import 'dart:convert';

import 'package:flutter/material.dart';

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

void main() {

  runApp(MyApp());

}

class MyApp extends StatelessWidget {

  @override

  Widget build(BuildContext context) {

    return MaterialApp(

      title: 'GitHub API Example',

      theme: ThemeData(

        primarySwatch: Colors.blue,

      ),

      home: HomePage(),

    );

  }

}

class Repository {

  final String name;

  final String description;

  final int stargazersCount;

  Repository(

      {required [this.name](http://this.name/),

      required this.description,

      required this.stargazersCount});

  factory Repository.fromJson(Map<String, dynamic> json) {

    return Repository(

      name: json['name'],

      description: json['description'] ?? 'No description available',

      stargazersCount: json['stargazers\_count'],

    );

  }

}

class HomePage extends StatefulWidget {

  @override

  \_HomePageState createState() => \_HomePageState();

}

class \_HomePageState extends State<HomePage> {

  late Future<List<Repository>> \_futureRepositories;

  final TextEditingController \_usernameController = TextEditingController();

  @override

  void initState() {

    super.initState();

    \_futureRepositories = fetchRepositories('');

  }

  Future<List<Repository>> fetchRepositories(String username) async {

    final response = await http

        .get(Uri.parse('<https://api.github.com/users/$username/repos')>);

    if (response.statusCode == 200) {

      final List<dynamic> jsonData = jsonDecode(response.body);

      return jsonData.map((repo) => Repository.fromJson(repo)).toList();

    } else {

      throw Exception('Failed to fetch repositories');

    }

  }

  @override

  void dispose() {

    \_usernameController.dispose();

    super.dispose();

  }

  @override

  Widget build(BuildContext context) {

    return Scaffold(

      appBar: AppBar(

        title: Text('GitHub API Example'),

      ),

      body: Column(

        children: [

          Padding(

            padding: const EdgeInsets.all(8.0),

            child: TextField(

              controller: \_usernameController,

              decoration: InputDecoration(

                labelText: 'GitHub Username',

              ),

            ),

          ),

          ElevatedButton(

            onPressed: () {

              setState(() {

                \_futureRepositories =

                    fetchRepositories(\_usernameController.text);

              });

            },

            child: Text('Fetch Repositories'),

          ),

          Expanded(

            child: FutureBuilder<List<Repository>>(

              future: \_futureRepositories,

              builder: (context, snapshot) {

                if (snapshot.connectionState == ConnectionState.waiting) {

                  return Center(

                    child: CircularProgressIndicator(),

                  );

                } else if (snapshot.hasData) {

                  final repositories = snapshot.data!;

                  return ListView.builder(

                    itemCount: repositories.length,

                    itemBuilder: (context, index) {

                      final repository = repositories[index];

                      return ListTile(

                        title: Text([repository.name](http://repository.name/)),

                        subtitle: Text(repository.description),

                        trailing: Text('Stars: ${repository.stargazersCount}'),

                      );

                    },

                  );

                } else if (snapshot.hasError) {

                  return Center(

                    child: Text('Error: ${snapshot.error}'),

                  );

                } else {

                  return Container();

                }

              },

            ),

          ),

        ],

      ),

    );

  }

}