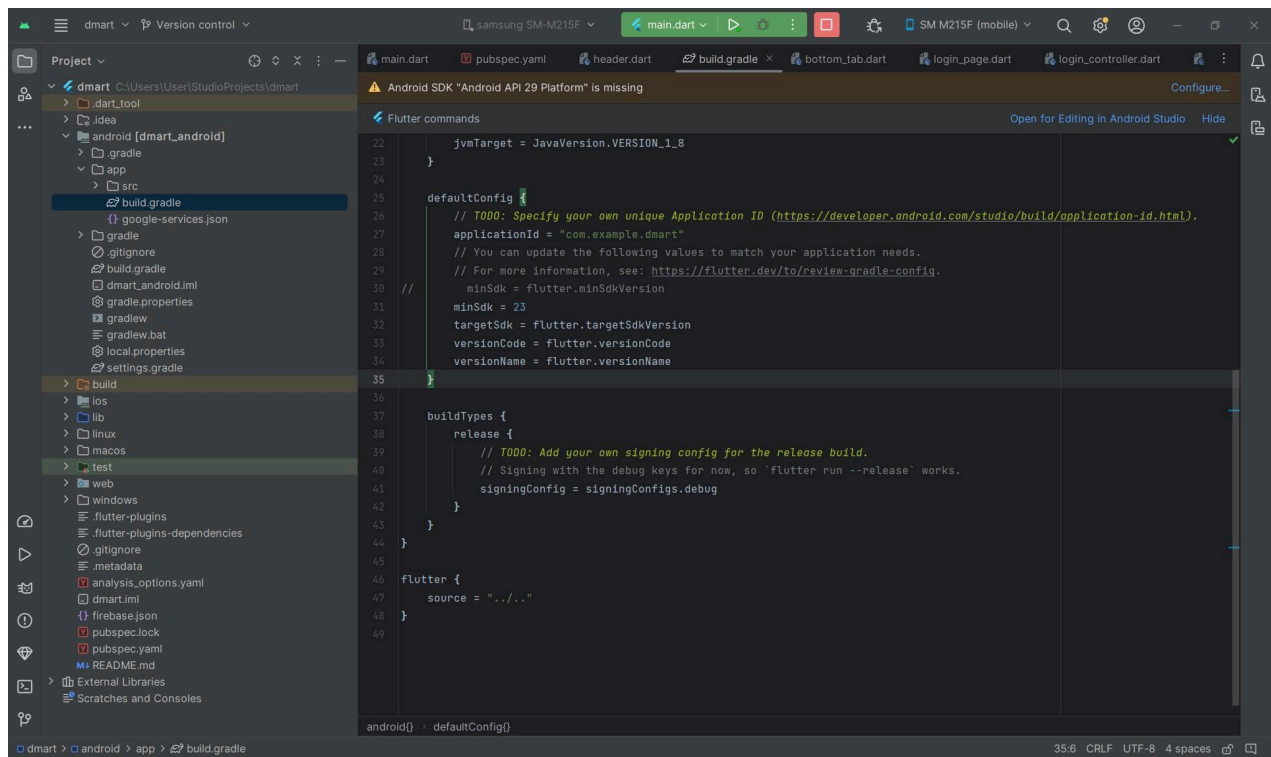


Step 3:

If you encounter any issues, add the following configuration to your `android/app/build.gradle` file:

```
android {  
    defaultConfig {  
        minSdk =  
            23  
        targetSdk =  
            flutter.targetSdkVersion  
        versionCode = flutter.versionCode  
        versionName = flutter.versionName  
    }  
}
```

This ensures that the Firebase dependencies are compatible with your Android app.

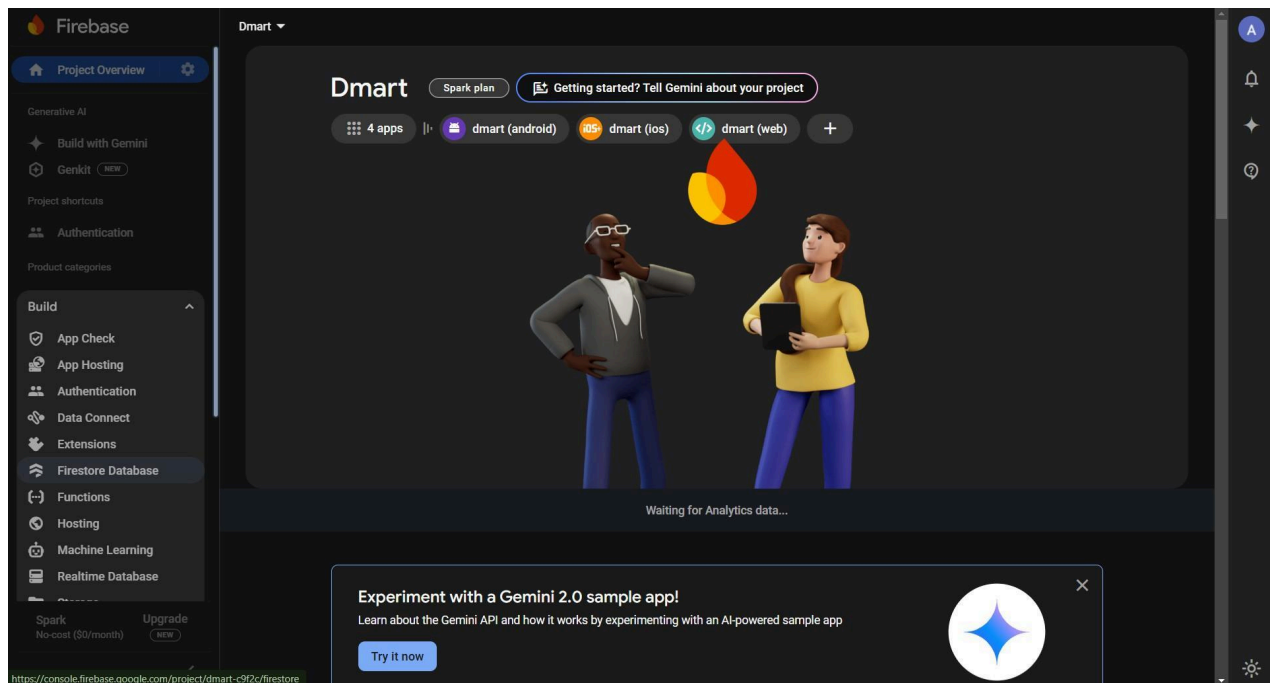


Now, firebase is successfully connected to our app.

Firestore Database Setup

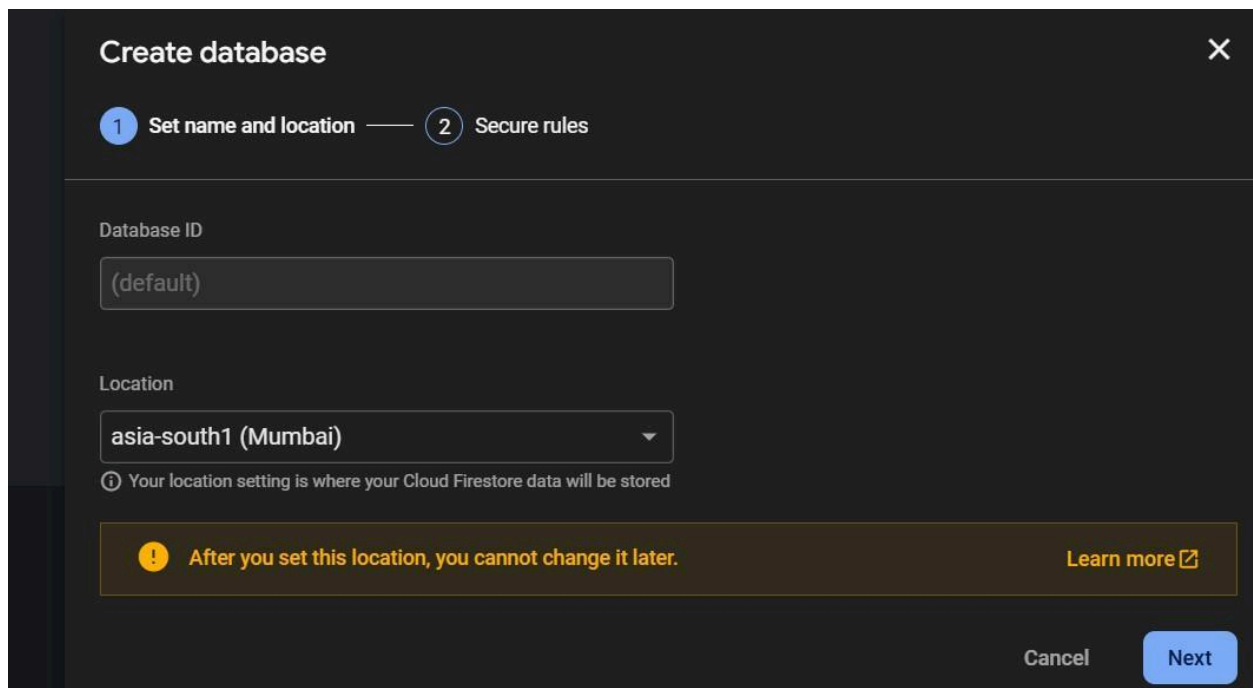
Step 1:

Select **Firestore Database** from **Build** in the sidebar.



Then click on Create database. For location select asia-south1(Mumbai). Then choose **Start in test mode** (for development).

Click **Next** and choose a location, then **Enable**.



Create database

1 Set name and location

2 Secure rules

After you define your data structure, you will need to write rules to secure your data.
[Learn more](#)

☐ Start in **production mode**


Your data is private by default. Client read/write access will only be granted as specified by your security rules.

☒ Start in **test mode**

Your data is open by default to enable quick setup. However, you must update your security rules within 30 days to enable long-term client read/write access.

```
rules_version = '2';

service cloud.firestore {
  match /databases/{database}/documents {
    match /{document=**} {
      allow read, write: if
        request.time < timestamp.date(2025, 3, 10);
    }
  }
}
```

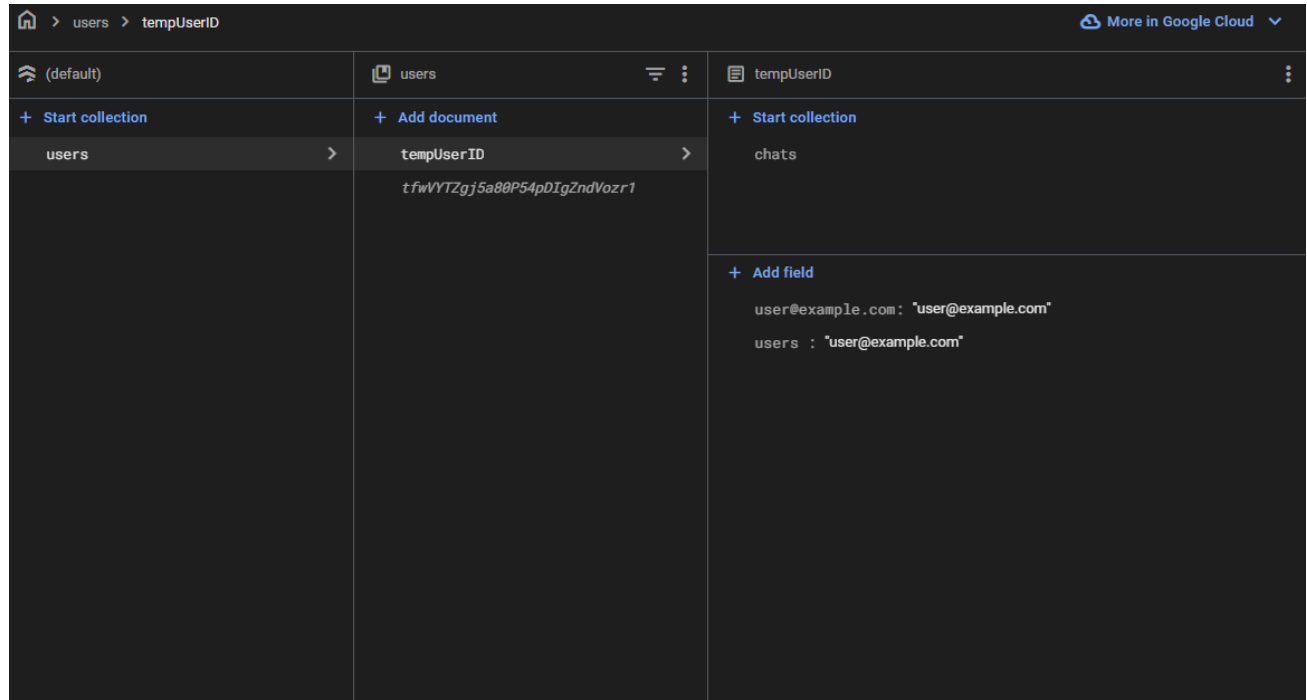
 The default security rules for test mode allow anyone with your database reference to view, edit and delete all data in your database for the next 30 days

Cancel

Create

Step 2:

Start by creating a **collection**. Add a **document** within the collection. Define the **fields** as per project



Step 3:

Run the following command to add Firestore to your Flutter app:

```
flutter pub add cloud_firestore
```

Step 4:

Go to Firebase Console → Firestore Database → Rules

Set the following rules:

```
rules_version = '2';
service cloud.firestore
{
  match /databases/{database}/documents {
    match /categories/{categoryId} {
      allow read, write: if true;
      match /subcategories/{subcategoryId}
        { allow read, write: if true;
      }
    }
  }
}
```

and then click **Publish** to apply changes.


```

Future<void> _authenticate() async {
  setState(() => isLoading = true);

  String email = _emailController.text.trim();
  String password = _passwordController.text.trim();
  String username = _usernameController.text.trim();

  if (email.isEmpty || password.isEmpty || (isSignUp && username.isEmpty)) {
    ScaffoldMessenger.of(context).showSnackBar(
      SnackBar(content: Text("All fields are required!")),
    );
    setState(() => isLoading = false);
    return;
  }

  try {
    UserCredential userCredential;
    if (isSignUp) {
      // Sign Up
      userCredential = await _auth.createUserWithEmailAndPassword(
        email: email,
        password: password,
      );
      await userCredential.user?.updateDisplayName(username);
      ScaffoldMessenger.of(context).showSnackBar(
        SnackBar(content: Text("Account Created. Please Login!")),
      );
      setState(() => isSignUp = false); // Switch to sign-in mode
    } else {
      // Sign In
      userCredential = await _auth.signInWithEmailAndPassword(
        email: email,
        password: password,
      );
      String chatId = userCredential.user?.uid ?? "defaultChatId";

      if (mounted) {
        Navigator.pushReplacement(
          context,
          MaterialPageRoute(builder: (context) => HomeScreen(chatId: chatId)), //
Pass chatId
        );
      }
    }
  }
  on FirebaseAuthException catch (e) {
    ScaffoldMessenger.of(context).showSnackBar(
      SnackBar(content: Text(e.message ?? "Authentication error")),
    );
  }
}

```

```

    } finally {
      if (mounted) setState(() => isLoading = false);
    }
  }

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      body: Padding(
        padding: EdgeInsets.symmetric(horizontal: 24),
        child: Column(
          mainAxisAlignment: MainAxisAlignment.center,
          crossAxisAlignment: CrossAxisAlignment.center,
          children: [
            Text(
              isSignUp ? "Sign Up" : "Sign In",
              style: TextStyle(fontSize: 28, fontWeight: FontWeight.bold),
            ),
            SizedBox(height: 30),
            if (isSignUp)
              TextField(
                controller: _usernameController,
                decoration: InputDecoration(
                  hintText: "Username",
                  filled: true,
                  fillColor: Colors.grey[850],
                  border: OutlineInputBorder(borderRadius:
BorderRadius.circular(10)),
                ),
              ),
            SizedBox(height: 12),
            TextField(
              controller: _emailController,
              decoration: InputDecoration(
                hintText: "Email",
                filled: true,
                fillColor: Colors.grey[850],
                border: OutlineInputBorder(borderRadius: BorderRadius.circular(10)),
              ),
            ),
            SizedBox(height: 12),
            TextField(
              controller: _passwordController,
              obscureText: !isPasswordVisible,
              decoration: InputDecoration(
                hintText: "Password",
                filled: true,
                fillColor: Colors.grey[850],

```

```

        border: OutlineInputBorder(borderRadius: BorderRadius.circular(10)),
        suffixIcon: IconButton(
          icon: Icon(
            isPasswordVisible ? Icons.visibility : Icons.visibility_off,
            color: Colors.white,
          ),
          onPressed: () {
            setState(() {
              isPasswordVisible = !isPasswordVisible;
            });
          },
        ),
      ),
    ),
    SizedBox(height: 20),
    isLoading
      ? CircularProgressIndicator()
      : ElevatedButton(
        onPressed: _authenticate,
        style: ElevatedButton.styleFrom(
          backgroundColor: Colors.blue,
          minimumSize: Size(double.infinity, 50),
        ),
        child: Text(isSignUp ? "Sign Up" : "Sign In", style:
TextStyle(fontSize: 16)),
      ),
    SizedBox(height: 16),
    TextButton(
      onPressed: () => setState(() => isSignUp = !isSignUp),
      child: Text(
        isSignUp ? "Already have an account? Sign In" : "Don't have an
account? Sign Up",
        style: TextStyle(color: Colors.white),
      ),
    ),
  ],
),
),
);
}
}

```

```
//home.dart
import 'package:flutter/material.dart';
import 'package:firebase_auth/firebase_auth.dart';
import 'package:cloud_firestore/cloud_firestore.dart';
import '../services/api_service.dart';
import '../services/firestore_service.dart';
import '../widgets/chat_bubble.dart';
import '../widgets/empty_state.dart';
import '../widgets/bottom_bar.dart';
import 'setting.dart';
import 'left_slidder.dart';

class HomeScreen extends StatefulWidget {
  final String chatId; // Accepts chatId dynamically

  HomeScreen({required this.chatId});

  @override
  _HomeScreenState createState() => _HomeScreenState();
}

class _HomeScreenState extends State<HomeScreen> {
  final TextEditingController _messageController = TextEditingController();
  final FirebaseAuth _auth = FirebaseAuth.instance;
  final FirestoreService _firestoreService = FirestoreService();
  final ApiService _apiService = ApiService();
  final ScrollController _scrollController = ScrollController();

  void _sendMessage() async {
    final user = _auth.currentUser;
    if (user == null || _messageController.text.trim().isEmpty) return;

    String userId = user.uid;
    String chatId = widget.chatId;
    String userMessage = _messageController.text.trim();

    try {
      // Store user message in Firestore
      await _firestoreService.addMessage(userId, chatId, userMessage, true);

      setState(() {
        _messageController.clear();
      });

      // Scroll to bottom after sending message
      Future.delayed(Duration(milliseconds: 300), () {
        _scrollToBottom();
      });
    }
  }
}
```

```

    });

    // Get response from Hugging Face API
    String aiResponse = await _apiService.getHuggingFaceResponse(userMessage);

    // Store AI response in Firestore
    await _firestoreService.addMessage(userId, chatId, aiResponse, false);

    print("✅ AI Response stored successfully!");
  } catch (e) {
    print("❌ Error sending message: $e");
  }
}

void _scrollToBottom() {
  if (_scrollController.hasClients) {
    _scrollController.animateTo(
      0, // Since we reversed ListView, scroll to position 0
      duration: Duration(milliseconds: 300),
      curve: Curves.easeOut,
    );
  }
}

@override
Widget build(BuildContext context) {
  final user = _auth.currentUser;
  if (user == null) {
    return Scaffold(
      body: Center(child: Text("User not logged in")),
    );
  }

  return Scaffold(
    backgroundColor: Colors.black,
    appBar: AppBar(
      backgroundColor: Colors.black,
      elevation: 0,
      leading: Builder(
        builder: (context) => IconButton(
          icon: Icon(Icons.menu, color: Colors.white),
          onPressed: () {
            Scaffold.of(context).openDrawer();
          },
        ),
      ),
    ),
    actions: [
      TextButton(

```



```

        onPressed: () {},
        child: Text("Get Plus ✨", style: TextStyle(color: Colors.white)),
      ),
      IconButton(
        icon: Icon(Icons.more_vert, color: Colors.white),
        onPressed: () {},
      ),
    ],
  ),
  drawer: LeftSlider(currentChatId: widget.chatId),
  body: Column(
    children: [
      Expanded(
        child: StreamBuilder<QuerySnapshot>(
          stream: _firestoreService.getMessage(user.uid, widget.chatId),
          builder: (context, snapshot) {
            if (snapshot.connectionState == ConnectionState.waiting) {
              return Center(child: CircularProgressIndicator());
            }

            if (!snapshot.hasData || snapshot.data!.docs.isEmpty) {
              return EmptyState(); // Show empty state if no messages
            }



            return ListView(
              controller: _scrollController,
              padding: EdgeInsets.all(16),
              reverse: true, // 🔥 Messages appear from bottom to top
              children: snapshot.data!.docs.map((doc) {
                Map<String, dynamic> data = doc.data() as Map<String, dynamic>;
                bool isUser = data['isUser'] ?? false;
                return ChatBubble(message: data['message'] ?? '', isUser:
isUser);
              }).toList(),
            );
          },
        ),
      ),
      BottomBar(
        messageController: _messageController,
        onSendPressed: _sendMessage,
      ),
    ],
  ),
);
}
}

```


Output:

Signup

Page:

8:28  



Sign Up




Sign Up

Already have an account? [Sign In](#)

Signin Page:

8:28  

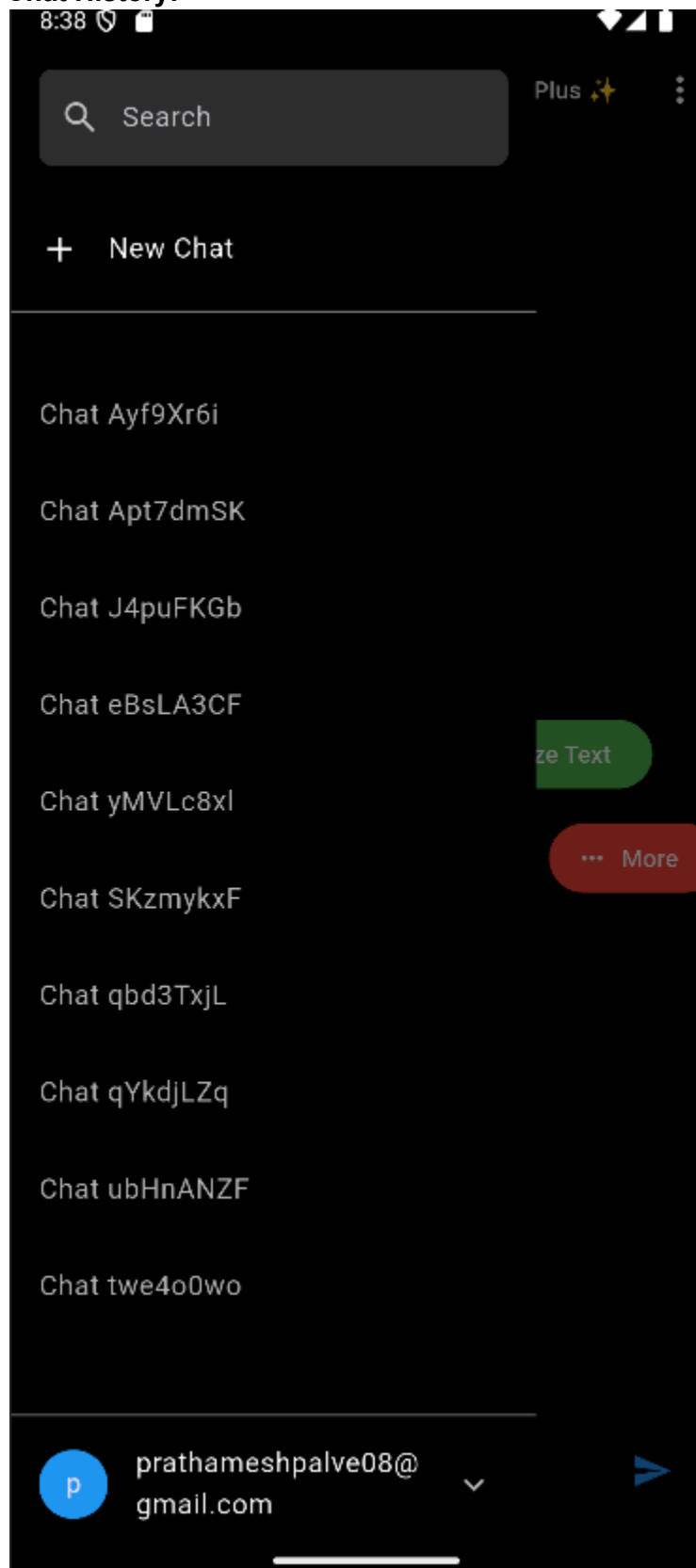
Sign In



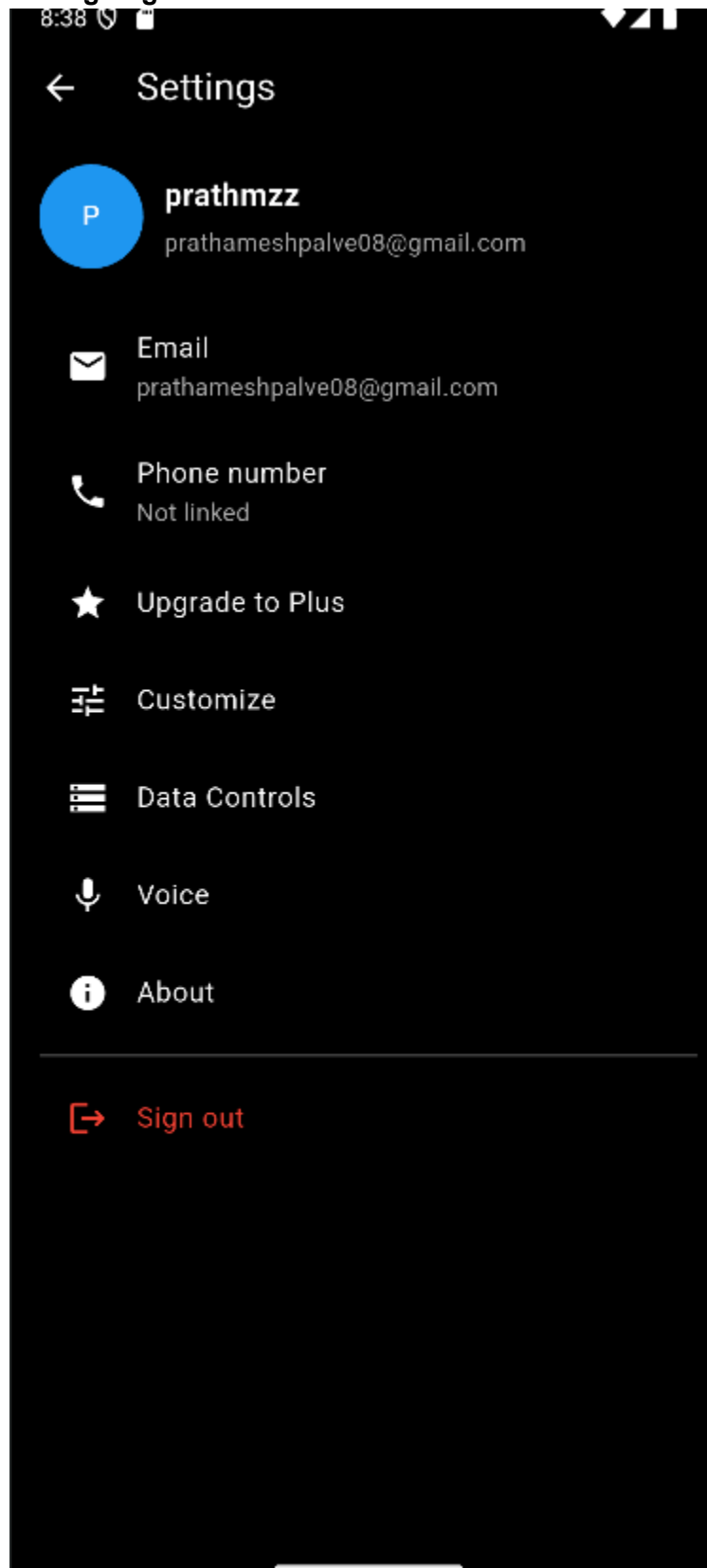
Sign In

Don't have an account? [Sign Up](#)

Chat History:



Setting Page:



home Page:

