PRESENTATION BY

Kelompok 6

FLUTTER

ANGGOTA KELOMPOK:

Alvin Triseptia Mairis	24060120130044
Farrel Andhika Rizky Putra	24060120130071
Liem, Roy Marcelino	24060120130059
M. Khoirul Ma'arif	24060120130116
Adriel Silaban	24060120140095

CONTENT

FUNDAMENTAL

FORM DEVELOPMENT

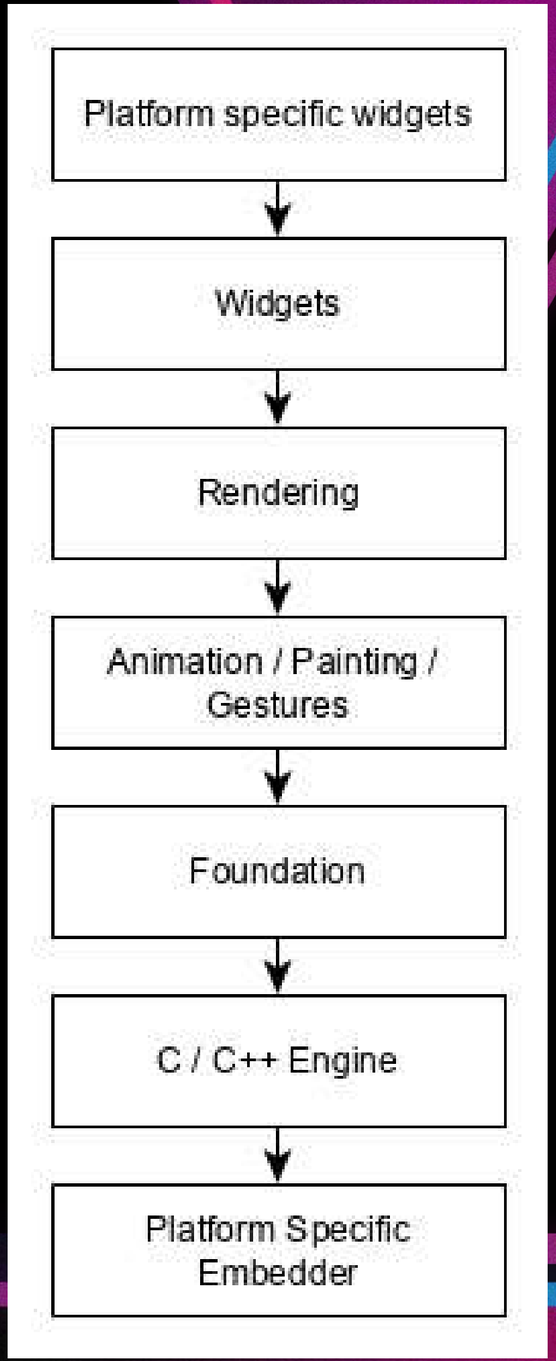
INTERAKSI DENGAN DATA

DEPLOYMENT

FUNDAMENTAL

ARSITEKTUR APLIKASI

- Di Flutter, semuanya adalah widget dan widget kompleks terdiri dari widget yang sudah ada.
- Fitur interaktif dapat digabungkan kapanpun diperlukan menggunakan widget GestureDetector .
- Status widget dapat dipertahankan kapanpun diperlukan menggunakan widget StatefulWidget .
- Flutter menawarkan desain berlapis sehingga setiap lapisan dapat diprogram tergantung pada kerumitan tugas.



BAHASA PEMROGRAMAN


```dart
void main() {
  print('Hello, World!');
}
```

Dart adalah bahasa pemrograman yang dikembangkan oleh Google untuk kebutuhan umum (general-purpose programming language) yang digunakan dalam mengembangkan aplikasi web, aplikasi seluler, aplikasi desktop, server, dll.

Secara umum, Dart menerapkan konsep pemrograman berorientasi objek (OOP) dimana struktur kode berada dalam class yang didalamnya berisi method maupun variabel. Dart sendiri menggunakan C-Style syntax dan Statically typed sehingga mekanisme dart mirip dengan bahasa pemrograman C, java, javascript, dan Swift.

PENGEMBANGAN APLIKASI

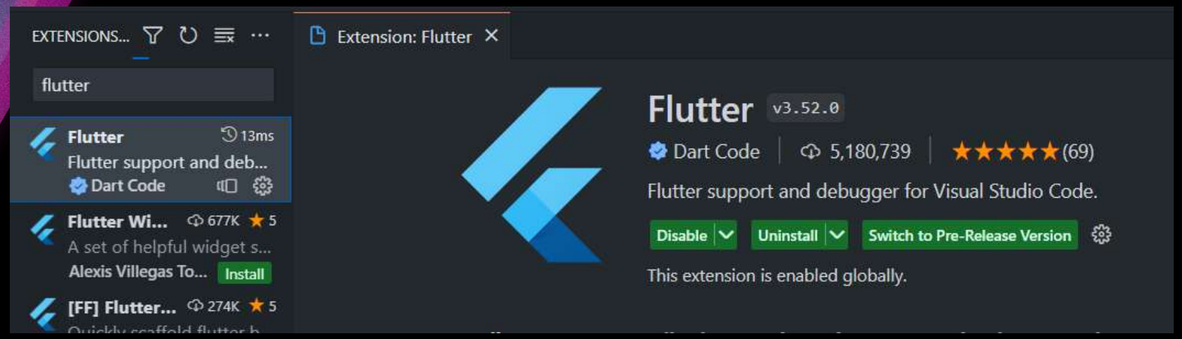
Flutter adalah SDK (Software Development Kit) yang dikembangkan oleh Google untuk membuat aplikasi yang bagus dan bisa berjalan pada berbagai platform. Berikut contoh pengembangan aplikasi menggunakan Visual Studio Code

1. Melakukan instalasi Flutter SDK.
2. Langkah selanjutnya kita akan melakukan update path supaya perintah Flutter bisa digunakan pada command prompt/terminal
3. Jalankan perintah '*flutter doctor*' untuk membuka flutter doctor:

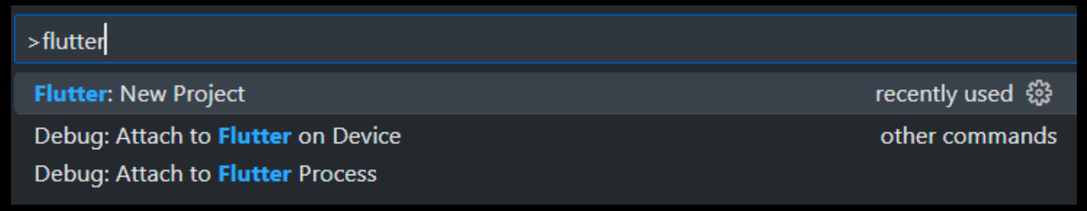
```
PS C:\Kuliah\Semester5\PengembanganBerbasisPlatform\PascaUTS\Tugas\FLUTTER_FSM> flutter doctor
Doctor summary (to see all details, run flutter doctor -v):
[√] Flutter (Channel stable, 3.3.0, on Microsoft Windows [Version 10.0.22621.819], locale en-ID)
[√] Android toolchain - develop for Android devices (Android SDK version 33.0.0)
[√] Chrome - develop for the web
[X] Visual Studio - develop for Windows
    X Visual Studio not installed; this is necessary for Windows development.
      Download at https://visualstudio.microsoft.com/downloads/.
      Please install the "Desktop development with C++" workload, including all of its default components
[√] Android Studio (version 2021.3)
[√] VS Code (version 1.73.1)
[√] Connected device (3 available)
[√] HTTP Host Availability
```

Flutter doctor adalah perintah untuk mengecek kelengkapan framework flutter yang digunakan, seperti versi flutter yang digunakan, Android SDK yang digunakan, iOS SDK yang digunakan (hanya pada MacOS), perangkat yang sudah terhubung, dan semacamnya.

PENGEMBANGAN APLIKASI



3. Membuka VS Code
4. Kemudian instal extension Flutter



5. Buka Tab View>Command Palette (Ctrl+Shift+P)
6. Ketik 'flutter', kemudian pilih 'Flutter: New Project'
7. Memberi nama pada project dan menentukan lokasi project ingin disimpan
8. Klik OK, project akan digenerate dan aplikasi siap dikembangkan

FORM DEVELOPMENT

CARA MENYUSUN FORM

```dart
class FormRuang extends StatefulWidget {
  final Ruangan? updatedRuangan;

  const FormRuang({super.key, this.updatedRuangan});

  @override
  State<FormRuang> createState() => _FormRuangState();
}

class _FormRuangState extends State<FormRuang> {
  final _formKey = GlobalKey<FormState>();
  final GedungController _gedungController =
      GedungController(Hive.box('gedung'));

```

Menggunakan StatefulWidget. Isi dari form harus di-manage menggunakan state

CARA MENYUSUN FORM

```dart
@override
Widget build(BuildContext context) {
  return Form(
    key: _formKey,
    child: Column(
```

```dart
class _FormRuangState extends State<FormRuang> {
  final _formKey = GlobalKey<FormState>();
```

Widget Form merupakan container dari form. Isi dari form dimasukkan ke dalam child

Key dari form merupakan unique identifier dari form yang dapat digunakan untuk validasi dan manipulasi form

FORM FIELD

```
FormField(
  builder: ((state) {
    return TextField();
  }),
  validator: (value) {
    if (value == null || value == "") {
      return "Value can't be empty!";
    }
    return null;
  },
), // FormField
ElevatedButton(
```

- Field input dari form dapat dibungkus dengan widget FormField.
- Builder: widget yang akan ditampilkan
- Validator: untuk memvalidasi input user
- Validator mengembalikan null bila input valid dan String error message bila tidak

KOMPONEN DALAM FORM

TextFormField

- Gabungan dari TextField dan FormField
- Komponen untuk menerima input teks.
- Dapat diberi atribut controller untuk mempermudah state management dari text field dan validator untuk validasi

```
TextFormField(
  controller: _kodeRuangController,
  decoration: const InputDecoration(
    icon: Icon(Icons.person),
    hintText: 'Enter kode ruang',
    labelText: 'Kode Ruang',
  ), // InputDecoration
  validator: (value) {
    if (value == null || value.isEmpty) {
      return 'Kode ruang tidak boleh kosong';
    }
    return null;
  },
), // TextFormField
```

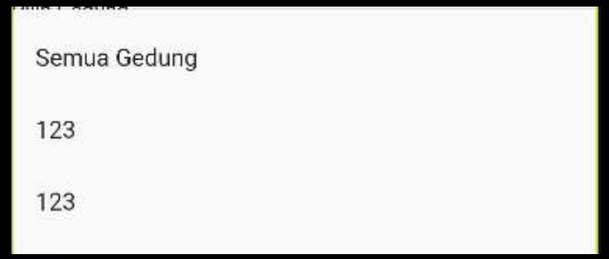

```
final TextEditingController _kodeRuangController = TextEditingController();
final TextEditingController _namaRuangController = TextEditingController();
final TextEditingController _kapasitasController = TextEditingController();
```

KOMPONEN DALAM FORM

DropdownButtonFormField

- DropdownButton merupakan komponen SELECT di dalam Flutter
- Gabungan dari DropdownButton dan FormField
- Atribut items berisikan option-option dalam dropdown. Diisi dengan List<DropdownMenuItem<T>>

```
DropdownButtonFormField<Gedung>(
  value: _gedung,
  items: _gedungList.map((Gedung e) {
    return DropdownMenuItem<Gedung>(
      value: e,
      child: Text(e.namaGedung),
    ); // DropdownMenuItem
  }).toList(),
  onChanged: (value) {
    setState(
      () {
        _gedung = value!;
      },
    );
  },
), // DropdownButtonFormField
```

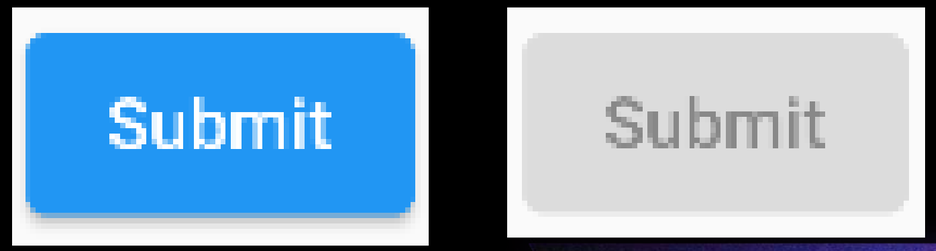


KOMPONEN DALAM FORM

ElevatedButton

- Komponen BUTTON di dalam Flutter.
- Atribut onPressed digunakan untuk menentukan aksi yang dilakukan ketika button di klik
- Child merupakan isi dari button. Biasanya berupa Text atau Icon

```
Container(
  alignment: Alignment.center,
  child: ElevatedButton(
    onPressed: onSubmit,
    child: const Text('Submit'),
  ), // ElevatedButton
), // Container
```



KOMPONEN DALAM FORM

Checkbox

- Komponen CHECKBOX di dalam Flutter.
- Value dari checkbox disimpan dalam state (boolean) menggunakan setState()


```
children: [
  Checkbox(
      value: checkboxVal,
      onChanged: (value) {
        setState(() {
          checkboxVal = value!;
        });
      }), // Checkbox
  Text("Centang ini")
]); // Row
```

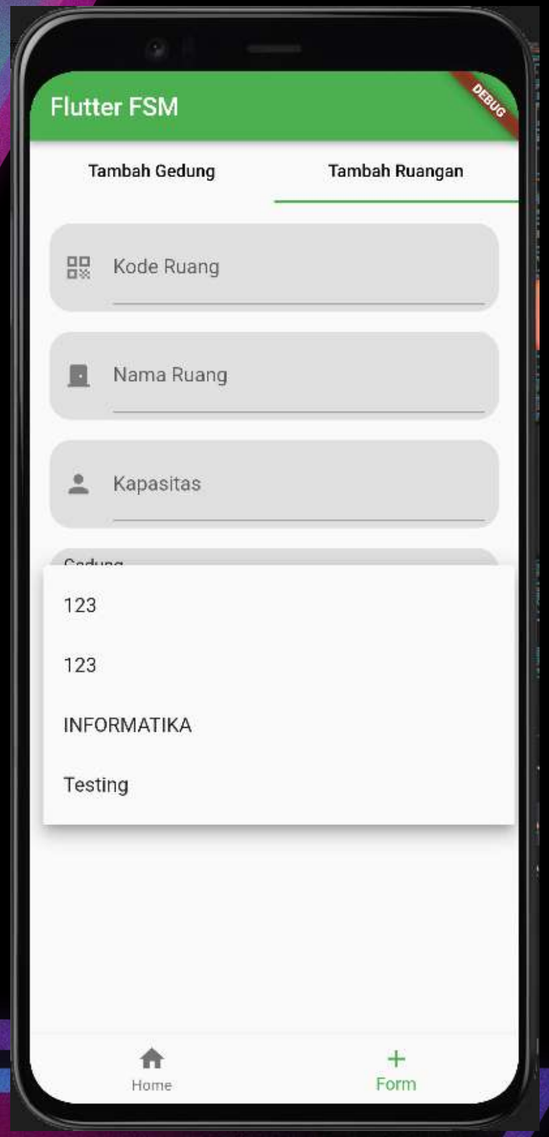
KOMPONEN DALAM FORM

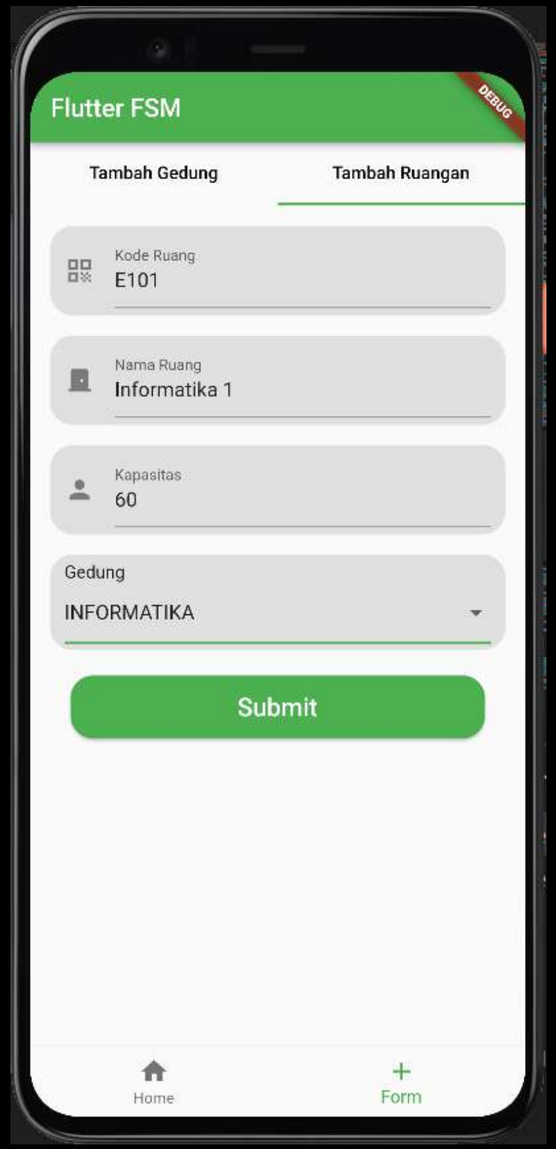
Radio

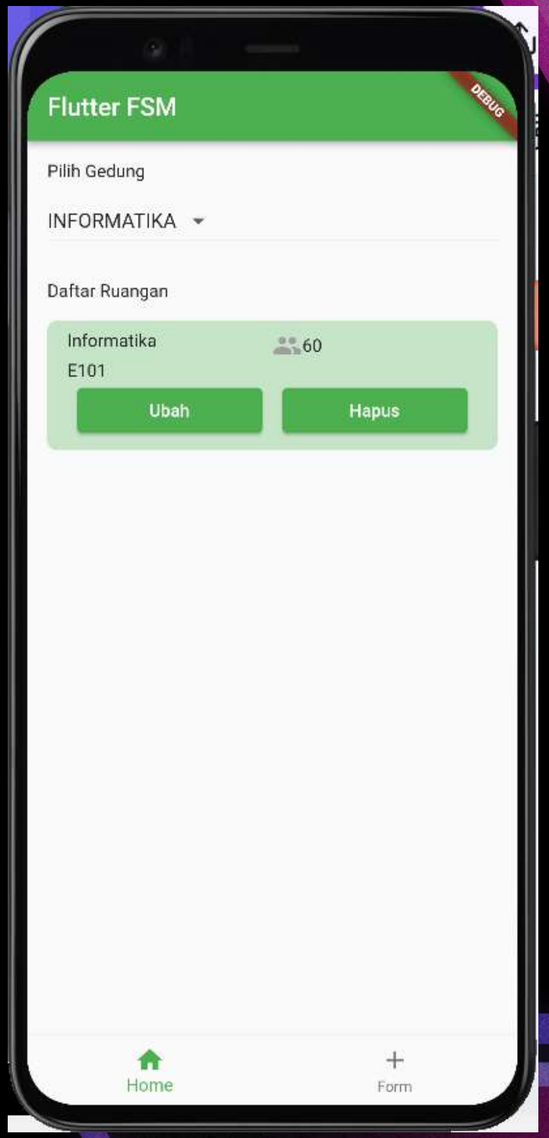
- Komponen RADIO di dalam Flutter.
- Value dari radio disimpan dalam state (sesuai tipe data item yang dipilih).
- Atribut groupValue menunjukkan state mana yang menyimpan value untuk radio button tersebut

```
children: [
  Radio(
    value: "Lab",
    groupValue: _radioValue,
    onChanged: (value) {
      setState(() {
        _radioValue = value!;
      });
    },
  ), // Radio
  Text("Lab"),
  Radio(
    value: "Kelas",
    groupValue: _radioValue,
    onChanged: (value) {
      setState(() {
        _radioValue = value!;
      });
    },
  ), // Radio
  Text("Kelas"),
]); // Row
```

CONTOH FORM DAN INTERAKSI







INTERAKSI DENGAN DATA

CARA PENGELOLAAN DATA

Pengelolaan database pada flutter dapat dilakukan dengan banyak cara, baik local storage maupun internet. Berikut merupakan contoh pengelolaan data di local storage menggunakan Hive.

Hive adalah database offline NoSQL yang cepat, ringan, untuk aplikasi flutter dan dart. Hive sangat membantu jika diperlukan basis data kunci dan value sederhana tanpa banyak relasi.

INSTALASI HIVE

1. Menambahkan dependencies pada file pubspec.yaml, yaitu build_runner, hive, hive_flutter dan hive_generator

```yaml
dependencies:
  build_runner: ^2.3.2
  cupertino_icons: ^1.0.2
  flutter:
    sdk: flutter
  hive: ^2.2.3
  hive_flutter: ^1.1.0
  hive_generator: ^2.0.0
```

2. Melakukan import package hive pada setiap file yang memerlukan hive

```dart
import 'package:hive/hive.dart';

import 'package:hive_flutter/hive_flutter.dart';
import 'package:hive_flutter/adapters.dart';
```

INSTALASI HIVE

3. Melakukan instalasi hive pada file main dengan code await Hive.initFlutter();

```
void main() async {
  WidgetsFlutterBinding.ensureInitialized();

  // init Hive
  await Hive.initFlutter();
```

4. Melakukan register adapter dari model database yang telah dibuat pada main()

```
// register adapter
Hive.registerAdapter<Gedung>(GedungAdapter());
Hive.registerAdapter<Ruangan>(RuanganAdapter());
```

5. Membuka Box untuk menyimpan semua data pada main()

```
// open box
await Hive.openBox<Gedung>('gedung');
await Hive.openBox<Ruangan>('ruangan');
```

PEMBUATAN MODEL DATABASE

1. Mengimport package hive

```
import 'package:hive/hive.dart';
```

2. Memberi HiveType pada class tabel dan HiveField pada setiap kolomnya

```
@HiveType(typeId: 0)
class Gedung {
  @HiveField(0)
  final String kodeGedung;
  @HiveField(1)
  final String namaGedung;
```

3. Membuat konstruktor class

```
const Gedung({required this.kodeGedung, required this.namaGedung});
```

PEMBUATAN MODEL DATABASE

4. Melakukan build dengan build_runner pada file model
   Menambahkan part pada file sesuai nama file dengan tambahan .g.dart,

```
part 'gedung.g.dart';
```

   lalu perintah “flutter packages pub run build_runner build” pada terminal. Otomatis akan terbuat file baru dengan nama sesuai part yang ditambahkan.

5. Mengeksport semua tabel model yang telah dibuat pada file models.dart

```
export 'gedung.dart';
export 'ruangan.dart';
```

CONTOH PENGELOLAAN DATA "DEMO"

DEPLOYEMENT

Flutter 3 yang merupakan versi terbaru memberikan dukungan untuk membangun aplikasi pada sistem operasi :

Android, iOS, Web, Windows, Linux, dan MacOS.

Dengan ini, developer cukup coding sekali saja, atau dikenal dengan single codebase.

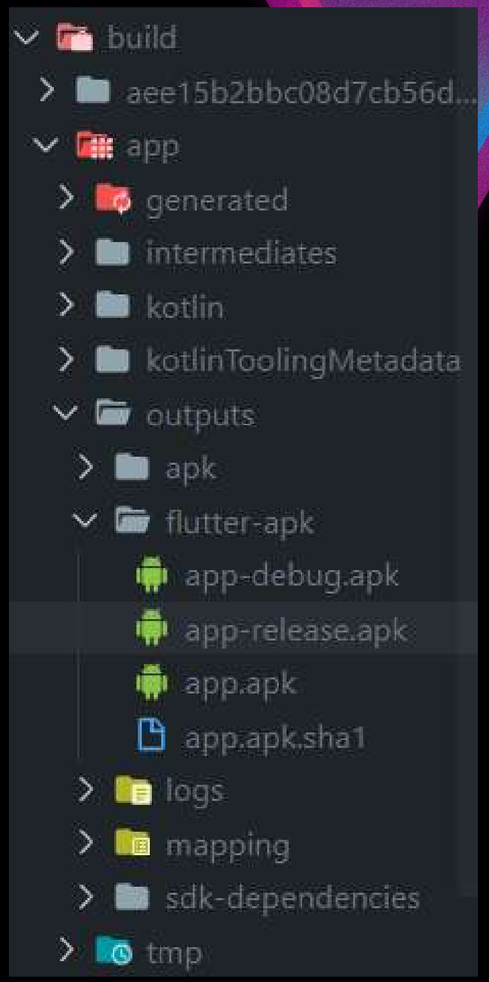
```
Available subcommands:
  aar          Build a repository containing an AAR and a POM file.
  apk          Build an Android APK file from your app.
  appbundle    Build an Android App Bundle file from your app.
  bundle       Build the Flutter assets directory from your app.
  web          Build a web application bundle.
  windows      Build a Windows desktop application.
```

Untuk melakukan deployement di flutter, cukup menggunakan command 'flutter build' diikuti subcommands target platform, yaitu pada kasus ini menggunakan subcommand apk.

```
flutter build apk
```

```
Running Gradle task 'assembleRelease'...                        80.3s
✓  Built build\app\outputs\flutter-apk\app-release.apk (18.6MB).
```

Setelah berhasil build apk, maka hasil build untuk aplikasi tersebut akan berada di folder build/app/outputs/flutter-apk



TERIMAKASIH!