

# **LAPORAN JOBSHEET 6**

## **Layout & Navigasi**

Mata Kuliah: Pemrograman Mobile

Dosen Pengampu: Ade Ismail, S.Kom., M.TI.



Disusun Oleh:

Nama: Syava Aprilia P

NIM: 2241760129

Absen: 24

**PROGRAM STUDI SISTEM INFORMASI BISNIS**

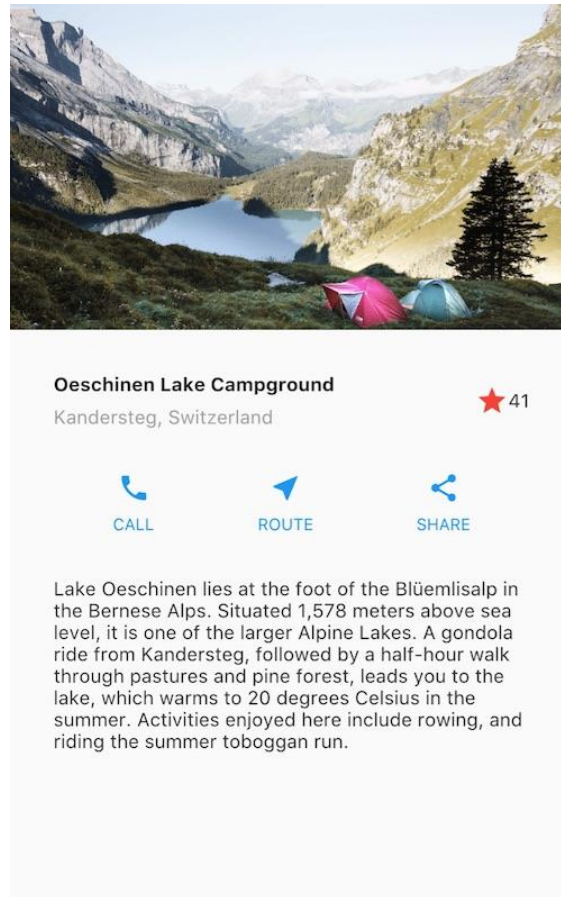
**JURUSAN TEKNOLOGI INFORMASI**

**POLITEKNIK NEGERI MALANG**

**2024**

# Praktikum 1: Membangun Layout di Flutter

Tampilan akhir yang akan Anda buat



## Langkah 1: Buat Project Baru

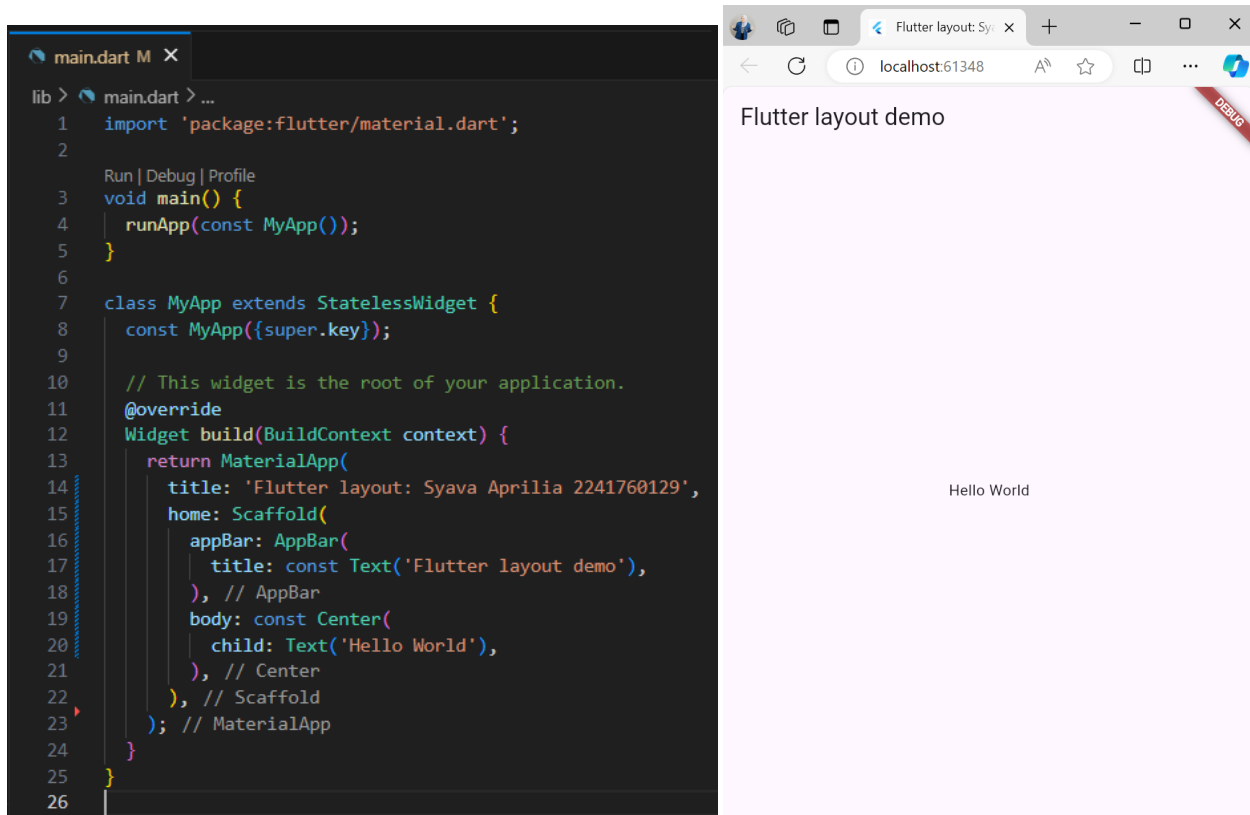
Buatlah sebuah project flutter baru dengan nama **layout\_flutter**. Atau sesuaikan style laporan praktikum yang Anda buat.

```
EXPLORER  ...  main.dart U X
/ LAYOUT_FLUTTER
> .dart_tool
> .idea
> android
> ios
> lib
  main.dart U
> linux
> macos
> test
> web
> windows
> .gitignore
> .metadata
! analysis_options... U
! layout_flutter.iml
! pubspec.lock
! pubspec.yaml
! README.md U

lib > main.dart > ...
1  import 'package:flutter/material.dart';
2
3  Run | Debug | Profile
4  void main() {
5    runApp(const MyApp());
6  }
7
8  class MyApp extends StatelessWidget {
9    const MyApp({super.key});
10
11    // This widget is the root of your application.
12    @override
13    Widget build(BuildContext context) {
14      return MaterialApp(
15        title: 'Flutter Demo',
16        theme: ThemeData(
17          // This is the theme of your application.
18          // TRY THIS: Try running your application with "flutter run". You
19          // the application has a purple toolbar. Then, without quitting
20          // try changing the seedColor in the colorScheme below to Color
21          // and then invoke "Hot reload" from your IDE to see the changes.
```

## Langkah 2: Buka file lib/main.dart

Buka file main.dart lalu ganti dengan kode berikut. Isi nama dan NIM Anda di text title.

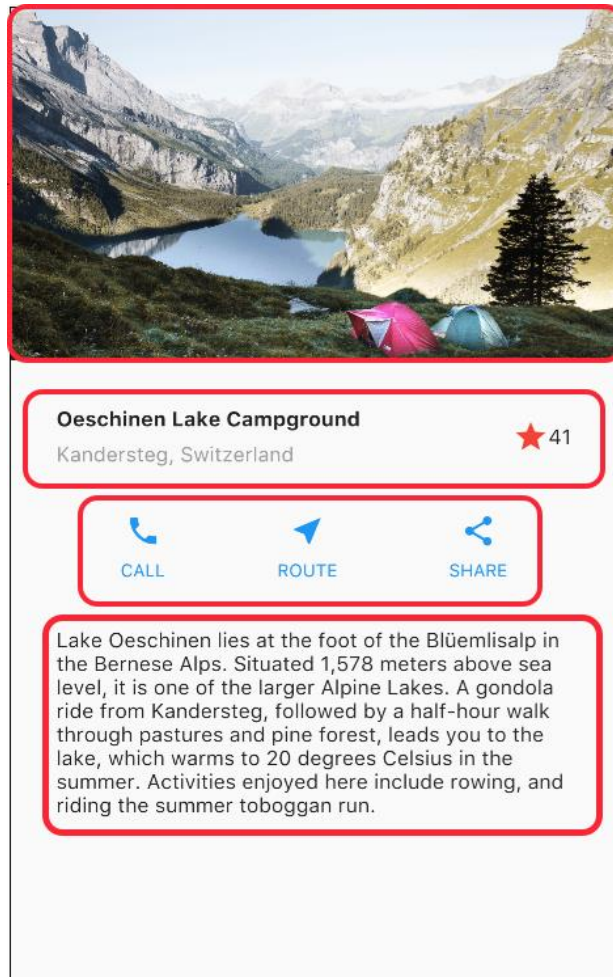


## Langkah 3: Identifikasi layout diagram

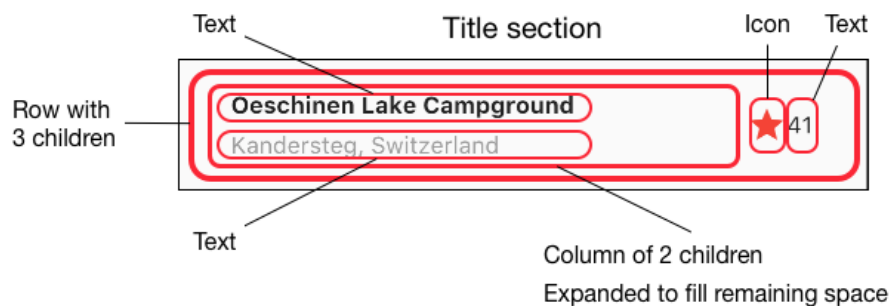
Langkah pertama adalah memecah tata letak menjadi elemen dasarnya:

- Identifikasi baris dan kolom.
- Apakah tata letaknya menyertakan kisi-kisi (grid)?
- Apakah ada elemen yang tumpang tindih?
- Apakah UI memerlukan tab?
- Perhatikan area yang memerlukan alignment, padding, atau borders.

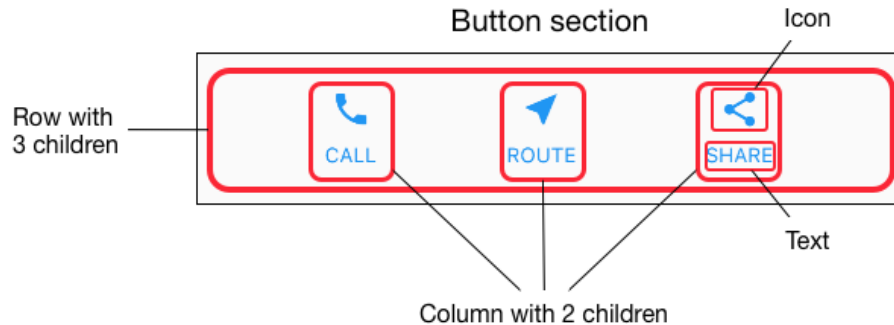
Pertama, identifikasi elemen yang lebih besar. Dalam contoh ini, empat elemen disusun menjadi sebuah kolom: sebuah gambar, dua baris, dan satu blok teks.



Selanjutnya, buat diagram setiap baris. Baris pertama, disebut bagian Judul, memiliki 3 anak: kolom teks, ikon bintang, dan angka. Anak pertamanya, kolom, berisi 2 baris teks. Kolom pertama itu memakan banyak ruang, sehingga harus dibungkus dengan widget yang Diperluas.



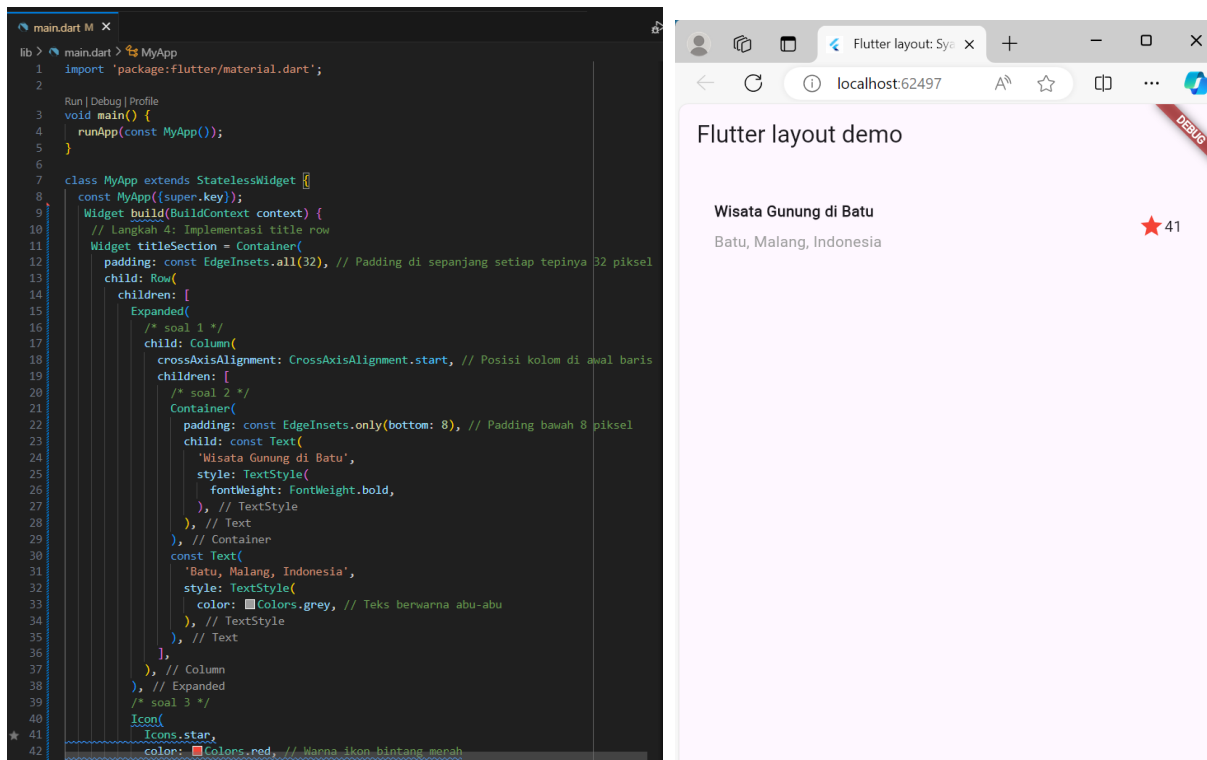
Baris kedua, disebut bagian Tombol, juga memiliki 3 anak: setiap anak merupakan kolom yang berisi ikon dan teks.



Setelah tata letak telah dibuat diagramnya, cara termudah adalah dengan menerapkan pendekatan bottom-up. Untuk meminimalkan kebingungan visual dari kode tata letak yang banyak bertumpuk, tempatkan beberapa implementasi dalam variabel dan fungsi.

#### Langkah 4: Implementasi title row

Pertama, Anda akan membuat kolom bagian kiri pada judul. Tambahkan kode berikut di bagian atas metode build() di dalam kelas MyApp:



## Praktikum 2: Implementasi button row

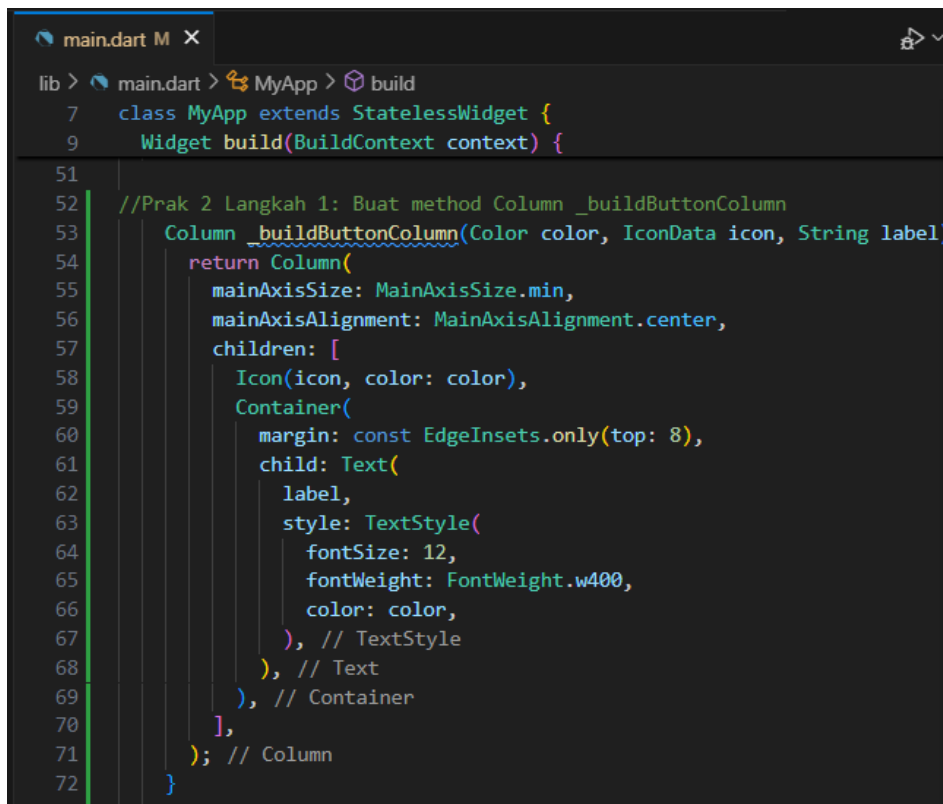
Selesaikan langkah-langkah praktikum berikut ini dengan melanjutkan dari praktikum sebelumnya.

#### Langkah 1: Buat method Column \_buildButtonColumn

Bagian tombol berisi 3 kolom yang menggunakan tata letak yang sama—sebuah ikon di atas baris teks. Kolom pada baris ini diberi jarak yang sama, dan teks serta ikon diberi warna primer.

Karena kode untuk membangun setiap kolom hampir sama, buatlah metode pembantu pribadi bernama `buildButtonColumn()`, yang mempunyai parameter warna, Icon dan Text, sehingga dapat mengembalikan kolom dengan widgetnya sesuai dengan warna tertentu.

#### lib/main.dart (`_buildButtonColumn`)



```
main.dart M x
lib > main.dart > MyApp > build
7 class MyApp extends StatelessWidget {
9   Widget build(BuildContext context) {
51
52   //Prak 2 Langkah 1: Buat method Column _buildButtonColumn
53   Column _buildButtonColumn(Color color, IconData icon, String label)
54   {
55     return Column(
56       mainAxisAlignment: MainAxisAlignment.min,
57       mainAxisAlignment: MainAxisAlignment.center,
58       children: [
59         Icon(icon, color: color),
60         Container(
61           margin: const EdgeInsets.only(top: 8),
62           child: Text(
63             label,
64             style: TextStyle(
65               fontSize: 12,
66               fontWeight: FontWeight.w400,
67               color: color,
68             ), // TextStyle
69           ), // Text
70         ), // Container
71       ],
72     ); // Column
73   }
```

#### Langkah 2: Buat widget `buttonSection`

Buat Fungsi untuk menambahkan ikon langsung ke kolom. Teks berada di dalam Container dengan margin hanya di bagian atas, yang memisahkan teks dari ikon.

Bangun baris yang berisi kolom-kolom ini dengan memanggil fungsi dan set warna, Icon, dan teks khusus melalui parameter ke kolom tersebut. Sejajarkan kolom di sepanjang sumbu utama menggunakan `MainAxisAlignment.spaceEvenly` untuk mengatur ruang kosong secara merata sebelum, di antara, dan setelah setiap kolom. Tambahkan kode berikut tepat di bawah deklarasi `titleSection` di dalam metode `build()`:

#### lib/main.dart (`buttonSection`)

#### Langkah 3: Tambah `button section` ke `body`

Tambahkan variabel `buttonSection` ke dalam `body` seperti berikut:

```

main.dart M X
lib > main.dart > MyApp > build
7   class MyApp extends StatelessWidget {
9     Widget build(BuildContext context) {
53       Column _buildButtonColumn(Color color, IconData icon, String label
71     ); // Column
72   }
73
74   //Prak 2 langkah 2: Buat widget buttonSection
75   Color color = Theme.of(context).primaryColor;
76
77   Widget buttonSection = Row(
78     mainAxisAlignment: MainAxisAlignment.spaceEvenly,
79     children: [
80       _buildButtonColumn(color, Icons.call, 'CALL'),
81       _buildButtonColumn(color, Icons.near_me, 'ROUTE'),
82       _buildButtonColumn(color, Icons.share, 'SHARE'),
83     ],
84   );
85
86   return MaterialApp(
87     title: 'Flutter layout: Syava Aprilia - 2241760129', // Nama dan
88     home: Scaffold(

```

## Praktikum 3: Implementasi text section

Selesaikan langkah-langkah praktikum berikut ini dengan melanjutkan dari praktikum sebelumnya.

### Langkah 1: Buat widget textSection

Tentukan bagian teks sebagai variabel. Masukkan teks ke dalam Container dan tambahkan padding di sepanjang setiap tepinya. Tambahkan kode berikut tepat di bawah deklarasi buttonSection:

```

lib > main.dart > MyApp > build
7   class MyApp extends StatelessWidget {
9     Widget build(BuildContext context) {
86   //Prak 3 langkah 1: Buat widget textSection
87   Widget textSection = Container(
88     padding: const EdgeInsets.all(32),
89     child: const Text(
90       'Pura Luhur Uluwatu terletak di Desa Pecatu, Kuta Selatan, Badung, Bali. '
91       'Pura ini berdiri di atas tebing setinggi 97 meter yang menjorok ke laut dan merupakan salah satu Pura Sad Kayangan, dipercaya sebagai penyangga 9 mata angin oleh umat Hindu. '
92       'Awalnya, pura ini digunakan untuk memuja Empu Kuturan, seorang pendeta suci abad ke-11, dan kemudian Dang Hyang Nirartha yang mencapai moksa di sini, yang menginspirasi nama "Luhur Uluwatu." '
93       'Selain menjadi tempat ibadah, Pura Uluwatu juga terkenal dengan Pantai Pecatu di bawahnya, yang menjadi destinasi selancar internasional.'
94       'Syava Aprilia 2241760129',
95       softWrap: true,
96     ), // Text
97   ); // Container
98

```

Dengan memberi nilai softWrap = true, baris teks akan memenuhi lebar kolom sebelum membungkusnya pada batas kata.

### Langkah 2: Tambahkan variabel text section ke body

Tambahkan widget variabel textSection ke dalam body seperti berikut:

```

100   return MaterialApp(
101     title: 'Flutter layout: Syava Aprilia - 2241760129', // Nama da
102     home: Scaffold(
103       appBar: AppBar(
104         title: const Text('Flutter layout demo'),
105       ), // AppBar
106       body: Center(
107         child: Column(
108           children: [
109             titleSection, // Bagian judul
110             buttonSection, // Bagian tombol
111             textSection, // Bagian teks
112           ],
113         ), // Column
114       ), // Center
115     ),

```



## Praktikum 4: Implementasi image section

Selesaikan langkah-langkah praktikum berikut ini dengan melanjutkan dari praktikum sebelumnya.

### Langkah 1: Siapkan aset gambar



Anda dapat mencari gambar di internet yang ingin ditampilkan. Buatlah folder images di root project **layout\_flutter**. Masukkan file gambar tersebut ke folder images, lalu set nama file tersebut ke file pubspec.yaml seperti berikut:

```
59 uses-material-design: true
60 assets:
61 - images/bali.jpeg
62
```

## Langkah 2: Tambahkan gambar ke body

Tambahkan aset gambar ke dalam body seperti berikut:

```
lib > main.dart > MyApp > build
7 class MyApp extends StatelessWidget {
9   Widget build(BuildContext context) {
100    ); // Container
101    return MaterialApp(
102      title: 'Flutter layout: Syava Aprilia - 2241760129',
103      home: Scaffold(
104        appBar: AppBar(
105          title: const Text('Flutter layout demo'),
106        ), // AppBar
107        body: Center(
108          child: Column(
109            children: [
110              Image.asset(
111                'images/bali.jpeg',
112                width: 600,
113                height: 240,
114                fit: BoxFit.cover,
115              ), // Image.asset
116              titleSection, // Bagian judul
117              buttonSection, // Bagian tombol
118              textSection, // Bagian teks
119            ],
120          ), // Column
121        ), // Center
122      ), // Scaffold
123    ); // MaterialApp
124  }
```

BoxFit.cover memberi tahu kerangka kerja bahwa gambar harus sekecil mungkin tetapi menutupi seluruh kotak rendernya.


## Langkah 3: Terakhir, ubah menjadi ListView

Pada langkah terakhir ini, atur semua elemen dalam ListView, bukan Column, karena ListView mendukung scroll yang dinamis saat aplikasi dijalankan pada perangkat yang resolusinya lebih kecil.

```




99   return MaterialApp(
100     title: 'Flutter layout: Syava Aprilia - 2241760129',
101     home: Scaffold(
102       appBar: AppBar(
103         title: const Text('Flutter layout demo'),
104       ), // AppBar
105       body: ListView(
106         children: [
107           Image.asset(
108             'images/bali.jpeg',
109             width: 600,
110             height: 270,
111             fit: BoxFit.cover,
112           ), // Image.asset
113           titleSection, // Bagian judul
114           buttonSection, // Bagian tombol
115           textSection, // Bagian teks
116         ],
117       ), // ListView
118     ), // Scaffold
119   ); // MaterialApp
120 }
121

```



### Pura Uluwatu Bali

Pecatu, Kec. Kuta Sel,Kab. Badung, Bali, Indonesia

 CALL
  ROUTE
  SHARE

Pura Luhur Uluwatu terletak di Desa Pecatu, Kuta Selatan, Badung, Bali. Pura ini berdiri di atas tebing setinggi 97 meter yang menjorok ke laut dan merupakan salah satu Pura Sad Kayangan, dipercaya sebagai penyangga 9 mata angin oleh umat Hindu. Awalnya, pura ini digunakan untuk memuja Empu Kuturan, seorang pendeta suci abad ke-11, dan kemudian Dang Hyang Nirartha yang mencapai moksa di sini, yang menginspirasi nama "Luhur Uluwatu." Selain menjadi tempat ibadah, Pura Uluwatu juga terkenal dengan Pantai Pecatu di bawahnya, yang menjadi destinasi selancar internasional.Syava Aprilia 2241760129

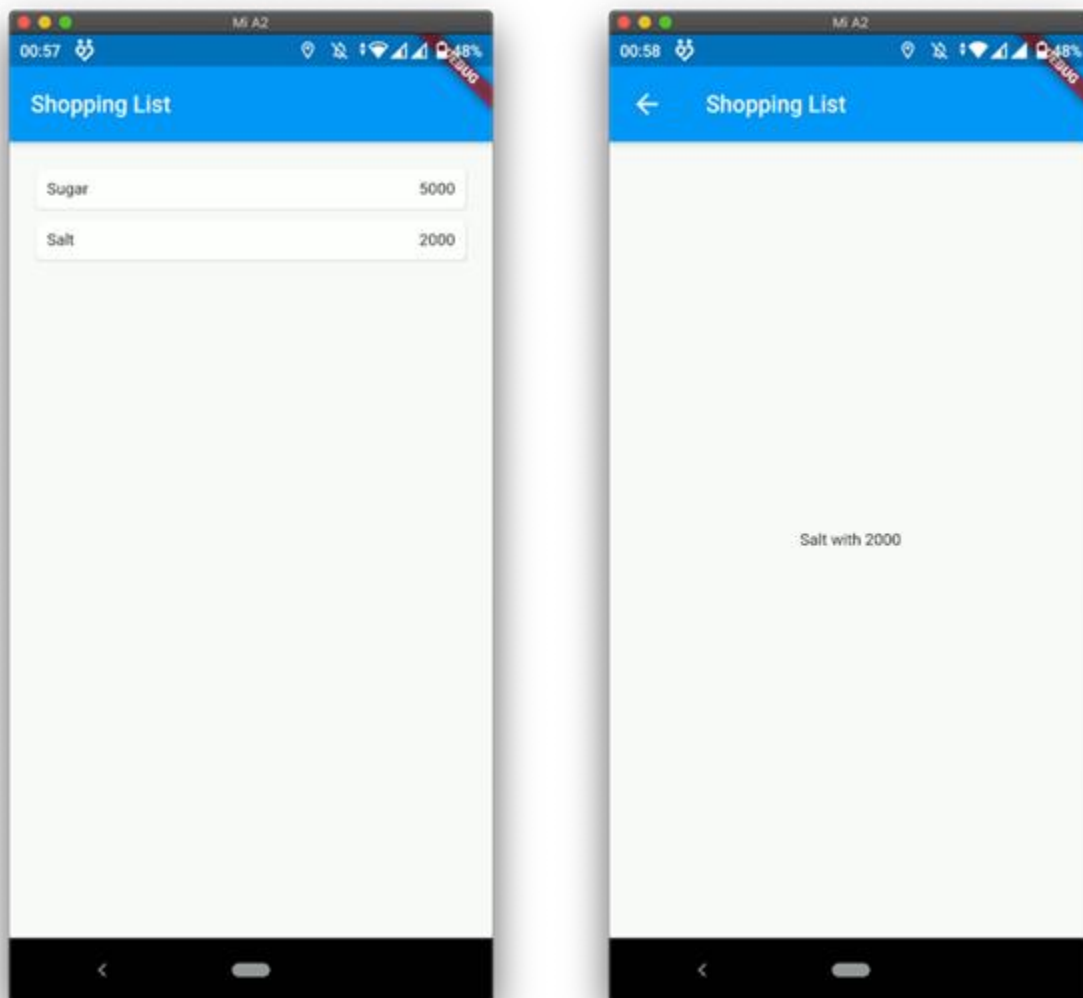
## Praktikum 5: Membangun Navigasi di Flutter

### Apa yang akan Anda pelajari

- Cara kerja mekanisme navigation dan route di Flutter.
- Cara membuat navigation dan route di Flutter.

Selesaikan langkah-langkah praktikum berikut ini menggunakan editor Visual Studio Code (VS Code) atau Android Studio atau code editor lain kesukaan Anda. Materi ini dapat dimasukkan ke Laporan Praktikum folder **Week** atau **Pertemuan 06**.

### Tampilan akhir yang akan Anda buat

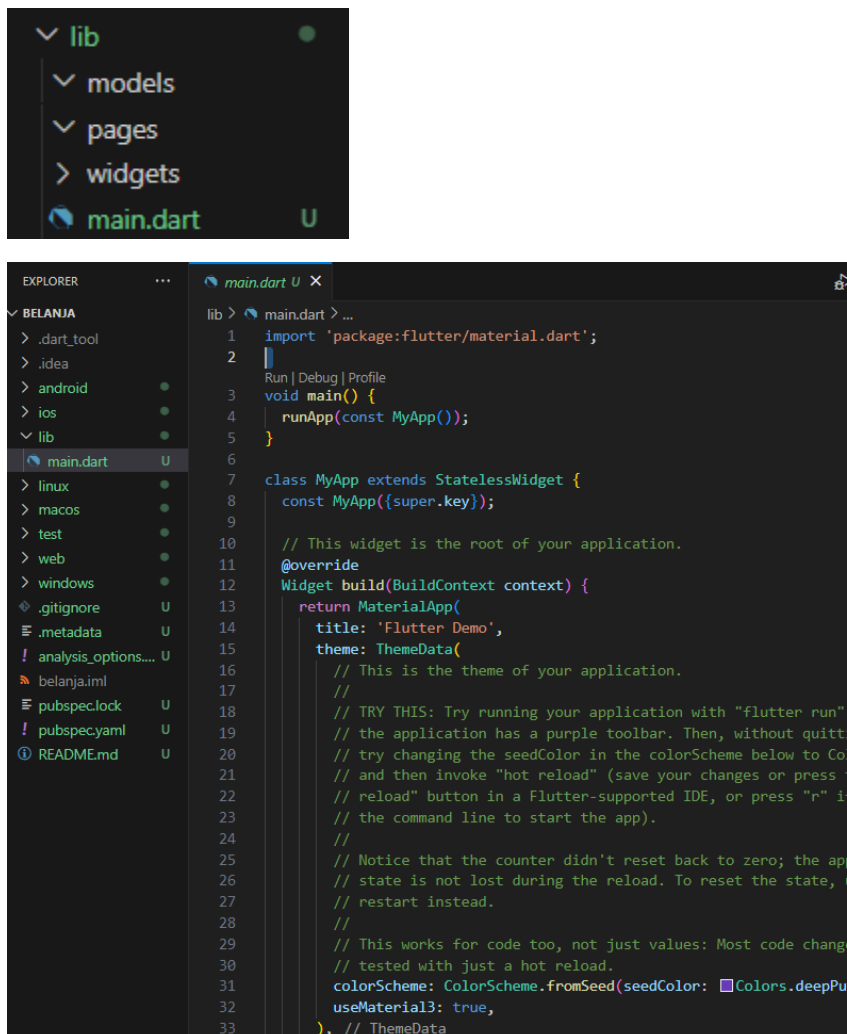


Pada praktikum 5 ini anda akan belajar mengenai pembangunan aplikasi bergerak multi halaman. Aplikasi yang dikembangkan berupa kasus daftar barang belanja. Pada aplikasi ini anda akan belajar untuk berpindah halaman dan mengirimkan data ke halaman lainnya. Gambaran mockup hasil akhir

aplikasi dapat anda lihat pada gambar di atas (mockup dibuat sederhana, sehingga Anda mempunyai banyak ruang untuk berkreasi). Desain aplikasi menampilkan sebuah ListView widget yang datanya bersumber dari List. Ketika item ditekan, data akan dikirimkan ke halaman berikutnya.

### Langkah 1: Siapkan project baru

Sebelum melanjutkan praktikum, buatlah sebuah project baru Flutter dengan nama **belanja** dan susunan folder seperti pada gambar berikut. Penyusunan ini dimaksudkan untuk mengorganisasi kode dan widget yang lebih mudah.



### Langkah 2: Mendefinisikan Route

Buatlah dua buah file dart dengan nama home\_page.dart dan item\_page.dart pada folder **pages**. Untuk masing-masing file, deklarasikan class HomePage pada file home\_page.dart dan ItemPage pada item\_page.dart. Turunkan class dari StatelessWidget. Gambaran potongan kode dapat anda lihat sebagai berikut.

### Langkah 3: Lengkapi Kode di main.dart

Setelah kedua halaman telah dibuat dan didefinisikan, bukalah file main.dart. Pada langkah ini anda akan mendefinisikan **Route** untuk kedua halaman tersebut. Definisi penamaan **route** harus bersifat **unique**. Halaman **HomePage** didefinisikan sebagai **/**. Dan halaman **ItemPage** didefinisikan sebagai **/item**. Untuk mendefinisikan halaman awal, anda dapat menggunakan named argument **initialRoute**. Gambaran tahapan ini, dapat anda lihat pada potongan kode berikut.

## Tugas Praktikum 2

1. Untuk melakukan pengiriman data ke halaman berikutnya, cukup menambahkan informasi arguments pada penggunaan Navigator. Perbarui kode pada bagian Navigator menjadi seperti berikut.

```
// Navigasi ke halaman ItemPage dengan mengirim data item
Navigator.pushNamed(
  context,
  '/item',
  arguments: item, // Kirim data item sebagai argument
);
},
child: Card(
```

2. Pembacaan nilai yang dikirimkan pada halaman sebelumnya dapat dilakukan menggunakan ModalRoute. Tambahkan kode berikut pada blok fungsi build dalam halaman ItemPage. Setelah nilai didapatkan, anda dapat menggunakannya seperti penggunaan variabel pada umumnya.  
(<https://docs.flutter.dev/cookbook/navigation/navigate-with-arguments>)

```
final itemArgs = ModalRoute.of(context)?.settings.arguments as Item;
```

Main.dart

```
lib > main.dart M x item.dart U home_page.dart 1, U item_page.dart U
1 import 'package:flutter/material.dart';
2 import 'package:belanja/pages/home_page.dart';
3 import 'package:belanja/pages/item_page.dart';
4 import 'package:belanja/models/item.dart';
5
Run | Debug | Profile
6 void main() {
7   runApp(MaterialApp(
8     initialRoute: '/',
9     onGenerateRoute: (settings) {
10      if (settings.name == '/item') {
11        final item = settings.arguments as Item; // Ambil argumen item
12        return MaterialPageRoute(
13          builder: (context) => ItemPage(item: item), // Kirim item ke ItemPage
14        ); // MaterialPageRoute
15      }
16      return null;
17    },
18    routes: {
19      '/': (context) => HomePage(),
20    },
21  )); // MaterialApp
22 }
```

Item.dart

```
main.dart M  item.dart U  home_page.dart 1, U  item_page.dart U

lib > models > item.dart > ...
1 class Item {
2   String name; // Nama item
3   double price; // Harga item
4
5   Item({required this.name, required this.price});
6 }
7
```

## Home\_page.dart

```
main.dart M  item.dart U  home_page.dart 1, U  item_page.dart U

lib > pages > home_page.dart > HomePage > build
1 import 'package:flutter/material.dart';
2 import 'package:belanja/models/item.dart';
3 import 'package:belanja/pages/item_page.dart';
4
5 class HomePage extends StatelessWidget {
6   final List<Item> items = [
7     Item(name: 'Sugar', price: 5000),
8     Item(name: 'Salt', price: 2000),
9   ];
10
11   Widget build(BuildContext context) {
12     return Scaffold(
13       appBar: AppBar(
14         title: Text('Shopping List'),
15       ), // AppBar
16       body: Container(
17         margin: EdgeInsets.all(8), // Memberikan margin di sekitar ListView
18         child: ListView.builder(
19           padding: EdgeInsets.all(8), // Padding untuk item di dalam ListView
20           itemCount: items.length,
21           itemBuilder: (context, index) {
22             final item = items[index];
23             return InkWell(
24               onTap: () {
25                 // Navigasi ke halaman ItemPage dengan mengirim data item
26                 Navigator.pushNamed(
27                   context,
28                   '/item',
29                   arguments: item, // Kirim data item sebagai argument

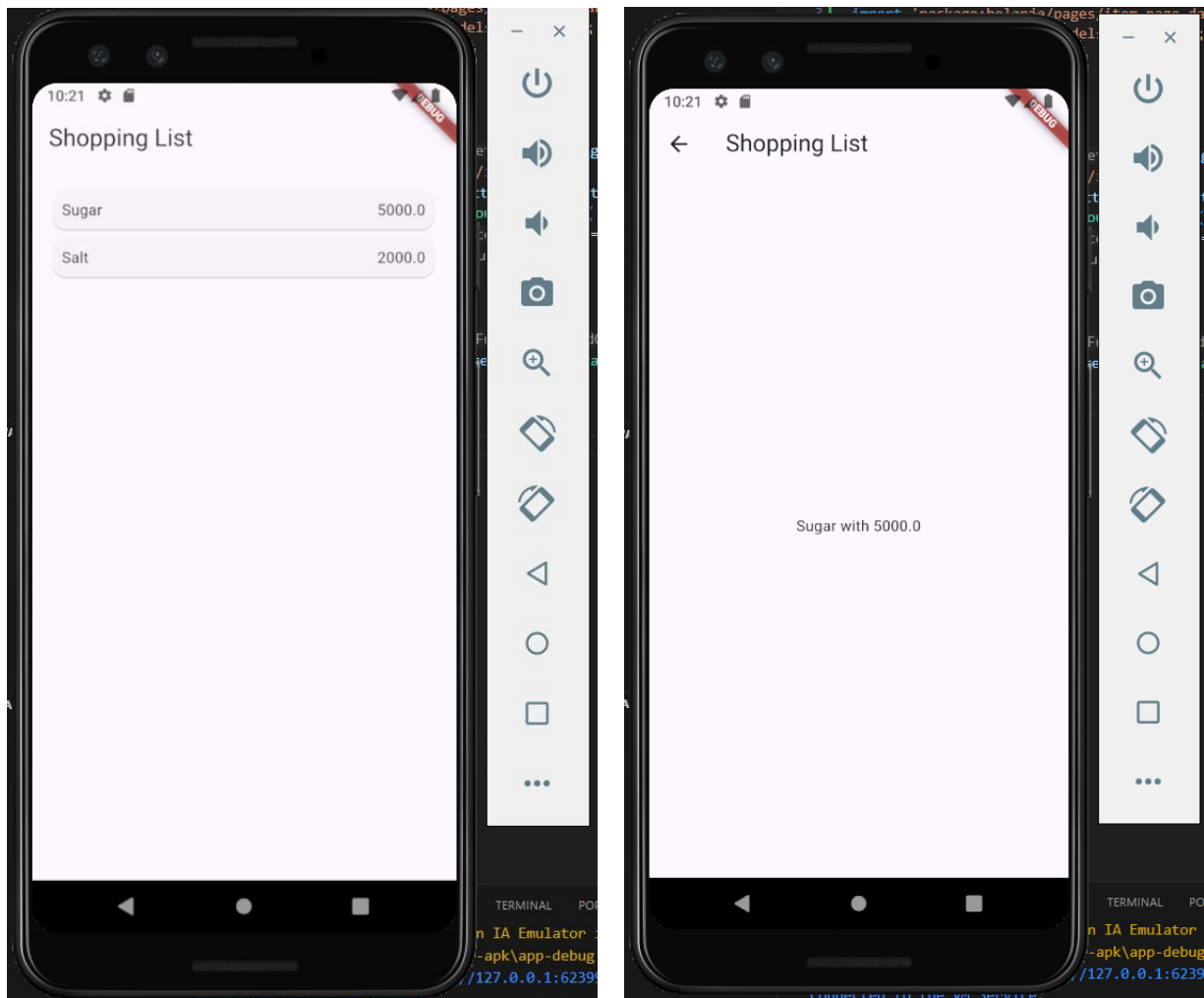
```

## Item\_page.dart

```
main.dart M  item.dart U  home_page.dart 1, U  item_page.dart U

lib > pages > item_page.dart > ItemPage > build
1 import 'package:flutter/material.dart';
2 import 'package:belanja/models/item.dart'; // Pastikan model Item diimport
3 class ItemPage extends StatelessWidget {
4   final Item item; // Parameter untuk menerima data item
5   // Constructor untuk menerima data item
6   ItemPage({required this.item});
7   @override
8   Widget build(BuildContext context) {
9     return Scaffold(
10       appBar: AppBar(
11         title: Text('Shopping List'), // Menampilkan nama item di AppBar
12       ), // AppBar
13       body: Center(
14         child: Column(
15           mainAxisAlignment: MainAxisAlignment.center,
16           children: [
17             Text('${item.name} with ${item.price}'),
18             //Text('Item: ${item.name} with Price: Rp${item.price}'), // Menampilkan nama item
19             //Text('Price: Rp${item.price}'), // Menampilkan harga item
20           ],
21         ), // Column
22       ), // Center
23     ); // Scaffold
24
25 }
```

Hasil



3. Pada hasil akhir dari aplikasi **belanja** yang telah anda selesaikan, tambahkan atribut foto produk, stok, dan rating. Ubahlah tampilan menjadi GridView seperti di aplikasi marketplace pada umumnya.

```
main.dart  item.dart M  home_page.dart 1, M  item_page.dart M
lib > models > item.dart > ...
1  class Item {
2    String name; // Nama item
3    double price; // Harga item
4    String imageUrl; // URL gambar produk
5    int stock; // Stok barang
6    double rating; // Rating produk
7
8    Item({
9      required this.name,
10     required this.price,
11     required this.imageUrl,
12     required this.stock,
13     required this.rating,
14   });
15 }
16
```

```
main.dart  item.dart M  home_page.dart 1, M X  item_page.dart M X
lib > pages > home_page.dart > HomePage > items
1 import 'package:flutter/material.dart';
2 import 'package:belanja/models/item.dart';
3 import 'package:belanja/pages/item_page.dart';
4
5 class HomePage extends StatelessWidget {
6   final List<Item> items = [
7     Item(
8       name: 'Sugar',
9       price: 5000,
10      imageUrl: 'https://via.placeholder.com/150',
11      stock: 10,
12      rating: 4.5,
13    ),
14    Item(
15      name: 'Salt',
16      price: 2000,
17      imageUrl: 'https://via.placeholder.com/150',
18      stock: 20,
19      rating: 4.2,
20    ),
21  ];
22
23  @override
24  Widget build(BuildContext context) {
25    return Scaffold(
26      appBar: AppBar(
27        title: Text('Shopping List'),
28      ), // AppBar
29      body: GridView.builder(
30        padding: EdgeInsets.all(8),
31        gridDelegate: SliverGridDelegateWithFixedCrossAxisCount(
32          crossAxisCount: 2,
33          crossAxisSpacing: 8,
34          mainAxisSpacing: 8,
35        ), // SliverGridDelegateWithFixedCrossAxisCount
36        itemCount: items.length,
37        itemBuilder: (context, index) {
38          final item = items[index];
```

```
main.dart  item.dart M  home_page.dart 1, M  item_page.dart M X
lib > pages > item_page.dart > ItemPage
1 import 'package:flutter/material.dart';
2 import 'package:belanja/models/item.dart'; // Pastikan model Item diimport
3 class ItemPage extends StatelessWidget {
4   final Item item; // Parameter untuk menerima data item
5   // Constructor untuk menerima data item
6   ItemPage({required this.item});
7   @override
8   Widget build(BuildContext context) {
9     return Scaffold(
10      appBar: AppBar(
11        title: Text('Detail ${item.name}'),
12      ), // AppBar
13      body: Center(
14        child: Column(
15          mainAxisAlignment: MainAxisAlignment.center,
16          children: [
17            Hero(
18              tag: item.name,
19              child: Image.network(item.imageUrl, height: 200, width: 200),
20            ), // Hero
21            SizedBox(height: 20),
22            Text('Name: ${item.name}', style: TextStyle(fontSize: 24)),
23            Text('Price: Rp${item.price}', style: TextStyle(fontSize: 24)),
24            Text('Stock: ${item.stock}', style: TextStyle(fontSize: 24)),
25            Text('Rating: ${item.rating}', style: TextStyle(fontSize: 24)),
26          ],
27        ), // Column
28      ), // Center
29    ); // Scaffold
30  }
31
32  // @override
```

4. Silakan implementasikan Hero widget pada aplikasi **belanja** Anda dengan mempelajari dari sumber ini: <https://docs.flutter.dev/cookbook/navigation/hero-animations>
5. Sesuaikan dan modifikasi tampilan sehingga menjadi aplikasi yang menarik. Selain itu, pecah widget menjadi kode yang lebih kecil. Tambahkan **Nama** dan **NIM** di footer aplikasi **belanja** Anda.



```
main.dart  item.dart M  home_page.dart 1, M  X  item_page.dart M
lib > pages > home_page.dart > HomePage > items
5  class HomePage extends StatelessWidget {
24  Widget build(BuildContext context) {
56      Text('Rating: ${item.rating}'),
57      ],
58      ), // Column
59      ), // Card
60      ), // Hero
61      ); // GestureDetector
62  },
63  ), // GridView.builder
64  bottomNavigationBar: Container(
65      padding: EdgeInsets.all(8),
66      color: Colors.grey[200],
67      child: Text('Syava Aprilia | NIM 2241760129'),
68  ), // Container
69  ); // Scaffold
70  }
71  }
72  // Item(name: "Sugar", price: 5000),
```

6. Selesaikan Praktikum 5: Navigasi dan Rute tersebut. Cobalah modifikasi menggunakan plugin [go\\_router](#), lalu dokumentasikan dan push ke repository Anda berupa screenshot setiap hasil pekerjaan beserta penjelasannya di file README.md. Kumpulkan link commit repository GitHub Anda kepada dosen yang telah disepakati!

