

Modul 2

Pengenalan Layout

Grid, Column, Row, Container dan Navigator

Module Overview

Flutter sebagai kerangka kerja memiliki sistem layouting yang lengkap untuk menyusun widget pada screen atau page. Layout pada flutter dapat disusun menggunakan widget Grid, Column, Row, dan Container, kemudian diakhir sesi akan ada pengenalan Navigator untuk mengarahkan pengguna dari satu halaman ke halaman lainnya.

Module Objectives

Setelah mempelajari dan mempraktikkan modul ini, mahasiswa diharapkan dapat:

- Memahami konsep Layout pada flutter
- Memahami struktur layout pada kumpulan widget flutter
- Menerapkan konsep layout pada flutter
- Memprogram layout menggunakan Grid, Column, Row dan Container.

Layout pada Flutter

Di Flutter, hampir semuanya adalah widget—bahkan model tata letak (layout) adalah widget. Widget Gambar (Image), ikon (Icon), dan teks (Text) pada aplikasi Flutter semuanya adalah widget. Bahkan baris (Row), kolom (Column), dan Grid, Constrain, dan Align adalah "widget yang terlihat".

Karena konsep utama Flutter adalah widget, maka Flutter menggabungkan pengguna fungsionalitas tata letak antarmuka ke dalam widget itu sendiri. Flutter menyediakan cukup banyak widget khusus yang dirancang seperti Container, Center, Align, Column, Row, Grid dll., yang bertujuan mengatur tata letak (layout) antarmuka pengguna. Widget yang dibuat dengan menyusun widget lain biasanya menggunakan widget tata letak (layout).

Tipe Widget Layout:

Layout dapat dikelompokkan ke dalam dua kategori berbeda berdasarkan turunannya:

- Single child Widget
- Multiple child Widget

Single Child Widget (Child):

Dalam kategori ini, widget hanya akan memiliki satu widget sebagai turunannya dan setiap widget akan memiliki fungsi tata letak khusus. Misalnya, widget **Center** hanya mengatur letak ke tengah widget child-nya dari widget induknya (parent widget). Selain itu, widget **Container** memberikan fleksibilitas untuk menempatkan child-nya

di tempat mana pun di dalam penampung-nya, seperti mengatur padding, margin, dan melakukan dekorasi, dll.,

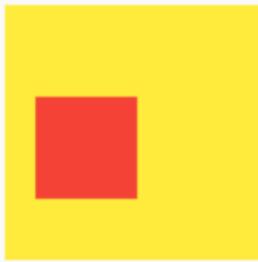
Contoh:

```
Center(
    child: Container(
        height: 100.0,
        width: 100.0,
        color: Colors.yellow,
        child: Align(
            alignment: FractionalOffset(0.2, 0.6),
            child: Container(
                height: 40.0,
                width: 40.0,
                color: Colors.red,
            ),
        ),
    ),
)
```

Keterangan:

- Widget **Center** adalah root widget yang menampung 1 (satu) child yaitu container. Widget ini mengatur posisi child-nya berada di tengah dari penampung (contoh: Posisi tengah dari body scaffold)
- Widget **Container** adalah penampung widget child dengan memberi pengaturan ukuran lebar, tinggi, warna atau **decoration** lainnya. Gunakan **Container** saat ingin menambahkan padding, margin, border, atau warna latar belakang, dan beberapa dekorasi lainnya.
- Widget **Align** mengatur posisi dari childnya berdasarkan posisi X dan Y dari induknya.

Result:



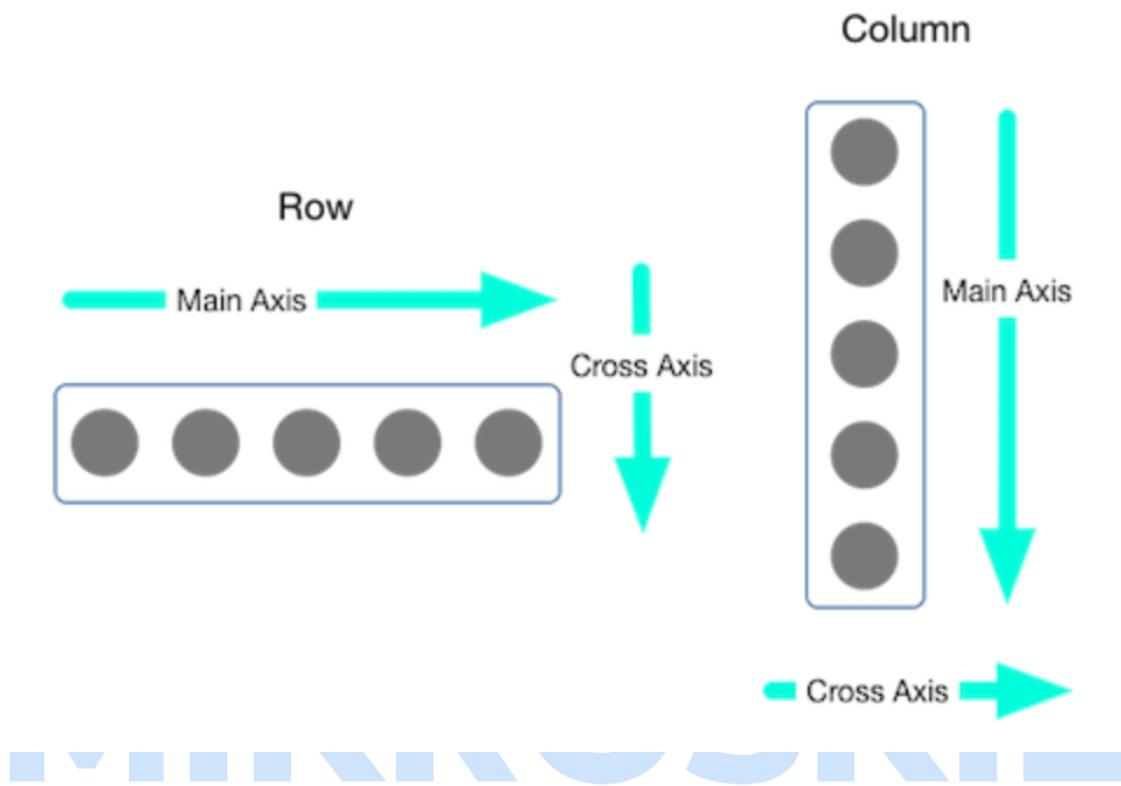
Multiple Child Widget (Children):

Dalam kategori, widget memiliki lebih dari 1 (satu) widget child dan tata letak yang unik (dapat di atur) dari setiap widget childnya. Contoh, Widget **Row** dapat mengatur letak widget children secara horizontal, begitu juga dengan widget **Column** yang mengatur tata letak children secara vertikal. Berikut beberapa multiple child Widget yang sering digunakan:

- Row - mengatur tata letak widget children-nya secara horizontal
- Column - mengatur tata letak widget children-nya secara vertikal
- Listview - mengatur tata letak widget children-nya sebagai list (horizontal / vertikal)
- Gridview - mengatur tata letak widget children-nya sebagai gallery
- Expanded - digunakan untuk membuat widget children Row dan Column untuk menempati daerah semaksimal mungkin.
- Table - widget berbentuk table
- Flow - widget berdasarkan flow
- Stack - widget berdasarkan tumpukan.

Aligning widgets:

Pada multiple widget terdapat kontrol bagaimana baris atau kolom menyelaraskan children-nya menggunakan properti **mainAxisAlignment** dan **crossAxisAlignment**. Untuk **Row**, sumbu utama berjalan horizontal dan sumbu silang berjalan vertikal. Untuk **Column**, sumbu utama berjalan secara vertikal dan sumbu silang berjalan secara horizontal.



1. Row

Mengatur tata letak widget children-nya secara baris.

Contoh:

```
Row(  
    mainAxisAlignment: MainAxisAlignment.spaceEvenly,  
    children: [  
        Image.asset('assets/images/pic1.jpg'),  
        Image.asset('assets/images/pic2.jpg'),  
        Image.asset('assets/images/pic3.jpg'),  
    ],  
)
```

Keterangan:

- *MainAxisAlignment.spaceEvenly*, memberi jarak antar, sebelum, dan sesudah children-nya.
- Children menampung 3 child yaitu Widget Image yang menampilkan asset gambar.

Hasil:



2. Column

Sama seperti Row, Column mengatur tata letak children-nya secara kolom.

Contoh:

```
Column(  
    mainAxisAlignment: MainAxisAlignment.spaceEvenly,  
    children: [  
        Image.asset('images/pic1.jpg'),  
        Image.asset('images/pic2.jpg'),  
        Image.asset('images/pic3.jpg'),  
    ],  
)
```

Hasil:



VERSITAS
ROSKIL

3. GridView

Widget ini mengatur letak widget dalam bentuk kotak / gallery. Widget ini memungkinkan untuk mengatur otomatis kontennya / children dalam bentuk column sehingga memungkinkan untuk scrolling secara horizontal atau vertikal (tergantung pengaturan *direction*-nya).

Terdapat 2 jenis GridView:

- `GridView.count` - dapat mengatur total kolom yang dibutuhkan.
- `GridView.extent` - dapat ukuran width (lebar) spesifik dari konten / children-nya.

Contoh:

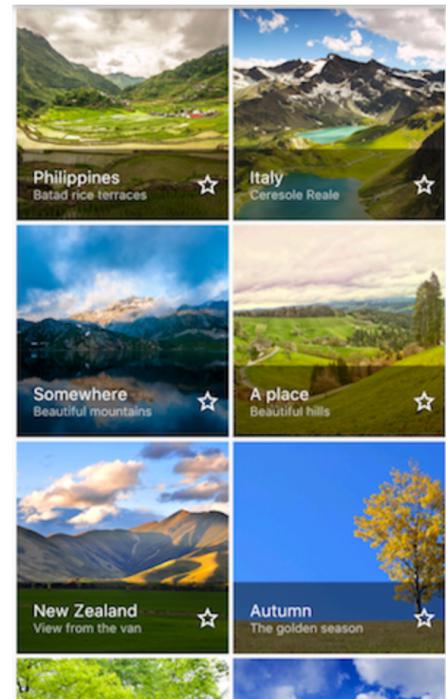
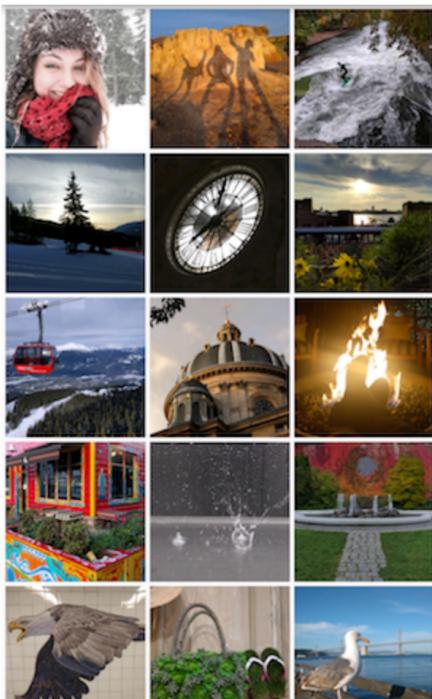
Gunakan widget `_buildGrid()` dan letakkan di dalam body **scaffold**.

```
//Buat widget berikut dan letakkan di dalam class _MyHomePageState

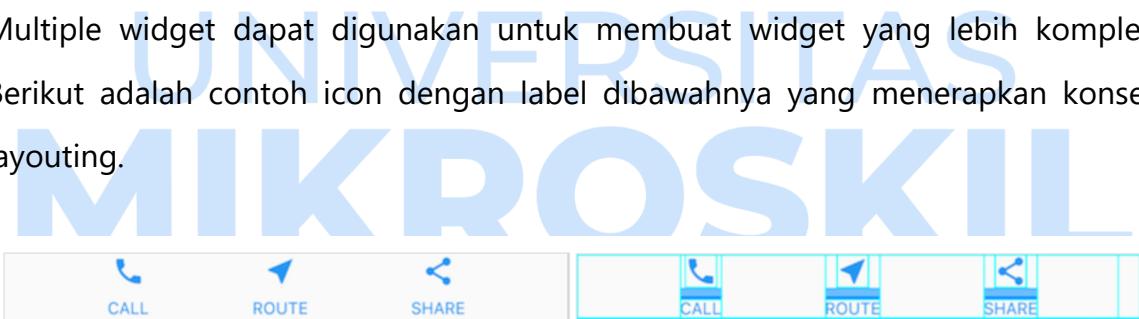
Widget _buildGrid() => GridView.extent(
  maxCrossAxisExtent: 150,
  padding: const EdgeInsets.all(4),
  mainAxisSpacing: 4,
  crossAxisSpacing: 4,
  children: _buildGridTileList(5));

// Pastikan memiliki 5 gambar dengan penamaan pic0.jpg,
// pic1.jpg...pic4.jpg.
// List.generate() adalah fungsi untuk membuat list
// a list when objects have a predictable naming pattern.
List<Container> _buildGridTileList(int count) => List.generate(
  count, (i) => Container(child:
    Image.asset('assets/images/pic${i}.jpg')));
```

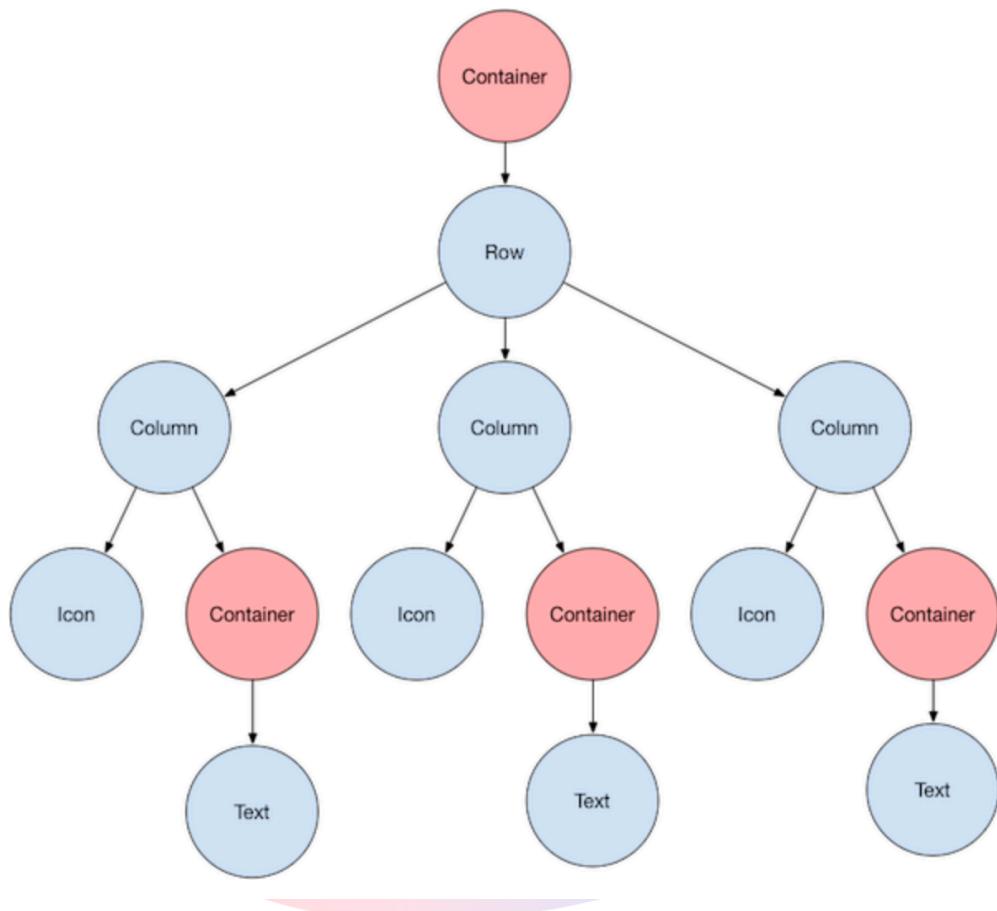
Hasil:



Multiple widget dapat digunakan untuk membuat widget yang lebih komplek. Berikut adalah contoh icon dengan label dibawahnya yang menerapkan konsep layouting.



Pada gambar sebelah kanan terlihat visual layout yang menampilkan 1 baris (Row) yang memiliki 3 kolom (Column) yang mana setiap kolom memiliki 1 (satu) icon dan label. Berikut adalah widget tree atau diagram dari layout di atas:



TODO: Silahkan buat tampilan layout berdasarkan tree widget di atas!

UNIVERSITAS
MIKROSKIL

Addtional:

Navigator

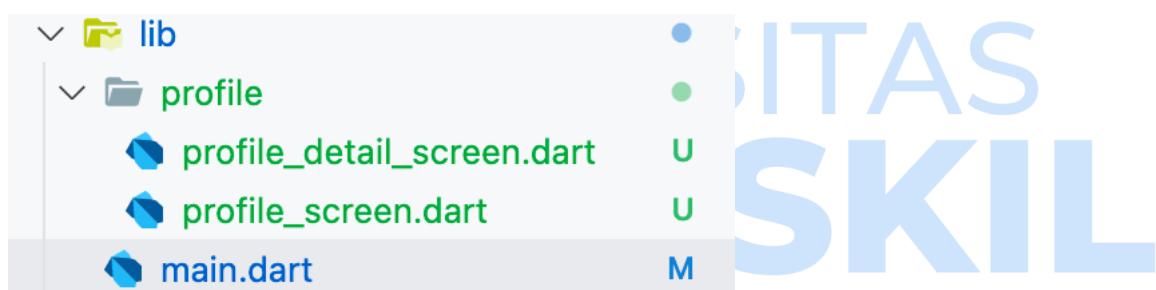
Sebagian besar aplikasi berisi beberapa layar untuk menampilkan berbagai jenis informasi. Misalnya, aplikasi mungkin memiliki layar yang menampilkan produk atau detail produk. Saat pengguna klik gambar produk, layar baru menampilkan detail tentang produk.

Untuk melakukan hal tersebut maka kita menggunakan navigator. Navigator pada flutter merupakan sebuah fungsi untuk melakukan navigasi dari satu halaman ke halaman lain (route).

Pada bagian ini, terdapat bagian yang dipelajari yaitu:

- Membuat halaman / screen pada flutter (routes)
- Melakukan navigasi dari 1 halaman ke halaman berikutnya **Navigator.push()**.
- Kembali pada halaman sebelumnya menggunakan **Navigator.pop()**.

Buatlah folder baru "profile" di dalam folder Lib pada project flutter seperti tampilan berikut:



Pada file "**profile_screen.dart**" buat halaman baru menggunakan stateless widget dan scaffold. Berikut contoh:

```
import 'package:flutter/material.dart';
import 'package:flutter_application/profile/profile_detail_screen.dart';

class ProfileScreen extends StatelessWidget {
    const ProfileScreen({Key? key}) : super(key: key);

    @override
    Widget build(BuildContext context) {
        return Scaffold(
            appBar: AppBar(
                title: const Text('Profile'),
            ),
            body: Center(
                child: ElevatedButton(
                    onPressed: () {
                        Navigator.push(
                            context,
                            MaterialPageRoute(
                                builder: (context) => const ProfileDetailScreen()));
                    },
                    child: const Text('Profile Detail'),
                ),
            ),
        );
    }
}
```

Keterangan:

- Class **ProfileScreen** menggunakan StatelessWidget
- Widget **build** menggunakan Scaffold dari flutter material yang akan menampilkan appbar dan body.
- ElevatedButton diberikan fungsi untuk navigasi ke halaman **ProfileDetailScreen()**

Pada file "**profile_detail_screen.dart**" buat halaman baru menggunakan stateless widget dan scaffold. Berikut contoh:

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

class ProfileDetailScreen extends StatelessWidget {
    const ProfileDetailScreen({Key? key}) : super(key: key);

    @override
    Widget build(BuildContext context) {
        return Scaffold(
            appBar: AppBar(
                title: const Text('Profile Detail'),
            ),
            body: Center(
                child: Column(children: [
                    const Text('Ini adalah detail Profile'),
                    ElevatedButton(
                        onPressed: () {
                            Navigator.pop(context);
                        },
                        child: const Text('Kembali'))
                ]),
            ),
        );
    }
}
```

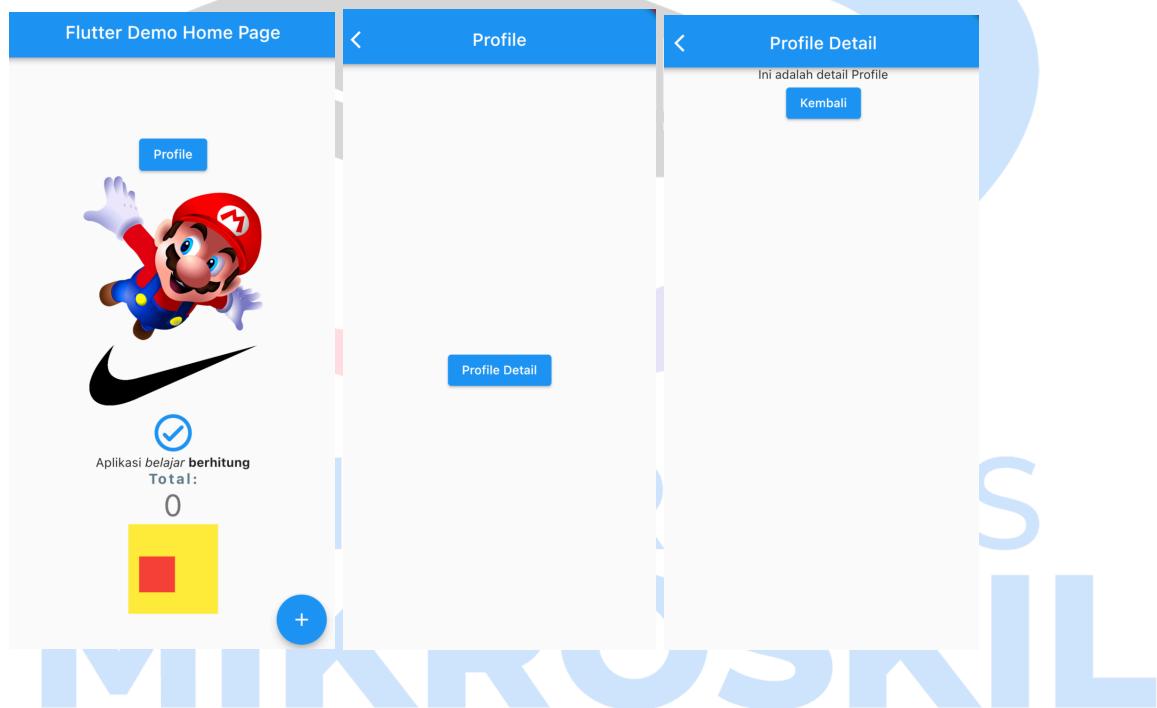
Keterangan: 

- ElevatedButton diberikan fungsi untuk navigasi ke halaman sebelumnya **ProfileScreen()** menggunakan fungsi Navigator.pop.

Untuk navigasi ke halaman ProfileScreen tambahkan button pada Widget Column di **main.dart**

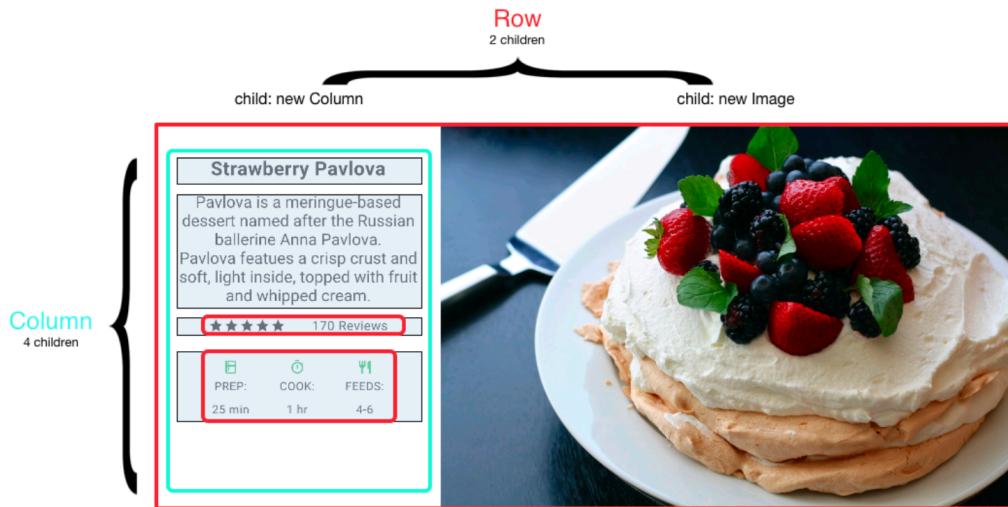
```
ElevatedButton(  
    onPressed: () {  
        Navigator.push(  
            context,  
            MaterialPageRoute(  
                builder: (context) => const ProfileScreen()));  
    },  
    child: const Text('Profile')),
```

Hasil:



Latihan:

Silahkan buat tampilan berikut dengan menerapkan layout pada flutter pada halaman profile_detail_screen.dart. Ganti detail seperti foto dan deskripsi dengan profile masing-masing:



The left column's widget tree nests rows and columns.

