

# Laporan Praktikum

---

## Pertemuan 7

---

### Layout & Navigasi

---

#### Data Mahasiswa

Nama : Fanesabhirawaning Sulistyo

NIM : 2241720027

Kelas : TI-3C

---

### Praktikum 1: Membangun Layout di Flutter

Apa yang akan Anda pelajari

1. Cara kerja mekanisme tata letak Flutter.
2. Cara menata widget secara vertikal dan horizontal.
3. Cara membuat tata letak Flutter.

Selesaikan langkah-langkah praktikum berikut ini menggunakan editor Visual Studio Code (VS Code) atau Android Studio atau code editor lain kesukaan Anda.

**Perhatian:**

Diasumsikan Anda telah berhasil melakukan setup environment Flutter SDK, VS Code, Flutter Plugin, dan Android SDK pada pertemuan pertama.

**Tampilan akhir yang akan Anda buat**



### Oeschinen Lake Campground

Kandersteg, Switzerland

★ 41



CALL



ROUTE



SHARE

Lake Oeschinen lies at the foot of the Blüemlisalp in the Bernese Alps. Situated 1,578 meters above sea level, it is one of the larger Alpine Lakes. A gondola ride from Kandersteg, followed by a half-hour walk through pastures and pine forest, leads you to the lake, which warms to 20 degrees Celsius in the summer. Activities enjoyed here include rowing, and riding the summer toboggan run.

## Langkah 1 Buat Project Baru

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

## Langkah 2 Buka file lib/main.dart

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

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

void main() => runApp(const MyApp());

class MyApp extends StatelessWidget {
    const MyApp({super.key});

    @override
    Widget build(BuildContext context) {
        return MaterialApp(
            title: 'Flutter layout: Fanesabhirawaning Sulistyo | 2241720027',
            home: Scaffold(
                appBar: AppBar(
                    title: const Text('Flutter layout demo'),
                ),
                body: const Center(
                    child: Text('Hello World'),
                ),
            ),
        );
    }
}
```

### 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.

**Oeschinen Lake Campground**

41

Kandersteg, Switzerland

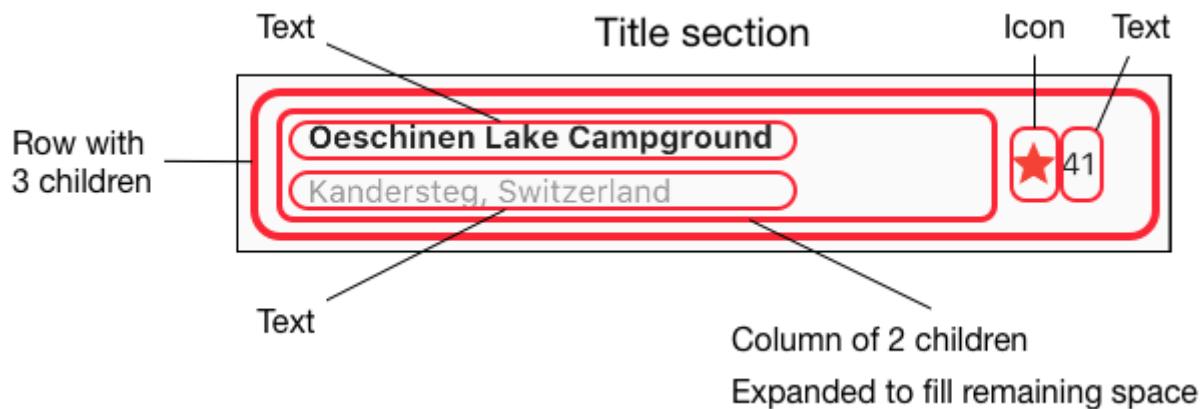
CALL

ROUTE

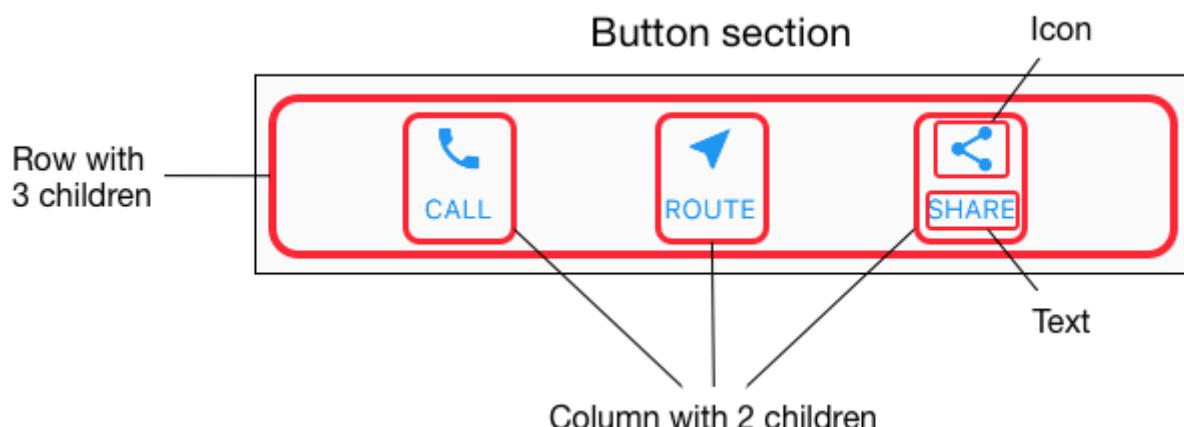
SHARE

Lake Oeschinen lies at the foot of the Blüemlisalp in the Bernese Alps. Situated 1,578 meters above sea level, it is one of the larger Alpine Lakes. A gondola ride from Kandersteg, followed by a half-hour walk through pastures and pine forest, leads you to the lake, which warms to 20 degrees Celsius in the summer. Activities enjoyed here include rowing, and riding the summer toboggan run.

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:

```
Widget titleSection = Container(
  padding: const EdgeInsets.all(32),
  child: Row(
    children: [
      Expanded(
        /* Jawaban soal 1 */
        child: Column(
          mainAxisAlignment: MainAxisAlignment.start,
          children: [
            /*Jawaban soal 2*/
            Container(
              padding: const EdgeInsets.only(bottom: 8),
              child: const Text(
                'Wisata Gunung di Batu',
                style: TextStyle(

```

```
        fontWeight: FontWeight.bold,
    ),
),
),
),
Text(
    'Batu, Malang, Indonesia',
    style: TextStyle(
        color: Colors.grey[500],
    ),
),
],
),
),
),
/*Jawaban soal 3*/
Icon(
    Icons.star,
    color: Colors.red[500],
),
const Text('41'),
],
),
);
```

**soal 1** Letakkan widget Column di dalam widget Expanded agar menyesuaikan ruang yang tersisa di dalam widget Row. Tambahkan properti mainAxisAlignment ke CrossAxisAlignment.start sehingga posisi kolom berada di awal baris.

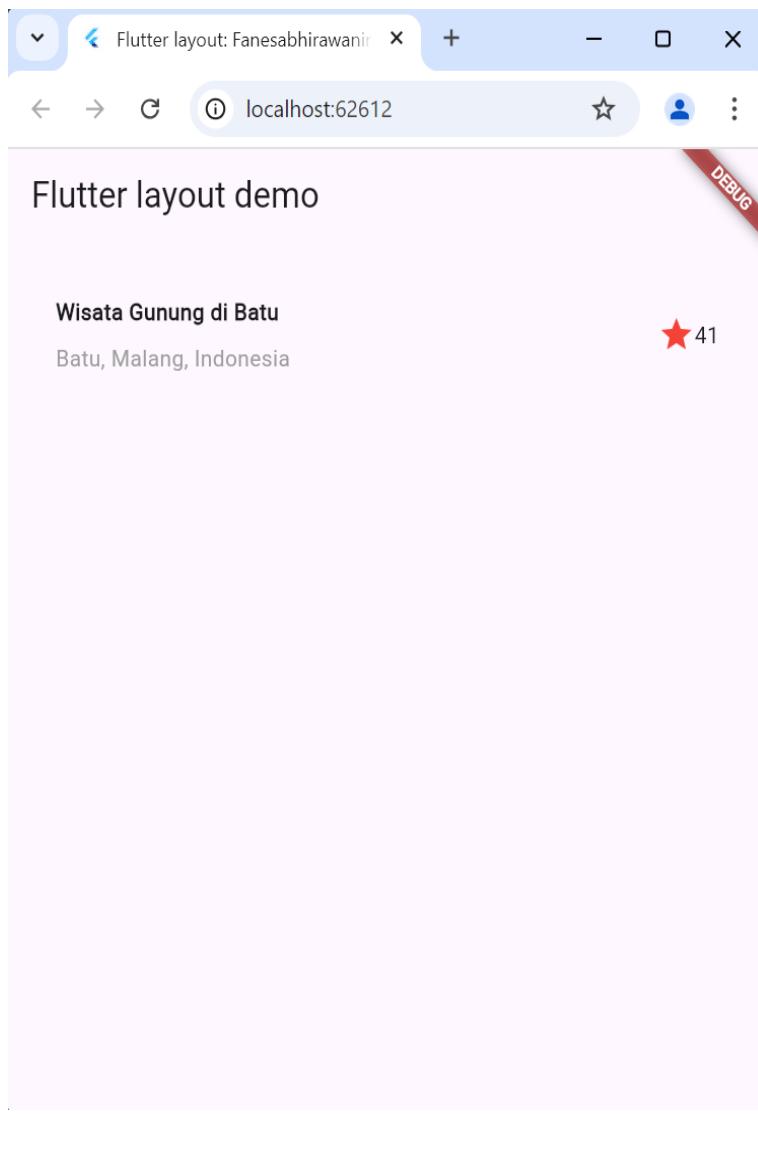
**soal 2** Letakkan baris pertama teks di dalam Container sehingga memungkinkan Anda untuk menambahkan padding = 8. Teks 'Batu, Malang, Indonesia' di dalam Column, set warna menjadi abu-abu.

**soal 3** Dua item terakhir di baris judul adalah ikon bintang, set dengan warna merah, dan teks "41". Seluruh baris ada di dalam Container dan beri padding di sepanjang setiap tepinya sebesar 32 piksel. Kemudian ganti isi body text 'Hello World' dengan variabel titleSection seperti berikut:

☰ {./base → step2}/lib/main.dart  
□ Viewed

		@@ -14,11 +48,13 @@
14	48	return MaterialApp(
15	49	title: 'Flutter layout demo',
16	50	home: Scaffold(
17	51	appBar: AppBar(
18	52	title: const Text('Flutter layout demo'),
19	53	),
20	-	body: const Center(
21	-	child: Text('Hello World'),
54	+	body: Column(
55	+	children: [
56	+	titleSection,
57	+	],
22	58	),
23	59	),
24	60	);

## Output dari hasil praktikum 01



## 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)

```
class MyApp extends StatelessWidget {  
  const MyApp({super.key});  
  
  @override  
  Widget build(BuildContext context) {
```

```
// ...
}

Column _buildButtonColumn(Color color, IconData icon, String label) {
    return Column(
        mainAxisAlignment: MainAxisAlignment.min,
        mainAxisSize: MainAxisSize.center,
        children: [
            Icon(icon, color: color),
            Container(
                margin: const EdgeInsets.only(top: 8),
                child: Text(
                    label,
                    style: TextStyle(
                        fontSize: 12,
                        fontWeight: FontWeight.w400,
                        color: color,
                    ),
                ),
            ),
        ],
    );
}
```

## 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)

```
Color color = Theme.of(context).primaryColor;

Widget buttonSection = Row(
    mainAxisAlignment: MainAxisAlignment.spaceEvenly,
    children: [
        _buildButtonColumn(color, Icons.call, 'CALL'),
        _buildButtonColumn(color, Icons.near_me, 'ROUTE'),
        _buildButtonColumn(color, Icons.share, 'SHARE'),
    ],
);
```

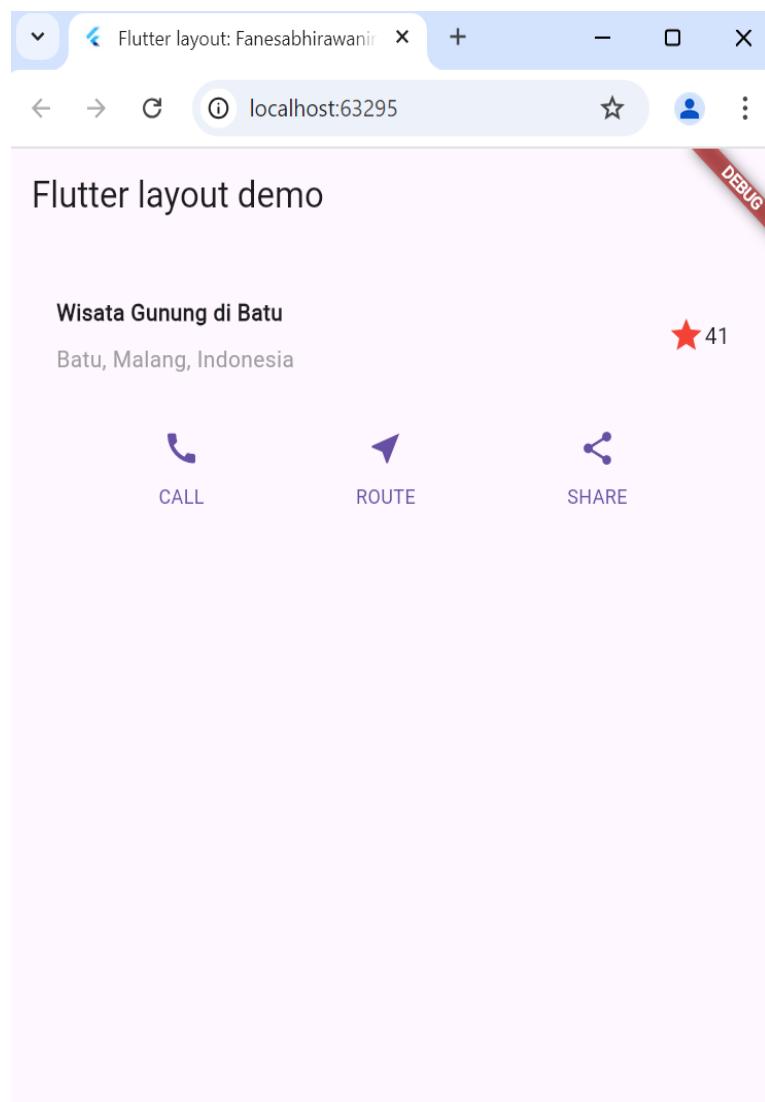
## Langkah 3 Tambah button section ke body

Tambahkan variabel buttonSection ke dalam body seperti berikut:

```
④ {step2 → step3}/lib/main.dart
□ Viewed

@@ -48,3 +59,3 @@
48   59     return MaterialApp(
49   60       title: 'Flutter layout demo',
50   61       home: Scaffold(
@@ -54,8 +65,9 @@
54   65         body: Column(
55   66           children: [
56   67             titleSection,
57   68             +         buttonSection,
58   69             ],
59   70             ),
60   71             ),
61   72             );
62   73 }
```

## Output dari hasil praktikum 02



## 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:

```
Widget textSection = Container(  
    padding: const EdgeInsets.all(32),  
    child: const Text(  
        'Carilah teks di internet yang sesuai '  
        'dengan foto atau tempat wisata yang ingin '  
        'Anda tampilkan. '  
        'Tambahkan nama dan NIM Anda sebagai '  
        'identitas hasil pekerjaan Anda. '  
        'Selamat mengerjakan ☺.',  
        softWrap: true,  
    ),  
);
```

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:

The screenshot shows a code diff interface comparing two versions of a file. The left column shows line numbers from 59 to 71. The right column shows the corresponding code. A green highlight covers the line where 'textSection' is added to the 'children' list. The code is as follows:

```
@@ -59,3 +72,3 @@  
 59   72      return MaterialApp(  
 60   73          title: 'Flutter layout demo',  
 61   74          home: Scaffold(  
@@ -66,6 +79,7 @@  
 66   79              children: [  
 67   80                  titleSection,  
 68   81                  buttonSection,  
 69   82 +                  textSection,  
 70   83              ],  
 71   84          ),  
 72   85      ),
```

### Berikut adalah hasil dari Praktikum saya



## Praktikum 4: Implementasi image section

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

### Langkah 1 Siapkan asset 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:

```
//To add assets to your application, add an assets section, like this:  
assets:  
- images/gunungkawi.jpg
```

#### Tips

- Perhatikan bahwa pubspec.yaml sensitif terhadap huruf besar-kecil, jadi tulis assets: dan URL gambar seperti yang ditunjukkan di atas.

- File pubspec juga sensitif terhadap spasi, jadi gunakan indentasi yang tepat.
- Anda mungkin perlu memulai ulang program yang sedang berjalan (baik di simulator atau perangkat yang terhubung) agar perubahan pubspec dapat diterapkan.

## Langkah 2 Tambahkan gambar ke body

Tambahkan asset gambar ke dalam body seperti berikut:

```
body: Column(
    children: [
        Image.asset(
            'images/gunungkawi.jpg',
            width: 600,
            height: 240,
            fit: BoxFit.cover,
        ),
        titleSection,
        buttonSection,
        textSection,
    ],
),
```

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

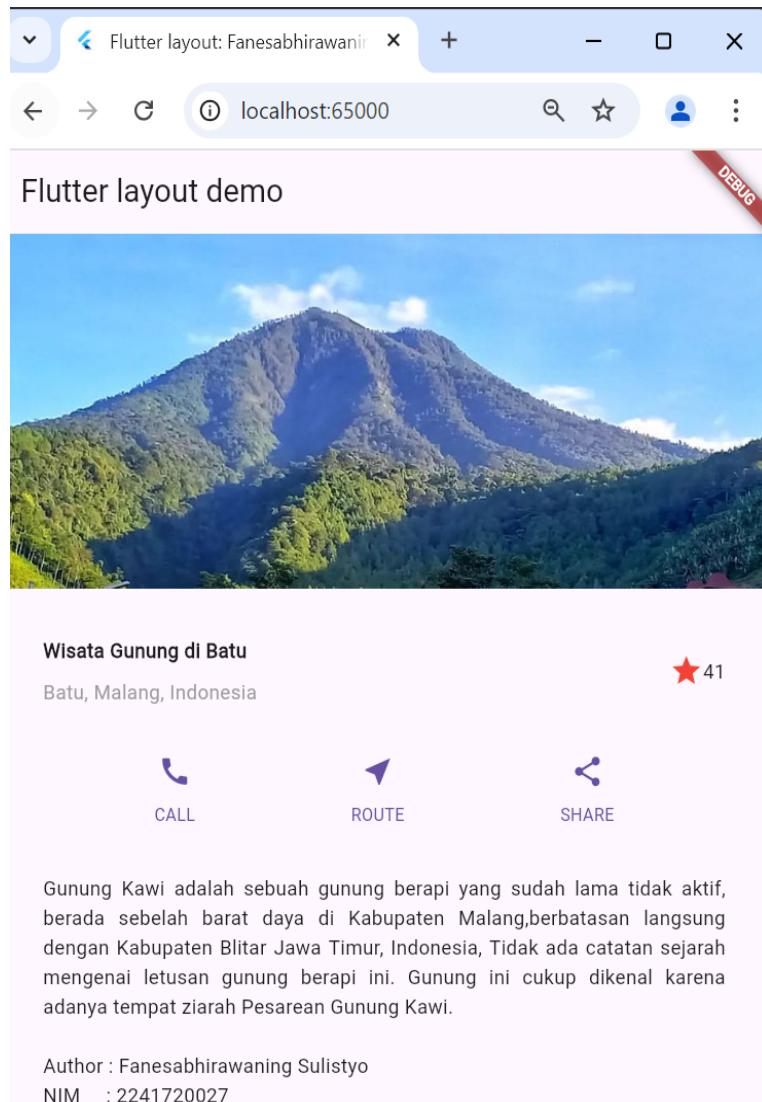
## Langkah 3 Terakhir, ubah menjadi ListView

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.

```
return MaterialApp(
    title: 'Flutter layout: Fanesabhirawaning Sulistyo | 2241720027',
    home: Scaffold(
        appBar: AppBar(
            title: const Text('Flutter layout demo'),
        ),
        body: ListView(
            children: [
                Image.asset(
                    'images/gunungkawi.jpg',
                    width: 600,
                    height: 240,
                    fit: BoxFit.cover,
                ),
                titleSection,
                buttonSection,
                textSection
            ],
        ),
    ),
)
```

```
)  
);
```

## Berikut adalah hasil dari Praktikum saya



## Tugas Praktikum

1. Selesaikan Praktikum 1 sampai 4, lalu dokumentasikan dan push ke repository Anda berupa screenshot setiap hasil pekerjaan beserta penjelasannya di file README.md!

### Jawab

Sudah saya jawab dan jelaskan pada praktikum 1, 2, 3 dan 4 tadi diatas atau juga bisa mengklik link dibawah kemudia diarahkan keatas secara otomatis

- [Praktikum 1 \(disini\)](#)
- [Praktikum 2 \(disini\)](#)
- [Praktikum 3 \(disini\)](#)

- Praktikum 4 (disini)

2. Silakan implementasikan di project baru "basic\_layout\_flutter" dengan mengakses sumber ini:  
<https://docs.flutter.dev/codelabs/layout-basics>

**Jawab**

tidak perlu dikerjakan untuk bahan belajar

3. Kumpulkan link commit repository GitHub Anda ke spreadsheet yang telah disediakan!

**Jawab**

[Berikut Link Repository saya \(Disini\)](#)