LAPORAN PRAKTIKUM

PEMROGRAMAN PERANGKAT BERGERAK

MODUL 14

DATA STORAGE BAGIAN 3

Disusun Oleh :

YOGI HAFIDH MAULANA / 2211104061

SE06-02

Asisten Praktikum :

Muhammad Faza Zulian Gesit Al Barru

Aisyah Hasna Aulia

Dosen Pengampu :

Yudha Islami Sulistya, S.Kom., M.Cs.

PROGRAM STUDI S1 SOFTWARE ENGINEERING

FAKULTAS INFORMATIKA

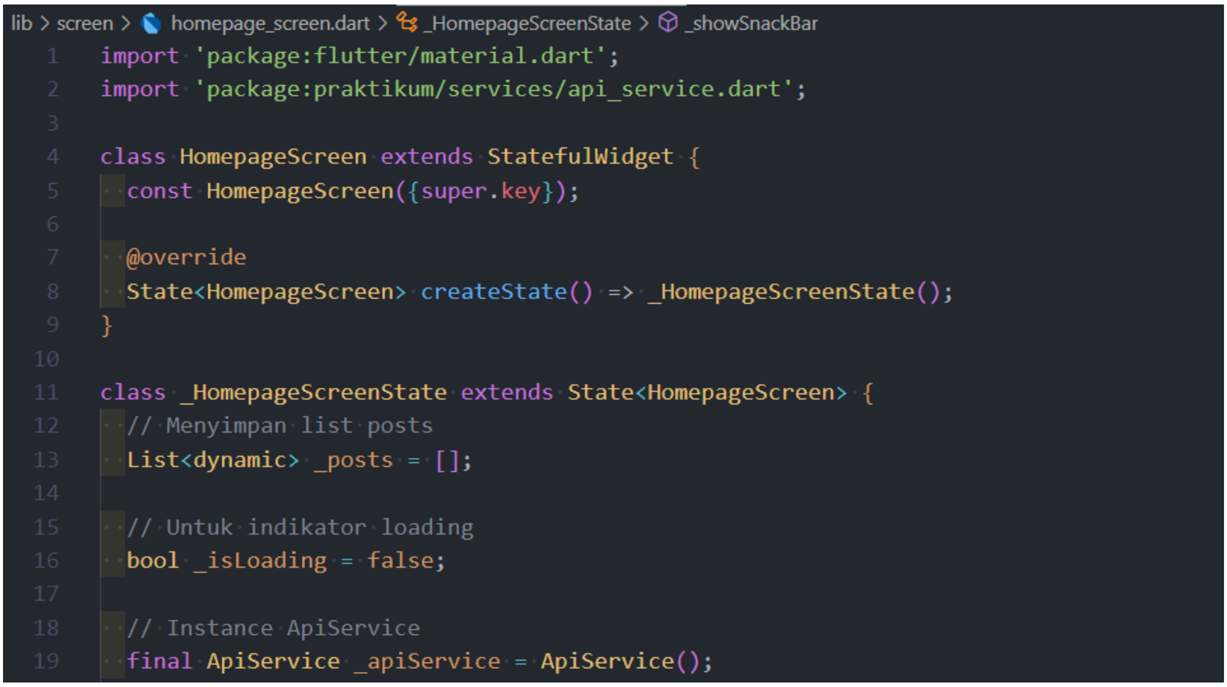
TELKOM UNIVERSITY PURWOKERTO

2024

A. GUIDED

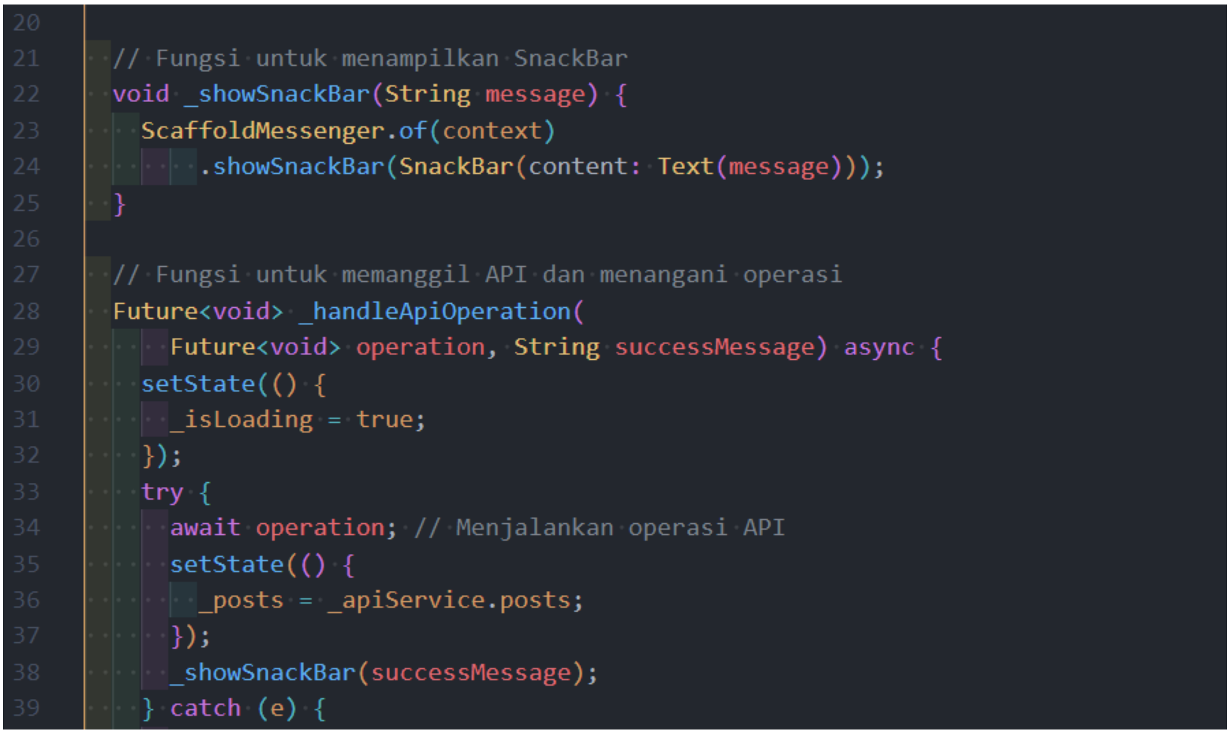
- **Homepage_screen.dart**

Code

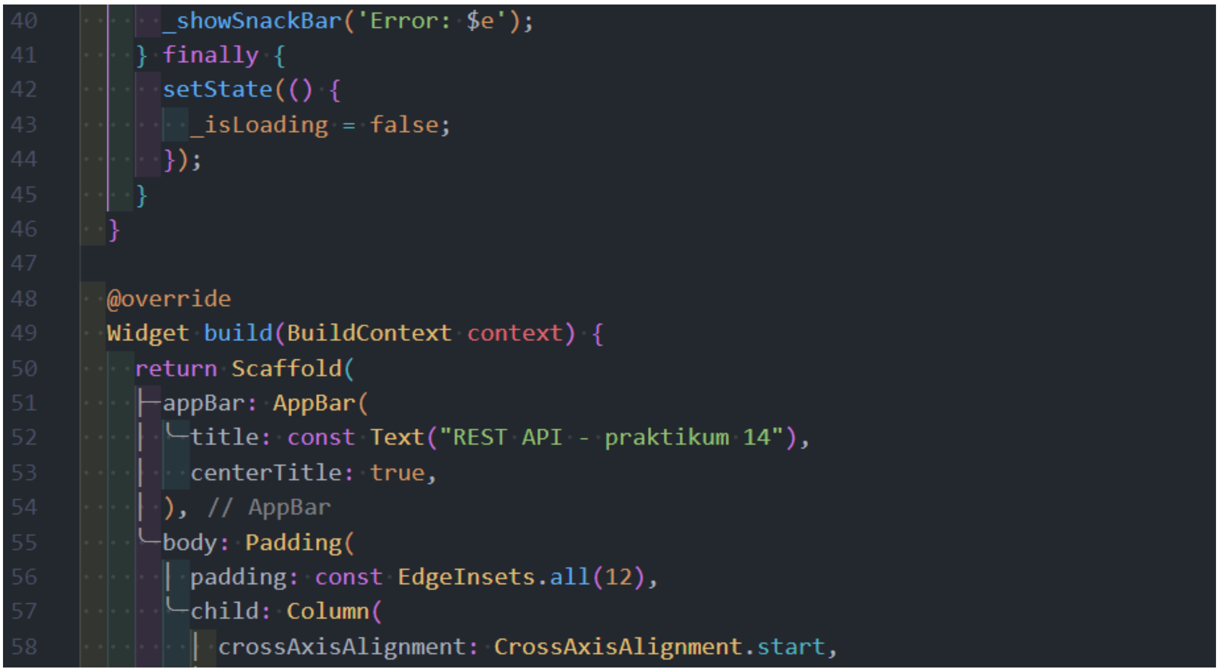


lib > screen > homepage_screen.dart > _HomepageScreenState > _showSnackBar

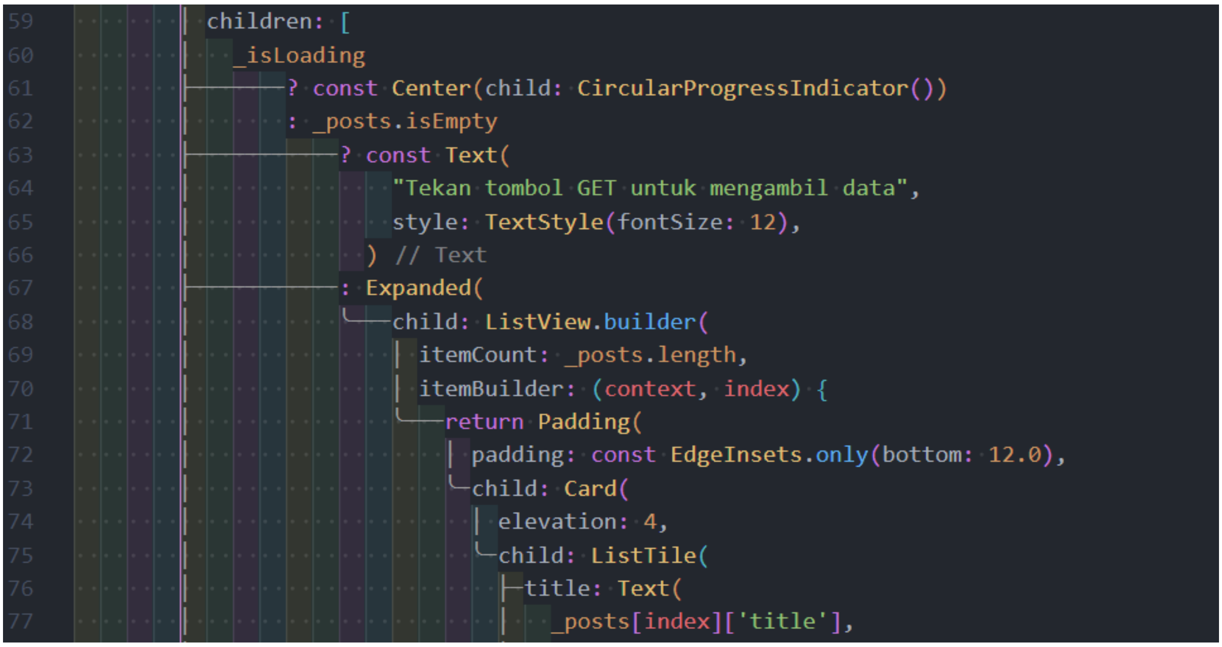
```dart
import 'package:flutter/material.dart';
import 'package:praktikum/services/api_service.dart';

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

  @override
  State<HomepageScreen> createState() => _HomepageScreenState();
}

class _HomepageScreenState extends State<HomepageScreen> {
  // Menyimpan list posts
  List<dynamic> _posts = [];

  // Untuk indikator loading
  bool _isLoading = false;

  // Instance ApiService
  final ApiService _apiService = ApiService();
```



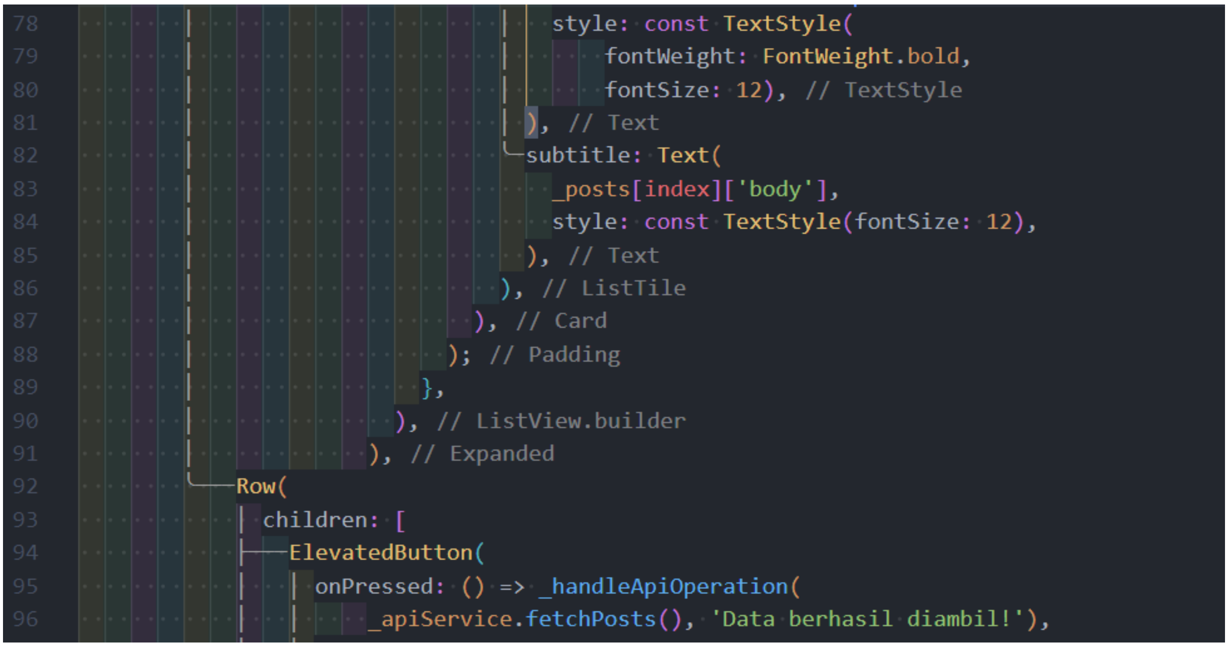
```dart

  // Fungsi untuk menampilkan SnackBar
  void _showSnackBar(String message) {
    ScaffoldMessenger.of(context)
        .showSnackBar(SnackBar(content: Text(message)));
  }

  // Fungsi untuk memanggil API dan menangani operasi
  Future<void> _handleApiOperation(
      Future<void> operation, String successMessage) async {
    setState(() {
      _isLoading = true;
    });
    try {
      await operation; // Menjalankan operasi API
      setState(() {
        _posts = _apiService.posts;
      });
      _showSnackBar(successMessage);
    } catch (e) {
```



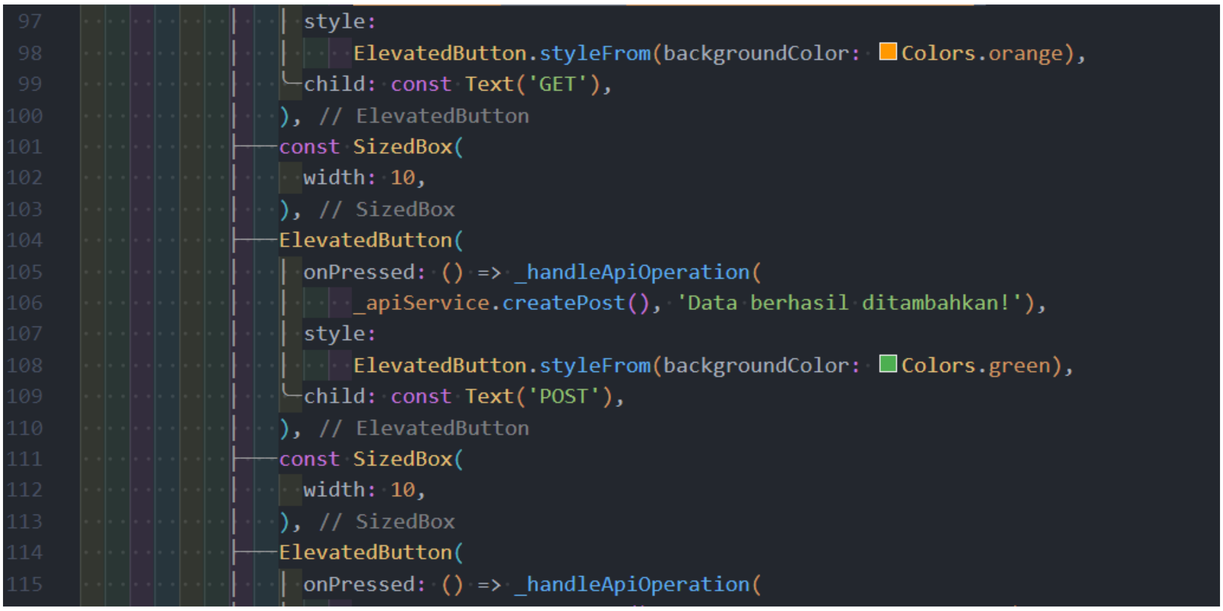
```dart
      _showSnackBar('Error: $e');
    } finally {
      setState(() {
        _isLoading = false;
      });
    }
  }

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: const Text("REST API - praktikum 14"),
        centerTitle: true,
      ), // AppBar
      body: Padding(
        padding: const EdgeInsets.all(12),
        child: Column(
          crossAxisAlignment: CrossAxisAlignment.start,
```



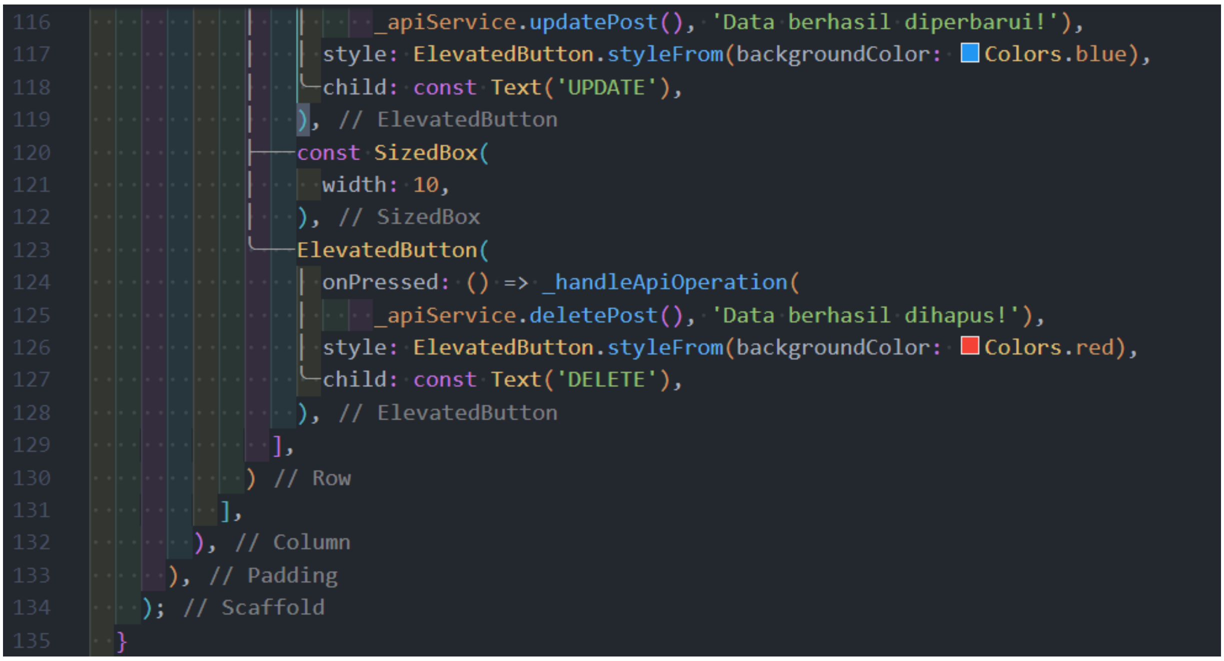
```dart
          children: [
            _isLoading
                ? const Center(child: CircularProgressIndicator())
                : _posts.isEmpty
                    ? const Text(
                        "Tekan tombol GET untuk mengambil data",
                        style: TextStyle(fontSize: 12),
                      ) // Text
                    : Expanded(
                        child: ListView.builder(
                          itemCount: _posts.length,
                          itemBuilder: (context, index) {
                            return Padding(
                              padding: const EdgeInsets.only(bottom: 12.0),
                              child: Card(
                                elevation: 4,
                                child: ListTile(
                                  title: Text(
                                    _posts[index]['title'],
```



```dart
                                    style: const TextStyle(
                                        fontWeight: FontWeight.bold,
                                        fontSize: 12), // TextStyle
                                  ), // Text
                                  subtitle: Text(
                                    _posts[index]['body'],
                                    style: const TextStyle(fontSize: 12),
                                  ), // Text
                                ), // ListTile
                              ), // Card
                            ); // Padding
                          },
                        ), // ListView.builder
                      ), // Expanded
            Row(
              children: [
                ElevatedButton(
                  onPressed: () => _handleApiOperation(
                      _apiService.fetchPosts(), 'Data berhasil diambil!'),
```

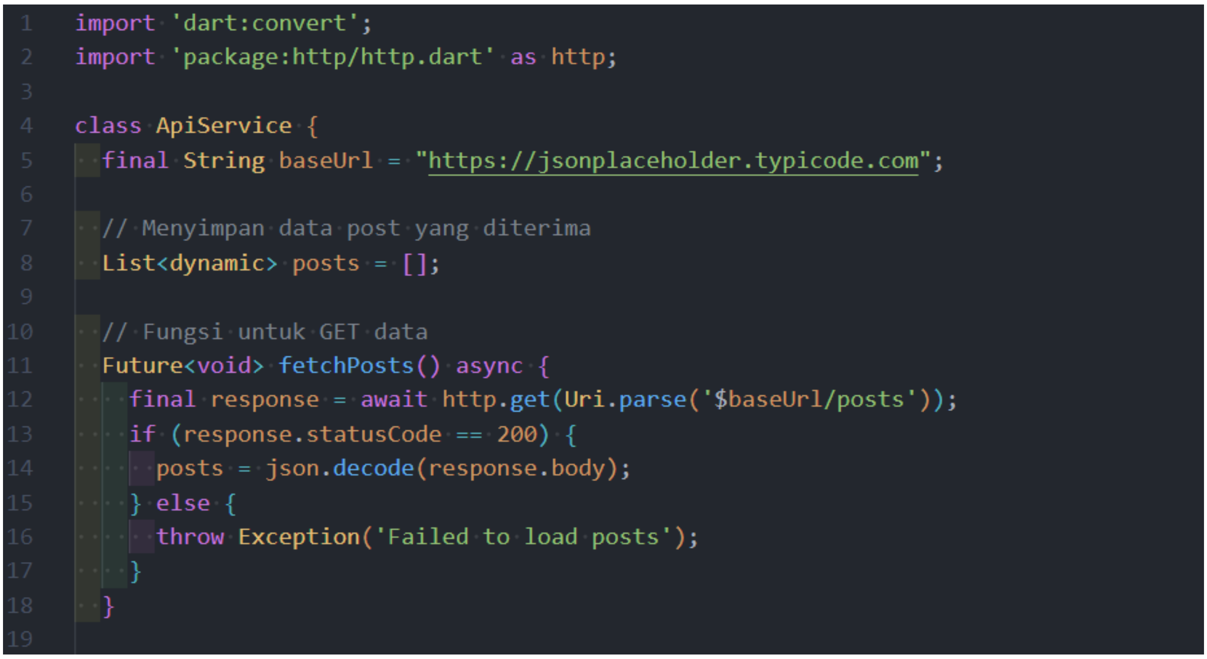


```dart
                  style:
                      ElevatedButton.styleFrom(backgroundColor: 🟧Colors.orange),
                  child: const Text('GET'),
                ), // ElevatedButton
                const SizedBox(
                  width: 10,
                ), // SizedBox
                ElevatedButton(
                  onPressed: () => _handleApiOperation(
                      _apiService.createPost(), 'Data berhasil ditambahkan!'),
                  style:
                      ElevatedButton.styleFrom(backgroundColor: 🟩Colors.green),
                  child: const Text('POST'),
                ), // ElevatedButton
                const SizedBox(
                  width: 10,
                ), // SizedBox
                ElevatedButton(
                  onPressed: () => _handleApiOperation(
```

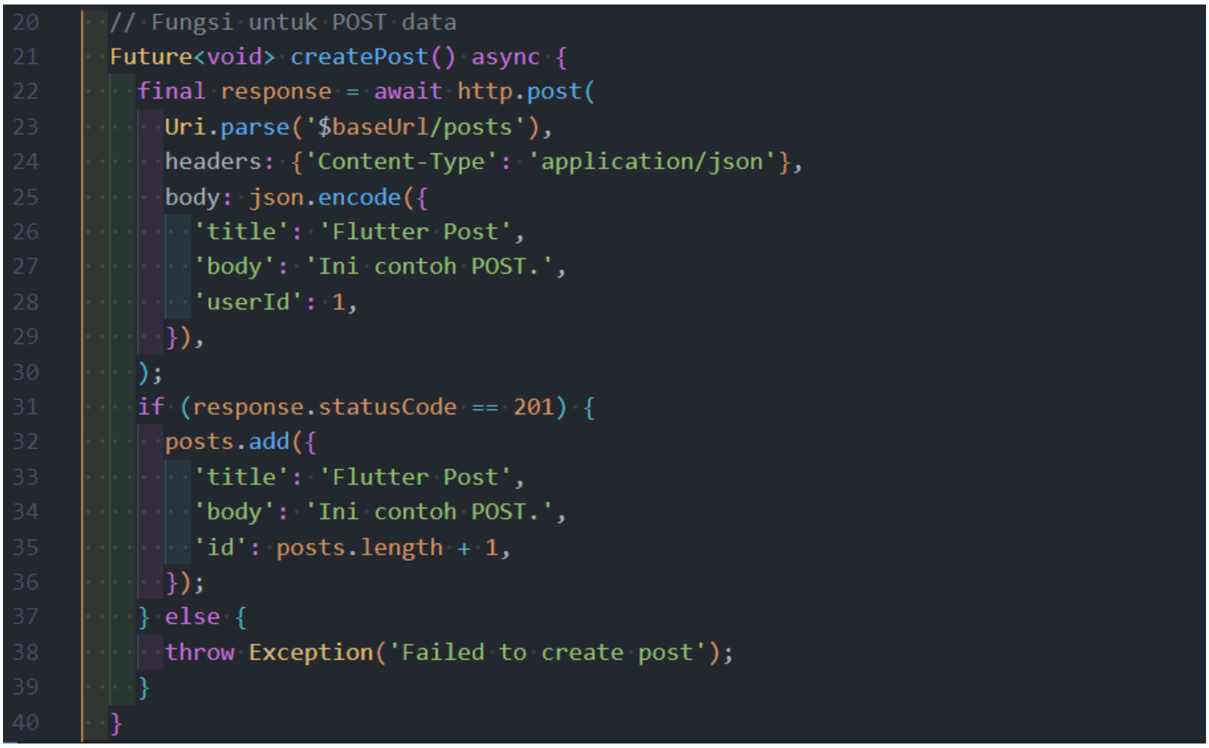


```dart
                      _apiService.updatePost(), 'Data berhasil diperbarui!'),
                  style: ElevatedButton.styleFrom(backgroundColor: 🟦Colors.blue),
                  child: const Text('UPDATE'),
                ), // ElevatedButton
                const SizedBox(
                  width: 10,
                ), // SizedBox
                ElevatedButton(
                  onPressed: () => _handleApiOperation(
                      _apiService.deletePost(), 'Data berhasil dihapus!'),
                  style: ElevatedButton.styleFrom(backgroundColor: 🟥Colors.red),
                  child: const Text('DELETE'),
                ), // ElevatedButton
              ],
            ) // Row
          ],
        ), // Column
      ), // Padding
    ); // Scaffold
  }
```

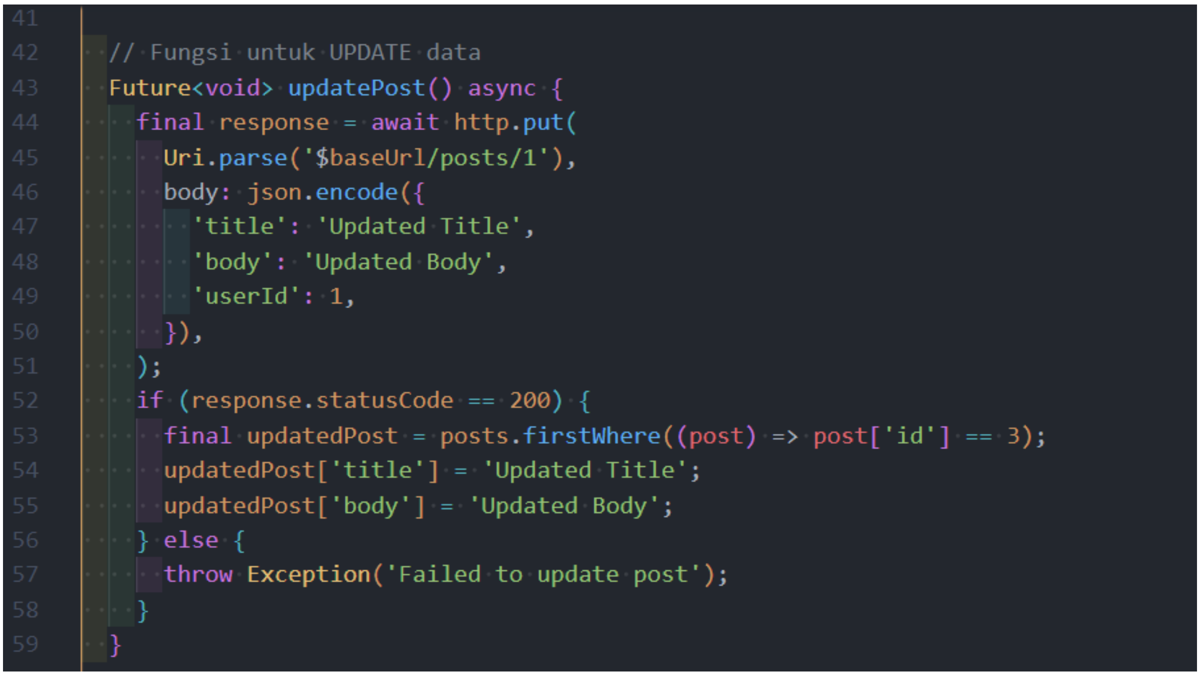
- **Api_service.dart**



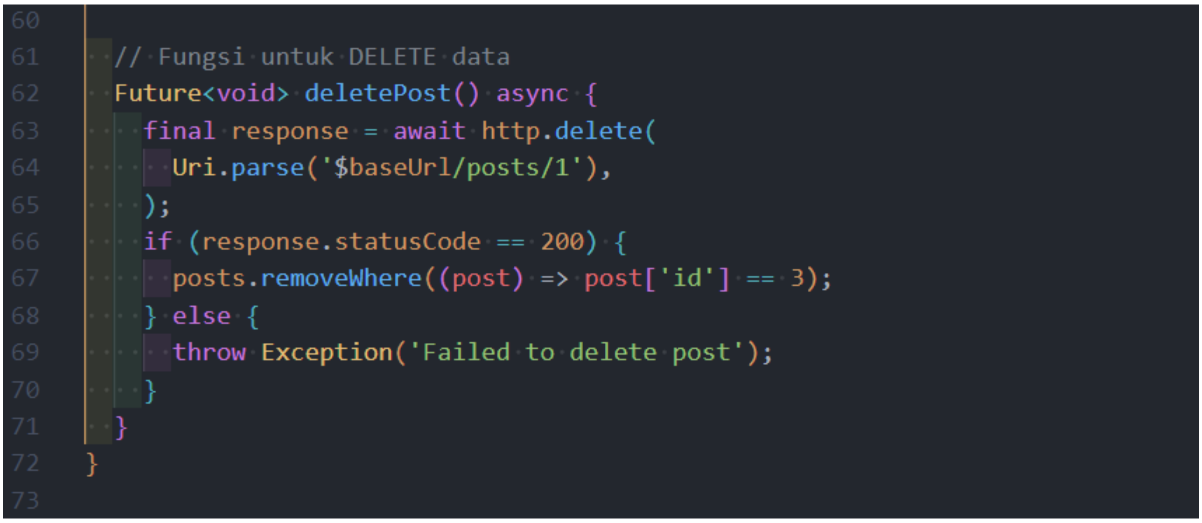
```dart
import 'dart:convert';
import 'package:http/http.dart' as http;

class ApiService {
  final String baseUrl = "https://jsonplaceholder.typicode.com";

  // Menyimpan data post yang diterima
  List<dynamic> posts = [];

  // Fungsi untuk GET data
  Future<void> fetchPosts() async {
    final response = await http.get(Uri.parse('$baseUrl/posts'));
    if (response.statusCode == 200) {
      posts = json.decode(response.body);
    } else {
      throw Exception('Failed to load posts');
    }
  }

```



```dart
  // Fungsi untuk POST data
  Future<void> createPost() async {
    final response = await http.post(
      Uri.parse('$baseUrl/posts'),
      headers: {'Content-Type': 'application/json'},
      body: json.encode({
        'title': 'Flutter Post',
        'body': 'Ini contoh POST.',
        'userId': 1,
      }),
    );
    if (response.statusCode == 201) {
      posts.add({
        'title': 'Flutter Post',
        'body': 'Ini contoh POST.',
        'id': posts.length + 1,
      });
    } else {
      throw Exception('Failed to create post');
    }
  }
```

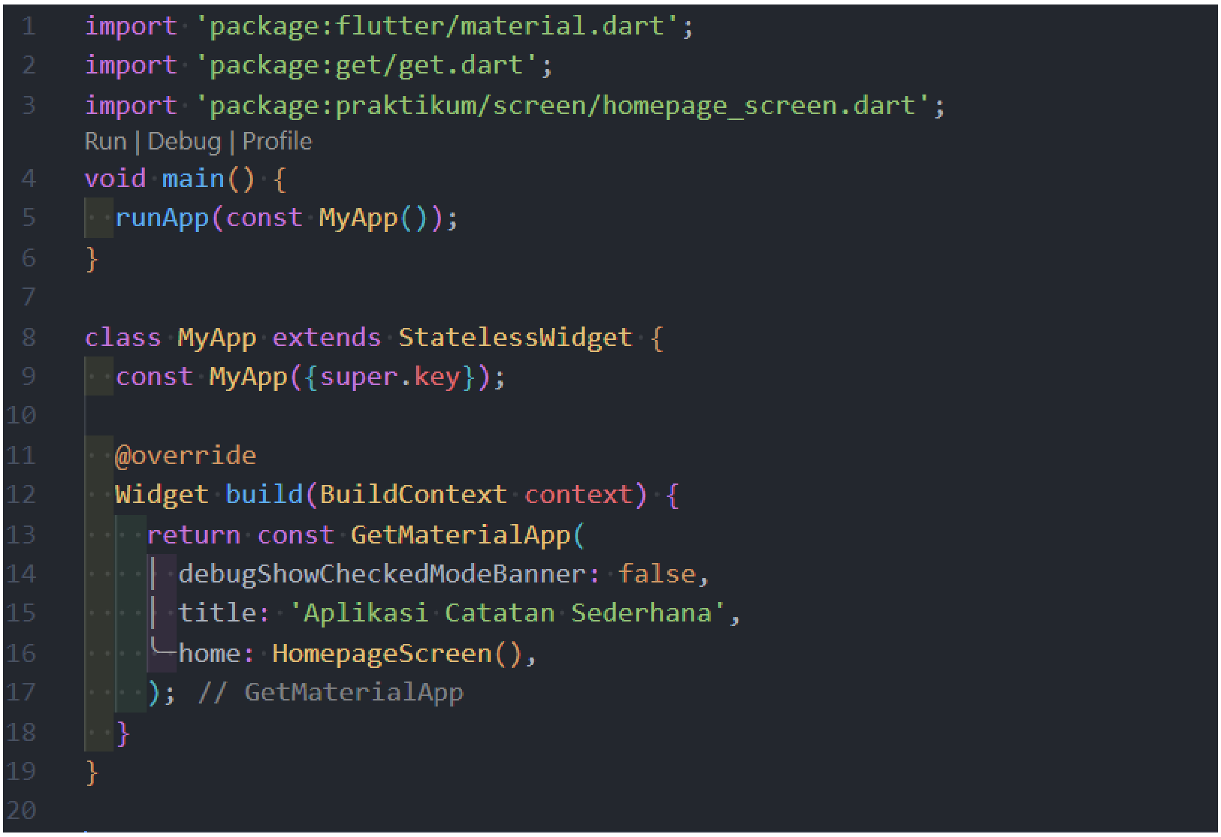


```dart

  // Fungsi untuk UPDATE data
  Future<void> updatePost() async {
    final response = await http.put(
      Uri.parse('$baseUrl/posts/1'),
      body: json.encode({
        'title': 'Updated Title',
        'body': 'Updated Body',
        'userId': 1,
      }),
    );
    if (response.statusCode == 200) {
      final updatedPost = posts.firstWhere((post) => post['id'] == 3);
      updatedPost['title'] = 'Updated Title';
      updatedPost['body'] = 'Updated Body';
    } else {
      throw Exception('Failed to update post');
    }
  }
```



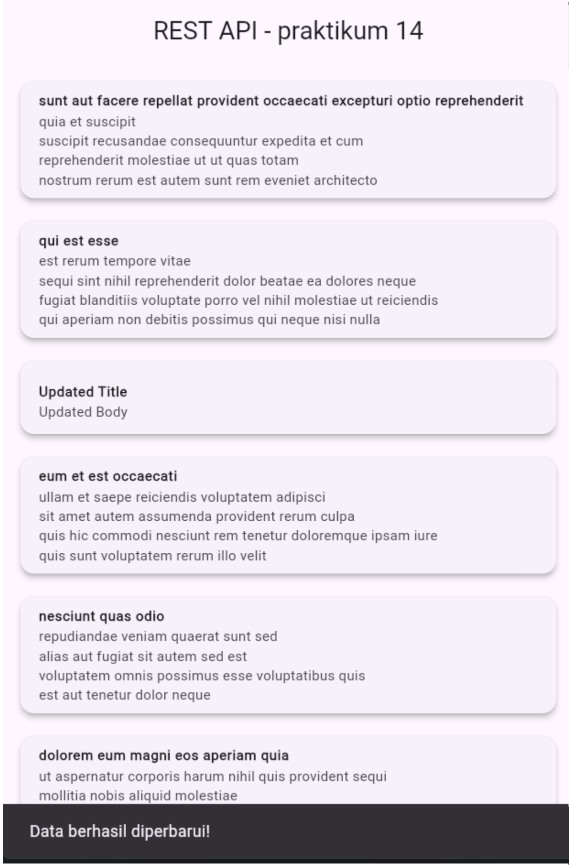
```dart

  // Fungsi untuk DELETE data
  Future<void> deletePost() async {
    final response = await http.delete(
      Uri.parse('$baseUrl/posts/1'),
    );
    if (response.statusCode == 200) {
      posts.removeWhere((post) => post['id'] == 3);
    } else {
      throw Exception('Failed to delete post');
    }
  }
}

```

- **Main.dart**

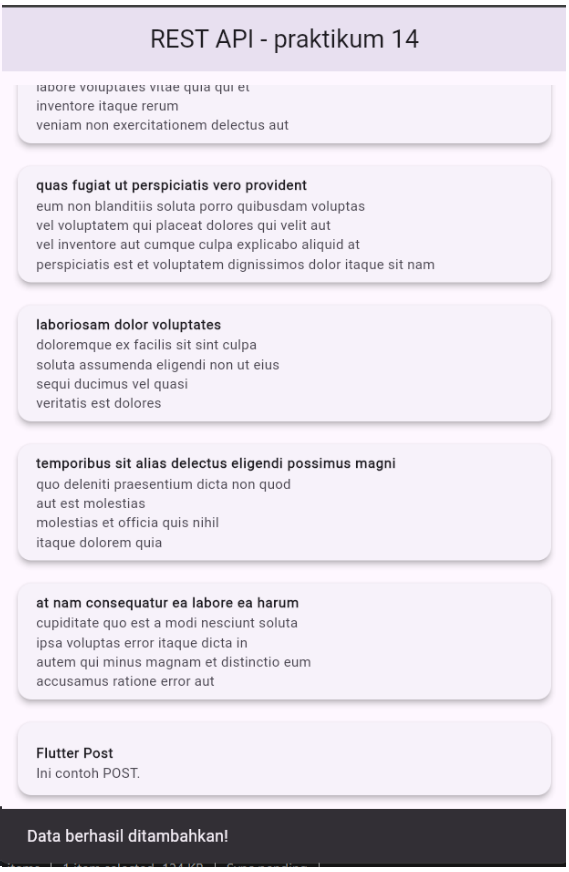


```dart
import 'package:flutter/material.dart';
import 'package:get/get.dart';
import 'package:praktikum/screen/homepage_screen.dart';
Run | Debug | Profile
void main() {
  runApp(const MyApp());
}

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

  @override
  Widget build(BuildContext context) {
    return const GetMaterialApp(
      debugShowCheckedModeBanner: false,
      title: 'Aplikasi Catatan Sederhana',
      home: HomepageScreen(),
    ); // GetMaterialApp
  }
}

```

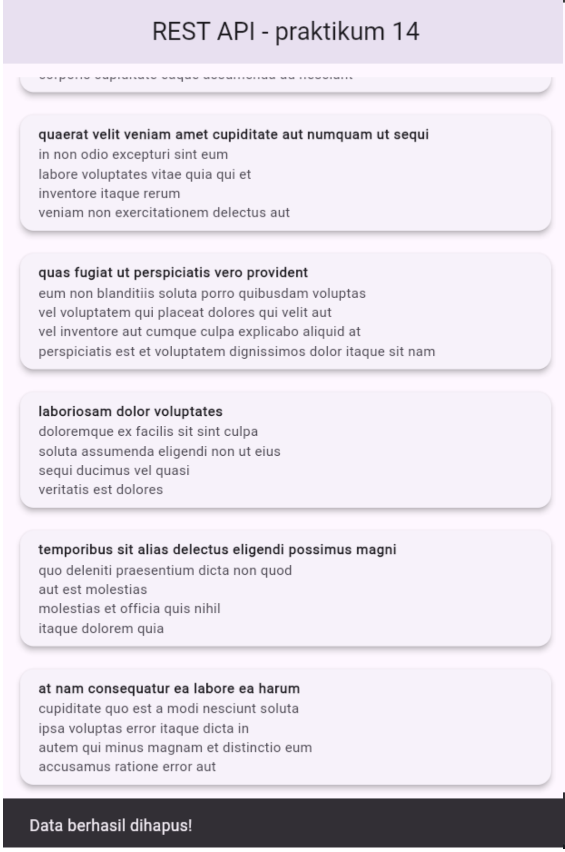
Output



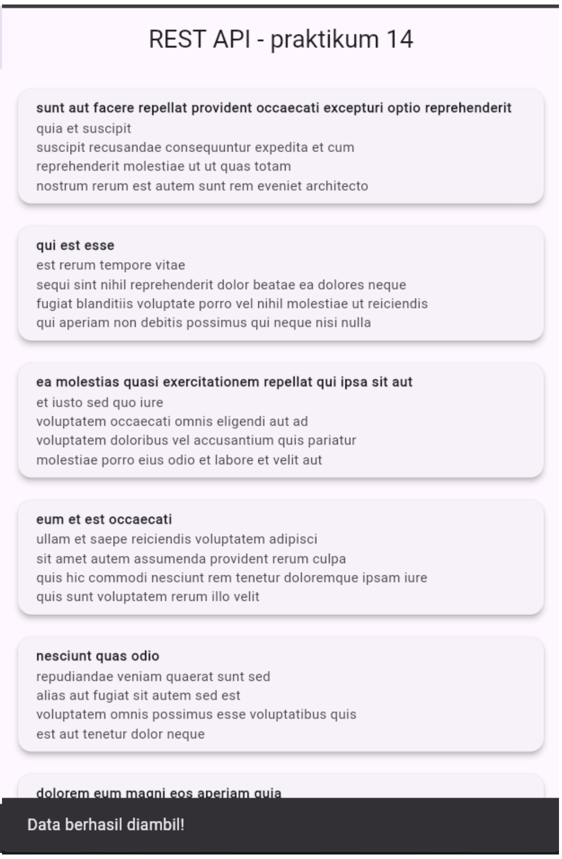
```
REST API - praktikum 14

sunt aut facere repellat provident occaecati excepturi optio reprehenderit
quia et suscipit
suscipit recusandae consequuntur expedita et cum
reprehenderit molestiae ut ut quas totam
nostrum rerum est autem sunt rem eveniet architecto

qui est esse
est rerum tempore vitae
sequi sint nihil reprehenderit dolor beatae ea dolores neque
fugiat blanditiis voluptate porro vel nihil molestiae ut reiciendis
qui aperiam non debitis possimus qui neque nisi nulla

Updated Title
Updated Body

eum et est occaecati
ullam et saepe reiciendis voluptatem adipisci
sit amet autem assumenda provident rerum culpa
quis hic commodi nesciunt rem tenetur doloremque ipsam iure
quis sunt voluptatem rerum illo velit

nesciunt quas odio
repudiandae veniam quaerat sunt sed
alias aut fugiat sit autem sed est
voluptatem omnis possimus esse voluptatibus quis
est aut tenetur dolor neque

dolorem eum magni eos aperiam quia
ut aspernatur corporis harum nihil quis provident sequi
mollitia nobis aliquid molestiae

Data berhasil diperbarui!
```



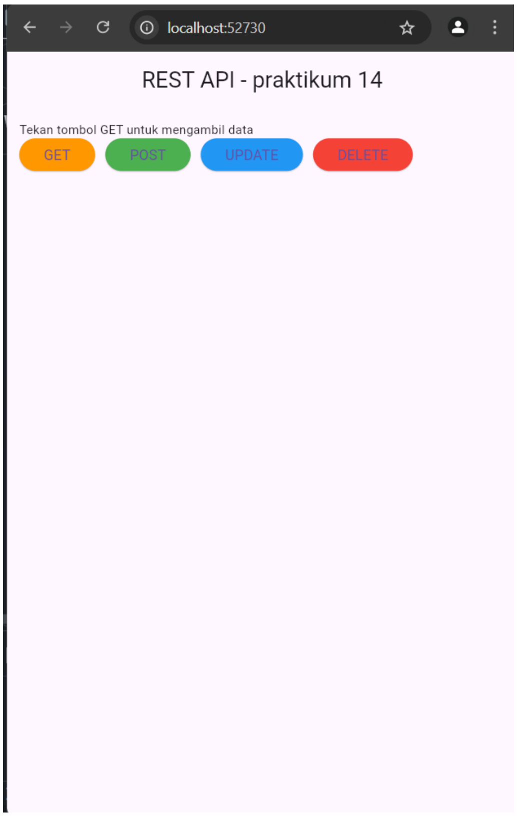
```
REST API - praktikum 14

labore voluptates vitae quia qui et
inventore itaque rerum
veniam non exercitationem delectus aut

quas fugiat ut perspiciatis vero provident
eum non blanditiis soluta porro quibusdam voluptas
vel voluptatem qui placeat dolores qui velit aut
vel inventore aut cumque culpa explicabo aliquid at
perspiciatis est et voluptatem dignissimos dolor itaque sit nam

laboriosam dolor voluptates
doloremque ex facilis sit sint culpa
soluta assumenda eligendi non ut eius
sequi ducimus vel quasi
veritatis est dolores

temporibus sit alias delectus eligendi possimus magni
quo deleniti praesentium dicta non quod
aut est molestias
molestias et officia quis nihil
itaque dolorem quia

at nam consequatur ea labore ea harum
cupiditate quo est a modi nesciunt soluta
ipsa voluptas error itaque dicta in
autem qui minus magnam et distinctio eum
accusamus ratione error aut

Flutter Post
Ini contoh POST.

Data berhasil ditambahkan!
```



```
REST API - praktikum 14

quaerat velit veniam amet cupiditate aut numquam ut sequi
in non odio excepturi sint eum
labore voluptates vitae quia qui et
inventore itaque rerum
veniam non exercitationem delectus aut

quas fugiat ut perspiciatis vero provident
eum non blanditiis soluta porro quibusdam voluptas
vel voluptatem qui placeat dolores qui velit aut
vel inventore aut cumque culpa explicabo aliquid at
perspiciatis est et voluptatem dignissimos dolor itaque sit nam

laboriosam dolor voluptates
doloremque ex facilis sit sint culpa
soluta assumenda eligendi non ut eius
sequi ducimus vel quasi
veritatis est dolores

temporibus sit alias delectus eligendi possimus magni
quo deleniti praesentium dicta non quod
aut est molestias
molestias et officia quis nihil
itaque dolorem quia

at nam consequatur ea labore ea harum
cupiditate quo est a modi nesciunt soluta
ipsa voluptas error itaque dicta in
autem qui minus magnam et distinctio eum
accusamus ratione error aut

Data berhasil dihapus!
```



```
REST API - praktikum 14

sunt aut facere repellat provident occaecati excepturi optio reprehenderit
quia et suscipit
suscipit recusandae consequuntur expedita et cum
reprehenderit molestiae ut ut quas totam
nostrum rerum est autem sunt rem eveniet architecto

qui est esse
est rerum tempore vitae
sequi sint nihil reprehenderit dolor beatae ea dolores neque
fugiat blanditiis voluptate porro vel nihil molestiae ut reiciendis
qui aperiam non debitis possimus qui neque nisi nulla

ea molestias quasi exercitationem repellat qui ipsa sit aut
et iusto sed quo iure
voluptatem occaecati omnis eligendi aut ad
voluptatem doloribus vel accusantium quis pariatur
molestiae porro eius odio et labore et velit aut

eum et est occaecati
ullam et saepe reiciendis voluptatem adipisci
sit amet autem assumenda provident rerum culpa
quis hic commodi nesciunt rem tenetur doloremque ipsam iure
quis sunt voluptatem rerum illo velit

nesciunt quas odio
repudiandae veniam quaerat sunt sed
alias aut fugiat sit autem sed est
voluptatem omnis possimus esse voluptatibus quis
est aut tenetur dolor neque

dolorem eum magni eos aperiam quia

Data berhasil diambil!
```



REST API - praktikum 14

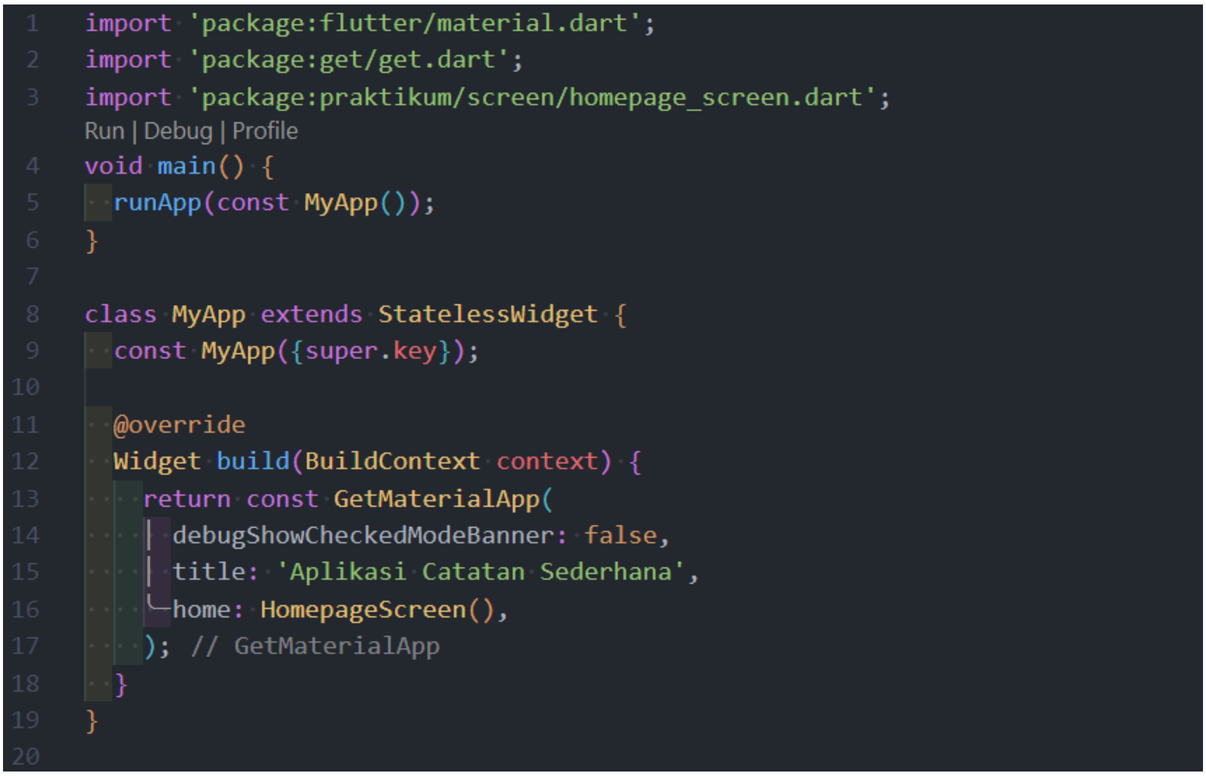
Tekan tombol GET untuk mengambil data

Deskripsi Program

Kode pada ApiService, bertugas menangani komunikasi dengan API, dan kode di HomepageScreen merupakan antarmuka pengguna (UI) untuk berinteraksi dengan data dari API. Di dalam ApiService, terdapat fungsi untuk mengambil data (GET), menambah data (POST), memperbarui data (PUT), dan menghapus data (DELETE) dari sebuah REST API menggunakan paket http. Data yang diterima dari API disimpan di dalam sebuah variabel posts sebagai list. Pada sisi UI, HomepageScreen menampilkan data yang disimpan dalam posts menggunakan widget seperti ListView untuk menampilkan daftar, dan ElevatedButton untuk memicu operasi API. Ketika tombol ditekan, fungsi _handleApiOperation dijalankan untuk memanggil metode API terkait, menangani proses loading, dan menampilkan pesan keberhasilan atau kesalahan menggunakan SnackBar. Dengan pendekatan ini, kode memastikan data dari API dapat diambil, diperbarui, atau dihapus secara efisien, sekaligus menyinkronkan tampilan dengan data terbaru.

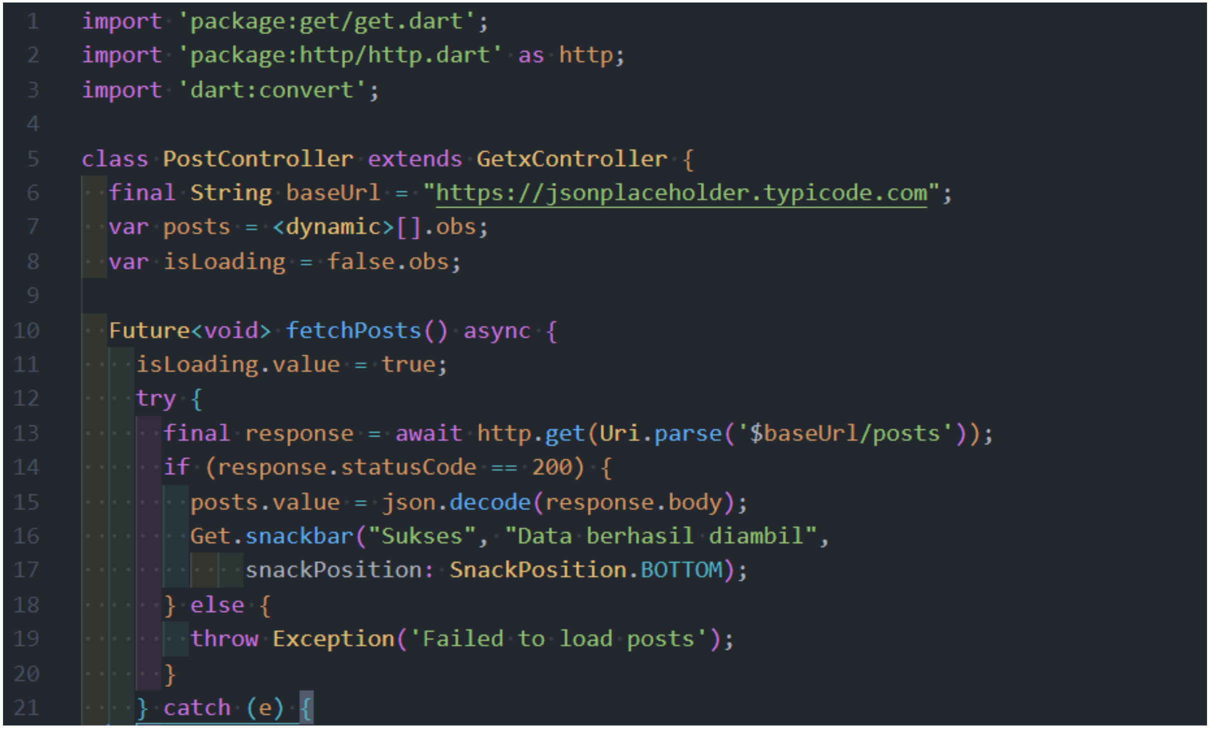
B. Unguided

- **Main.dart**

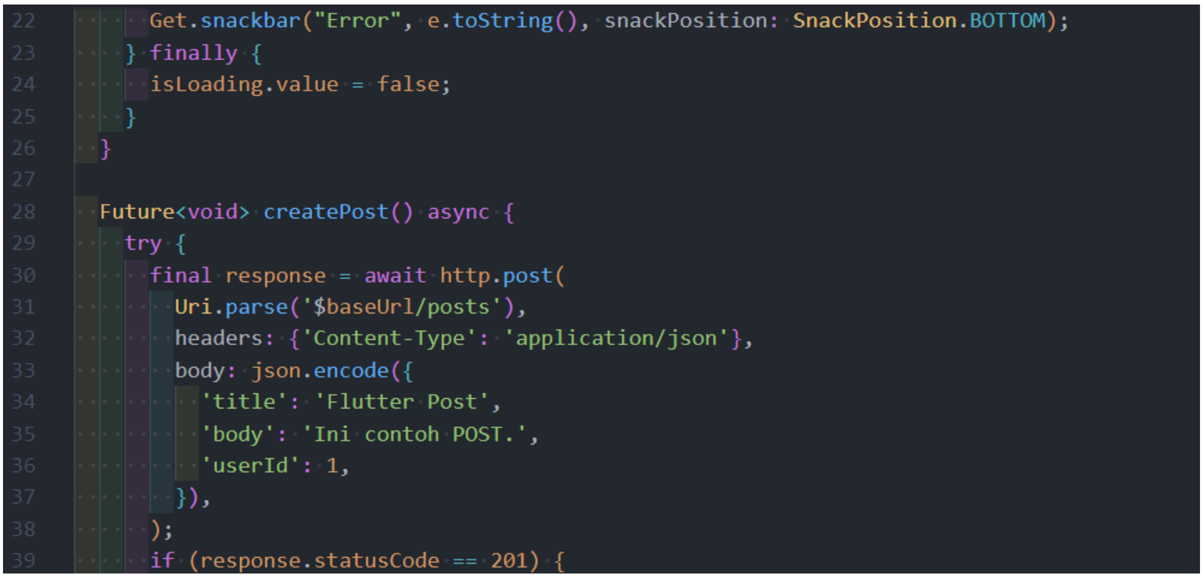


```dart
import 'package:flutter/material.dart';
import 'package:get/get.dart';
import 'package:praktikum/screen/homepage_screen.dart';
void main() {
  runApp(const MyApp());
}

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

  @override
  Widget build(BuildContext context) {
    return const GetMaterialApp(
      debugShowCheckedModeBanner: false,
      title: 'Aplikasi Catatan Sederhana',
      home: HomepageScreen(),
    ); // GetMaterialApp
  }
}

```

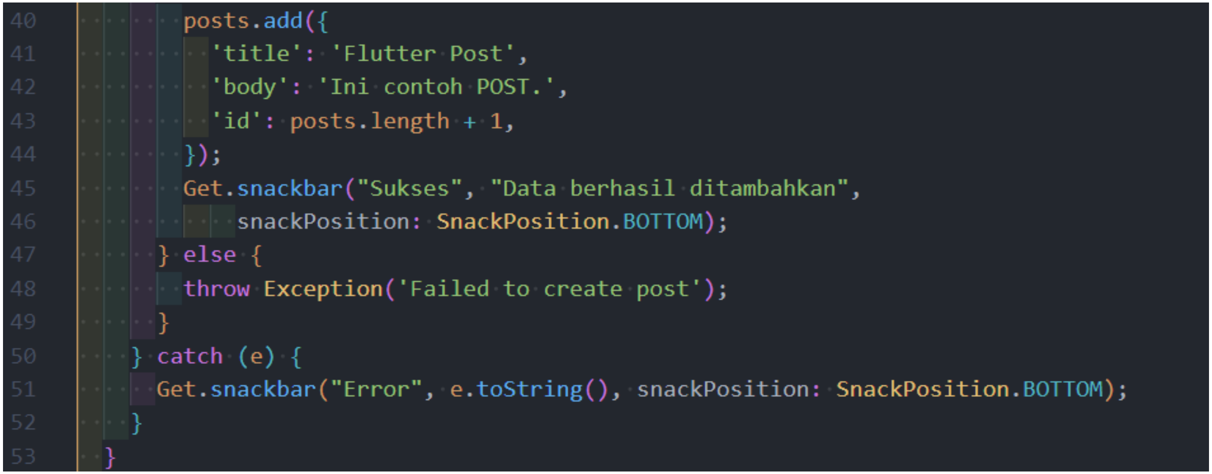
- **Api_controller.dart**



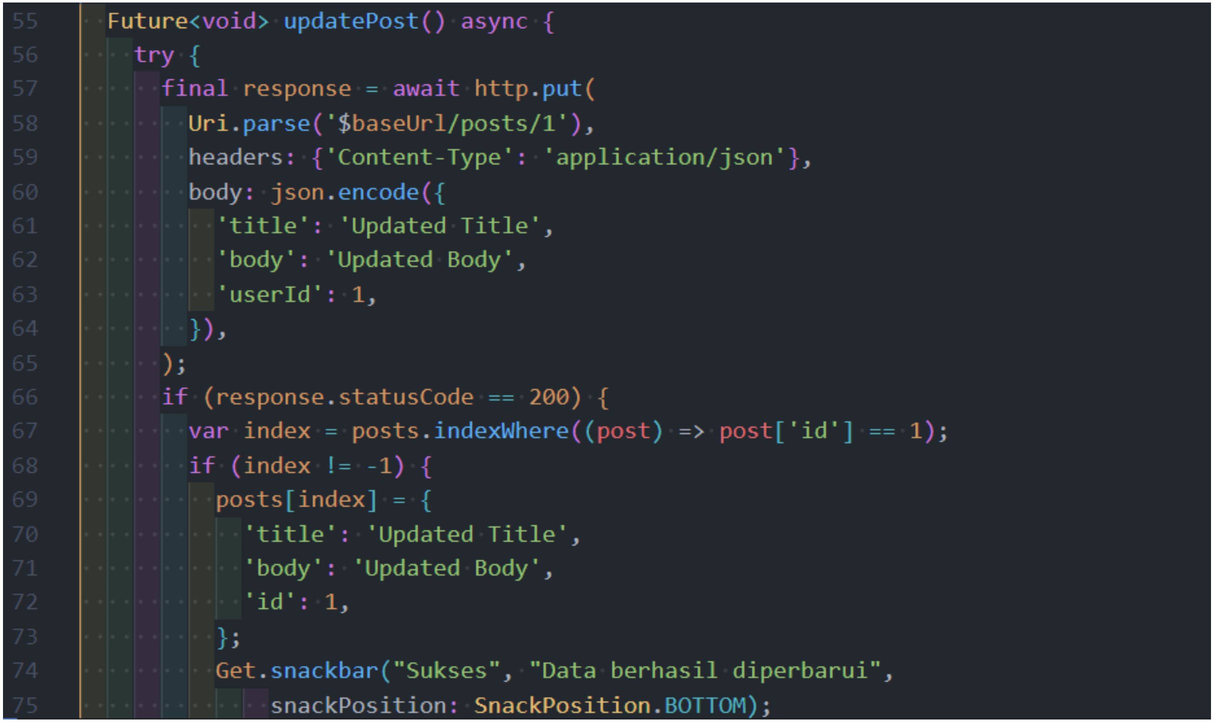
```dart
import 'package:get/get.dart';
import 'package:http/http.dart' as http;
import 'dart:convert';

class PostController extends GetxController {
  final String baseUrl = "https://jsonplaceholder.typicode.com";
  var posts = <dynamic>[].obs;
  var isLoading = false.obs;

  Future<void> fetchPosts() async {
    isLoading.value = true;
    try {
      final response = await http.get(Uri.parse('$baseUrl/posts'));
      if (response.statusCode == 200) {
        posts.value = json.decode(response.body);
        Get.snackbar("Sukses", "Data berhasil diambil",
            snackPosition: SnackPosition.BOTTOM);
      } else {
        throw Exception('Failed to load posts');
      }
    } catch (e) {
```



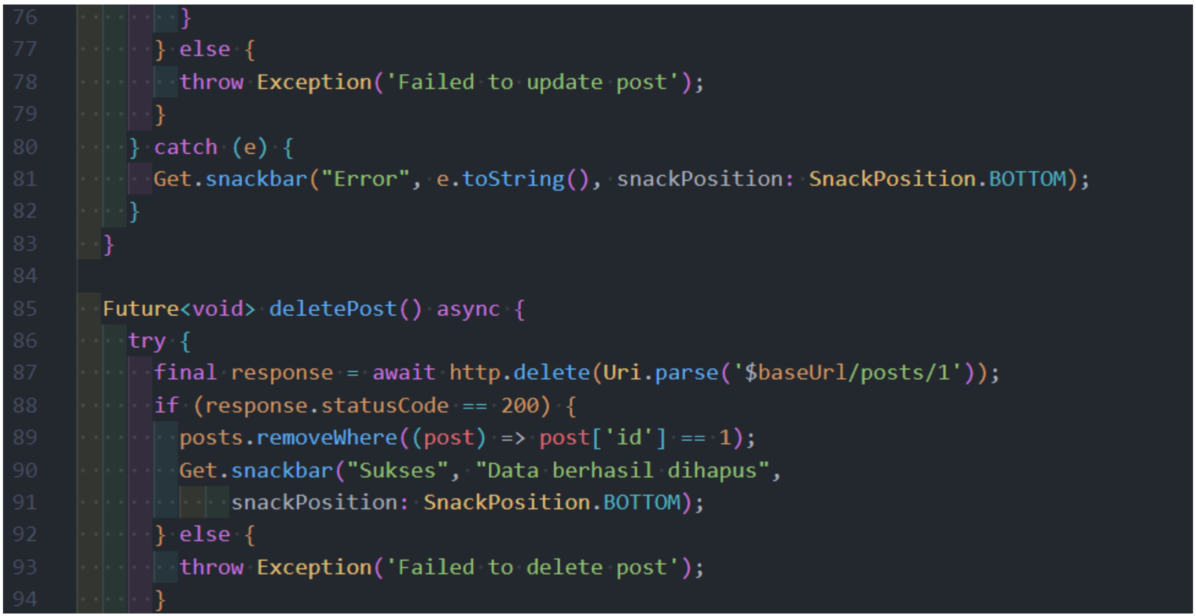
```dart
      Get.snackbar("Error", e.toString(), snackPosition: SnackPosition.BOTTOM);
    } finally {
      isLoading.value = false;
    }
  }

  Future<void> createPost() async {
    try {
      final response = await http.post(
        Uri.parse('$baseUrl/posts'),
        headers: {'Content-Type': 'application/json'},
        body: json.encode({
          'title': 'Flutter Post',
          'body': 'Ini contoh POST.',
          'userId': 1,
        }),
      );
      if (response.statusCode == 201) {
```



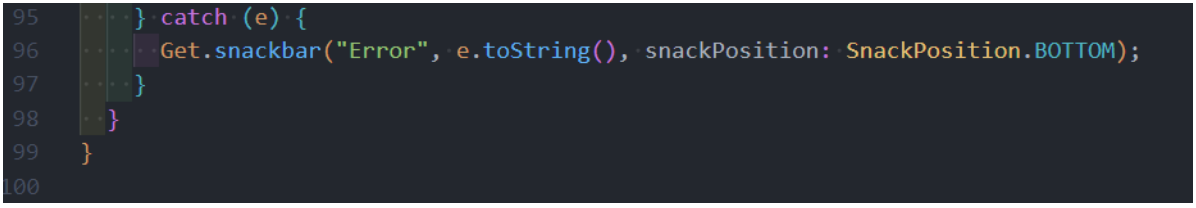
```dart
        posts.add({
          'title': 'Flutter Post',
          'body': 'Ini contoh POST.',
          'id': posts.length + 1,
        });
        Get.snackbar("Sukses", "Data berhasil ditambahkan",
            snackPosition: SnackPosition.BOTTOM);
      } else {
        throw Exception('Failed to create post');
      }
    } catch (e) {
      Get.snackbar("Error", e.toString(), snackPosition: SnackPosition.BOTTOM);
    }
  }
```



```dart
  Future<void> updatePost() async {
    try {
      final response = await http.put(
        Uri.parse('$baseUrl/posts/1'),
        headers: {'Content-Type': 'application/json'},
        body: json.encode({
          'title': 'Updated Title',
          'body': 'Updated Body',
          'userId': 1,
        }),
      );
      if (response.statusCode == 200) {
        var index = posts.indexWhere((post) => post['id'] == 1);
        if (index != -1) {
          posts[index] = {
            'title': 'Updated Title',
            'body': 'Updated Body',
            'id': 1,
          };
          Get.snackbar("Sukses", "Data berhasil diperbarui",
              snackPosition: SnackPosition.BOTTOM);
```

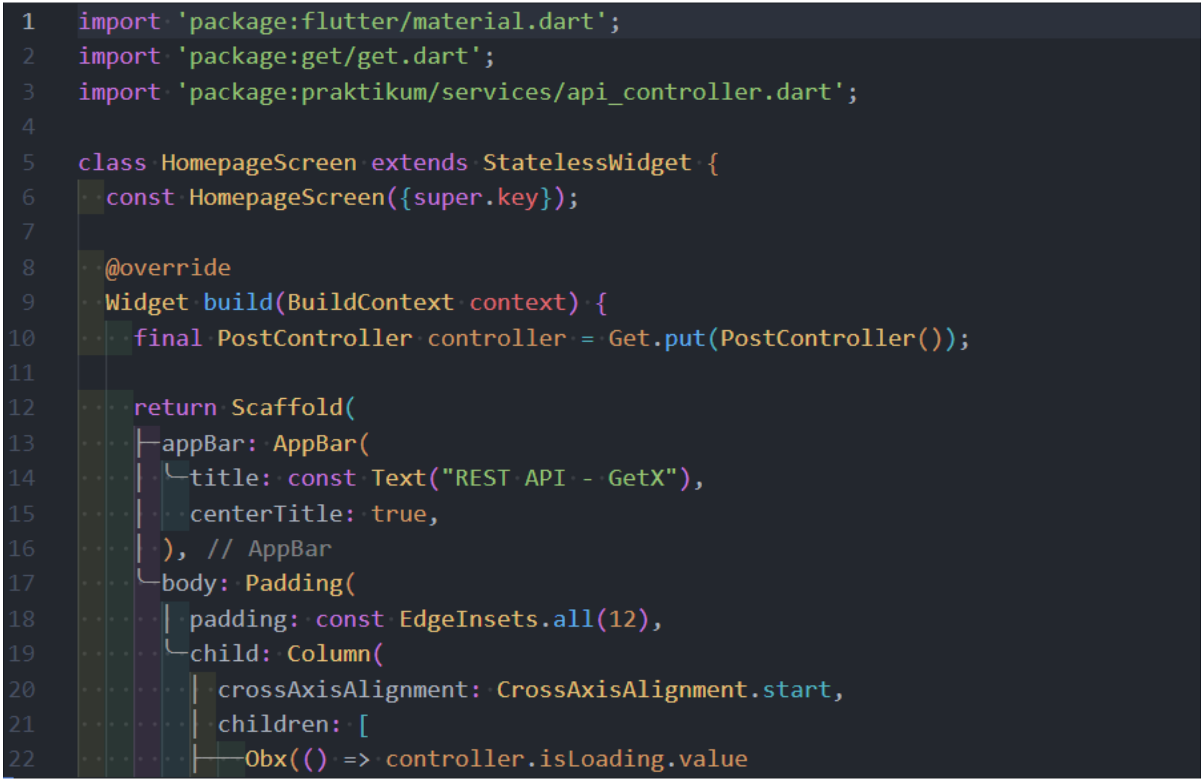


```dart
        }
      } else {
        throw Exception('Failed to update post');
      }
    } catch (e) {
      Get.snackbar("Error", e.toString(), snackPosition: SnackPosition.BOTTOM);
    }
  }

  Future<void> deletePost() async {
    try {
      final response = await http.delete(Uri.parse('$baseUrl/posts/1'));
      if (response.statusCode == 200) {
        posts.removeWhere((post) => post['id'] == 1);
        Get.snackbar("Sukses", "Data berhasil dihapus",
            snackPosition: SnackPosition.BOTTOM);
      } else {
        throw Exception('Failed to delete post');
      }
```

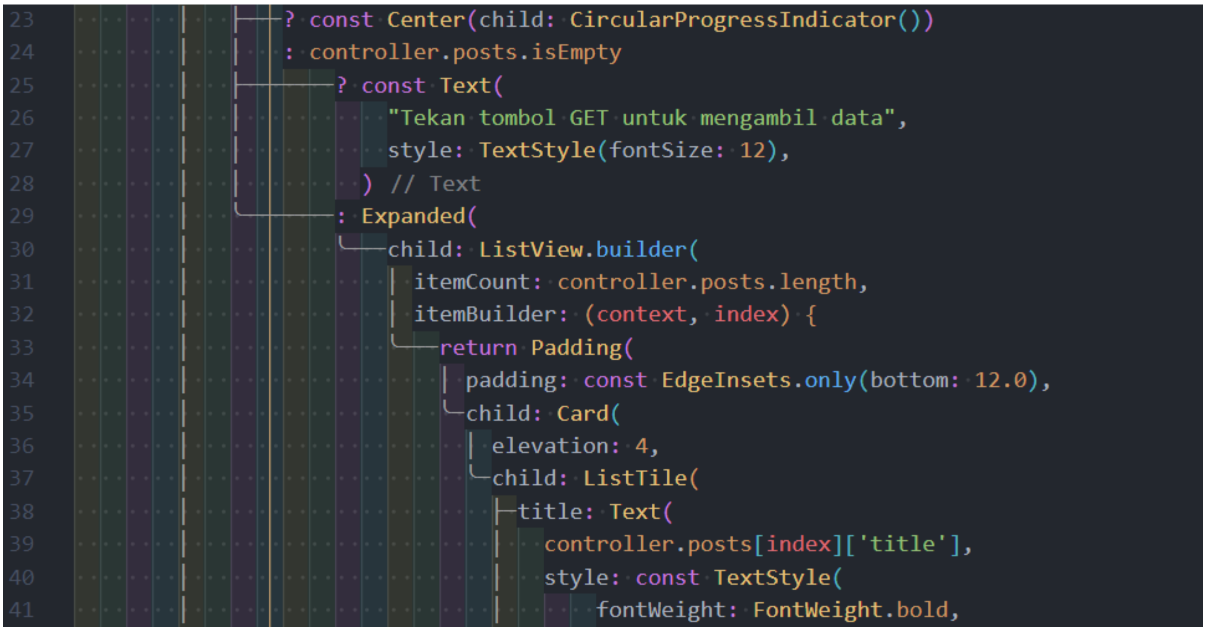


```dart
    } catch (e) {
      Get.snackbar("Error", e.toString(), snackPosition: SnackPosition.BOTTOM);
    }
  }
}
```

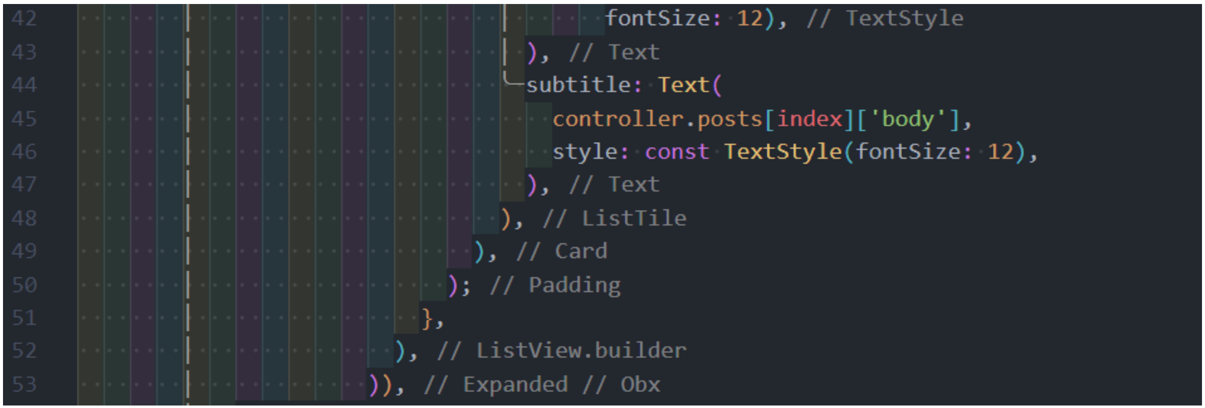
- **Homepage.dart**



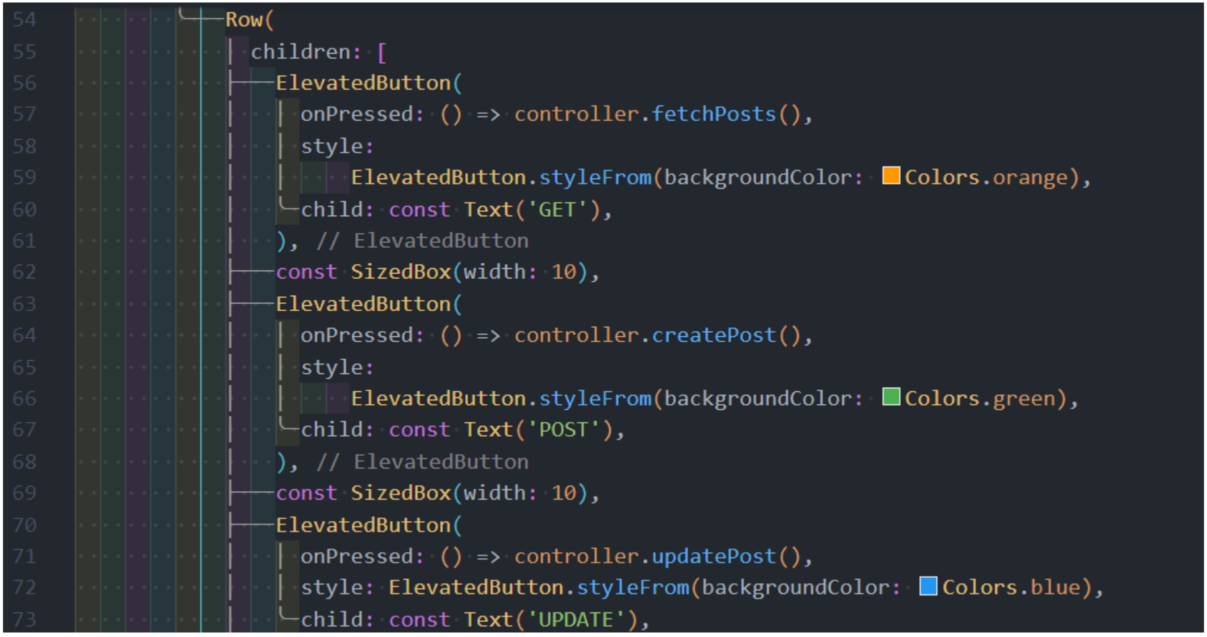
```dart
import 'package:flutter/material.dart';
import 'package:get/get.dart';
import 'package:praktikum/services/api_controller.dart';

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

  @override
  Widget build(BuildContext context) {
    final PostController controller = Get.put(PostController());

    return Scaffold(
      appBar: AppBar(
        title: const Text("REST API - GetX"),
        centerTitle: true,
      ), // AppBar
      body: Padding(
        padding: const EdgeInsets.all(12),
        child: Column(
          crossAxisAlignment: CrossAxisAlignment.start,
          children: [
            Obx(() => controller.isLoading.value
```



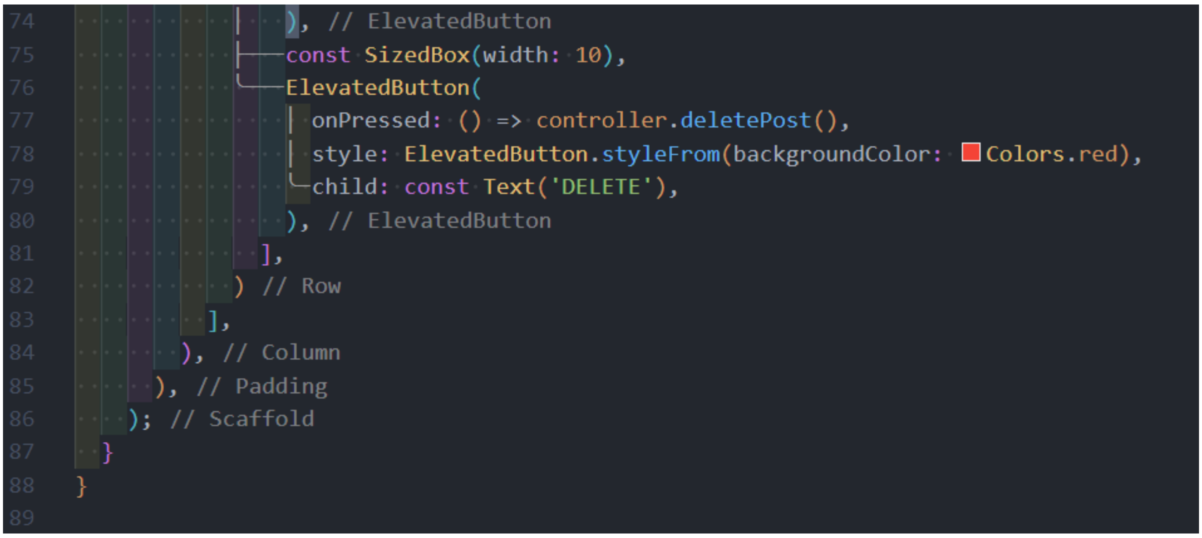
```dart
        ? const Center(child: CircularProgressIndicator())
        : controller.posts.isEmpty
          ? const Text(
            "Tekan tombol GET untuk mengambil data",
            style: TextStyle(fontSize: 12),
          ) // Text
          : Expanded(
            child: ListView.builder(
             itemCount: controller.posts.length,
             itemBuilder: (context, index) {
              return Padding(
               padding: const EdgeInsets.only(bottom: 12.0),
               child: Card(
                elevation: 4,
                child: ListTile(
                 title: Text(
                  controller.posts[index]['title'],
                  style: const TextStyle(
                    fontWeight: FontWeight.bold,
```



```dart
                    fontSize: 12), // TextStyle
                 ), // Text
                 subtitle: Text(
                  controller.posts[index]['body'],
                  style: const TextStyle(fontSize: 12),
                 ), // Text
                ), // ListTile
               ), // Card
              ); // Padding
             },
            ), // ListView.builder
           )), // Expanded // Obx
```

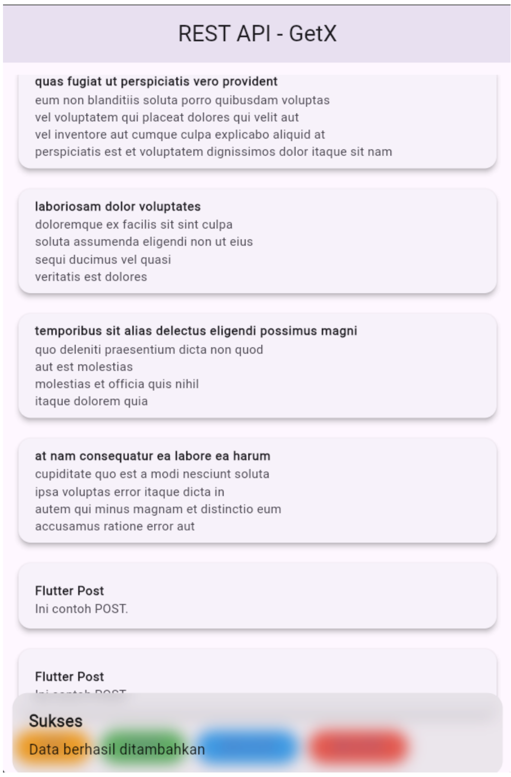


```dart
      Row(
       children: [
        ElevatedButton(
         onPressed: () => controller.fetchPosts(),
         style:
           ElevatedButton.styleFrom(backgroundColor: 🟧Colors.orange),
         child: const Text('GET'),
        ), // ElevatedButton
        const SizedBox(width: 10),
        ElevatedButton(
         onPressed: () => controller.createPost(),
         style:
           ElevatedButton.styleFrom(backgroundColor: 🟩Colors.green),
         child: const Text('POST'),
        ), // ElevatedButton
        const SizedBox(width: 10),
        ElevatedButton(
         onPressed: () => controller.updatePost(),
         style: ElevatedButton.styleFrom(backgroundColor: 🟦Colors.blue),
         child: const Text('UPDATE'),
```



```
                ), // ElevatedButton
                const SizedBox(width: 10),
                ElevatedButton(
                  onPressed: () => controller.deletePost(),
                  style: ElevatedButton.styleFrom(backgroundColor: Colors.red),
                  child: const Text('DELETE'),
                ), // ElevatedButton
              ],
            ) // Row
          ],
        ), // Column
      ), // Padding
    ); // Scaffold
  }
}

```

Output



REST API - GetX

quas fugiat ut perspiciatis vero provident
eum non blanditiis soluta porro quibusdam voluptas
vel voluptatem qui placeat dolores qui velit aut
vel inventore aut cumque culpa explicabo aliquid at
perspiciatis est et voluptatem dignissimos dolor itaque sit nam

laboriosam dolor voluptates
doloremque ex facilis sit sint culpa
soluta assumenda eligendi non ut eius
sequi ducimus vel quasi
veritatis est dolores

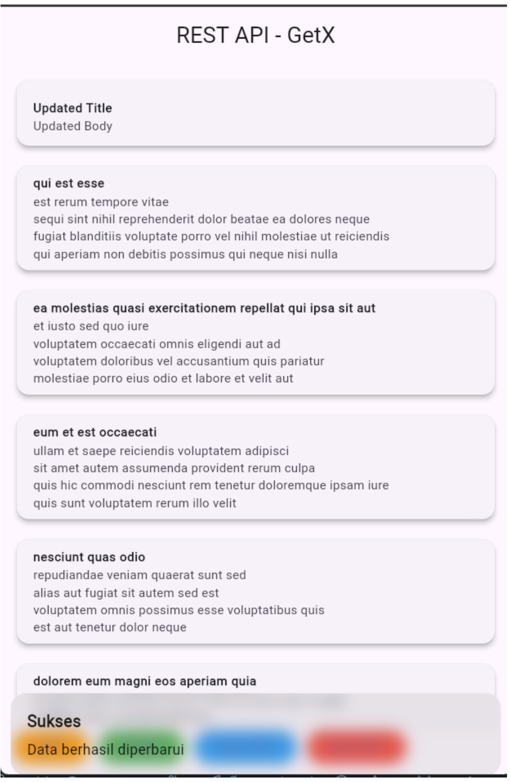
temporibus sit alias delectus eligendi possimus magni
quo deleniti praesentium dicta non quod
aut est molestias
molestias et officia quis nihil
itaque dolorem quia

at nam consequatur ea labore ea harum
cupiditate quo est a modi nesciunt soluta
ipsa voluptas error itaque dicta in
autem qui minus magnam et distinctio eum
accusamus ratione error aut

Flutter Post
Ini contoh POST.

Flutter Post
Ini contoh POST

Sukses
Data berhasil ditambahkan



REST API - GetX

Updated Title
Updated Body

qui est esse
est rerum tempore vitae
sequi sint nihil reprehenderit dolor beatae ea dolores neque
fugiat blanditiis voluptate porro vel nihil molestiae ut reiciendis
qui aperiam non debitis possimus qui neque nisi nulla

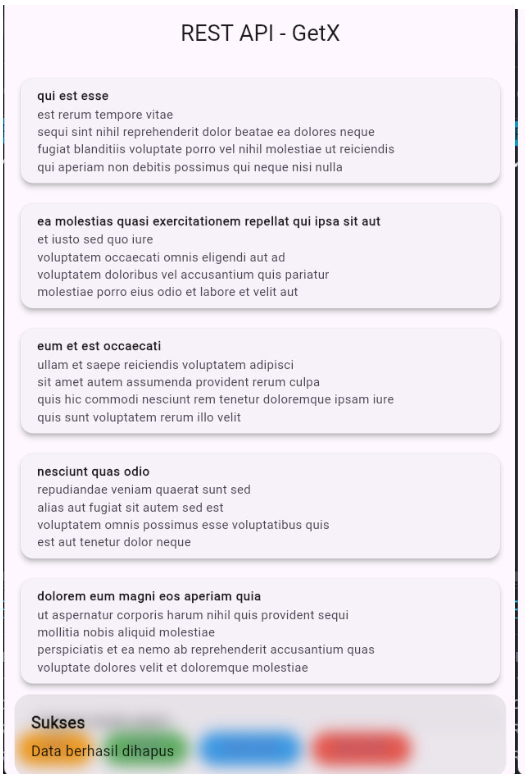
ea molestias quasi exercitationem repellat qui ipsa sit aut
et iusto sed quo iure
voluptatem occaecati omnis eligendi aut ad
voluptatem doloribus vel accusantium quis pariatur
molestiae porro eius odio et labore et velit aut

eum et est occaecati
ullam et saepe reiciendis voluptatem adipisci
sit amet autem assumenda provident rerum culpa
quis hic commodi nesciunt rem tenetur doloremque ipsam iure
quis sunt voluptatem rerum illo velit

nesciunt quas odio
repudiandae veniam quaerat sunt sed
alias aut fugiat sit autem sed est
voluptatem omnis possimus esse voluptatibus quis
est aut tenetur dolor neque

dolorem eum magni eos aperiam quia

Sukses
Data berhasil diperbarui



REST API - GetX

qui est esse
est rerum tempore vitae
sequi sint nihil reprehenderit dolor beatae ea dolores neque
fugiat blanditiis voluptate porro vel nihil molestiae ut reiciendis
qui aperiam non debitis possimus qui neque nisi nulla

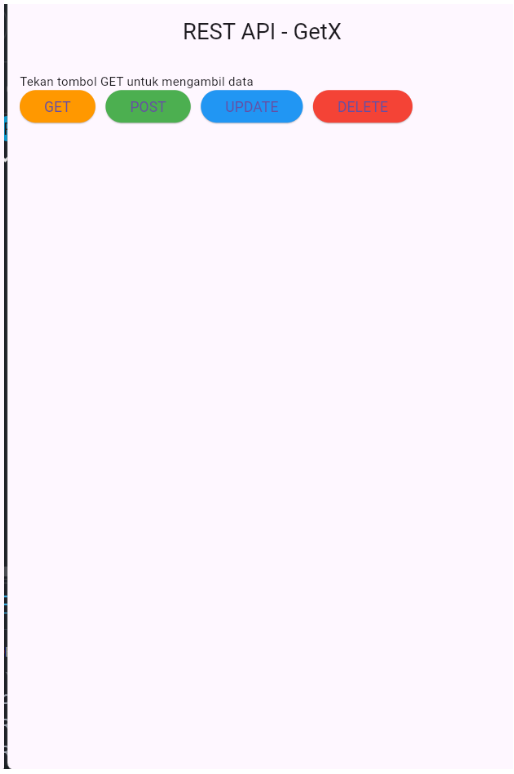
ea molestias quasi exercitationem repellat qui ipsa sit aut
et iusto sed quo iure
voluptatem occaecati omnis eligendi aut ad
voluptatem doloribus vel accusantium quis pariatur
molestiae porro eius odio et labore et velit aut

eum et est occaecati
ullam et saepe reiciendis voluptatem adipisci
sit amet autem assumenda provident rerum culpa
quis hic commodi nesciunt rem tenetur doloremque ipsam iure
quis sunt voluptatem rerum illo velit

nesciunt quas odio
repudiandae veniam quaerat sunt sed
alias aut fugiat sit autem sed est
voluptatem omnis possimus esse voluptatibus quis
est aut tenetur dolor neque

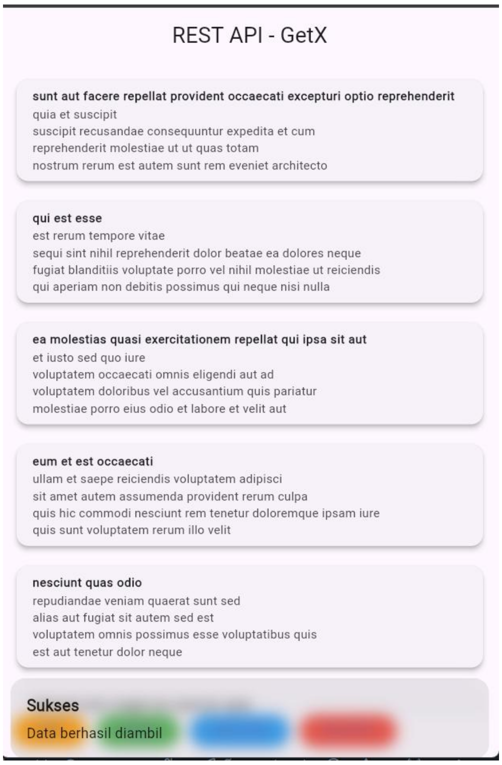
dolorem eum magni eos aperiam quia
ut aspernatur corporis harum nihil quis provident sequi
mollitia nobis aliquid molestiae
perspiciatis et ea nemo ab reprehenderit accusantium quas
voluptate dolores velit et doloremque molestiae

Sukses
Data berhasil dihapus



REST API - GetX

Tekan tombol GET untuk mengambil data



REST API - GetX

sunt aut facere repellat provident occaecati excepturi optio reprehenderit
quia et suscipit
suscipit recusandae consequuntur expedita et cum
reprehenderit molestiae ut ut quas totam
nostrum rerum est autem sunt rem eveniet architecto

qui est esse
est rerum tempore vitae
sequi sint nihil reprehenderit dolor beatae ea dolores neque
fugiat blanditiis voluptate porro vel nihil molestiae ut reiciendis
qui aperiam non debitis possimus qui neque nisi nulla

ea molestias quasi exercitationem repellat qui ipsa sit aut
et iusto sed quo iure
voluptatem occaecati omnis eligendi aut ad
voluptatem doloribus vel accusantium quis pariatur
molestiae porro eius odio et labore et velit aut

eum et est occaecati
ullam et saepe reiciendis voluptatem adipisci
sit amet autem assumenda provident rerum culpa
quis hic commodi nesciunt rem tenetur doloremque ipsam iure
quis sunt voluptatem rerum illo velit

nesciunt quas odio
repudiandae veniam quaerat sunt sed
alias aut fugiat sit autem sed est
voluptatem omnis possimus esse voluptatibus quis
est aut tenetur dolor neque

Sukses
Data berhasil diambil

Deskripsi Program

Pada kode unguided ini kode diimprovisasi menggunakan state management GetX untuk mengelola data dan state dengan lebih efisien saat berinteraksi dengan API. PostController bertanggung jawab atas operasi API seperti GET, POST, UPDATE, dan DELETE. Data yang diterima dari API disimpan dalam variabel posts yang dikelola dengan RxList, sehingga perubahan data secara otomatis memperbarui UI tanpa memerlukan pengelolaan state manual. Pada UI, widget Obx memastikan tampilan ter-update saat ada perubahan state seperti loading atau perubahan data. Saat tombol aksi ditekan, fungsi di dalam PostController dipanggil untuk memodifikasi data, dan Get.snackbar digunakan untuk menampilkan notifikasi interaktif yang memberikan umpan balik tentang hasil operasi kepada pengguna. Pendekatan ini membuat pengelolaan data dan sinkronisasi UI lebih sederhana, responsif, dan terstruktur.