

PEMROGRAMAN PERANGKAT BERGERAK

MODUL XIII
NETWORKING



Disusun oleh:
Salman Alfa R
2211104056

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

NETWORKING

Tujuan Praktikum
Mahasiswa mampu memahami state management dalam Flutter
Mahasiswa mampu mengimplementasikan state management dalam Flutter

NETWORKING/STATE MANAGEMENT

State management dalam Flutter adalah proses mengelola state atau status dari aplikasi, yaitu data atau informasi yang dapat berubah sepanjang siklus hidup aplikasi. State ini mencakup segala hal yang memengaruhi tampilan antarmuka pengguna (UI), seperti input pengguna, data dari API, dan status internal widget. Ketika aplikasi semakin kompleks dibuat, maka pasti akan ada saatnya dimana harus dibagikan state aplikasi ke berbagai halaman yang ada.

Flutter adalah deklaratif, sehingga Flutter membangun user interface berdasarkan state saat ini. Dengan menggunakan state management, dapat dilakukan sentralisasi semua state dari berbagai macam UI Control untuk mengendalikan aliran data lintas aplikasi.

State management penting karena aplikasi Flutter sering kali terdiri dari banyak *widget* yang saling terkait. Dengan mengelola state dengan baik, kita dapat memastikan :

- Sinkronisasi UI dan data, karena selalu mencerminkan data terkini.
- Organisasi kode yang baik untuk mempermudah pengembangan dan pemeliharaan.
- Pengurangan bug, karena state yang dikelola dengan benar mengurangi kemungkinan terjadinya bug.

Jenis State dalam Flutter

1. Ephemeral State (State Lokal)

State ini hanya relevan untuk widget tertentu dan tidak dibagikan ke widget lain. Contohnya adalah state untuk *TextField* atau *Checkbox*. Dan kita dapat menggunakan

StatefulWidget untuk mengelola *ephemeral state*. Pendekatannya state management-nya ada dua, yakni *StatefulWidget* (untuk *ephemeral state*) dan *InheritedWidget* (untuk berbagai state antar widget).

2. App State (State Global)

State ini digunakan di berbagai widget dalam aplikasi. Contohnya adalah informasi pengguna yang masuk, data keranjang belanja, atau tema aplikasi. App state biasanya membutuhkan pendekatan state management yang lebih kompleks. *Package/library* pendukung Flutter memiliki berbagai framework atau *package* untuk *state management*, seperti :

A. Provider

Provider adalah library state management yang didukung resmi oleh tim Flutter. Provider memanfaatkan kemampuan bawaan Flutter seperti *InheritedWidget*, tetapi dengan cara yang lebih sederhana dan efisien.

B. Bloc/Cubit

Bloc (Business Logic Component) adalah pendekatan state management berbasis pola *stream*. Bloc memisahkan *business logic* dari UI, sehingga cocok untuk aplikasi yang besar dan kompleks.

C. Riverpod

Riverpod adalah framework state management modern yang dirancang sebagai pengganti atau alternatif untuk Provider. Riverpod lebih fleksibel dan mengatasi beberapa keterbatasan Provider.

D. GetX

GetX adalah framework Flutter serbaguna yang menyediakan solusi lengkap untuk *state management*, *routing*, dan *dependency injection*. GetX dirancang untuk meminimalkan *boilerplate code*, meningkatkan efisiensi, dan mempermudah pengembangan aplikasi Flutter, terutama yang memerlukan reaktivitas tinggi.

Berikut cara instalasi GetX :

1) Tambahkan GetX ke dalam proyek Flutter melalui pubspec.yaml :

```
dependencies:  
  flutter:  
    sdk: flutter  
  get: ^4.6.5
```

2) Konfigurasi dasar

Untuk menggunakan GetX, ubah root aplikasi dengan mengganti MaterialApp menjadi GetMaterialApp :

```
import 'package:flutter/material.dart';  
import 'package:get/get.dart';  
  
void main() {  
  runApp(MyApp());  
}  
  
class MyApp extends StatelessWidget {  
  @override  
  Widget build(BuildContext context) {  
    return GetMaterialApp(  
      title: 'Praktikum 13 - GetX',  
      home: HomePage(),  
    );  
  }  
}
```

3) State Management dengan GetX

a. Membuat Controller

Buat class controller untuk mengelola state. Misalnya, untuk counter sederhana :

```
import 'package:get/get.dart';  
  
class CounterController extends GetxController {  
  var count = 0.obs; // State yang reaktif
```

```
void increment() => count++;  
}
```

b. Menggunakan Controller di UI

- Tambahkan controller ke dalam widget menggunakan `Get.put()` untuk dependency injection.
- Gunakan `Obx` untuk memantau perubahan state.

```
import 'package:flutter/material.dart';  
import 'package:get/get.dart';  
import 'counter_controller.dart';  
  
class HomePage extends StatelessWidget {  
  final CounterController controller =  
    Get.put(CounterController());  
  
  @override  
  Widget build(BuildContext context) {  
    return Scaffold(  
      appBar: AppBar(title: Text('GetX State  
Management')),  
      body: Center(  
        child: Obx(() => Text(  
          'Counter: ${controller.count}',  
          style: TextStyle(fontSize: 25),  
        )),  
      ),  
      floatingActionButton: FloatingActionButton(  
        onPressed: controller.increment,  
        child: Icon(Icons.add),  
      ),  
    );  
  }  
}
```

4)

Routing dengan GetX

a. Definisikan Route

Gunakan `GetPage` pada `main.dart` untuk mendefinisikan rute aplikasi :

```

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

void main() {
  runApp(MyApp());
}

class MyApp extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return GetMaterialApp(
      initialRoute: '/',
      getPages: [
        GetPage(name: '/', page: () => HomePage()),
        GetPage(name: '/details', page: () =>
DetailsPage()),
      ],
    );
  }
}

class HomePage extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return Scaffold(
      body: Center(
        child: ElevatedButton(
          onPressed: () {
            Get.toNamed('/details'); // Navigasi ke
halaman lain
          },
          child: Text('Go to Details'),
        ),
      ),
    );
  }
}

```

```

class DetailsPage extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return Scaffold(
      body: Center(
        child: Text('Details Page'),
      ),
    );
  }
}

```

b. Navigasi

- `Get.to()` : Navigasi ke halaman baru.
- `Get.back()` : Kembali ke halaman sebelumnya.
- `Get.off()` : Menghapus semua halaman sebelumnya.
- `Get.offAll()` : Menghapus semua halaman dalam stack.

5) Dependency Injection dengan GetX

1. Injeksi Sederhana

Gunakan `Get.put()` untuk membuat instance controller yang tersedia di mana saja :

```

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

```

2. Lazy Loading

Gunakan `Get.lazyPut()` jika ingin membuat instance hanya saat dibutuhkan :

```

Get.lazyPut(() => CounterController());

```

3. Mengambil Instance

Ambil instance di mana saja dalam aplikasi :

```

final CounterController controller = Get.find();

```

6) SnackBar

```
Get.snackbar('Title', 'This is a snackbar');
```

7) Dialog

```
Get.defaultDialog(  
  title: 'Dialog Title',  
  middleText: 'This is a dialog',  
);
```

8) BottomSheet

```
Get.bottomSheet(  
  Container(  
    child: Text('This is a bottom sheet'),  
  ),  
);
```

~ **SEKIAN & TERIMA KASIH** ~

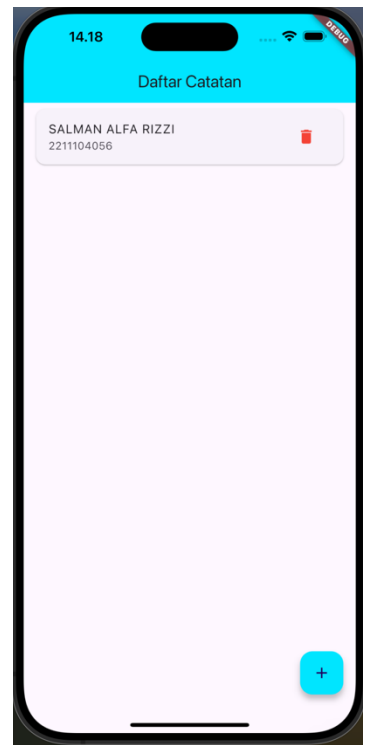
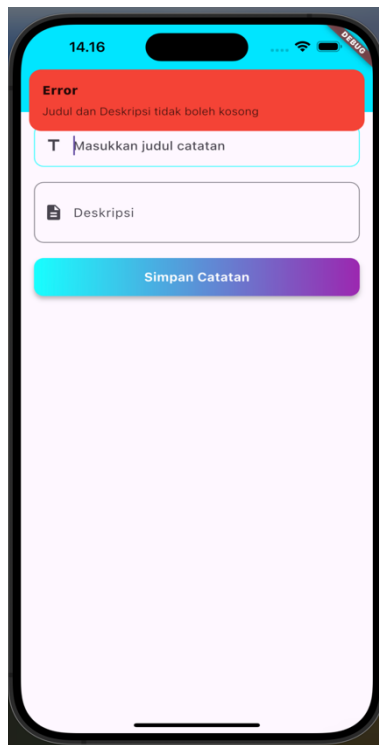
TUGAS MANDIRI (UNGUIDED)

SOAL

Buatlah Aplikasi Catatan Sederhana menggunakan GetX, dengan ketentuan sebagai berikut :

1. Halaman utama atau Homepage untuk menampilkan daftar catatan yang telah ditambahkan. Setiap catatan terdiri dari judul dan deskripsi singkat, serta terdapat tombol untuk menghapus catatan dari daftar.
2. Halaman kedua untuk menambah catatan baru, berisi : form untuk memasukkan judul dan deskripsi catatan, serta tombol untuk menyimpan catatan ke daftar (Homepage).
3. Menggunakan getx controller.
4. Menggunakan getx routing untuk navigasi halaman.

ScreenShoot:



SourceCode:

Main.dart

```
import 'package:flutter/material.dart';
import 'package:get/get.dart';
import 'package:praktikum_13/view/notes_page.dart';
import 'view/home_page.dart';
void main() {
  runApp(const MyApp());
}

class MyApp extends StatelessWidget {
  const MyApp({super.key});
  @override
  Widget build(BuildContext context) {
    return GetMaterialApp(
      initialRoute: '/home',
      getPages: [
        GetPage(name: '/home',
          page: ()=> HomePage()),
        GetPage(
          name: '/note',
          page: ()=> AddNotePage(),

        ],
    );
  }
}
```

Notes_controller.dart

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

class NotesController extends GetxController {
  var notes = <Map<String, String>>[].obs;

  void addNote(String title, String description) {
    notes.add({"title": title, "description": description});
  }

  void deleteNote(int index) {
    notes.removeAt(index);
  }
}
```

home_page.dart

```
import 'package:flutter/material.dart';
import 'package:get/get.dart';
import 'package:praktikum_13/view%20model/notes_controller.dart';
import 'package:praktikum_13/view/notes_page.dart';
class HomePage extends StatelessWidget {
  final NotesController notesController = Get.put(NotesController());

  HomePage({super.key});

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: Text('Daftar Catatan'),
        backgroundColor: Colors.cyanAccent[400],
        centerTitle: true,
      ),
      body: Obx(() {
        return notesController.notes.isEmpty
          ? Center(
              child: Text(
                'Belum ada catatan.',
                style: TextStyle(fontSize: 16),
              ),
            )
          : ListView.builder(
              itemCount: notesController.notes.length,
              itemBuilder: (context, index) {
                final note = notesController.notes[index];
                return Card(
                  margin: EdgeInsets.symmetric(vertical: 8, horizontal: 16),
                  child: ListTile(
                    title: Text(note['title']!),
                    subtitle: Text(note['description']!),
                    trailing: IconButton(
                      icon: Icon(Icons.delete, color: Colors.red),
                      onPressed: () => notesController.deleteNote(index),
                    ),
                  ),
                );
              },
            );
      }),
      floatingActionButton: FloatingActionButton(
        child: Icon(Icons.add),
        backgroundColor: Colors.cyanAccent[400],
        onPressed: () => Get.to(AddNotePage()),
      ),
    );
  }
}
```

notes_page.dart

```
import 'package:flutter/material.dart';
import 'package:get/get.dart';
import 'package:praktikum_13/view%20model/notes_controller.dart';

class AddNotePage extends StatelessWidget {
  final NotesController notesController = Get.find();
  final TextEditingController titleController = TextEditingController();
  final TextEditingController descriptionController = TextEditingController();

  AddNotePage({super.key});

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: const Text('Tambah Catatan'),
        backgroundColor: Colors.cyanAccent[400],
      ),
      body: Padding(
        padding: const EdgeInsets.all(16.0),
        child: Column(
          children: [
            TextField(
              controller: titleController,
              decoration: InputDecoration(
                labelText: 'Judul',
                hintText: 'Masukkan judul catatan',
                prefixIcon: Icon(Icons.title),
                border: OutlineInputBorder(
                  borderRadius: BorderRadius.circular(12),
                ),
                focusedBorder: OutlineInputBorder(
                  borderRadius: BorderRadius.circular(12),
                  borderSide: BorderSide(color: Colors.cyanAccent, width: 1),
                ),
              ),
            ),
            SizedBox(height: 20),
            // TextField untuk Deskripsi
            TextField(
              controller: descriptionController,
              decoration: InputDecoration(
                labelText: 'Deskripsi',
                hintText: 'Masukkan deskripsi singkat',
                prefixIcon: Icon(Icons.description),
                border: OutlineInputBorder(
                  borderRadius: BorderRadius.circular(12),
                ),
                focusedBorder: OutlineInputBorder(
```

```
borderRadius: BorderRadius.circular(12),
borderSide: BorderSide(color: Colors.cyanAccent, width: 1),
),
),
maxLines: 2,
),
  SizedBox(height: 20),
  Container(
    width: double.infinity,
    height: 50,
    decoration: BoxDecoration(
      gradient: LinearGradient(
        colors: [Colors.cyanAccent, Colors.purple],
        begin: Alignment.bottomLeft,
        end: Alignment.bottomRight,
      ),
      borderRadius: BorderRadius.circular(12),
      boxShadow: [
        BoxShadow(
          color: Colors.black26,
          offset: Offset(0, 4),
          blurRadius: 4,
        ),
      ],
    ),
    child: ElevatedButton(
      style: ElevatedButton.styleFrom(
        backgroundColor: Colors.transparent,
        shadowColor: Colors.transparent,
        shape: RoundedRectangleBorder(
          borderRadius: BorderRadius.circular(12),
        ),
      ),
      onPressed: () {
        if (titleController.text.isNotEmpty &&
            descriptionController.text.isNotEmpty) {
          notesController.addNote(
            titleController.text, descriptionController.text);
          Get.back();
        } else {
          Get.snackbar(
            'Error',
            'Judul dan Deskripsi tidak boleh kosong',
            snackPosition: SnackPosition.TOP,
            backgroundColor: Colors.red
          );
        }
      },
      child: Text(
        'Simpan Catatan',
        style: TextStyle(
          fontSize: 16,
          fontWeight: FontWeight.bold,
```

```

        color: Colors.white,)
      ),),
    ),
  ],
),
),
);
}
}

```

Penjelasan:

Program ini adalah aplikasi Flutter sederhana untuk mengelola catatan menggunakan GetX. Aplikasi terdiri dari dua halaman: halaman utama untuk menampilkan daftar catatan dan halaman kedua untuk menambah catatan baru. State management dikelola oleh NotesController yang menggunakan RxList agar data catatan dapat diperbarui secara otomatis pada UI. Halaman utama menggunakan Obx untuk memantau perubahan data dan menampilkan daftar catatan dengan fitur hapus. Navigasi antar halaman dilakukan menggunakan Get.to, sedangkan data ditambahkan melalui form dengan validasi input pada halaman tambah catatan. Aplikasi ini dirancang agar responsif, modern, dan mudah digunakan.

Note: Jangan lupa sertakan source code, screenshot output, dan deskripsi program. Kreativitas menjadi nilai tambah.