

PEMROGRAMAN PERANGKAT BERGERAK

MODUL VI NAVIGASI DAN NOTIFIKASI



Disusun Oleh :

Salman Alfa Rizzi

2211104056

Asisten Praktikum

Muhammad Faza Zulian

Gesit AlBarru

Aisyah Hasna Aulia

Dosen Pengampu :

Yudha Islami Sulistya, S.Kom., M.Cs.

PROGRAM STUDI S1 SOFTWARE

ENGINEERING FAKULTAS INFORMATIKA

TELKOM UNIVERSITY PURWOKERTO 2024

NAVIGASI DAN NOTIFIKASI

Tujuan Praktikum

Mahasiswa mampu memahami konsep layout pada Flutter

Mahasiswa dapat mengimplementasikan desain user interface pada Flutter

1. Model

Pada umumnya, hampir seluruh aplikasi yang dibuat akan bekerja dengan data. Data dalam sebuah aplikasi memiliki sangat banyak bentuk, tergantung dari aplikasi yang dibuat. Setiap data yang diterima atau dikirimkan akan lebih baik apabila memiliki standar yang sama. Hampir mustahil melakukan peneliharaan *project* yang kompleks tanpa model.

- **Membuat Model Class**

Untuk membuat model, buatlah direktori baru pada folder lib project flutter, kemudian buat sebuah file class dart dengan nama filenya adalah nama data yang ingin dijadikan model. Sebagai contoh, ketika membuat model user dengan response json di bawah ini:

```
{
  "user_id": 1,
  "id": 13,
  "title": "Bedroom Pop"
}
```

Maka buatlah file user.dart di dalam folder models, dengan code seperti di bawah ini:

```
class Album {
  final int userId;
  final int id;
  final String title;

  const Album({
    required this.userId,
    required this.id,
```

```

        required this.title,
    });

    factory Album.fromJson(Map<String, dynamic> json) {
        return Album(
            userId: json['user_id'],
            id: json['id'],
            title: json['title'],
        );
    }
}

```

2. Navigation

Dalam Flutter, navigation merujuk pada cara berpindah antar halaman (atau tampilan) di aplikasi. Sistem navigasi di Flutter berbasis route dan navigator. Setiap halaman atau layar di Flutter disebut sebagai route, dan Navigator adalah widget yang mengelola stack dari route tersebut.

- **Navigation Pindah Halaman**

Untuk melakukan navigasi ke halaman lain pada Flutter, dapat digunakan code seperti di bawah ini:

```

Navigator.push(
    Context, MaterialPageRoute(builder: (context) =>
    SecondRoute()),
);

```

Untuk melakukan navigasi kembali ke halaman sebelumnya, dapat digunakan code seperti di bawah ini:

```

Navigator.pop(context);

```

Potongan code diatas harus diletakkan dalam function sebuah widget, misalnya pada *onPressed* milik *ElevatedButton*. *SecondRoute* pada contoh dapat diubah menjadi halaman bari yang dituju.

- **Navigation Mengirim Data**

Untuk dapat melakukan navigasi dengan mengirimkan data ke halamanlain, perlu disiapkan 2 hal, yakni:

- 1) Halaman baru memiliki parameter data yang diminta.
- 2) Halaman awal mengirimkan data melalui parameter.

Untuk dapat melakukan hal tersebut, kita dapat membuat sebuahhalaman baru seperti di bawah ini:

```
class DetailScreen extends StatelessWidget {  
  const DetailScreen({Key? key, required this.title}) :  
    super(key: key);  
  
  final String title;  
  
  @override  
  Widget build(BuildContext context) {  
    return Scaffold(  
      appBar: AppBar(  
        title: Text(title),  
      ),  
    );  
  }  
}
```

Pada code di atas, telah dibuat halaman baru yang memiliki parameterdata yang diinginkan yaitu atribut title dengan tipe data string.

```
Navigator.push(  
  context,  
  MaterialPageRoute(builder: (context) => DetailScreen(title:  
    "Detail User")),  
);
```

Berbeda dengan contoh navigasi sebelumnya, pada navigasi di atas ditambahkan parameter title berisi string “Detail User” yang akan

dikirimkan ke halaman baru.

3. Notification

Untuk mengirimkan notifikasi dalam aplikasi Flutter, dapat digunakan package bernama `flutter_local_notifications`. Tambahkan terlebih dahulu package tersebut ke dalam `pubspec.yaml`.

```
dependencies:  
  flutter:  
    sdk: flutter  
  
  cupertino_icons: ^1.0.6  
  flutter_local_notifications: ^17.2.4
```

Konfigurasi Gradle

Pastikan `minSdkVersion` di `android/app/build.gradle` diatur ke 21 atau lebih tinggi:

```
defaultConfig {  
    minSdkVersion 21  
}
```

Konfigurasi iOS

Buka file `ios/Runner/Info.plist` dan tambahkan kode berikut untuk mendapatkan izin notifikasi pada iOS:

```
<key>UIBackgroundModes</key>  
<array>  
  <string>fetch</string>  
  <string>remote-notification</string>  
</array>  
<key>NSAppTransportSecurity</key>  
<dict>  
  <key>NSAllowsArbitraryLoads</key>  
  <true/>  
</dict>
```

```
<key>NSUserNotificationUsageDescription</key>
<string>We need your permission to send notifications.</string>
```

Setelah menambahkan package, ubah file *AndroidManifest.xml* dengan menambahkan barisan code seperti di bawah ini:

```
<!-- Izin untuk menghidupkan kembali notifikasi setelah reboot -->
<uses-permission
android:name="android.permission.RECEIVE_BOOT_COMPLETED" />
<!-- Izin untuk mengaktifkan getaran pada notifikasi --> <uses-
permission android:name="android.permission.VIBRATE" />
```

Tambahkan potongan kode di bawah ini di luar build dari stateful widget tersebut:

```
FlutterLocalNotificationsPlugin flutterLocalNotificationsPlugin
=FlutterLocalNotificationsPlugin();
```

Dengan membuat sebuah object Flutter LocalNotificationsPlugin, operasi yang terdapat dalam package flutter_notification dapat digunakan.

Override method initState dari widget tersebut dengan menambahkan code seperti di bawah ini:

```
void initState() {
  super.initState();
  var initializationSettingsAndroid =
    AndroidInitializationSettings('flutter_devs');
  var initializationSettingsIOs = IOSInitializationSettings();
  var initSetttings = InitializationSettings(
    initializationSettingsAndroid, initializationSettingsIOs);
  flutterLocalNotificationsPlugin.initialize(initSetttings,
    onSelectNotification: onSelectNotification);
}
```

Kemudian buat sebuah function yang akan mengendalikan ketika notifikasi

dipilih:

```
Future onSelectNotification(String payload) {  
    Navigator.of(context).push(MaterialPageRoute(builder: (_) {  
        return NewScreen(  
            payload: payload,  
        );  
    }));  
}
```

Buatlah sebuah widget baru yang akan menjadi halaman berikutnya setelah notifikasi dipilih:

```
class NewScreen extends StatelessWidget {  
    String payload;  
    NewScreen({  
        @required this.payload,  
    });  
    @override  
    Widget build(BuildContext context) {  
        return Scaffold(  
            appBar: AppBar(  
                title: Text(payload),  
            ),  
        );  
    }  
}
```

Kemudian buat sebuah function yang fungsinya untuk menampilkan notifikasi sederhana untuk android:

```
showNotification() async {  
    var android = new AndroidNotificationDetails(  
        'id', 'channel ', 'description',  
        priority: Priority.High, importance: Importance.Max);  
    var iOS = new IOSNotificationDetails();  
    var platform = new NotificationDetails(android, iOS);  
    await flutterLocalNotificationsPlugin.show(  
        0, 'title', 'body', platform);  
}
```

```
0, 'Flutter devs', 'Flutter Local Notification Demo',  
platform,  
  payload: 'Welcome to the Local Notification demo ');  
}
```

- `AndroidNotificationDetails` akan berisi mengenai detail notifikasi pada android.
- `IOSNotificationDetails` akan berisi mengenai detail notifikasi pada iOS.
- Kunci untuk menampilkan notifikasi terletak pada pemanggilan function `flutterLocalNotificationsPlugin` yang berfungsi untuk menampilkan notifikasi sesuai dengan platform yang digunakan.

Keseluruhan code di atas akan membentuk sebuah file seperti di bawah ini:

main.dart

```
import 'dart:async';  
import 'package:flutter/material.dart';  
import  
'package:flutter_local_notifications/flutter_local_notifications.d  
art';  
  
void main() => runApp(new MaterialApp(  
  theme: ThemeData(  
    appBarTheme: AppBarTheme(  
      color: Colors.amber,  
    )),  
  home: new MyApp(),  
  debugShowCheckedModeBanner: false,  
));  
  
class MyApp extends StatefulWidget {  
  @override  
  _MyAppState createState() => _MyAppState();  
}  
  
class _MyAppState extends State<MyApp> {
```



```

FlutterLocalNotificationsPlugin flutterLocalNotificationsPlugin
=FlutterLocalNotificationsPlugin();

@override
void initState() { super.initState();
    var initializationSettingsAndroid = AndroidInitializationSettings('flutter_devs');
    var initializationSettingsIOs=IOSInitializationSettings();var initSettings =
    InitializationSettings(
        initializationSettingsAndroid, initializationSettingsIOs);

    flutterLocalNotificationsPlugin.initialize(initSettings,onSelectNotification:
        onSelectNotification);
}

Future onSelectNotification(String payload) { Navigator.of(context).push(MaterialPageRoute(builder: ()
{
    return NewScreen( payload:
        payload,
    );
}));
}

@override
Widget build(BuildContext context) {return Scaffold(
    appBar: AppBar( backgroundColor:
        Colors.amber,
        title: new Text('Flutter notification demo'),
    ),
    body: new Center(child:
        Column(
            children: <Widget>[ButtonTheme(
                minWidth: 250.0, child:
                RaisedButton(
                    color: Colors.blueAccent,

```

```

        onPressed: showNotification, child: new Text(
          'showNotification',
        ),
      ),
    ),
  ],
),
),
);
}
showNotification() async {
  var android = new AndroidNotificationDetails('id', 'channel ', 'description',
    priority: Priority.High, importance: Importance.Max); var iOS = new
  IOSNotificationDetails();
  var platform = new NotificationDetails(android, iOS); await
  flutterLocalNotificationsPlugin.show(
    0, 'Flutter devs', 'Flutter Local Notification Demo', platform,
    payload: 'Welcome to the Local Notification demo ');
}

class NewScreen extends StatelessWidget {String payload;

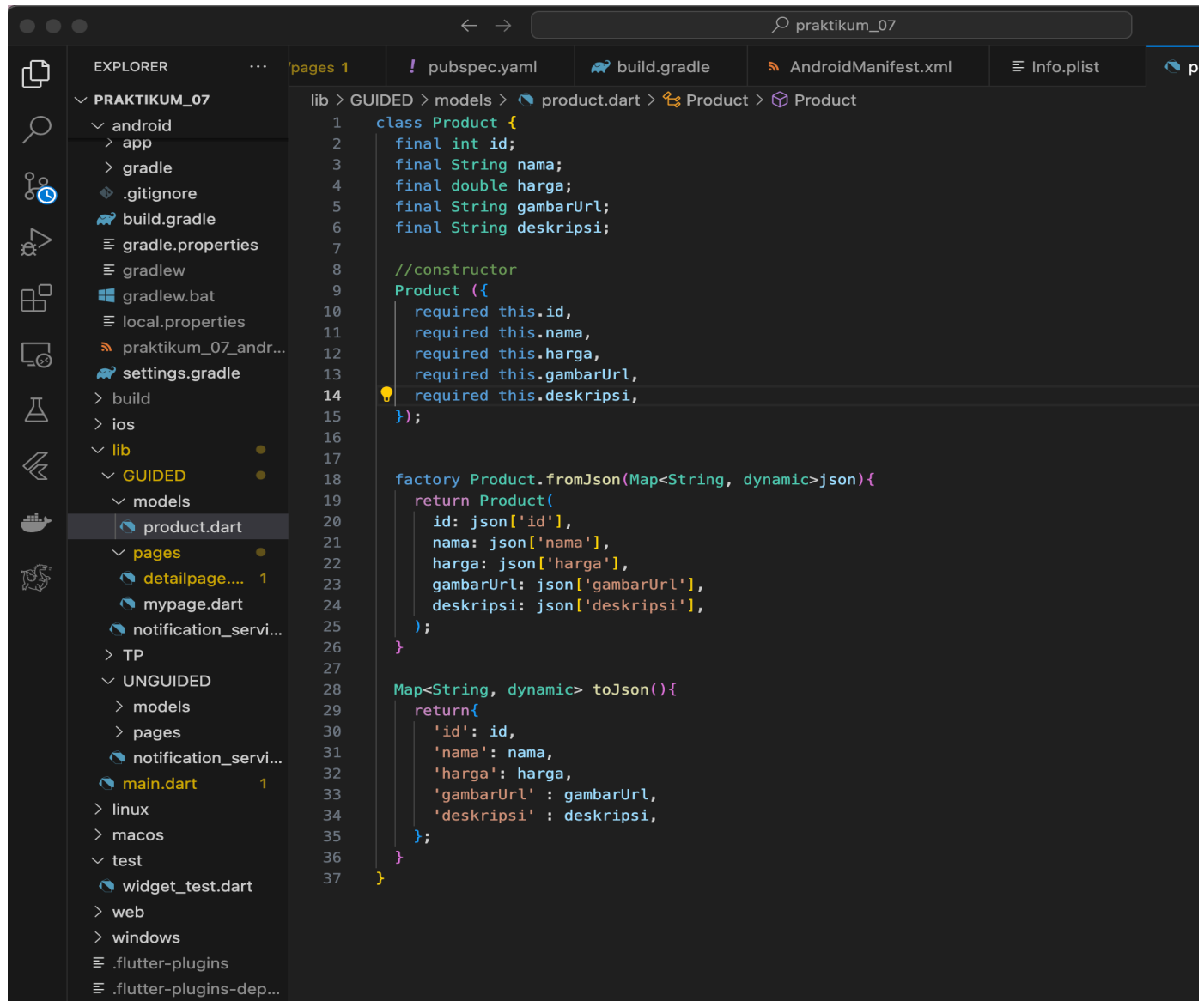
  NewScreen({
    @required this.payload,
  });

  @override
  Widget build(BuildContext context) {return Scaffold(
    appBar: AppBar(
      title: Text(payload),
    ),
  );
}

```

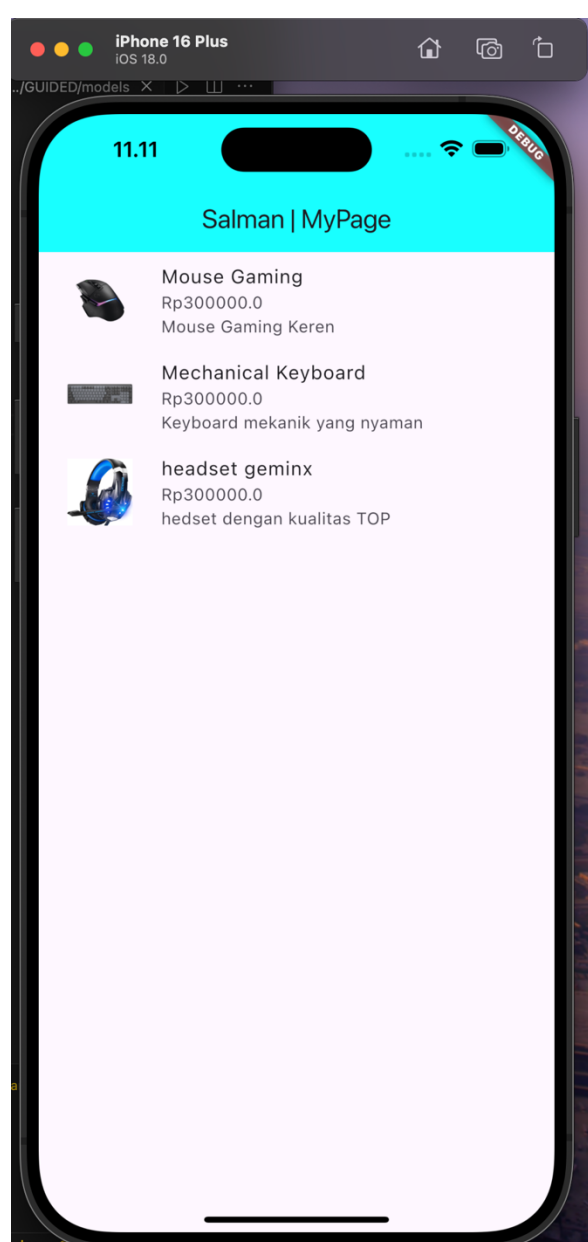
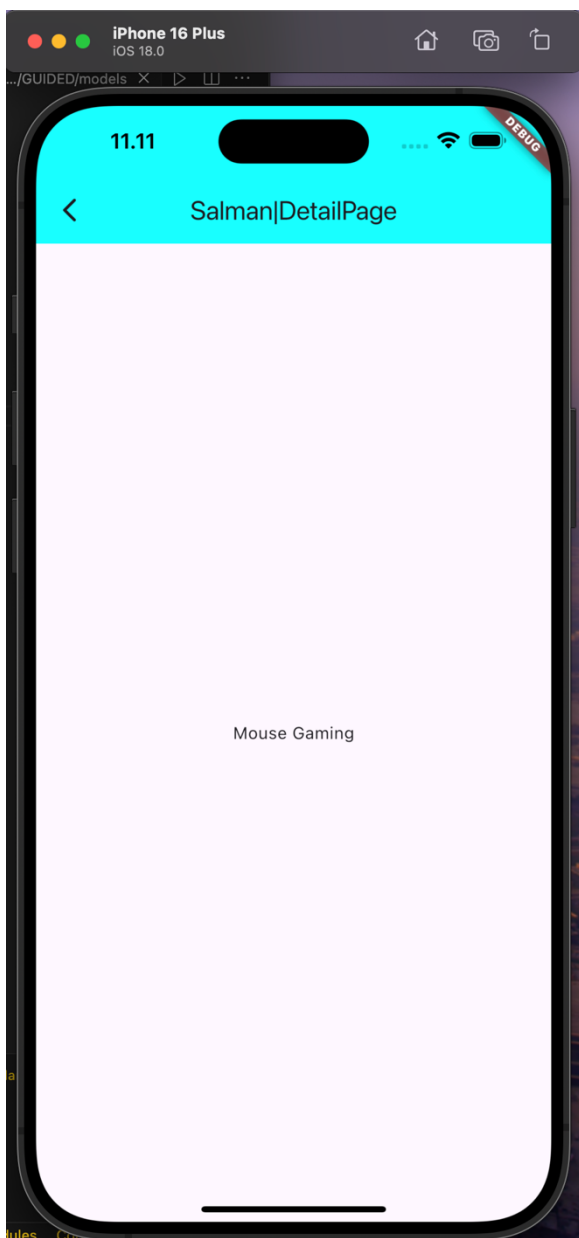
```
}
```

Di atas adalah keseluruhan kode yang digunakan untuk menampilkan sebuah notifikasi sederhana.



The screenshot shows an IDE interface with a file explorer on the left and a code editor on the right. The file explorer shows a project named 'PRAKTIKUM_07' with various files and folders. The code editor displays the following Dart code:

```
lib > GUIDED > models > product.dart > Product > Product
1  class Product {
2      final int id;
3      final String nama;
4      final double harga;
5      final String gambarUrl;
6      final String deskripsi;
7
8      //constructor
9      Product ({
10         required this.id,
11         required this.nama,
12         required this.harga,
13         required this.gambarUrl,
14         required this.deskripsi,
15     });
16
17
18     factory Product.fromJson(Map<String, dynamic> json){
19         return Product(
20             id: json['id'],
21             nama: json['nama'],
22             harga: json['harga'],
23             gambarUrl: json['gambarUrl'],
24             deskripsi: json['deskripsi'],
25         );
26     }
27
28     Map<String, dynamic> toJson(){
29         return{
30             'id': id,
31             'nama': nama,
32             'harga': harga,
33             'gambarUrl' : gambarUrl,
34             'deskripsi' : deskripsi,
35         };
36     }
37 }
```



Tugas Mandiri (Unguided)

1. (Soal) Buatlah satu project untuk menampilkan beberapa produk dan halaman *e-commerce* dengan menerapkan class model serta navigasi halaman.

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

Product.dart

```
class Product {  
  final int id;  
  final String nama;  
  final double harga;  
  final String gambarUrl;  
  final String deskripsi;  
  
  //constructor  
  Product ({
```

```

        required this.id,
        required this.nama,
        required this.harga,
        required this.gambarUrl,
        required this.deskripsi,
    });

    factory Product.fromJson(Map<String, dynamic>json){
        return Product(
            id: json['id'],
            nama: json['nama'],
            harga: json['harga'],
            gambarUrl: json['gambarUrl'],
            deskripsi: json['deskripsi'],
        );
    }

    Map<String, dynamic> toJson(){
        return{
            'id': id,
            'nama': nama,
            'harga': harga,
            'gambarUrl' : gambarUrl,
            'deskripsi' : deskripsi,
        };
    }
}

```

Detailpage.dart

```

import 'package:flutter/material.dart';
import 'package:praktikum_07/UNGUIDED/models/product.dart'; // Pastikan impor ini
konsisten

class DetailUnguided extends StatelessWidget {
    final Product product; // Properti untuk menyimpan objek Product

    const DetailUnguided({super.key, required this.product}); // Konstruktor

    @override
    Widget build(BuildContext context) {
        return Scaffold(
            appBar: AppBar(
                title: Text(product.nama),
            ),
            body: Padding(
                padding: const EdgeInsets.all(16.0),
                child: Column(
                    crossAxisAlignment: CrossAxisAlignment.start,
                    children: [

```

```

        if (product.gambarUrl.isNotEmpty)
          Image.network(product.gambarUrl),
        SizedBox(height: 10),
        Text('Nama: ${product.nama}'),
        SizedBox(height: 10),
        Text('Harga: Rp${product.harga}'),
        SizedBox(height: 10),
        Text('Deskripsi: ${product.deskripsi}'),
      ],
    ),
  ),
);
}
}

```

mypage.dart

```

import 'package:flutter/material.dart';
import 'package:praktikum_07/UNGUIDED/models/product.dart';
import 'package:praktikum_07/UNGUIDED/notification_service.dart';
import 'package:praktikum_07/UNGUIDED/pages/detailpage.dart';
// Impor file notifikasi

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

  final NotificationService notificationService = NotificationService(); // Buat instance dari NotificationService

  final List<Product> products = [
    Product(
      id: 1,
      nama: 'KAWASAKI D-Tracker',
      harga: 35000000,
      gambarUrl:
        'https://content2.kawasaki.com/ContentStorage/KMI/ProductTrimGroup/2020/ea9c6e8-1768-41be-a176-cf3e674c721c.jpg?w=750', // Isi dengan URL gambar sesuai kebutuhan
      deskripsi: 'Motor Trail Berbasis Aspal karena Ban nya di desain untuk aspal dan dibekali tenaga 150cc',
    ),
    Product(
      id: 2,
      nama: 'KAWASAKI Ninja 250',
      harga: 68000000,
      gambarUrl:
        'https://content2.kawasaki.com/ContentStorage/KMI/ProductTrimGroup/2007/6616e8ca-ff48-4631-bdb6-4a09c1c2630d.jpg?w=750', // Isi dengan URL gambar sesuai kebutuhan
      deskripsi: 'Sportbike yang menawarkan performa tinggi dan desain aerodinamis.',
    ),
  ],

```

```

    ),
    Product(
        id: 3,
        nama: 'KAWASAKI Z900',
        harga: 270000000,
        gambarUrl:
'https://content2.kawasaki.com/ContentStorage/KMI/ProductTrimGroup/74/2863a45f-f060-4272-8f3d-78821d361e82.jpg?w=750', // Isi dengan URL gambar sesuai kebutuhan
        deskripsi: 'Motor naked dengan mesin 948cc, ideal untuk pengendara yang menyukai kecepatan.',
    ),
    Product(
        id: 4,
        nama: 'KAWASAKI Versys 650',
        harga: 165000000,
        gambarUrl:
'https://content2.kawasaki.com/ContentStorage/KMI/ProductTrimGroup/34/33d4e683-be1e-4217-aaa1-0754dfcc8f6d.jpg?w=750', // Isi dengan URL gambar sesuai kebutuhan
        deskripsi: 'Touring bike yang nyaman untuk perjalanan jauh dan berkendara di berbagai medan.',
    ),
    Product(
        id: 5,
        nama: 'KAWASAKI W175',
        harga: 30000000,
        gambarUrl:
'https://content2.kawasaki.com/ContentStorage/KMI/ProductTrimGroup/2009/76903c4c-1d25-45d6-8bf1-0aa6d4c3b3a3.jpg?w=750', // Isi dengan URL gambar sesuai kebutuhan
        deskripsi: 'Motor klasik dengan desain retro, cocok untuk berkendara santai di kota.',
    ),
    Product(
        id: 6,
        nama: 'KAWASAKI KLX 150',
        harga: 38000000,
        gambarUrl:
'https://content2.kawasaki.com/ContentStorage/KMI/ProductTrimGroup/2019/9dcd6dc7-eaba-4586-8736-d5c6c96f6416.jpg?w=750', // Isi dengan URL gambar sesuai kebutuhan
        deskripsi: 'Motor off-road yang ringan dan lincah, ideal untuk petualangan di alam terbuka.',
    ),
    Product(
        id: 7,
        nama: 'KAWASAKI Ninja H2',
        harga: 920000000,
        gambarUrl:
'https://content2.kawasaki.com/ContentStorage/KMI/ProductTrimGroup/2030/6278de43-fbd0-4242-bf3a-9d4c0cba72fb.jpg?w=750', // Isi dengan URL gambar sesuai kebutuhan
        deskripsi: 'Motor super sport dengan teknologi supercharged untuk performa maksimal.',
    ),
    Product(

```

```

        id: 8,
        nama: 'KAWASAKI Z125 Pro',
        harga: 40000000,
        gambarUrl:
'https://content2.kawasaki.com/ContentStorage/KMI/ProductTrimGroup/71/303a68e2-b7f0-43f3-b147-eb90f4926a3e.jpg?w=750', // Isi dengan URL gambar sesuai kebutuhan
        deskripsi: 'Mini bike yang menyenangkan untuk berkendara di dalam kota.',
    ),
    Product(
        id: 9,
        nama: 'KAWASAKI KLR 650',
        harga: 130000000,
        gambarUrl:
'https://content2.kawasaki.com/ContentStorage/KMC/ProductTrimGroup/51/b9f7b4db-44bf-4c2f-913f-33216d454abe.jpg?w=750', // Isi dengan URL gambar sesuai kebutuhan
        deskripsi: 'Dual-purpose bike yang siap menjelajahi jalan raya dan off-road.',
    ),
    Product(
        id: 10,
        nama: 'KAWASAKI Ninja 400',
        harga: 70000000,
        gambarUrl:
'https://content2.kawasaki.com/ContentStorage/KMI/ProductTrimGroup/2058/934269dc-4cb2-4840-9a10-d04a7482ebec.jpg?w=750', // Isi dengan URL gambar sesuai kebutuhan
        deskripsi: 'Motor sport ringan dengan performa tinggi untuk pengendara pemula.',
    ),
];

```

```

void _showNotification() {
    notificationService.showNotification(
        0,
        'Notifikasi Baru!',
        'Anda telah melihat produk baru.',
    );
}

```

```

@override
Widget build(BuildContext context) {
    return Scaffold(
        appBar: AppBar(
            title: const Text("Salman | MyPage"),
            backgroundColor: Colors.cyanAccent,
        ),
        body: ListView.builder(
            itemCount: products.length,
            itemBuilder: (context, index) {
                final product = products[index];

                return ListTile(
                    leading: Image.network(
                        product.gambarUrl,

```



```

        width: 70,
        height: 70,
      ),
      title: Text(product.nama),
      subtitle: Column(
        crossAxisAlignment: CrossAxisAlignment.start,
        children: [
          Text('Rp${product.harga}'),
          Text(product.deskripsi),
        ],
      ),
      onTap: () {
        _showNotification(); // Panggil fungsi untuk menampilkan notifikasi saat item ditekan
        Navigator.push(
          context,
          MaterialPageRoute(
            builder: (_) => DetailUnguided(product: product), // Mengirim objek produk
          ),
        );
      },
    ),
  ),
),
);
}
}
}

```

Output code:



Deskripsi Program:

Program ini adalah aplikasi Flutter yang menampilkan daftar item menggunakan `ListView.builder`, dengan setiap item menampilkan ikon, judul, dan membuka dialog detail saat ditekan. Aplikasi dimulai dengan `MyApp`, yang menavigasi ke `MyPageUnguided`, dan `HomeScreen` menampilkan daftar 100 item, masing-masing berisi ikon dan detail. Program ini juga memiliki model `Product` untuk menyimpan data produk, lengkap dengan metode `fromJson` dan `toJson` untuk serialisasi data. Halaman `DetailUnguided` digunakan untuk menampilkan detail produk, termasuk nama, harga, deskripsi, dan gambar, jika tersedia.