Sujal Taktani **Batch :- C**
D15A_60

Experiment no. :- 06

Aim: To connect firebase database with flutter ui

Theory:

Firebase is a great backend solution for anyone that wants to use authentication, databases, cloud functions, ads, and countless other features within an app.

Prerequisites

To complete this tutorial, you will need:

- A Google account to use Firebase.
- Developing for iOS will require XCode.
- To download and install Flutter.
- To download and install Android Studio and Visual Studio Code.
- It is recommended to install plugins for your code editor:
    - Flutter and Dart plugins installed for Android Studio.
    - Flutter extension installed for Visual Studio Code

This tutorial was verified with Flutter v2.0.6, Android SDK v31.0.2, and Android Studio v4.1.

Creating a New Flutter Project

This tutorial will require the creation of an example Flutter app.

Once you have your environment set up for Flutter, you can run the following to create a new application:

flutter create flutterfirebaseexample

Using flutter create will produce a demo application that will display the number of times a button is clicked.

Now that we've got a Flutter project up and running, we can add Firebase.

Creating a New Firebase Project

First, log in with your Google account to manage your Firebase projects. From within the Firebase dashboard, select the Create new project button and give it a name

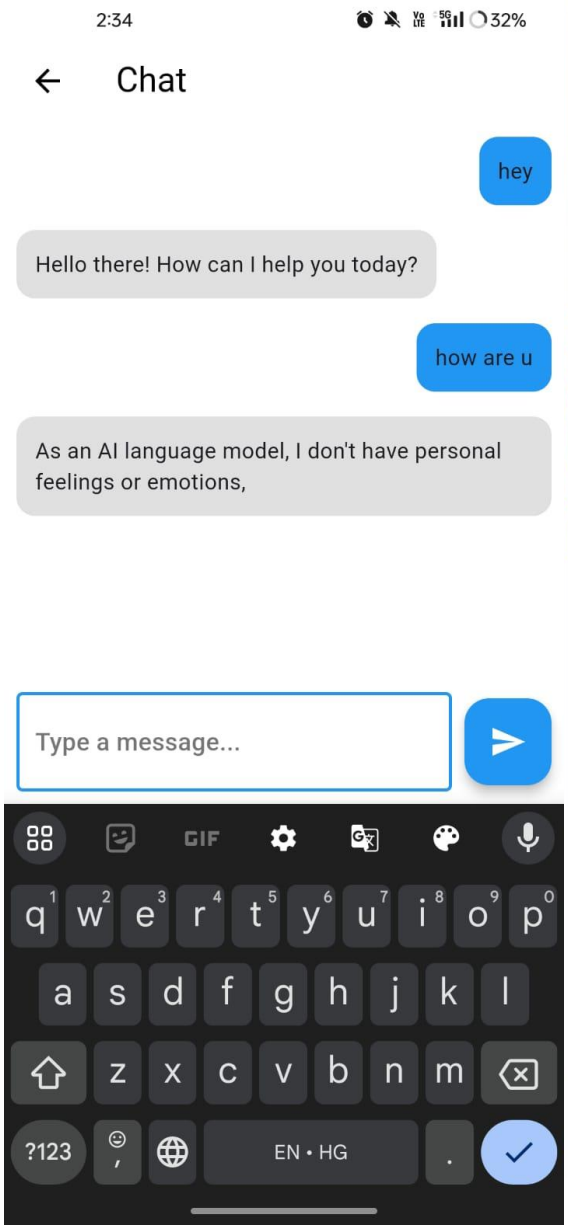
Code:

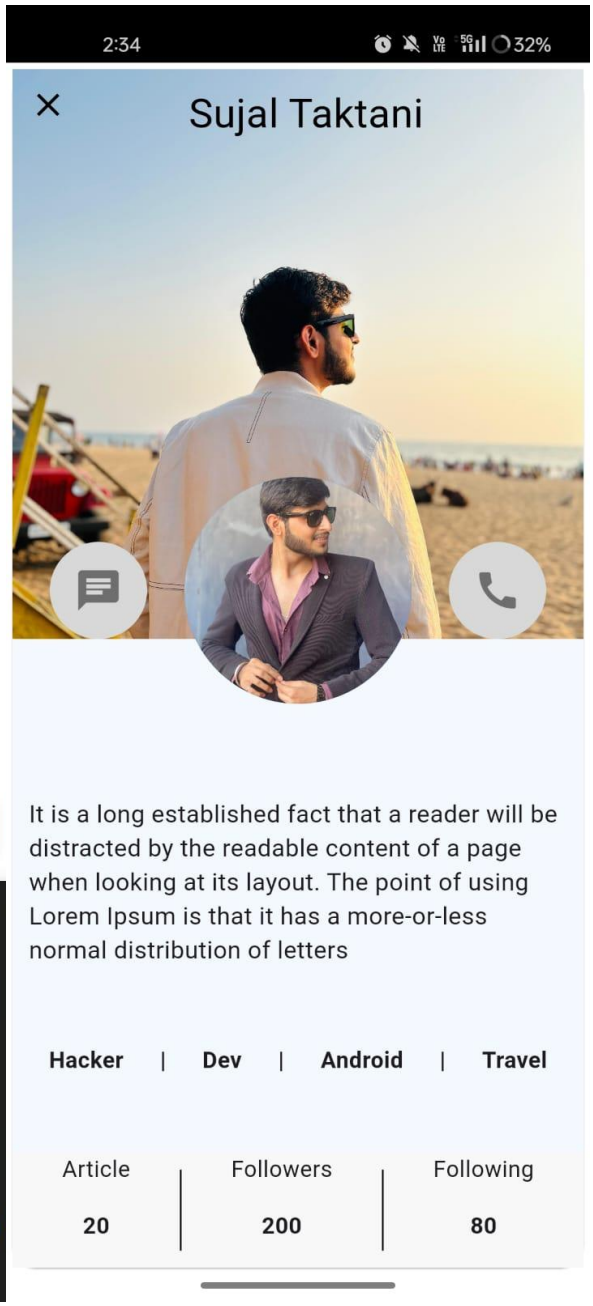
```
import 'dart:convert';

import 'package:flutter/material.dart';
import 'package:http/http.dart' as http;

class ChatScreen extends StatefulWidget {
  @override
  _ChatScreenState createState() => _ChatScreenState();
}

class _ChatScreenState extends State<ChatScreen> {
  List<Message> messages = [];
  TextEditingController _textEditingController = TextEditingController();

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: Text('Chat'),
      ),
      body: Column(
        children: <Widget>[
          Expanded(
            child: ListView.builder(
              itemCount: messages.length,
              itemBuilder: (context, index) {
                return _buildMessage(messages[index]);
              },
            ),
          ),
          _buildMessageInputArea(),
        ],
      ),
    );
  }

  Widget _buildMessage(Message message) {
    return Align(
      alignment: message.isUserMessage ? Alignment.centerRight : Alignment.centerLeft,
      child: Padding(
        padding: const EdgeInsets.all(8.0),
        child: Container(
          padding: EdgeInsets.all(12),
          decoration: BoxDecoration(
            color: message.isUserMessage ? Colors.blue : Colors.grey[300],
            borderRadius: BorderRadius.circular(12),
          ),
          child: Text(message.text),
        ),
```

```
    ),
   );
  }

  Widget _buildMessageInputArea() {
   return Padding(
    padding: const EdgeInsets.all(8.0),
    child: Row(
     children: <Widget>[
      Expanded(
       child: TextField(
        controller: _textEditingController,
        decoration: InputDecoration(
         hintText: 'Type a message...',
         border: OutlineInputBorder(),
        ),
       ),
      ),
      SizedBox(width: 8),
      FloatingActionButton(
       onPressed: () {
        _sendMessage(_textEditingController.text);
       },
       child: Icon(Icons.send),
      ),
     ],
    ),
   );
  }

  void _sendMessage(String message) async {
   setState(() {
    messages.add(Message(message, true)); // Add user's message
    _textEditingController.clear(); // Clear text field
   });

   // Define the API endpoint URL
   var apiUrl = 'https://generativelanguage.googleapis.com/v1/models/gemini-
pro:streamGenerateContent?key=AIzaSyBUA19PoTh7qUmJPnpDiow5n7R2lvIHCKM';

   // Define the request body
   var requestBody = json.encode({
    "contents": [
     {
      "role": "user",
      "parts": [{"text": message}]
     }
    ]
   });

   // Make the HTTP POST request
   var response = await http.post(
    Uri.parse(apiUrl),
    headers: {'Content-Type': 'application/json'},
    body: requestBody,
   );
```

```
    // Check if the request was successful
    if (response.statusCode == 200) {
      // Parse the response JSON
      var responseBody = json.decode(response.body);

      // Extract the API response text from the JSON and add it to the messages list
      if (responseBody is List && responseBody.isNotEmpty) {
        var candidates = responseBody[0]['candidates'] as List;
        if (candidates.isNotEmpty) {
          var content = candidates[0]['content'];
          var parts = content['parts'] as List;
          if (parts.isNotEmpty) {
            var apiResponseText = parts[0]['text'];
            setState(() {
              messages.add(Message(apiResponseText, false)); // Add API response
            });
          }
        }
      }
    } else {
      // Handle the error if the request fails
      print('Error: ${response.statusCode}');
    }
  }
}

class Message {
  final String text;
  final bool isUserMessage;

  Message(this.text, this.isUserMessage);
}
```

Implementation:


2:34
32%
Chat
hey
Hello there! How can I help you today?
how are u
As an AI language model, I don't have personal feelings or emotions,
Type a message...
GIF
EN • HG


2:34
32%
Sujal Taktani
It is a long established fact that a reader will be distracted by the readable content of a page when looking at its layout. The point of using Lorem Ipsum is that it has a more-or-less normal distribution of letters
Hacker | Dev | Android | Travel
Article
20
Followers
200
Following
80

```yaml
dependencies:
  any_link_preview: ^3.0.1
  cloud_firestore: ^4.14.0
  cupertino_icons: ^1.0.2
  dotted_border: ^2.1.0
  file_picker: ^6.1.1
  firebase_auth: ^4.16.0
  firebase_core: ^2.24.2
  firebase_storage: ^11.6.0
  flutter:
    sdk: flutter
  flutter_riverpod: ^2.4.9
  fpdart: ^1.1.0
  google_sign_in: ^6.2.1
  routemaster: ^1.0.1
  shared_preferences: ^2.2.2
  uuid: ^4.3.3
```