

**LAPORAN PRAKTIKUM  
PEMROGRAMAN PERANGKAT BERGERAK**

**MODUL 13  
NETWORKING**



**Disusun Oleh :  
Dzikri Naufal Wisnu Pravida/2211104063  
SE06-02**

**Asisten Praktikum :  
Muhammad Faza Zulian  
Aisyah Hasna Aulia**

**Dosen Pengampu :  
Yudha Islami Sulistya**

**PROGRAM STUDI S1 SOFTWARE ENGINEERING  
FAKULTAS INFORMATIKA  
TELKOM UNIVERSITY PURWOKERTO  
2024**

## **PRAKTIKUM**

### **A. GUIDED**

#### **1. Teori**

##### **1.1. Jenis-jenis state dalam flutter**

###### **1.1.1. 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).

###### **1.1.2. App state(state global)**

State ini biasanya digunakan di berbagai widget dalam sebuah aplikasi. Seperti contohnya theme app, data keranjang di aplikasi e-commerce,. 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.

## 2. Hasil Praktikum

### Source Code

#### Main



```
1  import 'package:flutter/material.dart';
2  import 'package:get/get.dart';
3  import 'package:prak_13/view_model/counter_controller.dart';
4  import 'package:prak_13/view/detail_page.dart';
5  import 'package:prak_13/view/my_home_page.dart';
6
7  void main() {
8    runApp(MyApp());
9  }
10
11 class MyApp extends StatelessWidget {
12   MyApp({super.key});
13   final CounterController controller = Get.put(CounterController());
14
15   @override
16   Widget build(BuildContext context) {
17     return GetMaterialApp(
18       initialRoute: '/',
19       getPages: [
20         GetPage(
21           name: '/',
22           page: () => MyHomePage(
23             title: 'Flutter Demo Home Page',
24           ),
25         GetPage(name: '/detail', page: () => DetailPage()),
26       ],
27     );
28   }
29 }
30
```

## Homepage

```
1 import 'package:flutter/material.dart';
2 import 'package:get/get.dart';
3 import 'package:prak_13/view_model/counter_controller.dart';
4
5 class MyHomePage extends StatelessWidget {
6   MyHomePage({super.key, required this.title});
7   final String title;
8   final CounterController controller = Get.put(CounterController());
9
10  @override
11  Widget build(BuildContext context) {
12    return Scaffold(
13      appBar: AppBar(
14        backgroundColor: Theme.of(context).colorScheme.inversePrimary,
15        title: Text(title),
16      ),
17      body: Center(
18        child: Obx(
19          () => Column(
20            mainAxisAlignment: MainAxisAlignment.center,
21            children: <Widget>[
22              const Text(
23                'You have pushed the button this many times:',
24              ),
25              Text(
26                controller.counter.toString(),
27                style: Theme.of(context).textTheme.headlineMedium,
28              ),
29              ElevatedButton(
30                onPressed: () {
31                  Get.toNamed('/detail');
32                },
33                child: Text('Go to Detail'))
34            ],
35          ),
36        ),
37      ),
38      floatingActionButton: Row(
39        mainAxisAlignment: MainAxisAlignment.spaceEvenly,
40        children: [
41          FloatingActionButton(
42            onPressed: controller.incrementCounter,
43            tooltip: 'Increment',
44            child: const Icon(Icons.add),
45          ),
46          FloatingActionButton(
47            onPressed: controller.decrementCounter,
48            tooltip: 'decrement',
49            child: const Icon(Icons.remove),
50          ),
51          FloatingActionButton(
52            onPressed: controller.getsnackbar,
53            tooltip: 'snackbar',
54            child: const Icon(Icons.message),
55          ),
56          FloatingActionButton(
57            onPressed: controller.getdialog,
58            tooltip: 'dialog',
59            child: const Icon(Icons.notifications_active),
60          ),
61          FloatingActionButton(
62            onPressed: controller.getbottomsheet,
63            tooltip: 'bottomsheet',
64            child: const Icon(Icons.arrow_upward),
65          ),
66        ],
67      ),
68    );
69  }
70 }
71
```

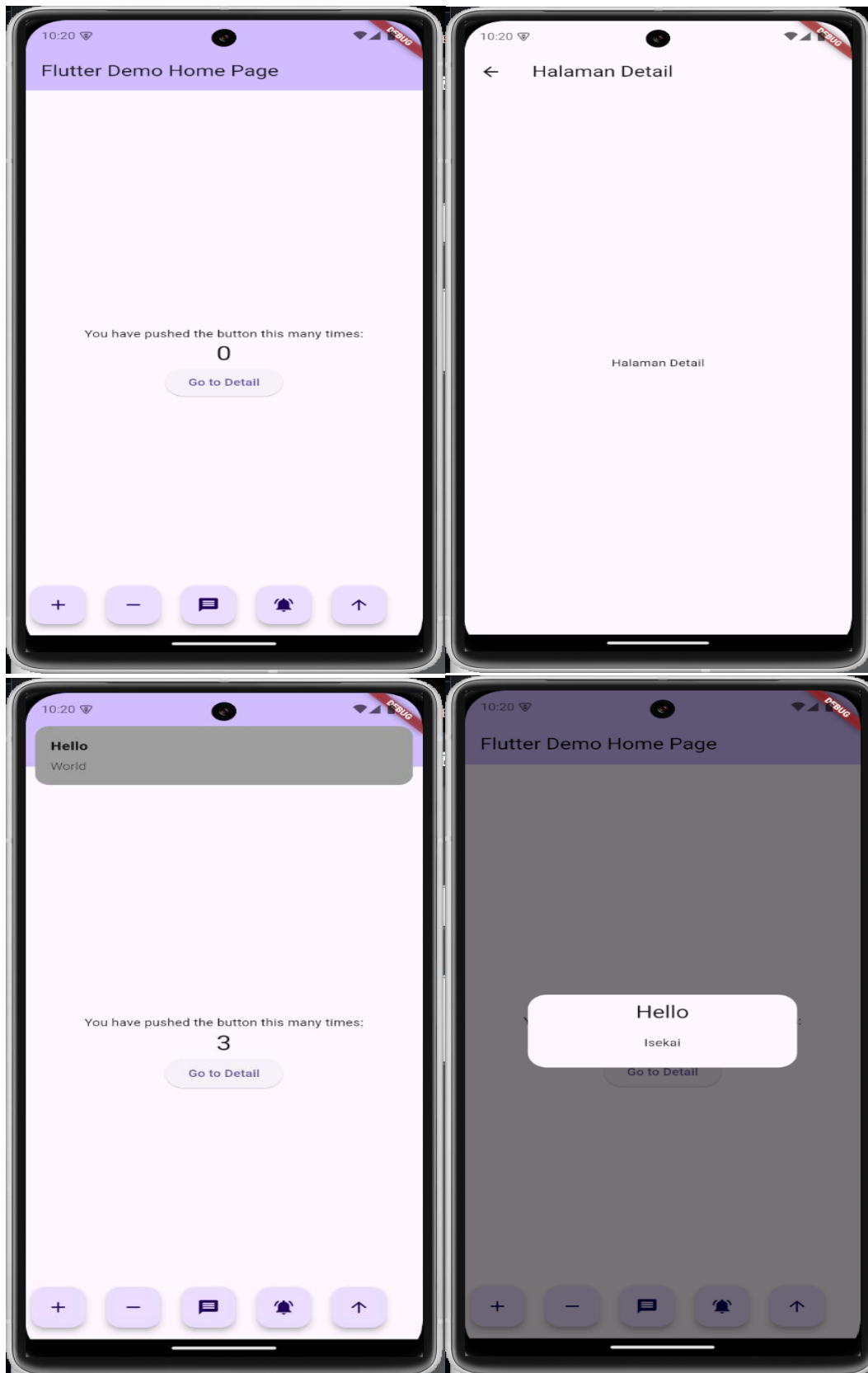


```
1  import 'package:flutter/material.dart';
2
3  class DetailPage extends StatelessWidget {
4    const DetailPage({super.key});
5
6    @override
7    Widget build(BuildContext context) {
8      return Scaffold(
9        appBar: AppBar(
10         title: const Text("Halaman Detail"),
11       ),
12       body: const Center(
13         child: Text("Halaman Detail"),
14       ),
15     );
16   }
17 }
18
```

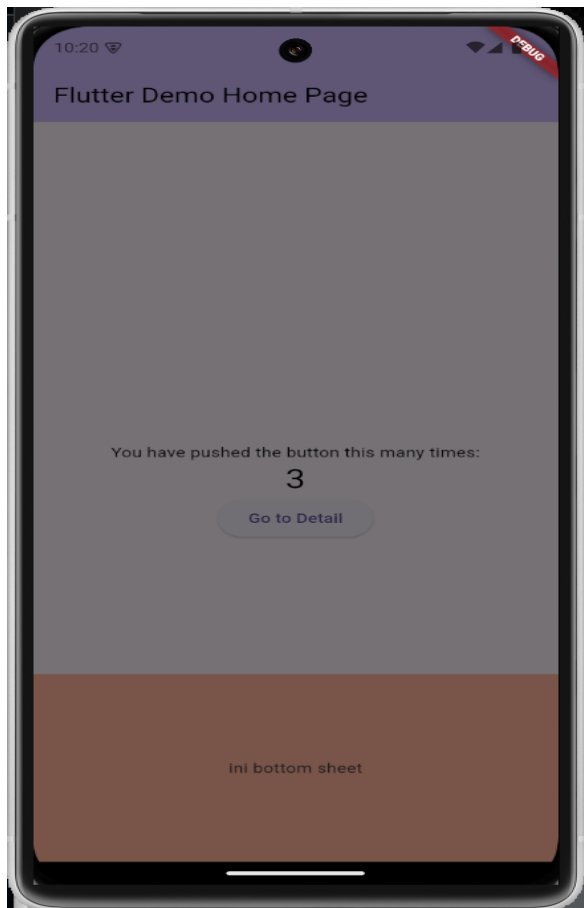
## Counter controller

```
1  import 'package:flutter/material.dart';
2  import 'package:get/get.dart';
3
4  class CounterController extends GetxController {
5      var counter = 0.obs;
6      void incrementCounter() {
7          counter++;
8      }
9
10     void decrementCounter() {
11         counter--;
12     }
13
14     void getsnackbar() {
15         Get.snackbar(
16             "Hello", "World",
17             //warna background snackbar
18             backgroundColor: Colors.grey,
19         );
20     }
21
22     void getdialog() {
23         Get.defaultDialog(
24             title: "Hello",
25             middleText: "Isekai",
26         );
27     }
28
29     void getbottomsheet() {
30         Get.bottomSheet(
31             Container(
32                 child: Center(child: Text("ini bottom sheet")),
33                 height: 200,
34                 width: double.infinity,
35                 color: Colors.brown,
36             ),
37         );
38     }
39 }
40
```

**Output :**







## **B. UNGUIDED**


### **1. Soal Studi Case**

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.

### **Sourcecode**

## Main



```
1 import 'package:flutter/material.dart';
2 import 'package:get/get.dart';
3 import 'views/home_page.dart';
4 import 'views/add_note_page.dart';
5
6 void main() {
7   runApp(MyApp());
8 }
9
10 class MyApp extends StatelessWidget {
11   @override
12   Widget build(BuildContext context) {
13     return GetMaterialApp(
14       title: 'Simple Notes App',
15       initialRoute: '/',
16       getPages: [
17         GetPage(name: '/', page: () => HomePage()),
18         GetPage(name: '/add_note', page: () => AddNotePage()),
19       ],
20     );
21   }
22 }
23
```

## Home page


```
1 import 'package:flutter/material.dart';
2 import 'package:get/get.dart';
3 import '../controllers/notes_controller.dart';
4
5 class HomePage extends StatelessWidget {
6   final NotesController _notesController = Get.put(NotesController());
7
8   @override
9   Widget build(BuildContext context) {
10    return Scaffold(
11      appBar: AppBar(
12        title: Text('My Notes'),
13        centerTitle: true,
14      ),
15      body: Obx(() => ListView.builder(
16        itemCount: _notesController.notes.length,
17        itemBuilder: (context, index) {
18          final note = _notesController.notes[index];
19          return ListTile(
20            title: Text(note.title),
21            subtitle: Text(note.description),
22            trailing: IconButton(
23              icon: Icon(Icons.delete, color: Colors.red),
24              onPressed: () => _notesController.deleteNote(index),
25            ),
26          );
27        },
28      )),
29      floatingActionButton: FloatingActionButton(
30        onPressed: () => Get.toNamed('/add_note'),
31        child: Icon(Icons.add),
32      ),
33    );
34  }
35 }
36
```

## Note controller



```
1  import 'package:get/get.dart';
2  import '../models/note.dart';
3
4  class NotesController extends GetxController {
5      RxList<Note> notes = <Note>[].obs;
6
7      void addNote(Note note) {
8          notes.add(note);
9      }
10
11     void deleteNote(int index) {
12         notes.removeAt(index);
13     }
14 }
15
```

## Note

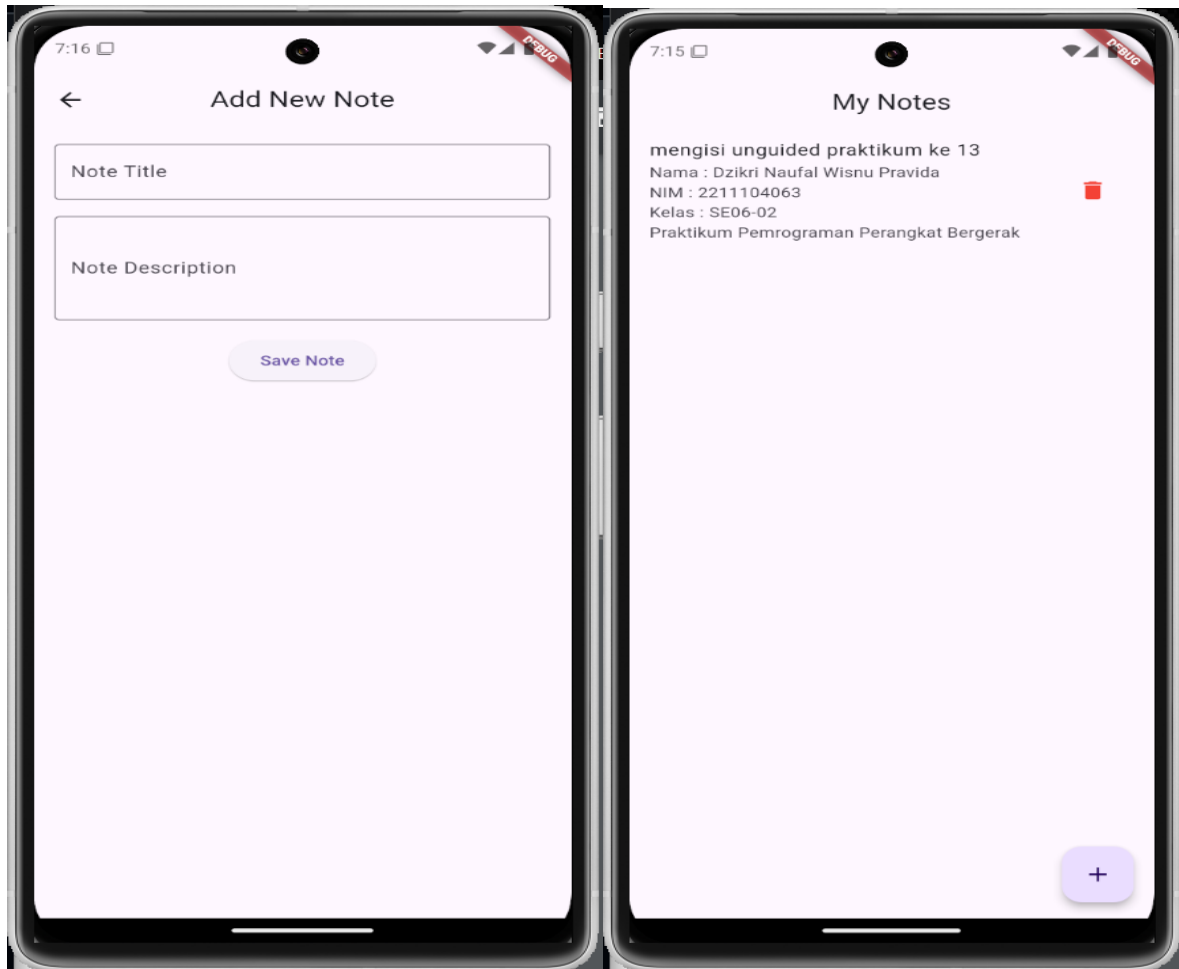


```
1  class Note {
2      String title;
3      String description;
4
5      Note({required this.title, required this.description});
6  }
7
```

## Add note

```
1 import 'package:flutter/material.dart';
2 import 'package:get/get.dart';
3 import '../controllers/notes_controller.dart';
4 import '../models/note.dart';
5
6 class AddNotePage extends StatelessWidget {
7   final NotesController _notesController = Get.find();
8   final TextEditingController _titleController = TextEditingController();
9   final TextEditingController _descriptionController = TextEditingController();
10
11   @override
12   Widget build(BuildContext context) {
13     return Scaffold(
14       appBar: AppBar(
15         title: Text('Add New Note'),
16         centerTitle: true,
17       ),
18       body: Padding(
19         padding: const EdgeInsets.all(16.0),
20         child: Column(
21           children: [
22             TextField(
23               controller: _titleController,
24               decoration: InputDecoration(
25                 labelText: 'Note Title',
26                 border: OutlineInputBorder(),
27               ),
28             ),
29             SizedBox(height: 16),
30             TextField(
31               controller: _descriptionController,
32               maxLines: 3,
33               decoration: InputDecoration(
34                 labelText: 'Note Description',
35                 border: OutlineInputBorder(),
36               ),
37             ),
38             SizedBox(height: 16),
39             ElevatedButton(
40               onPressed: () {
41                 if (_titleController.text.isNotEmpty &&
42                     _descriptionController.text.isNotEmpty) {
43                   _notesController.addNote(
44                     Note(
45                       title: _titleController.text,
46                       description: _descriptionController.text,
47                     ),
48                   );
49                   Get.back(); // Return to home page
50                 } else {
51                   Get.snackbar(
52                     'Error',
53                     'Please enter both title and description',
54                     snackPosition: SnackPosition.BOTTOM,
55                   );
56                 }
57             },
58             child: Text('Save Note'),
59           ],
60         ),
61       ),
62     );
63   }
64 }
65
66
```

## Screenshoot Output



## Deskripsi Program

Program ini berisikan aplikasi note sederhana menggunakan getx. Aplikasi terdiri dari empat komponen utama: model Note yang merepresentasikan struktur data catatan, NotesController yang mengelola state daftar catatan dengan metode reaktif menggunakan RxList, HomePage yang menampilkan daftar catatan yang sudah dibuat dengan kemampuan untuk menghapus catatan, serta AddNotePage yang memungkinkan pengguna menambahkan catatan baru melalui formulir sederhana.

Alur kerja aplikasi dimulai dengan inisialisasi GetMaterialApp yang mengatur routing antara halaman utama dan halaman tambah catatan. Pada halaman utama (HomePage), pengguna dapat melihat daftar catatan yang telah dibuat, dengan setiap item catatan menampilkan judul dan deskripsi singkat serta tombol hapus. Ketika tombol tambah (+) ditekan, aplikasi akan navigasi ke halaman AddNotePage di mana pengguna dapat memasukkan judul dan deskripsi catatan. Setelah menekan tombol simpan, catatan baru akan ditambahkan ke daftar menggunakan metode addNote() pada NotesController, dan pengguna dikembalikan ke halaman utama.