TUGAS PENDAHULUAN
PEMROGRAMAN PERANGKAT BERGERAK
MODUL XIII
NETWORKING

Disusun Oleh :

Lintang Suminar Tyas Wening / 2211104009

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

2024

TUGAS PENDAHULUAN

SOAL

1. Apa yang dimaksud dengan state management pada Flutter?
2. Sebut dan jelaskan komponen-komponen yang ada di dalam GetX.
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(),
  ));
}
```

JAWAB:

1. State management di Flutter adalah cara mengelola data atau "state" dalam aplikasi agar tampilan (UI) selalu sesuai dengan data terbaru. Misalnya, ketika pengguna menekan tombol yang mengubah data, tampilan aplikasi akan langsung diperbarui tanpa harus memuat ulang seluruh aplikasi. Ada beberapa cara mengelola state di Flutter, seperti menggunakan StatefulWidget untuk data lokal di satu widget, Provider untuk data yang digunakan oleh banyak widget, atau solusi lain seperti Riverpod, Bloc, dan GetX untuk kebutuhan yang lebih besar dan terstruktur. Intinya, state management memastikan data dan tampilan aplikasi selalu sinkron.

2. Berikut adalah komponen yang ada di dalam GetX:
   a. **State Management**
      Mengelola data (state) dan sinkronisasi UI secara otomatis. Contohnya, jika data berubah, tampilan langsung diperbarui tanpa kode tambahan. Menggunakan class seperti Rx atau GetBuilder untuk mengatur state.
      - Obx : Widget reaktif yang akan memperbarui tampilan secara otomatis setiap kali data (state) berubah.
      - Rx (Reactive Variables) : Variabel yang dibuat reaktif dengan menggunakan .obs.
      - Controller : Controller adalah kelas yang mengelola logika bisnis dan *state*. Biasanya, kita menggunakan GetxController untuk membuat controller.

   b. **Dependency Management**
      Komponen ini digunakan untuk mengatur *dependency* tanpa harus menggunakan InheritedWidget. GetX menggunakan metode *lazy loading*, sehingga *dependency* hanya di-*initialize* saat dibutuhkan.
      - Get,put() : Digunakan untuk mendaftarkan *dependency* agar dapat diakses di mana saja.

- Get.lazyPut() : Mendaftarkan *dependency* tetapi hanya di-*initialize* saat pertama kali digunakan.
- Get.find() : Mengambil *dependency* yang telah didaftarkan sebelumnya.
- Bindings : Bindings digunakan untuk mengatur *dependency* secara terpusat saat berpindah halaman.

c. Route Manajement

Digunakan untuk navigasi antar halaman tanpa boilerplate. Dengan Get.to() atau Get.off(), bisa berpindah halaman atau menggantinya tanpa menambahkan banyak kode di MaterialApp.

- Get.to() : Navigasi ke halaman lain
- Get.off() : Navigasi ke halaman lain dan menghapus halaman sebelumnya dari *stack*.
- Get.offAll() : Navigasi ke halaman lain dan menghapus semua halaman sebelumnya.
- Get.parameters : Digunakan untuk mengirim data saat navigasi.
- GetNamed : Navigasi menggunakan nama rute.

d. Snackbar, Dialog, BottomSheet

GetX menyediakan cara cepat untuk menampilkan notifikasi atau dialog seperti Get.snackbar(), Get.defaultDialog(), dan Get.bottomSheet() tanpa perlu context.

- Snackbar :
  ```
  Get.snackbar("Title", "Message", snackPosition: SnackPosition.BOTTOM);
  ```
- Dialog :
  ```
  Get.defaultDialog(
    title: "Dialog Title",
    middleText: "This is a dialog",
  );
  ```
- BottomSheet :
  ```
  Get.bottomSheet(
    Container(
      color: Colors.white,
      child: Text("This is a BottomSheet"),
    ),
  );
  ```

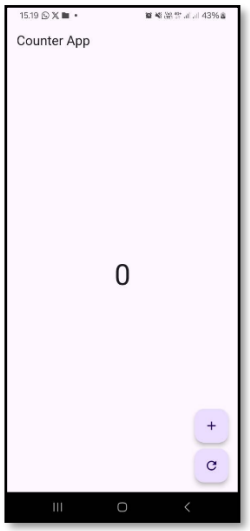
e. Middleware

Middleware digunakan untuk mengontrol logika sebelum sebuah rute ditampilkan. Contohnya, untuk otentikasi pengguna sebelum membuka halaman tertentu

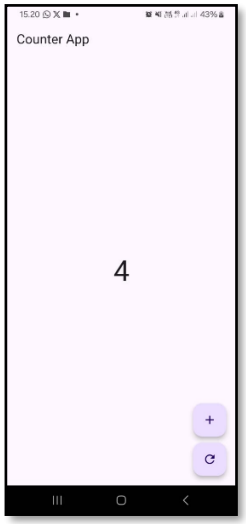
3. Lengkapilah code di bawah ini, dan tampilkan hasil outputnya serta jelaskan.
    - **Source Code**

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


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

/// Controller untuk mengelola state counter
class CounterController extends GetxController {
  // TODO: Tambahkan variabel untuk menyimpan nilai counter
  var counter = 0.obs;

  // TODO: Buat fungsi untuk menambah nilai counter
  void tambahCounter() => counter.value++;

  // TODO: Buat fungsi untuk mereset nilai counter
  void resetCounter() => counter.value = 0;
}

class HomePage extends StatelessWidget {
  HomePage({super.key});

  final CounterController controller = Get.put(CounterController());

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(title: const Text("Counter App")),
      body: Center(
        child: Obx(() {
          // TODO: Lengkapi logika untuk menampilkan nilai counter
          return Text(
            "${controller.counter.value}", // Ganti ini dengan nilai counter
            style: const TextStyle(fontSize: 48),
          );
        }),
      ),
      floatingActionButton: Column(
        mainAxisAlignment: MainAxisAlignment.end,
        children: [
          FloatingActionButton(
            onPressed: () {
              // TODO: Tambahkan logika untuk menambah nilai counter
              controller.tambahCounter();
            },
            child: const Icon(Icons.add),
          ),
          const SizedBox(height: 10),
          FloatingActionButton(
            onPressed: () {
              // TODO: Tambahkan logika untuk mereset nilai counter
              controller.resetCounter();
            },
            child: const Icon(Icons.refresh),
          ),
        ],
      ),
    );
  }
}
```

- **Screenshot Output**
  Pada Gambar 1 berupa Tampilan Awal Counter App. Awalnya, angka pada program menunjukkan 0. **Pada Gambar ke-2** Saat pengguna menekan tombol ”+”, angka akan bertambah atau naik. Seperti pada gambar dibawah ini , contohnya, jika tombol ditekan 4 kali, angka di layar akan menjadi 4. Namun, jika tombol reset ditekan, angka akan kembali menjadi 0.



Gambar 1



Gambar 2

- **Deskripsi Program**
  Program ini bernama Counter App yang menggunakan GetX untuk mengatur state. Aplikasi ini memiliki fitur untuk menambah nilai counter dan meresetnya ke nol. Nilai counter dikelola melalui CounterController, yang menggunakan variabel reaktif (obs) agar perubahan nilainya diperbarui secara otomatis di layar. Tampilan aplikasi terdiri dari halaman utama yang menampilkan nilai counter dengan widget Obx, serta dua tombol: tombol "+" untuk menambah angka atau nilai dan tombol "Refresh" untuk mereset angka atau nilai menjadi 0.