TUGAS PENDAHULUAN
PEMROGRAMAN PERANGKAT BERGERAK

MODUL XIII

NETWORKING

Disusun Oleh :

Dhiya Ulhaq Ramadhan / 2211104053
SE-06-02

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

TUGAS PENDAHULUAN

SOAL

1. Apa yang dimaksud dengan state management pada Flutter?

   Jawaban :

   Cara untuk mengelola dan mempertahankan data/state dalam aplikasi Flutter. State management diperlukan untuk mengatur bagaimana data mengalir dan berubah di berbagai bagian aplikasi, serta bagaimana UI bereaksi terhadap perubahan data tersebut. Ketika data berubah, state management memastikan bahwa UI diperbarui secara efisien untuk mencerminkan perubahan tersebut.

2. Sebut dan jelaskan komponen-komponen yang ada di dalam GetX.

   Jawaban :

   a. State Management

   GetX menyediakan sistem reactive state management melalui Rx Variables (.obs) dan GetxController. Controller berfungsi sebagai tempat untuk logika bisnis dan state management, sementara Rx Variables secara otomatis memicu pembaruan UI ketika nilainya berubah.

   b. Route Management

   GetX memiliki sistem navigasi yang powerful dengan Get.to(), Get.off(), Get.offAll() untuk mengelola perpindahan antar halaman. Sistem ini lebih sederhana dari Navigator bawaan Flutter dan mendukung navigasi dengan nama route.

   c. Dependency Injection

   Melalui Get.put(), Get.lazyPut(), dan Get.find(), GetX menyediakan sistem dependency injection yang memungkinkan sharing instance antar widget dan memori yang efisien.

3. Lengkapilah code di bawah ini, dan tampilkan hasil outputnya serta jelaskan.

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

/// Controller untuk mengelola state counter
class CounterController extends GetxController {
  // TODO: Tambahkan variabel untuk menyimpan nilai counter

  // TODO: Buat fungsi untuk menambah nilai counter

  // TODO: Buat fungsi untuk mereset nilai counter
}

class HomePage extends StatelessWidget {
  final CounterController controller =
Get.put(CounterController());

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(title: Text("Counter App")),
      body: Center(
        child: Obx(() {
          // TODO: Lengkapi logika untuk menampilkan nilai
counter
          return Text(
            "0", // Ganti ini dengan nilai counter
            style: TextStyle(fontSize: 48),
          );
        }),
      ),
      floatingActionButton: Column(
```

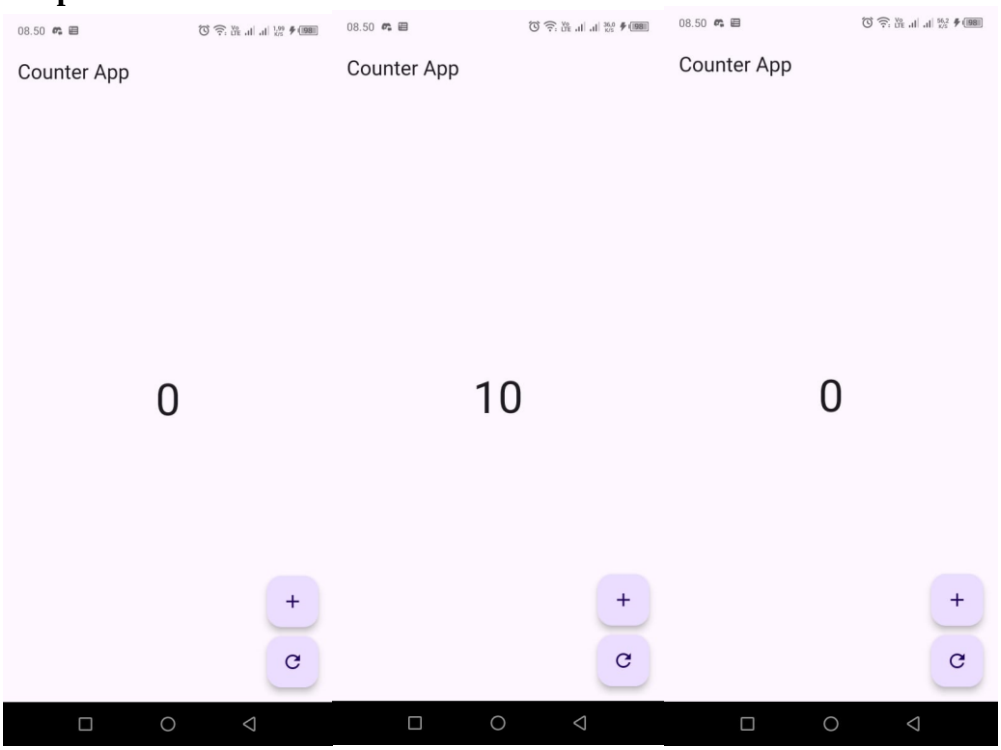
```
        mainAxisAlignment: MainAxisAlignment.end,
        children: [
          FloatingActionButton(
            onPressed: () {
              // TODO: Tambahkan logika untuk menambah nilai
counter
            },
            child: Icon(Icons.add),
          ),
          SizedBox(height: 10),
          FloatingActionButton(
            onPressed: () {
              // TODO: Tambahkan logika untuk mereset nilai
counter
            },
            child: Icon(Icons.refresh),
          ),
        ],
      ),
    );
  }
}

void main() {
  runApp(MaterialApp(
    debugShowCheckedModeBanner: false,
    home: HomePage(),
  ));
}
```

Jawaban :

lib > main.dart > CounterController

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

// Controller untuk mengelola state counter
class CounterController extends GetxController {
  // Membuat variable observable untuk nilai counter
  var count = 0.obs;

  // Fungsi untuk menambah nilai counter
  void increment() {
    count.value++;
  }

  // Fungsi untuk mereset nilai counter ke 0
  void reset() {
    count.value = 0;
  }
}

class HomePage extends StatelessWidget {
  // Inisialisasi dan inject controller
  final CounterController controller = Get.put(CounterController());

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: Text("Counter App"),
      ), // AppBar
      body: Center(
        child: Obx(() {
          // Menampilkan nilai counter yang reactive
          return Text(
            "${controller.count.value}",
            style: TextStyle(fontSize: 48),
          ); // Text
        }), // Obx
      ), // Center
      floatingActionButton: Column(
        mainAxisAlignment: MainAxisAlignment.end,
        children: [
          FloatingActionButton(
            onPressed: () {
              // Memanggil fungsi increment saat tombol + ditekan
              controller.increment();
            },
            child: Icon(Icons.add),
          ), // FloatingActionButton
          SizedBox(height: 10),
          FloatingActionButton(
            onPressed: () {
              // Memanggil fungsi reset saat tombol refresh ditekan
              controller.reset();
            },
            child: Icon(Icons.refresh),
          ), // FloatingActionButton
        ],
      ), // Column
    ); // Scaffold
  }
}

Run | Debug | Profile
void main() {
  runApp(MaterialApp(
    debugShowCheckedModeBanner: false,
    home: HomePage(),
  )); // MaterialApp
}
```

Output



Deskripsi Program

Program ini menggunakan GetX untuk state management dengan cara:

1. CounterController menyimpan state count sebagai observable (.obs) sehingga UI akan otomatis diperbarui ketika nilainya berubah
2. Controller di-inject ke dalam HomePage menggunakan Get.put()
3. Widget yang menampilkan nilai counter dibungkus dengan Obx() agar reactive terhadap perubahan nilai
4. Setiap kali tombol + ditekan, fungsi increment() dipanggil dan UI otomatis diperbarui
5. Setiap kali tombol refresh ditekan, fungsi reset() dipanggil dan counter kembali ke 0