TUGAS PENDAHULUAN
PEMROGRAMAN PERANGKAT BERGERAK

MODUL XIII
NETWORKING

Disusun Oleh :

Aorinka Anendya Chazanah / 2211104013

S1 SE 06-01

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

SOAL

1. **Apa yang dimaksud dengan state management pada Flutter?**

   State management di Flutter merupakan pengelolaan perubahan data atau status aplikasi dan bagaimana data tersebut disinkronkan dengan UI. State management membantu memisahkan logika bisnis dan UI sehingga memudahkan pengelolaan data, terutama saat aplikasi berkembang dan memerlukan pembaruan tampilan berdasarkan status tertentu. Beberapa pendekatan state management di Flutter termasuk menggunakan setState, Provider, Riverpod, BLoC, dan GetX.

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

   GetX adalah framework state management yang dikenal karena performanya yang tinggi, sistem dependency injection yang efisien, dan manajemen route yang cepat dan praktis. Berikut adalah beberapa komponen utama dalam GetX:

   a. **Pengelolaan State**

      GetX menyediakan dua pendekatan untuk mengelola state: **Reactive State** dan **Simple State**.

      - **Reactive State Management** menggunakan kelas Rx untuk menjadikan data reaktif, yang berarti UI akan otomatis diperbarui setiap kali data berubah. Komponen utama dalam pendekatan ini adalah:
        1. Rx<Type>: Digunakan untuk menjadikan data reaktif, seperti RxInt atau RxString.
        2. .obs: Merupakan cara singkat untuk membuat data reaktif.
      - **Simple State Management** menggunakan widget seperti GetBuilder atau GetX untuk memperbarui UI secara manual tanpa membuat data reaktif. Komponen utama di sini adalah GetBuilder: Digunakan untuk memperbarui widget hanya ketika ada perubahan pada state tertentu.

   b. **Manajemen Navigasi (Route Management)**

      GetX memungkinkan navigasi antar halaman tanpa perlu menggunakan Navigator bawaan Flutter. Beberapa komponen utama yang digunakan dalam manajemen route adalah:

      - Get.to(): Untuk berpindah ke halaman baru.
      - Get.off(): Untuk berpindah ke halaman baru dan menutup halaman sebelumnya.
      - Get.back(): Untuk kembali ke halaman sebelumnya.
      - Get.arguments: Digunakan untuk mengirimkan data antar halaman.

   c. **Dependency Injection (DI)**

      GetX menyediakan sistem dependency injection built-in untuk mengelola lifecycle objek dan membuatnya tersedia di seluruh aplikasi. Komponen utamanya meliputi:

- Get.put(): Untuk menyimpan instance controller yang bisa diakses secara global.
- Get.lazyPut(): Untuk menyimpan instance dan hanya membuatnya ketika diperlukan.
- Get.find(): Untuk mengambil instance yang telah disimpan.

d. Middleware

Middleware di GetX digunakan untuk mengontrol akses ke halaman tertentu, seperti memeriksa status login pengguna sebelum memasuki halaman tertentu. Komponen utamanya adalah:

- GetMiddleware: Kelas yang digunakan untuk membuat middleware.
- redirect: Fungsi untuk mengarahkan pengguna ke halaman lain jika syarat tertentu tidak terpenuhi.

e. Snackbar, Dialog, dan BottomSheet

GetX menyediakan utilitas built-in untuk menampilkan notifikasi, dialog, atau bottom sheet tanpa memerlukan BuildContext. Komponen utama yang digunakan adalah:

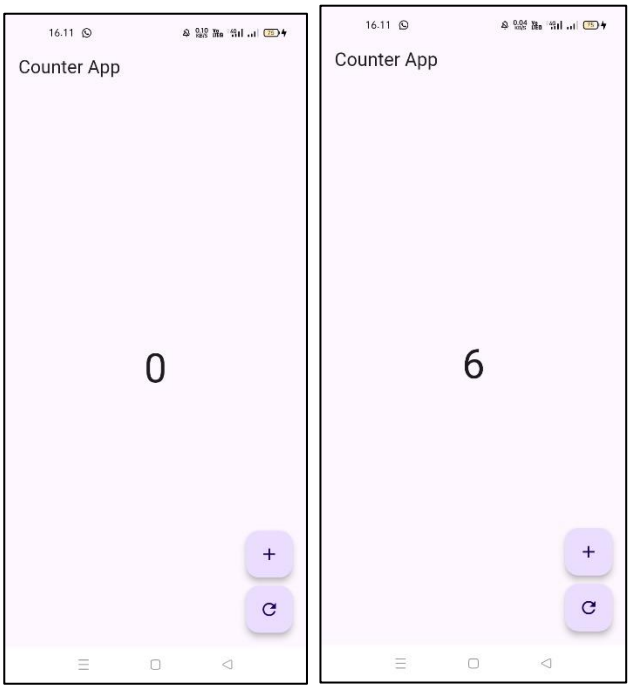
- Get.snackbar: Untuk menampilkan snackbar.
- Get.defaultDialog: Untuk menampilkan dialog.
- Get.bottomSheet: Untuk menampilkan bottom sheet.

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

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

/// Controller untuk mengelola state counter
class CounterController extends GetxController {
  // Variabel untuk menyimpan nilai counter
  var counter = 0.obs; // Menggunakan .obs untuk membuat variabel
reaktif

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

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

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(() {
          // Menampilkan nilai counter yang sudah diperbarui
          return Text(
            "${controller.counter.value}", // Menggunakan controller
untuk mendapatkan nilai counter
            style: TextStyle(fontSize: 48),
          );
        }),
      ),
      floatingActionButton: Column(
        mainAxisAlignment: MainAxisAlignment.end,
        children: [
          FloatingActionButton(
            onPressed: () {
              // Menambahkan nilai counter
              controller.increment();
            },
            child: Icon(Icons.add),
          ),
          SizedBox(height: 10),
          FloatingActionButton(
            onPressed: () {
              // Mereset nilai counter
              controller.reset();
            },
            child: Icon(Icons.refresh),
          ),
        ],
      ),
    );
  }
}

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

Hasil Output



Deskripsi Program

Aplikasi ini memiliki dua fungsi utama: menambah dan mereset nilai counter. Di dalamnya, terdapat dua bagian penting: **Controller** dan **UI**.

Pada bagian pertama, CounterController adalah kelas yang bertanggung jawab untuk mengelola nilai counter. Variabel counter didefinisikan sebagai RxInt dengan menggunakan .obs, menjadikannya reaktif, sehingga UI akan otomatis terupdate setiap kali nilai counter berubah. Dua fungsi utama ada di sini: increment() untuk menambah nilai counter dan reset() untuk mengatur nilai counter kembali ke nol.

Di bagian UI, terdapat kelas HomePage yang menampilkan antarmuka pengguna. Di dalam widget Scaffold, ada sebuah Text yang menampilkan nilai counter yang diperbarui secara reaktif menggunakan Obx(), yang secara otomatis merender ulang tampilan setiap kali nilai counter berubah. Ada dua tombol aksi (FloatingActionButton) yang masing-masing berfungsi untuk menambah nilai counter (increment()) atau mereset nilai counter (reset()).

Aplikasi ini dijalankan dengan memanggil main(), yang menggunakan MaterialApp untuk menampilkan HomePage sebagai halaman utama..