

BUKU AJAR

PEMROGRAMAN MOBILE DENGAN FLUTTER

JURUSAN TEKNOLOGI INFORMASI

Arie Rachmad Syulistyo, S.Kom., M.Kom.

NIP 198708242019031010

Dian Hanifudin Subhi, S.Kom., M.Kom.

NIP 198806102019031018

Putra Prima Arhandi ST.,M.Kom

NIP 198611032014041001



**POLITEKNIK NEGERI MALANG
MARET 2021**

DAFTAR ISI

Syukur alhamdulillah, akhirnya penulisan buku ajar yang dilakukan ini dapat diselesaikan dengan tepat waktu. Semoga dengan hasil yang dicapai pada penulisan ini dapat membantu pembaca dalam mempelajari dan memahami topik pengujian perangkat lunak.

Modul ajar kami yang berjudul **“Pemrograman Mobile dengan Flutter”** ini disusun sebagai salah satu tugas utama Dosen dalam poin Tri Dharma Perguruan Tinggi.

Pada kesempatan ini tidak lupa penulis mengucapkan terimakasih kepada semua yang telah membimbing dan memberikan dorongan pada penulis, agar buku ajar yang dibuat dapat terselesaikan tepat waktu. Maka dari itu, perkenankanlah peneliti untuk mengucapkan terimakasih sebesar besarnya kepada.

1. Allah Subhanah Wat'ala karena atas berkat rahmat-Nya penelitian ini dapat terselesaikan dengan baik.
2. Rasulullah Muhammad SAW yang telah menjadi penunjuk jalan bagi umat manusia. Tanpa beliau tentulah saat ini saya akan berjalan tanpa arah.
3. Bapak Drs. Awan Setiawan, MMT,MM selaku Direktur Politeknik Negeri Malang.
4. Bapak Rudy Ariyanto Selaku Ketua Jurusan Teknologi Informasi.
5. Serta teman teman sejawat yang membantu dan memberikan dorongan agar buku ajar ini dapat terselesaikan dengan baik.

Dalam penulisan ini sepenuhnya jauh dari sempurna. Oleh karena itu, kritik maupun saran Anda dapat membangun demi kelanjutan dan sempurnanya penulisan ini, terimakasih.

Malang, 24 Januari 2020

Tim Penulis

DAFTAR ISI

BAB 1. INSTALASI DAN PENGENALAN FLUTTER

1.1 Apa itu Flutter

Flutter adalah sebuah framework open source yang dibuat oleh Google. Google membuat flutter dengan tujuan membangun sebuah framework untuk membuat UI yang modern, native dan reactive yang dapat berjalan di sistem operasi iOS maupun Android. Tidak hanya pada smartphone google juga membuat flutter untuk desktop, web dan embedded device.

Flutter diprogram dengan menggunakan bahasa Dart sebuah bahasa moderen yang dapat dicompile ke arsitektur processor ARM atau javascript. Flutter menggunakan Skia 2D rendering engine yang dapat bekerja pada hardware atau software yang berbeda platform.

Dart menggunakan metode kompilasi ahead of time (AOT) untuk mengubah kode Dart menjadi kode native untuk sistem operasi yang digunakan, oleh karena itu aplikasi yang dibangun menggunakan flutter memiliki kecepatan yang hampir sama dengan aplikasi native. Dart juga menggunakan konsep just-in-time (JIT) sehingga memungkinkan programmer dapat membuat perubahan pada kode program dan langsung melihat hasilnya melalui fitur hot reload yang dimiliki Flutter.

Flutter menggunakan Dart untuk membuat User Interface, sehingga memudahkan dalam membuat aplikasi karena menggunakan satu bahasa (Dart) dalam pembuatan UI maupun logika program. Flutter menggunakan pendekatan declarative dimana Flutter membangun UI mengikuti “State” yang dimiliki oleh aplikasi. Ketika state berubah maka UI akan digambar ulang .

Flutter juga memudahkan programmer karena dari satu kode program dapat dikompilasi ke kode native ARM, menggunakan GPU dan mengakses fitur spesifik dari smartphone baik yang menggunakan sistem operasi iOS ataupun yang menggunakan sistem operasi Android. Jadi dengan satu kali membuat program dapat membuat 2 aplikasi yang sama untuk sistem operasi yang berbeda (iOS atau Android).

1.2 Widget dan Element pada Flutter

Gaya pengembangan aplikasi menggunakan flutter sedikit berbeda dengan gaya pengembangan aplikasi pada umumnya, dimana UI pada flutter dibuat menggunakan *Widget*. Widget adalah sebuah konsep dimana UI dapat dianggap sebagai sebuah balok LEGO, sebuah bentuk baru dapat disusun dari beberapa balok dan masing masing kumpulan balok dapat dikombinasikan dengan kumpulan balok lain sehingga membentuk sebuah bentuk baru yang lebih kompleks. Flutter menggunakan widget ini sebagai balok dasar pembangunan aplikasi.

Widget dapat disusun dan dikombinasikan dalam satu layar, sama halnya dengan xml pada pemrograman android native, widget dapat disusun dalam bentuk tree dimana satu widget menjadi parent dan widget lain menjadi child. Masing masing widget dapat diberikan konfigurasi sesuai dengan kebutuhan aplikasi.

Flutter memiliki dua jenis widget yaitu StatelessWidget dan StatefulWidget. StatelessWidget digunakan ketika value (state / konfigurasi) dari widget tersebut tidak pernah berubah, dan StatefulWidget digunakan ketika value (state / konfigurasi) dari widget dapat berubah. Baik StatelessWidget maupun StatefulWidget sama sama memiliki sebuah method bernama “build” yang memilikiBuildContext untuk mengatur posisi widget didalam widget tree detail mengenai widget dan bagaimana membuatnya akan dibahas pada bab selanjutnya.

1.3 Praktikum Instalasi Flutter

1.3.1 Alat dan Bahan

Pada praktikum kali ini anda akan melakukan instalasi flutter pada sistem operasi Windows. Berikut ini Alat dan bahan yang dibutuhkan untuk menginstall flutter :

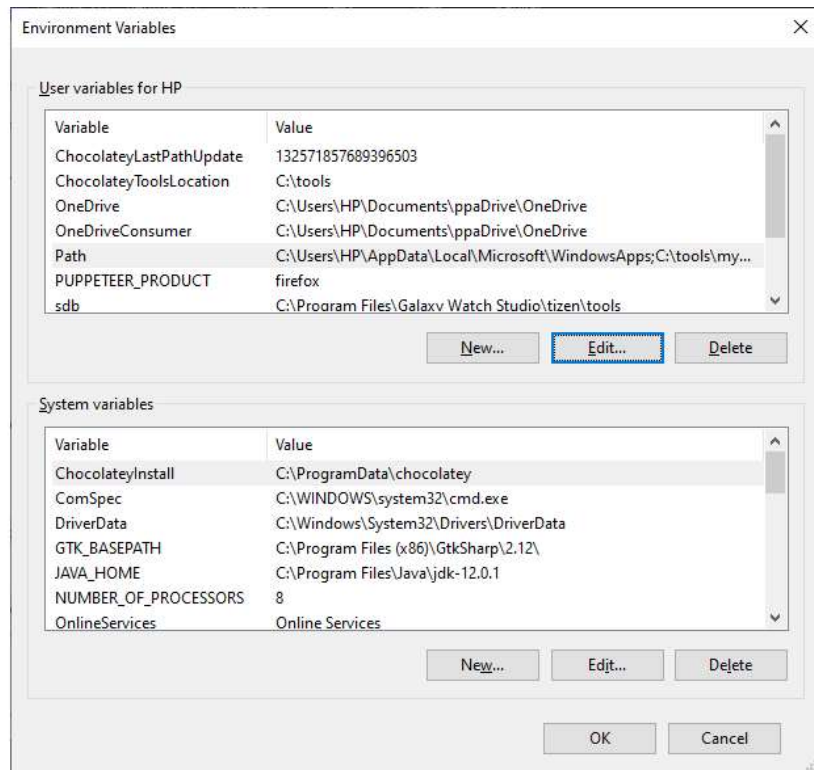
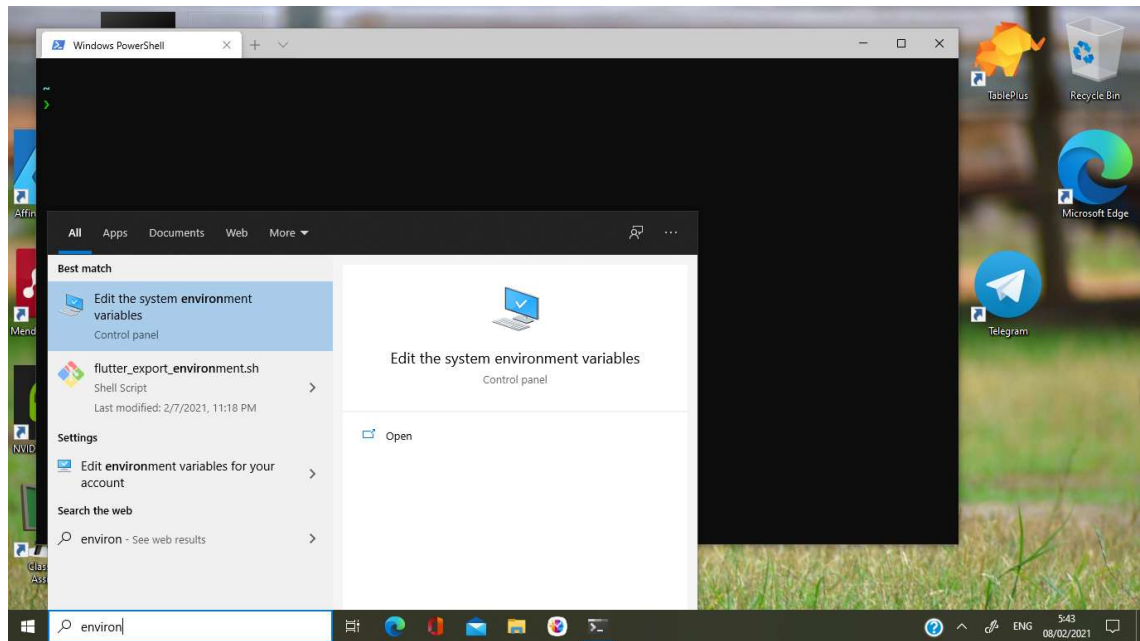
1. Sistem Operasi Windows minimal Windows 7 SP1 atau setelahnya 64 bit dan x86-64.
2. Harddisk 1,32 GB untuk Flutter saja selain IDE atau alat lain
3. Windows PowerShell 5.0 atau lebih baru <https://docs.microsoft.com/en-us/powershell/scripting/install/installing-windows-powershell>
4. Git For Windows 2.x <https://git-scm.com/download/win>

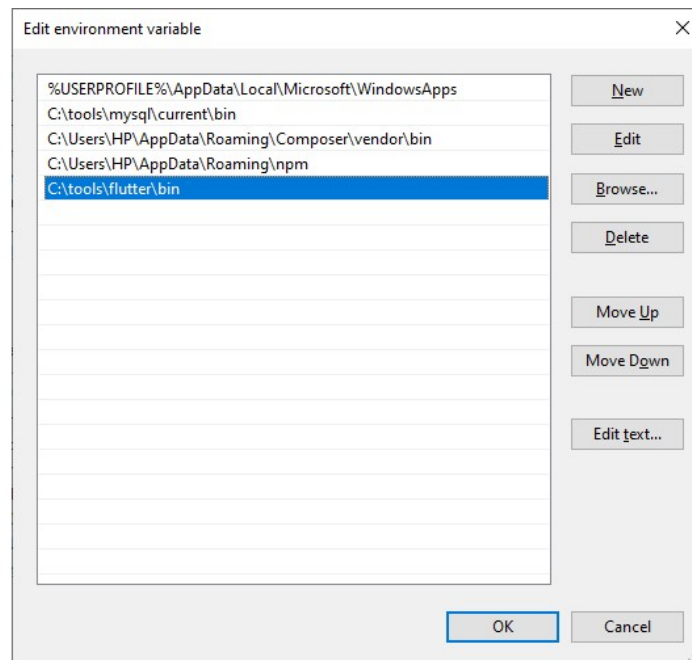
1.3.2 Langkah Langkah Praktikum

1. Download Flutter SDK, pada saat modul praktikum ini dibuat SDK untuk windows ada di tautan berikut
https://storage.googleapis.com/flutter_infra/releases/stable/windows/flutter_windows_1.22.6-stable.zip
2. Extract file yang di download ke harddisk anda contoh lokasi ke C:\src\flutter (**JANGAN** di install ke folder C:\Program Files\ karena membutuhkan akses admin)
3. Atau jika anda sudah menginstall git buatlah folder src di dalam drive C kemudian buka terminal di folder tersebut dan ketik

```
git clone https://github.com/flutter/flutter.git -b stable
```

4. Update Windows PATH tambahkan path menuju folder C:\src\flutter\bin





5. Install Android Studio
6. Setup Android Device / Emulator
7. Install Visual Studio Code <https://code.visualstudio.com/>
8. Install Flutter Plugin
9. Validasi Install dengan mengetikkan perintah berikut di terminal

```
flutter doctor
```

```
Windows PowerShell
Documents/2021/pemrogramanMobile
> flutter doctor
Doctor summary (to see all details, run flutter doctor -v):
[✓] Flutter (Channel dev, 1.26.0-17.2.pre, on Microsoft Windows [Version 10.0.19041.746], locale en-ID)
[!] Android toolchain - develop for Android devices (Android SDK version 30.0.2)
    x Android license status unknown.
      Run 'flutter doctor --android-licenses' to accept the SDK licenses.
      See https://flutter.dev/docs/get-started/install/windows#android-setup for more details.
[✗] Chrome - develop for the web (Cannot find Chrome executable at .\Google\Chrome\Application\chrome.exe)
    ! Cannot find Chrome. Try setting CHROME_EXECUTABLE to a Chrome executable.
[✓] Android Studio (version 4.1.0)
[✓] VS Code, 64-bit edition (version 1.53.0)
[✓] Connected device (1 available)

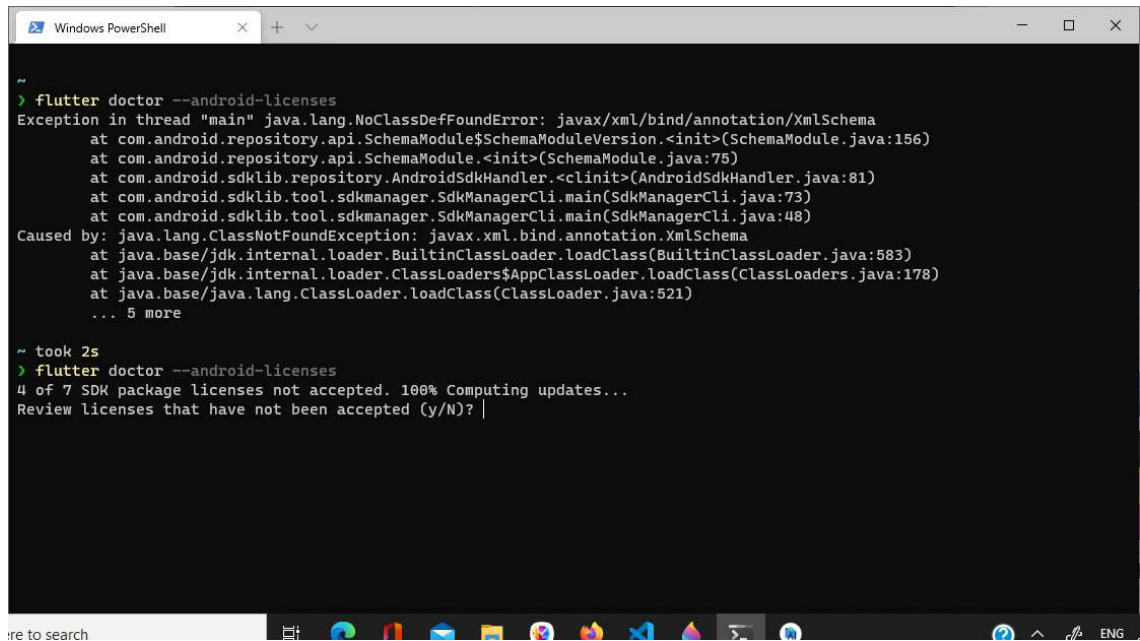
! Doctor found issues in 2 categories.

Documents/2021/pemrogramanMobile took 2s
> |
```

10. Jika perintah flutter doctor mengeluarkan hasil seperti gambar di atas maka anda perlu menerima lisensi android SDK dengan mengetikkan perintah berikut ini di terminal.

```
flutter doctor --android-licenses
```

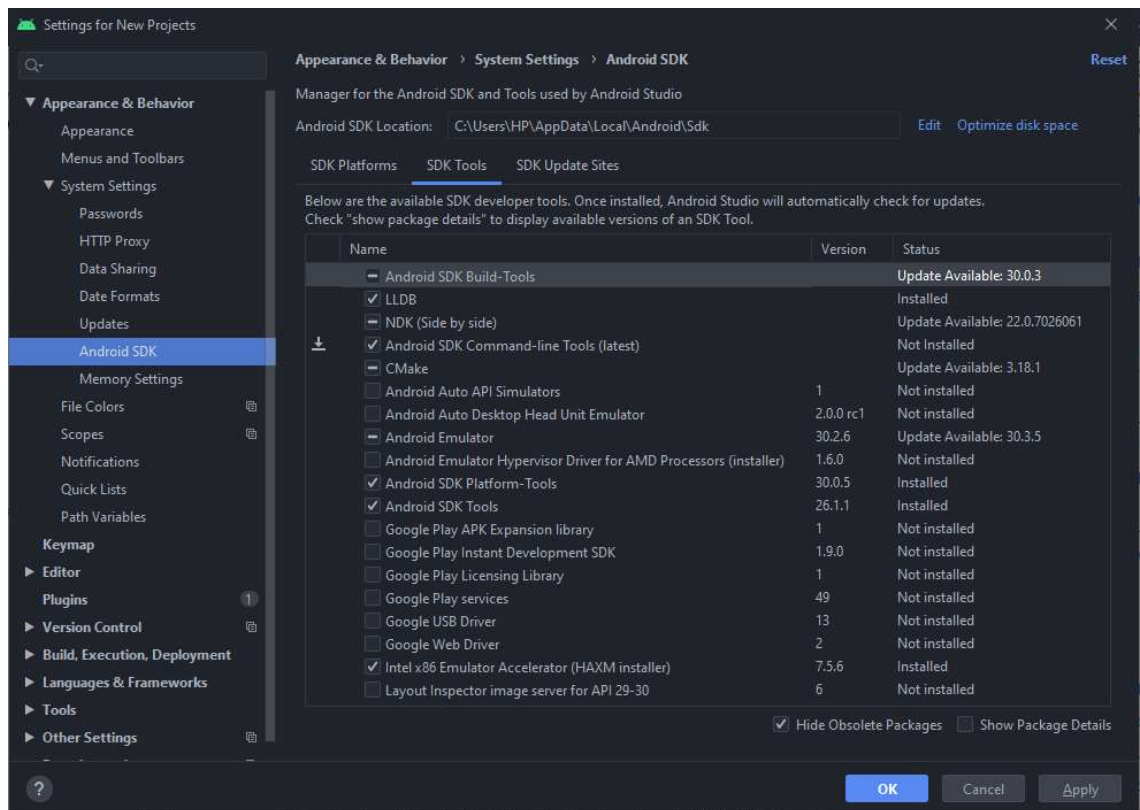
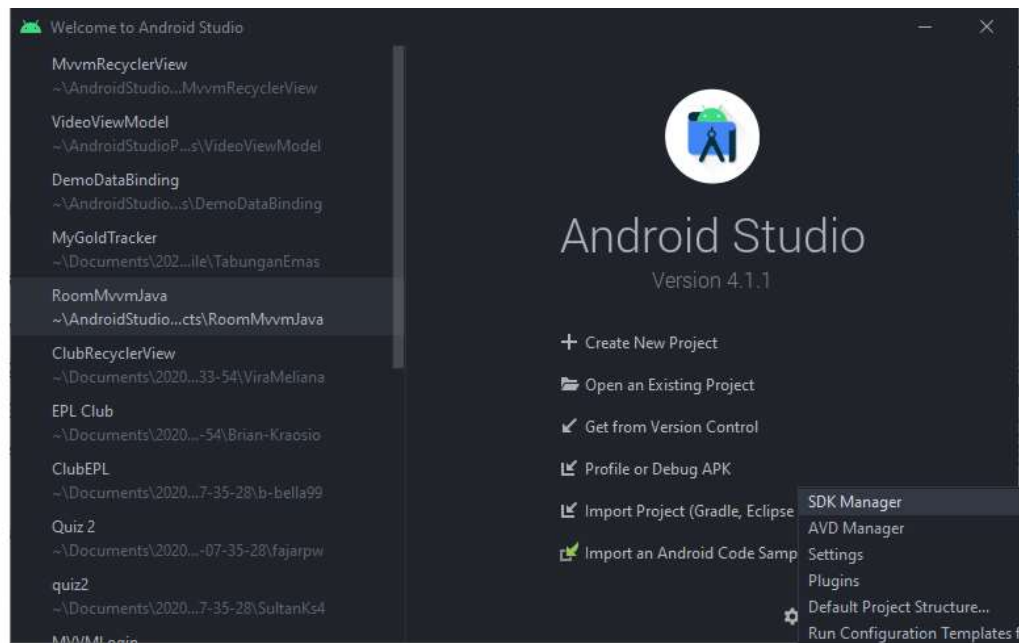
11. Jika tidak ada error lanjutkan ke langkah praktikum selanjutnya, jika muncul error seperti gambar dibawah ini.

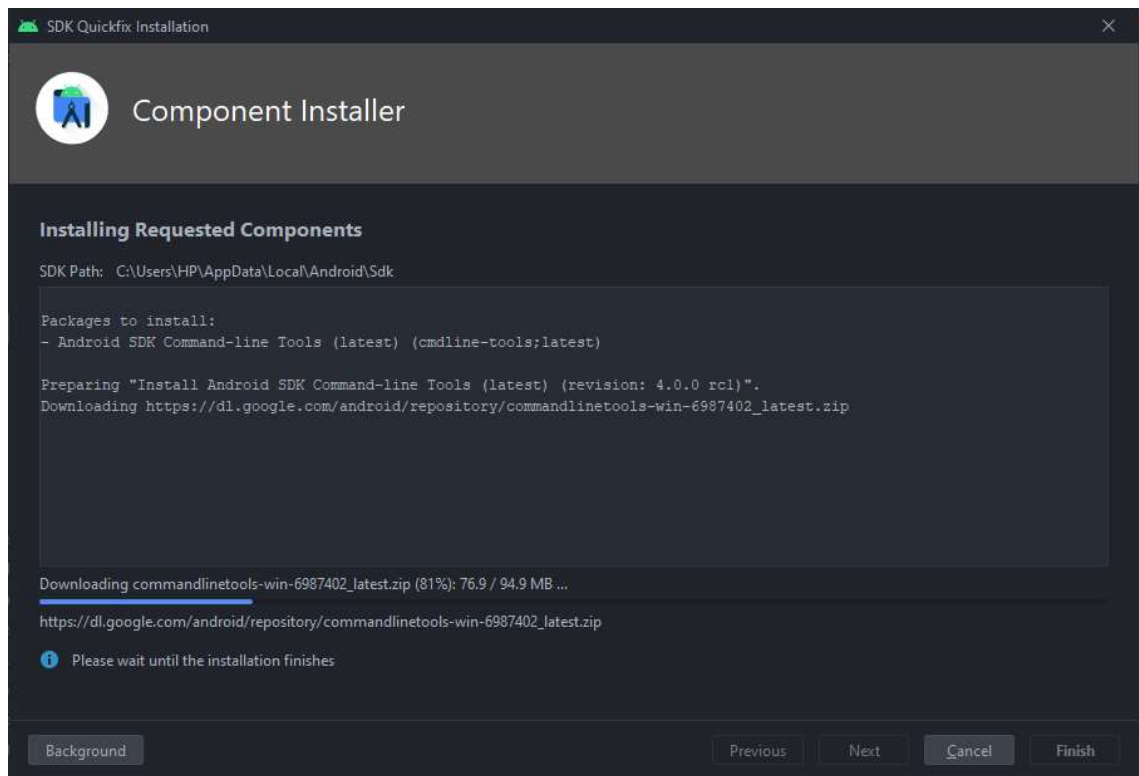


```
Windows PowerShell
> flutter doctor --android-licenses
Exception in thread "main" java.lang.NoClassDefFoundError: javax/xml/bind/annotation/XmlSchema
    at com.android.repository.api.SchemaModule$SchemaModuleVersion.<init>(SchemaModule.java:156)
    at com.android.repository.api.SchemaModule.<init>(SchemaModule.java:76)
    at com.android.sdklib.repository.AndroidSdkHandler.<clinit>(AndroidSdkHandler.java:81)
    at com.android.sdklib.tool.sdkmanager.SdkManagerCli.main(SdkManagerCli.java:73)
    at com.android.sdklib.tool.sdkmanager.SdkManagerCli.main(SdkManagerCli.java:48)
Caused by: java.lang.ClassNotFoundException: javax.xml.bind.annotation.XmlSchema
    at java.base/jdk.internal.loader.BuiltinClassLoader.loadClass(BuiltinClassLoader.java:583)
    at java.base/jdk.internal.loader.ClassLoaders$AppClassLoader.loadClass(ClassLoaders.java:178)
    at java.base/java.lang.ClassLoader.loadClass(ClassLoader.java:521)
    ... 5 more

~ took 2s
> flutter doctor --android-licenses
4 of 7 SDK package licenses not accepted. 100% Computing updates...
Review licenses that have not been accepted (y/N)? |
```

12. Lakukan update di Android SDK pada Android Studio dengan menginstall Android Command Line Tools pada android SDK langkah melakukannya lihat gamabar selanjutnya.





1.4 Praktikum Hello World

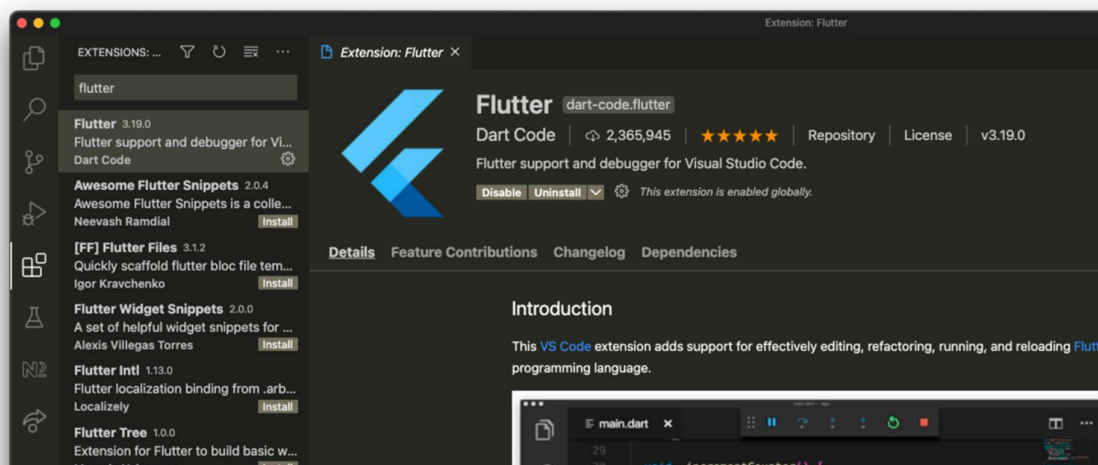
1.4.1 Alat dan Bahan

Pada praktikum kali ini anda akan membuat sebuah project hello world flutter

1. Visual Studio Code
2. Flutter SDK
3. Emulator

1.4.2 Langkah Langkah Praktikum

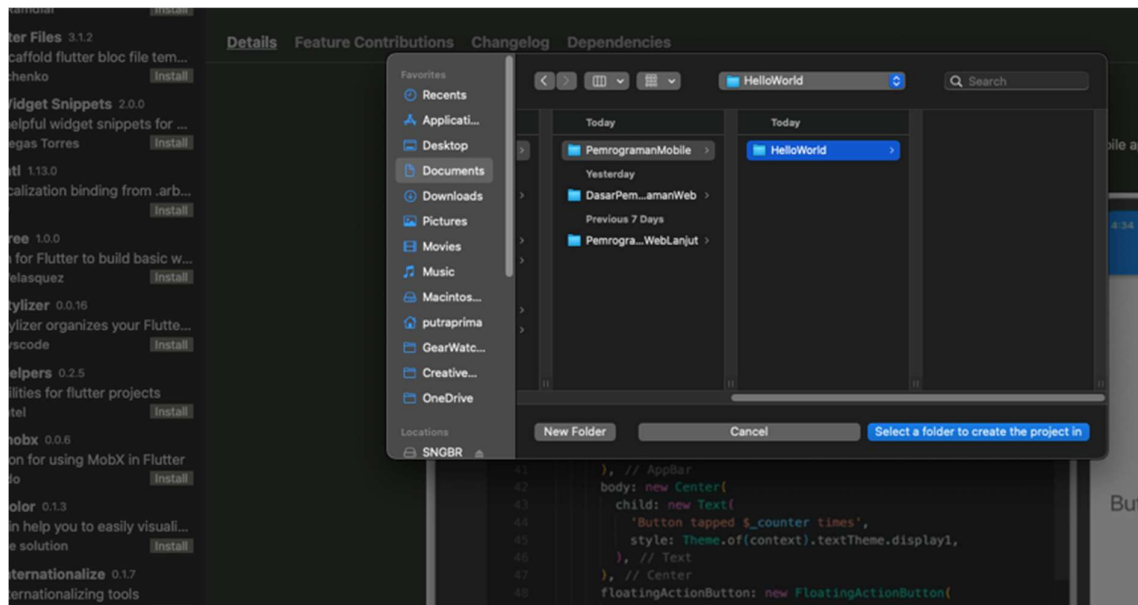
1. Install Flutter Plugin di Visual Studio Code



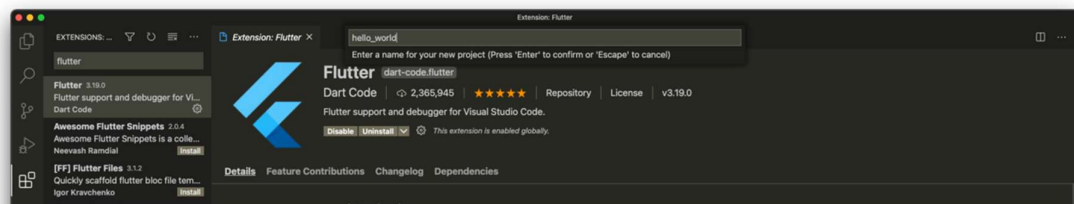
2. Tekan command Ctrl + Shift + P kemudian ketik perintah Flutter : New Application Project



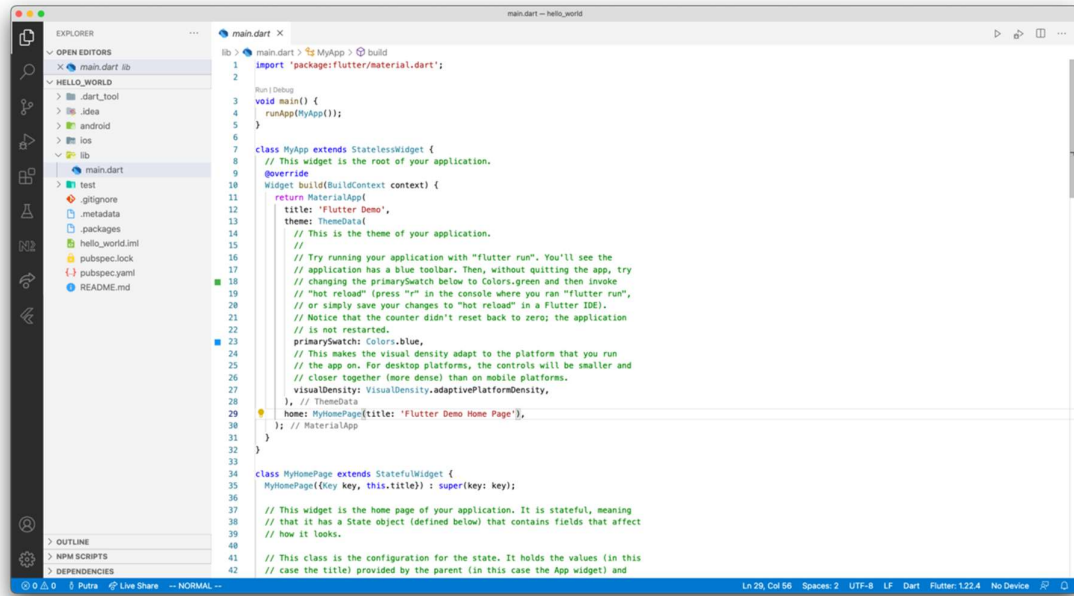
3. Pilih Folder tempat project akan dibuat



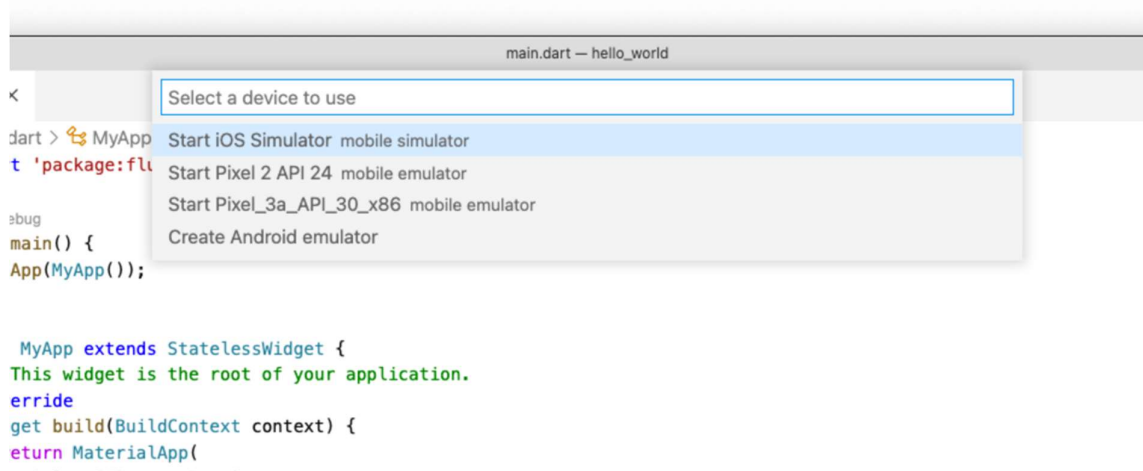
4. Berilah nama project yang sesuai Flutter mewajibkan nama project menggunakan huruf kecil dan dipisahkan dengan underscore contoh : hello_world



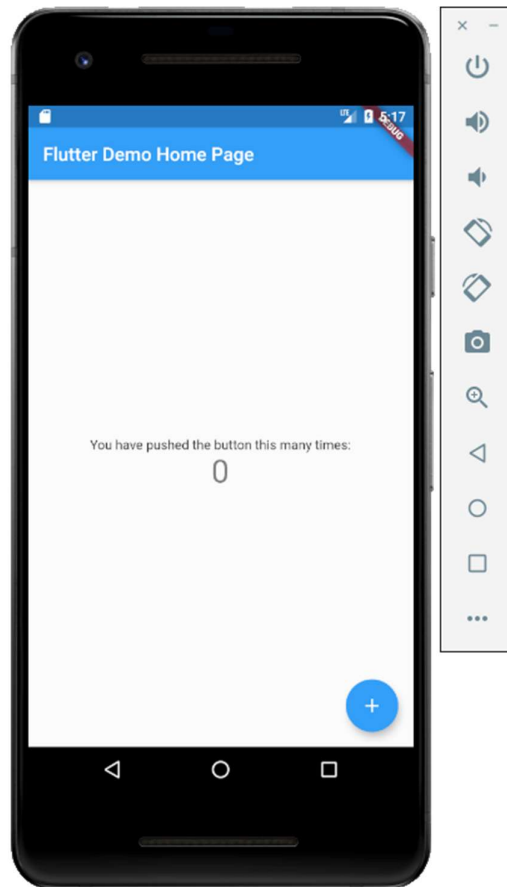
5. Berikut ini hasil project hello_world jika berhasil di generate



6. Untuk menjalankan project pastikan anda sudah melakukan setup emulator / setup device sesuai dengan perintah pada praktikum sebelumnya.
7. Jika emulator / device sudah disetup tekan tombol F5 untuk menjalankan aplikasi ke emulator / device.



8. Pilih emulator / device dan klik enter.



1.5 Praktikum 2 Membuat Git Repository dan Publish Ke Github

Pada praktikum kali ini anda akan mempublikasikan project flutter anda ke public repository github.

1.5.1 Alat dan Bahan

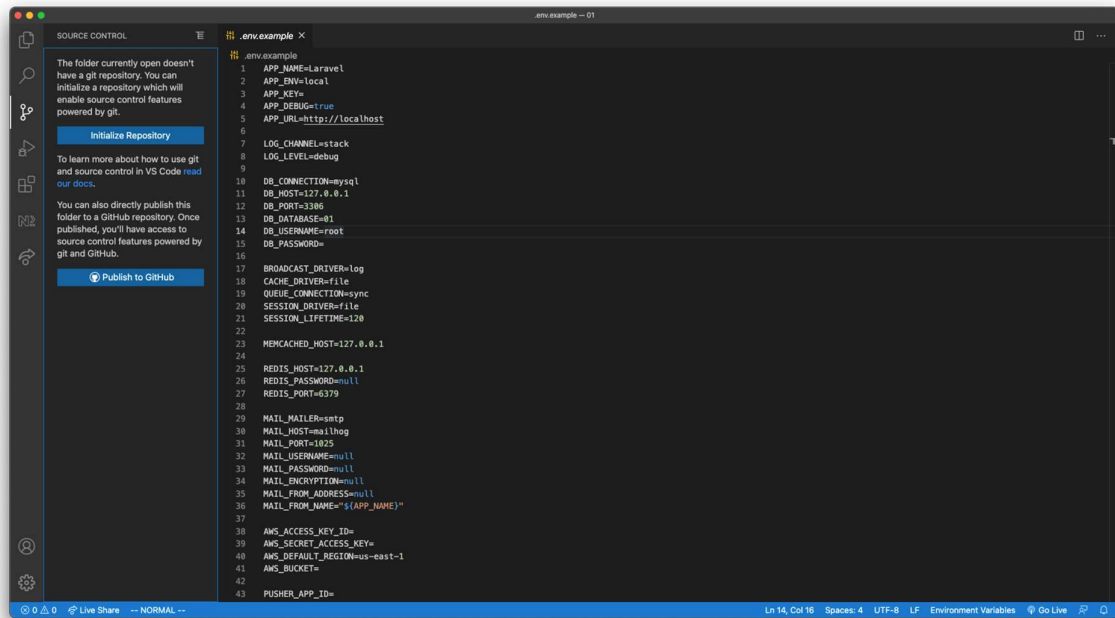
1. Visual Studio Code
2. Git
3. Akun Github
4. Source Code Flutter hello_world

1.5.2 Langkah Langkah praktikum

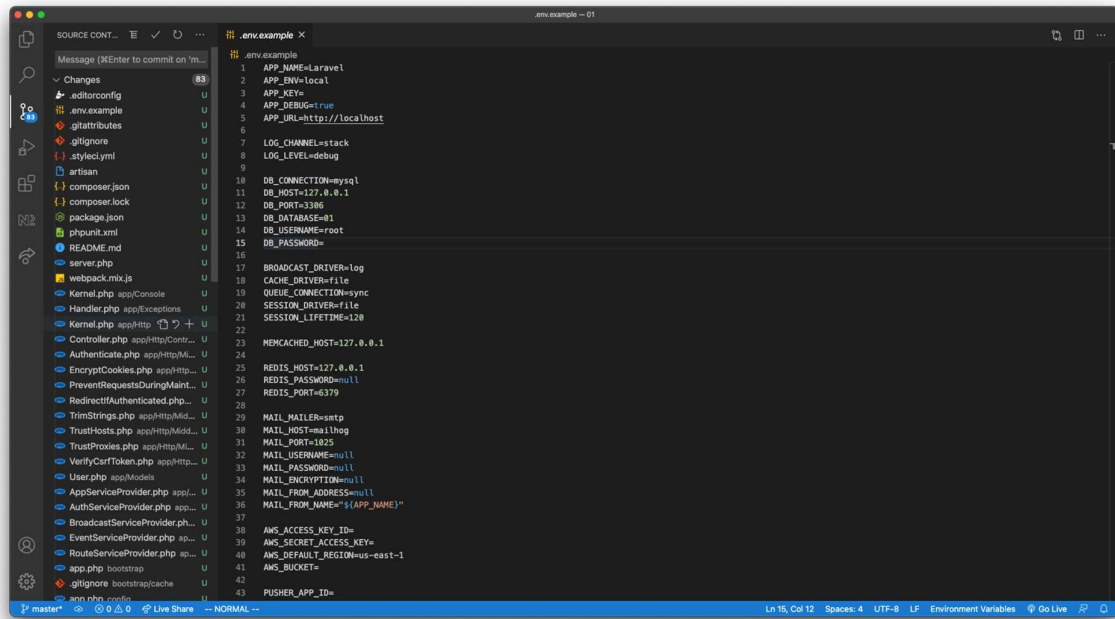
1. Mendaftar akun github ke <https://github.com/> , pilihlah username anda dengan bijaksana karena akun github anda akan sangat sering digunakan dan dapat dijadikan portofolio untuk mendaftar pekerjaan.
2. Update konfigurasi git di komputer anda sesuai dengan username dan email yang anda buat di github.com.
- 3.

```
git config --global user.name usernameAnda
git config --global user.email emailAnda
```

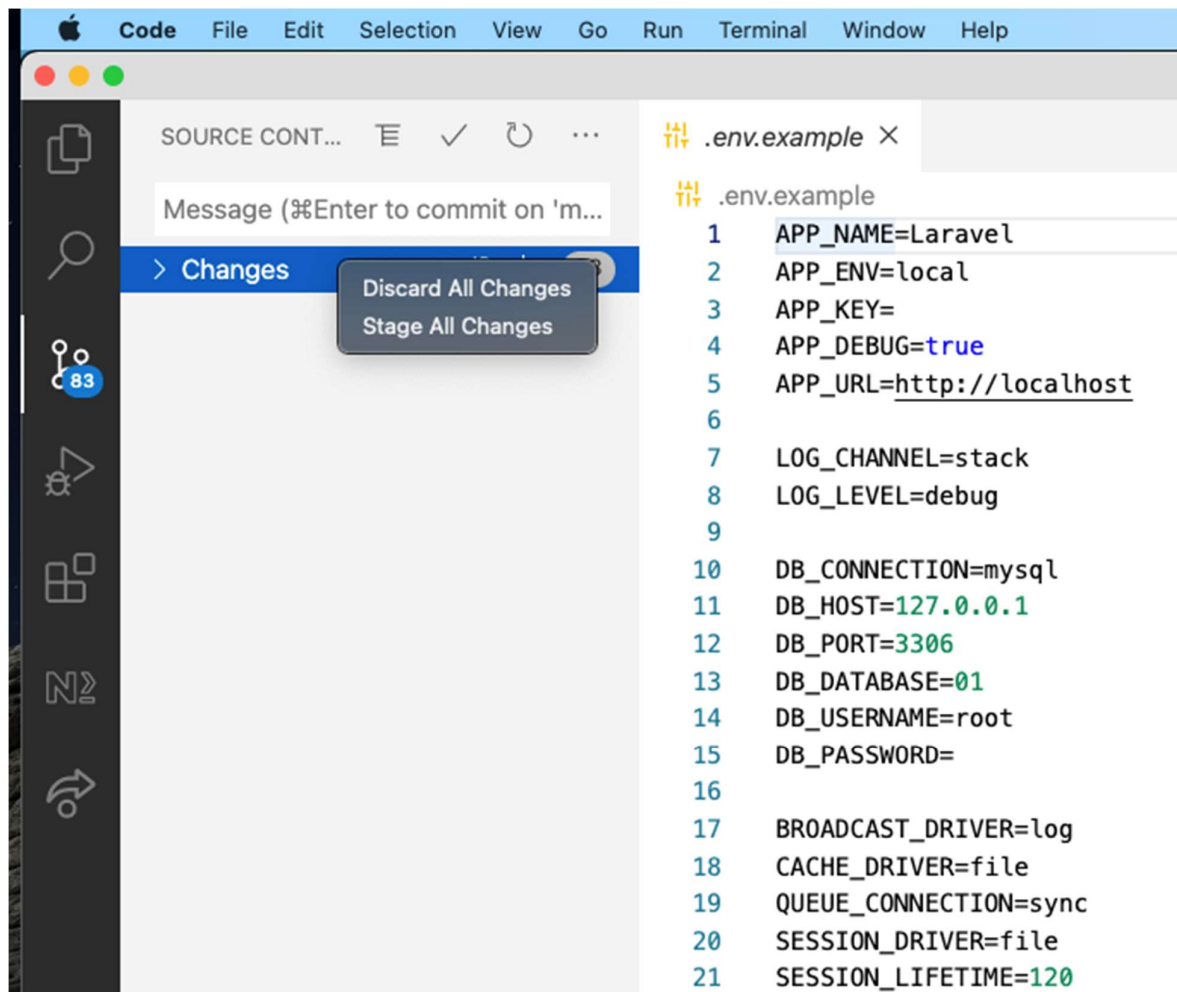
4. Bukalah project laravel yang anda buat sebelumnya pada Praktikum 2 menggunakan Visual Studio Code
5. Jika anda belum menginisialisasi repository git pada project tersebut akan keluar tampilan seperti ini ketika anda mengklik icon source control di sebelah kiri.



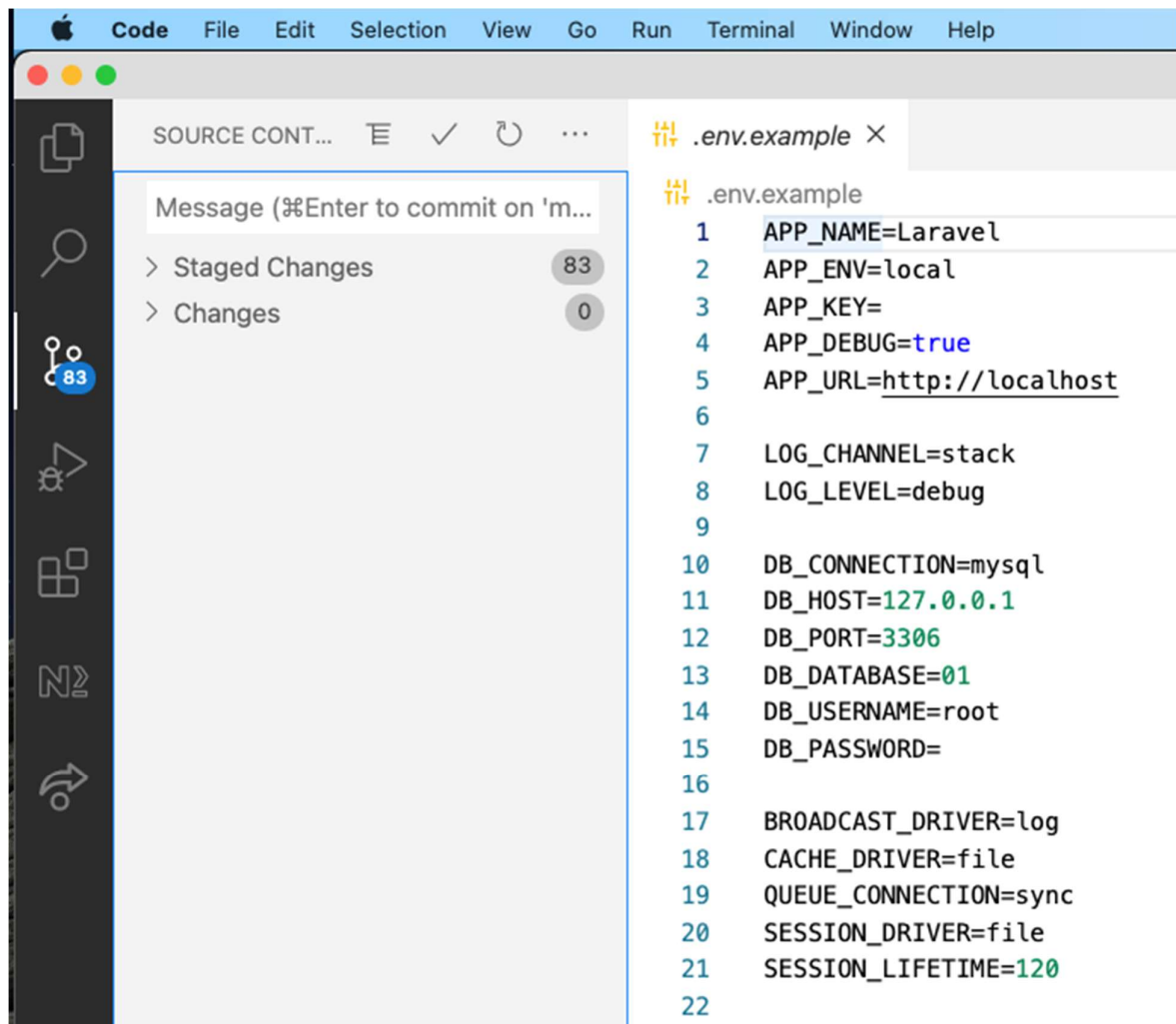
6. Perhatikan ada dua tombol yaitu Initialize Repository dan Publish to Github, Klik lah tombol Initialize Repository. Sidebar akan berubah menjadi seperti berikut ini.



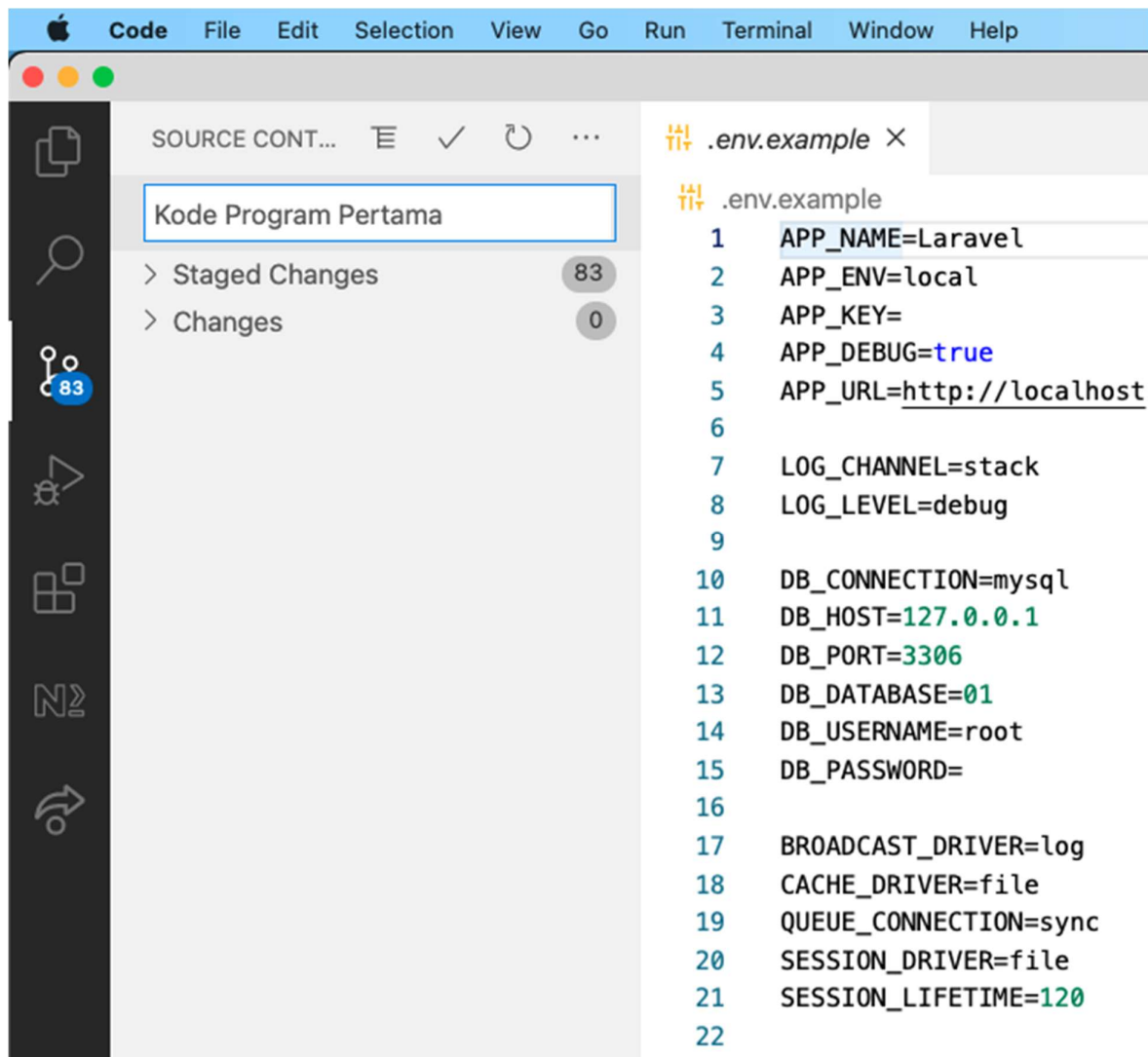
7. Perhatikan pada langkah ini kita sudah membuat repository git dan git mulai melacak perubahan yang ada pada root folder project kita. Perhatikan dengan seksama di sidebar terdapat daftar file yang berubah dibawah dropdown menu Changes dan status file nya memiliki status U dengan warna hijau.
8. Klik dropdown changes ini sehingga daftar file yang berubah diminimize, kemudian klik kanan dan klik menu “Stage All Changes”



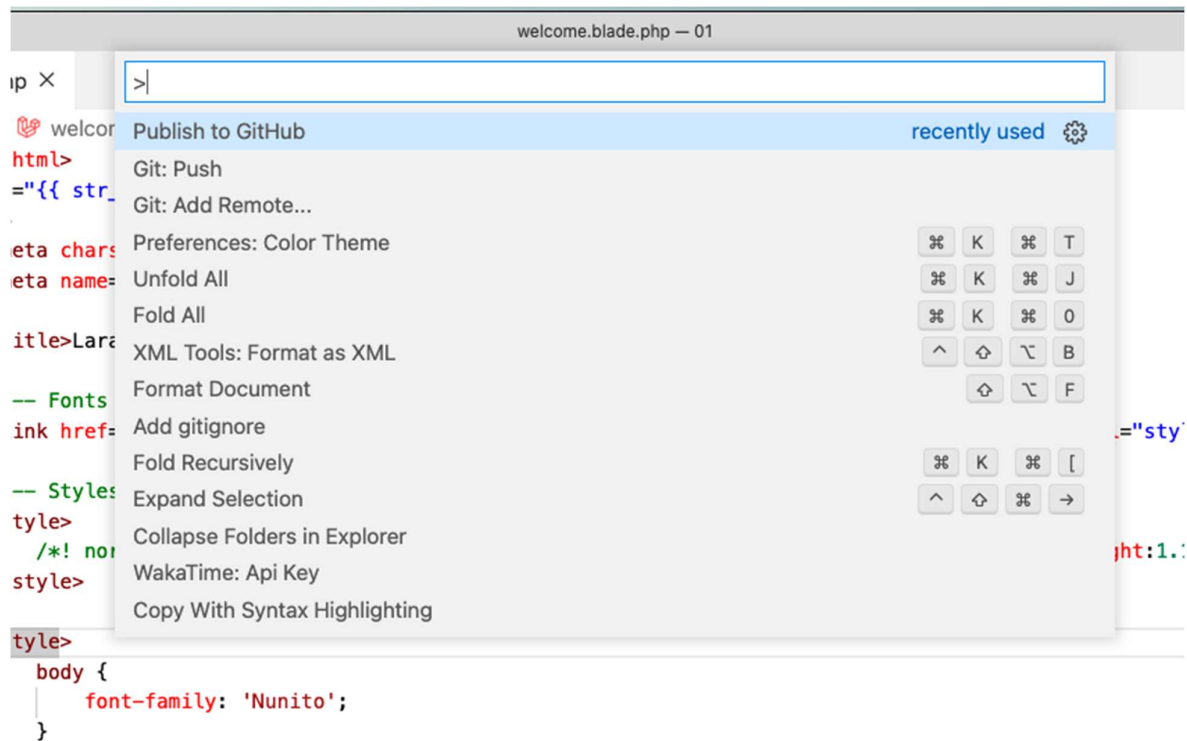
9. Setelah di klik git mencatat semua perubahan yang kita buat pada kode program dan sekarang semua file yang ada di Changes berpindah ke Dropdown Staged Changes.



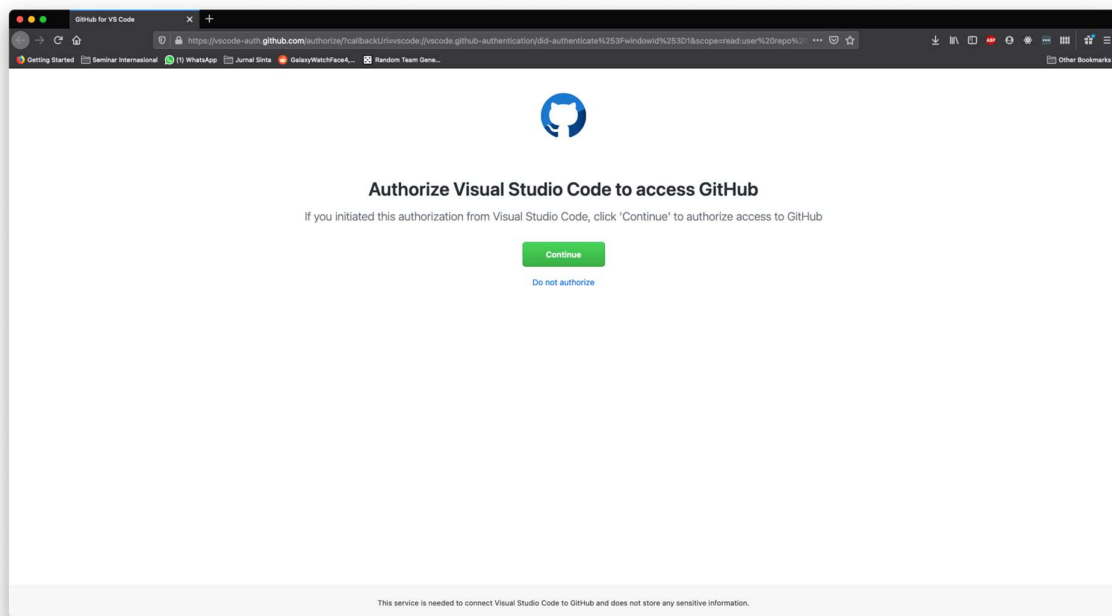
10. Isilah pesan tentang perubahan kode yang kita buat, pada kolom isian Message, kemudian klik tombol centang di atas.

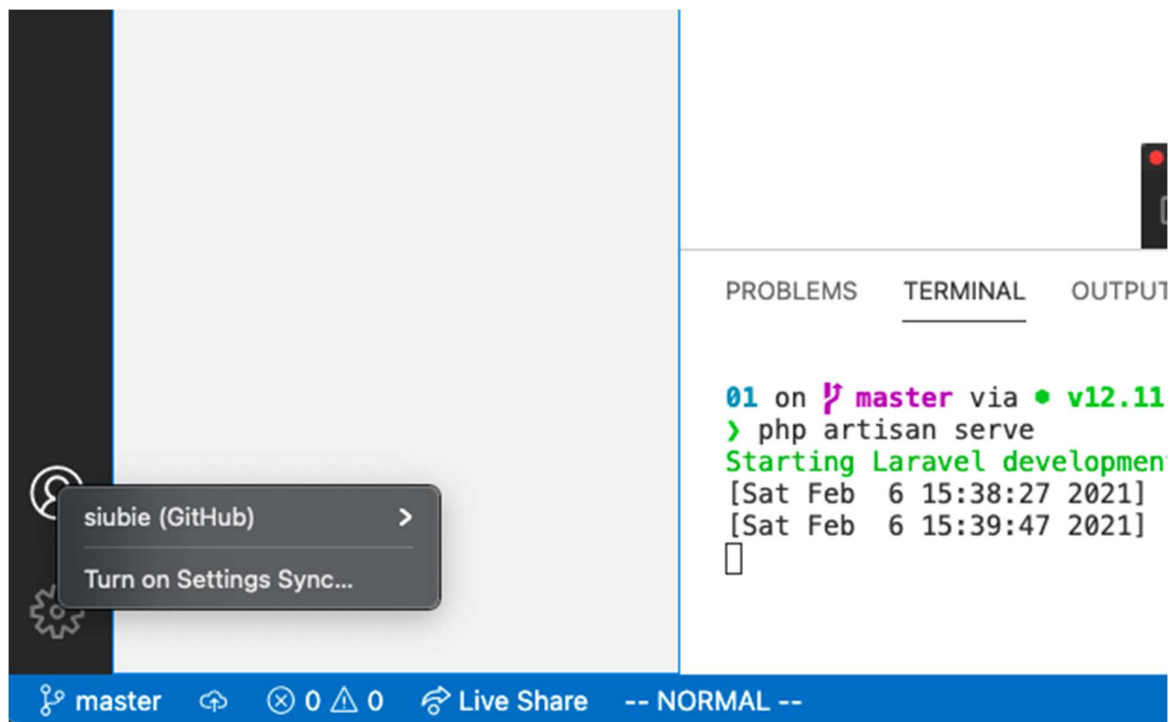
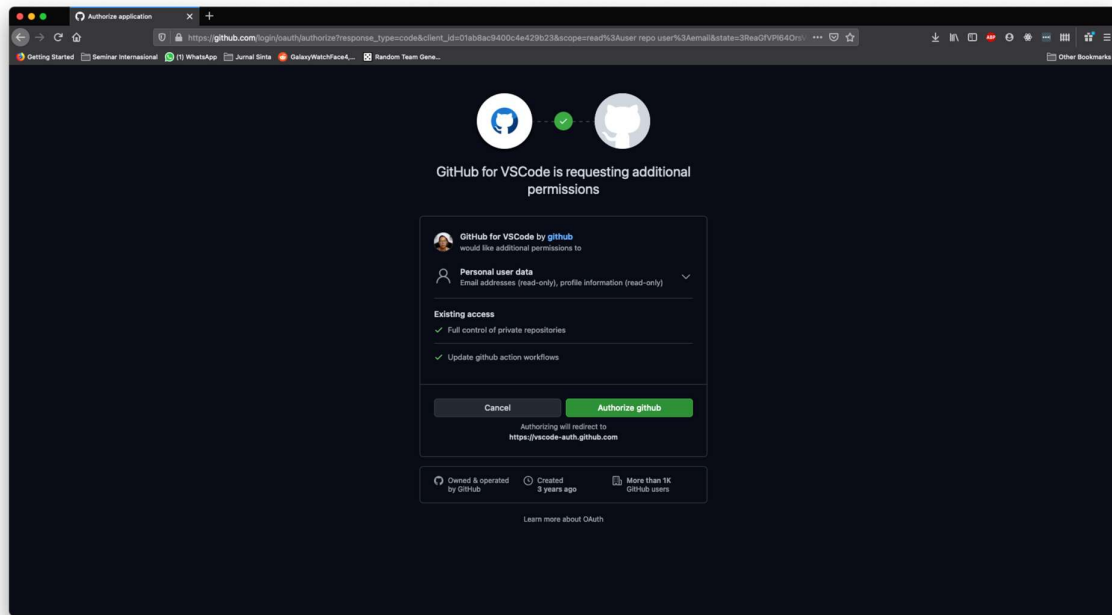


11. Jika di klik menu “Staged Change” akan hilang dan semua perubahan sudah di catat oleh git.
12. Selamat anda sudah berhasil menggunakan git pada repository lokal.
13. Selanjutnya untuk melakukan publikasi repository lokal ke github lakukan perintah berikut ini tekan Ctrl + Shift + P kemudian ketik publish to github

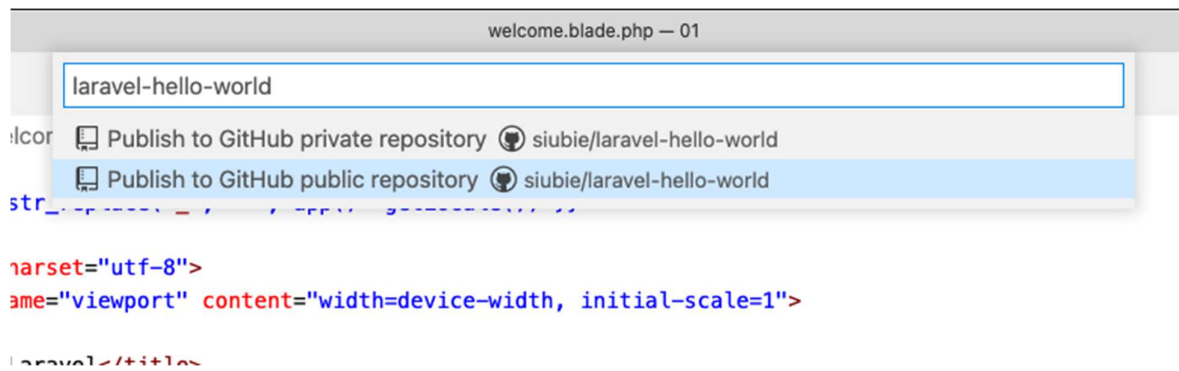


14. Visual studio code akan membuka browser dan meminta otorisasi untuk akun github



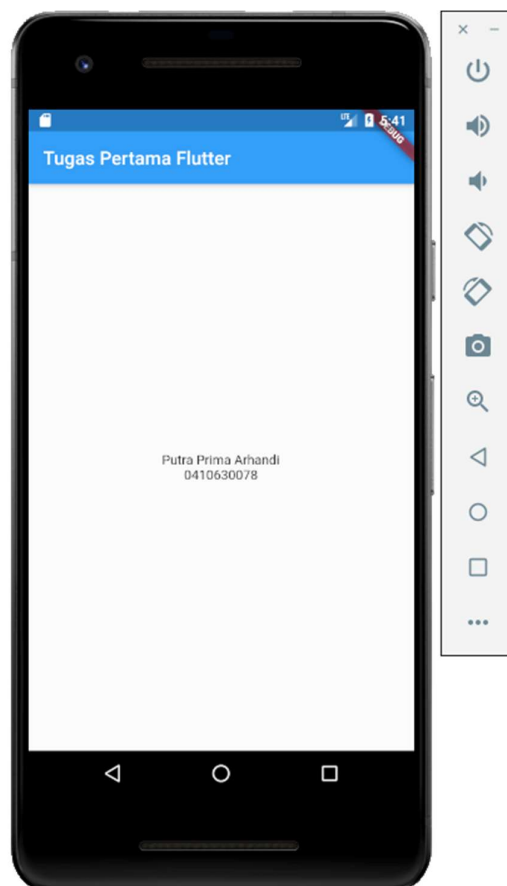


15. Setelah otorisasi berhasil anda dapat melanjutkan kembali command Ctrl + Shift + P dan pilih publish to github kemudian berilah nama flutter-hello-world



1.6 Tugas Praktikum

1. Pastikan anda dapat melakukan installasi flutter dan melakukan kompilasi project hello_world ke emulator / device android.
2. Push repository project hello_world anda ke github
3. Buat satu project lagi yang menggunakan flutter new project namun kali ini isi ubah lah tampilan aplikasi menjadi seperti gambar berikut dan publish ke repository github dengan nama repository polinema_mobile_flutter_hello.

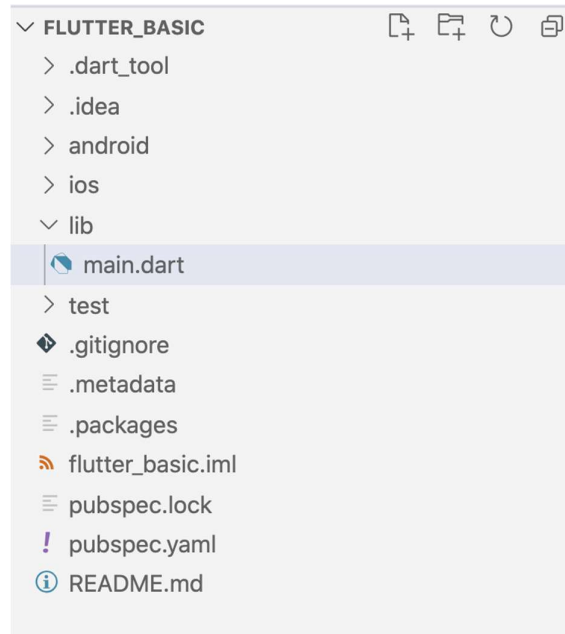


BAB 2. BASIC APLIKASI FLUTTER

2.1 Struktur Project Flutter

2.1.1 Struktur folder

Ketika membuat project flutter secara default berikut adalah struktur folder dan filenya. Dapat kita lihat strukturnya terdiri dari .dart_tool, .idea, android, ios, lib, test, .gitignore, .metadata, .packages, .flutter_basic.iml, pubspec.lock, pubspec.yaml, README.md.



Berikut adalah penjelasan yang lebih detail tentang struktur files dari flutter.

- .dart_tools : Konfigurasi untuk dart language
- .idea : Konfigurasi untuk android studio
- gitignore : File git yang digunakan untuk mengelola source code. Hal ini akan berguna jika developer sudah bekerja dengan git.
- metadata : File yang berisi metadata dari project
- packages : File yang berisi alamat path
- flutter_basic.iml: File yang berisi detail dari project.
- pubspec.lock : File yang berisi versi library atau package yang digunakan pada project yang degenerate sesuai dengan file pubspec.yaml.
- pubspec.yaml : File yang berisi library atau package yang dibutuhkan untuk pengembangan aplikasi.
- Readme.md : File markdown yang dapat digunakan untuk menjelaskan cara setup

aplikasi atau informasi penting yang perlu untuk diketahui oleh developer lain.

Untuk struktur folder akan dijelaskan pada subbab 2.1.2 sampai 2.1.7.

2.1.2 Multi os ios dan android

Perhatikan pada folder project flutter terdapat dua folder yaitu folder ios dan folder android, dengan menggunakan kedua folder tersebut flutter dapat membuat aplikasi berbasis ios dan berbasis android dalam satu project.

2.1.3 Folder android

Folder android berisi file file pendukung untuk mengenerate project android dan akan dikompilasi menjadi sebuah apk pada folder build. Namun anda jarang atau bahkan tidak perlu mengedit yang ada di folder android. Anda akan banyak bekerja di folder lib.

2.1.4 Folder ios

Berisi project ios, folder ini sama dengan folder android, sangat jarang dan bahkan tidak perlu untuk mengubah apapun pada folder ios. Folder ios dan android dikelola oleh flutter sdk yang akan dimerge (disatukan) dengan code yang ada di folder lib untuk membuat aplikasi ios dan android.

2.1.5 Folder lib

Folder lib berisi kode program dengan bahasa dart yang berupa widget yang dapat dibuat sesuai dengan kebutuhan aplikasi anda.

2.1.6 Folder test

Berisi source code dart yang digunakan untuk melakukan test secara otomatis pada aplikasi flutter.

2.1.7 Pubspec.yaml

Pada file ini berisi konfigurasi konfigurasi project flutter yang dibuat dimana anda dapat mendata asset berupa font, gambar dan lain lain. Pada file ini anda juga dapat mengkonfigurasi flutter sdk dan konfigurasi yang terkait flutter.

2.2 Flutter Hot Reload

Pada flutter terdapat fungsi hot reload dan hot restart yang digunakan untuk pengembangan aplikasi dengan flutter. Hote reload mencompile source code yang baru ditambahkan dan dikirimkan ke dart virtual machine diupdate. Setelah selesai update, dart virtual machine akan memperbarui UI sesuai dengan perubahan. Keunggulan hot reload adalah waktu prosenya cepat disbanding hot restart. Akan tetapi hot reload mempertahankan state yang ada sehingga jika menggunakan state maka nilai dari widget tidak akan berubah.

2.3 Flutter Hot Restart

Hot restart akan mencompile ulang aplikasi dan mereset (destroy) state yang ada. Jadi hot restart akan membuild ulang widget tree sesuai dengan code yang telah diperbarui.

2.4 Bedah Hello World Project

2.4.1 Import Statement

Seperti halnya kode program pada umumnya dart dapat menggunakan statement import untuk mengimport package, library, atau file lain yang digunakan pada file yang dieksekusi.

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

2.4.2 Main function

Main function pada flutter dibuat dengan menggunakan kode program berikut ini dimana semua proses aplikasi dimulai dari mengeksekusi fungsi main.

```
void main() {  
  runApp(MyApp());  
}
```

Perhatikan pada fungsi main ini yang di eksekusi adalah class MyApp dimana class MyApp harus mengextend salah satu StatelessWidget atau StatefulWidget, ingat bahwa flutter membangun UI menggunakan widget.

2.4.3 Stateless Widget

Flutter menggunakan Widget sebagai elemen elemen pembangun UI, widget ini adalah kode program yang diterjemahkan menjadi tampilan yang dapat dilihat dan diinteraksikan oleh pengguna. Stateless widget bersifat statis / final dimana nilai atau konfigurasi telah diinisiasi sejak awal, nilai variabel pada widget ini tidak dapat diubah oleh widget ini sendiri tetapi dapat diubah oleh parent widget nya jika parent nya adalah StatefulWidget. Struktur dasar stateless widget adalah sebagai berikut:


```
class exampleStateless extends StatelessWidget{  
  @override  
  Widget build(BuildContext context) {  
  
  }  
}
```

2.4.4 Statefull Widget

Statefull widget bersifat dinamis, widget ini dapat diperbarui ketika dibutuhkan sesuai dengan action pengguna atau jika ada perubahan data. Perubahan data pada statefull widget di trigger oleh perubahan state oleh karena itu sebuah StatefulWidget selalu memiliki State. Struktur dasar statefull widget adalah sebagai berikut:

```
class exampleStateless extends StatefulWidget{  
  @override  
  State<StatefulWidget> createState() {  
  }  
}
```

2.5 Build in Widget

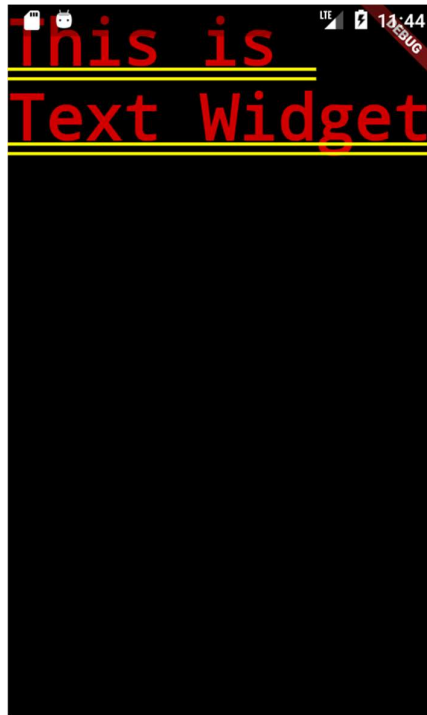
Pada framework flutter terdapat banyak widget. Widget yang telah disediakan dapat digunakan mengembangkan aplikasi yang mobile, desktop dan web yang memiliki tampilan menarik. Secara lebih detail widget yang tersedia di flutter dapat dicek pada <https://flutter.dev/docs/development/ui/widgets>. Berikut adalah beberapa contoh widget yang sering digunakan pada pembuatan aplikasi dengan flutter.

2.5.1 Text Widget

Text widget digunakan untuk menampilkan string yang dapat terdiri satu baris maupun beberapa baris. Contoh penggunaan text widget pada source code dan ouputnya adalah sebagai berikut:

```
class MyApp extends StatelessWidget {  
  @override  
  Widget build(BuildContext context) {  
    return MaterialApp(  
      home : Text('This is Text Widget',),  
    );  
  }  
}
```

```
}  
}
```



2.5.2 Image Widget

Image widget digunakan untuk menampilkan image. Contoh penggunaan image widget pada source code dan ouputnya adalah sebagai berikut:

```
class MyApp extends StatelessWidget {  
  @override  
  Widget build(BuildContext context) {  
    return MaterialApp(  
      home: Image(image: NetworkImage('https://flutter.github.io/assets-  
for-api-docs/assets/widgets/owl.jpg')),  
    );  
  }  
}
```



2.5.3 Material Design dan iOS Cupertino

Cupertino widget digunakan untuk mendesain sesuai dengan standar desain pada iOS.

Contoh penggunaan cupertino widget pada source code dan outputnya adalah sebagai berikut:

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

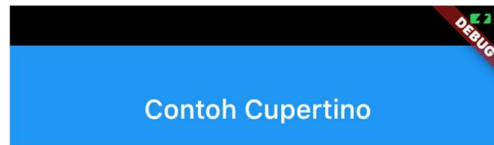
void main() => runApp(MyApp());

class MyApp extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      home: Container(
        margin: EdgeInsets.only(top: 30),
        color: Colors.white,
        child: Column(
          children: <Widget>[
            AppBar(title: Text('Contoh Cupertino')),
            CupertinoButton(
              child: Text("Contoh button"),
              onPressed: () {},
            ),
            CupertinoActivityIndicator(),
          ],
        ),
      ),
    );
  }
}
```

```

    ],
  ),
),
);
}
}

```



Contoh button



2.5.4 Button

Button widget terdapat beberapa macam pada flutter yaitu antara lain `AppBar`, `DropDownButton`, `FlatButton`, `FloatingActionButton`, `IconButton`, `OutlineButton`, `PopupMenuButton`, dan `RaisedButton`. Contoh penggunaan cupertino widget pada source code dan outputnya adalah sebagai berikut:

```

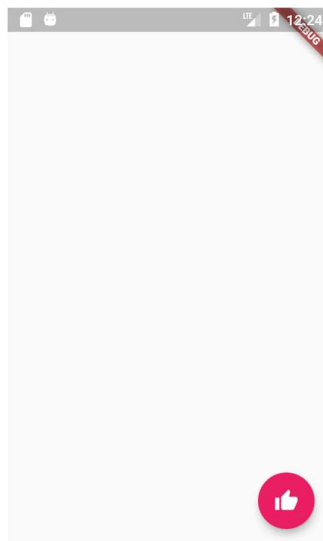
class MyApp extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      home: Scaffold(
        floatingActionButton:FloatingActionButton(

```

```

        onPressed: () {
          // Add your onPressed code here!
        },
        child: Icon(Icons.thumb_up),
        backgroundColor: Colors.pink,
      ),
    ),
  );
}
}

```



2.5.5 Scaffold

Scaffold widget digunakan untuk mengatur tata letak sesuai dengan material design. Contoh penggunaan scaffold widget pada source code dan ouputnya adalah sebagai berikut:

```

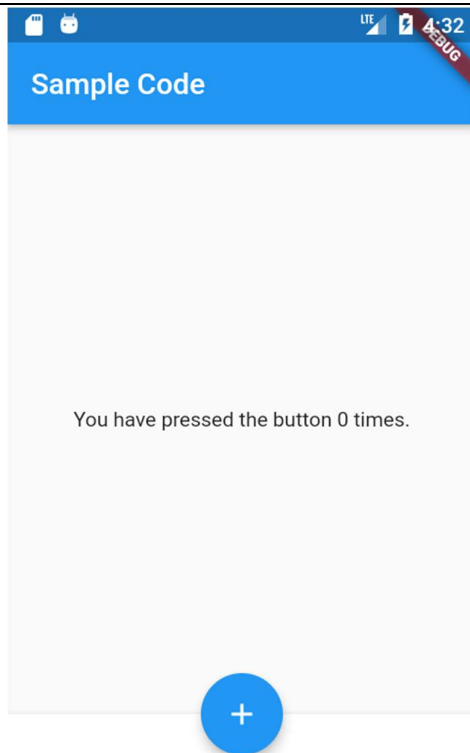
class MyApp extends StatelessWidget {
  int _count = 0;
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      home: Scaffold(
        appBar: AppBar(
          title: Text('Sample Code'),
        ),
        body: Center(
          child: Text('You have pressed the button $_count times.'),

```

```

    ),
    bottomNavigationBar: BottomAppBar(
      child: Container(
        height: 50.0,
      ),
    ),
    floatingActionButton: FloatingActionButton(
      onPressed: () => 0,
      tooltip: 'Increment Counter',
      child: Icon(Icons.add),
    ),
    floatingActionButtonLocation:
FloatingActionButtonLocation.centerDocked,
  ),
);
}
}

```



2.5.6 Dialog

Dialog widget pada flutter memiliki dua jenis dialog yaitu AlertDialog dan SimpleDialog. Contoh penggunaan AlertDialog widget pada source code dan ouputnya adalah sebagai berikut:

```
class MyApp extends StatelessWidget {
  int _count = 0;

  @override
  Widget build(BuildContext context) {
    return MaterialApp(

      home: Scaffold(
        body: MyLayout(),
      ),
    );
  }
}

class MyLayout extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return Padding(
      padding: const EdgeInsets.all(8.0),
      child: RaisedButton(
        child: Text('Show alert'),
        onPressed: () {
          showAlertDialog(context);
        },
      ),
    );
  }
}

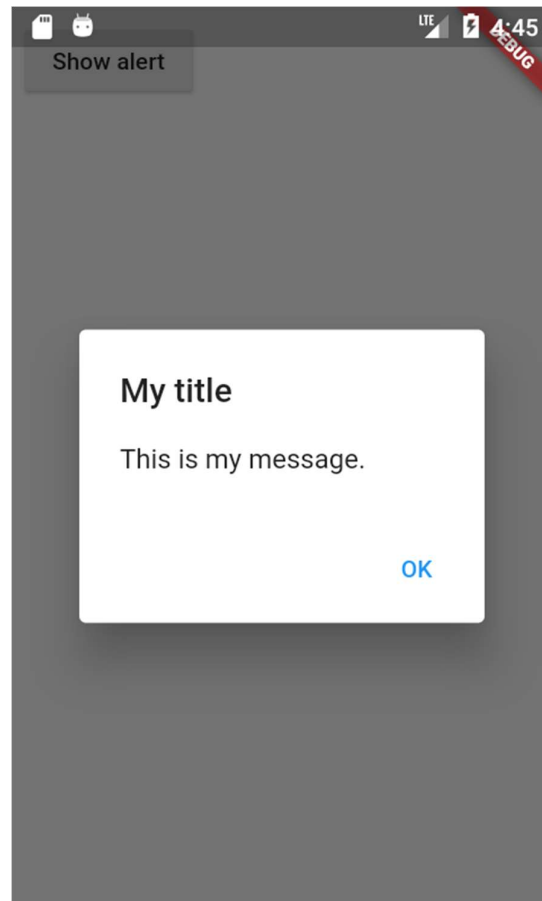
showAlertDialog(BuildContext context) {

  // set up the button
  Widget okButton = FlatButton(
    child: Text("OK"),
    onPressed: () { },
```

```
);

// set up the AlertDialog
AlertDialog alert = AlertDialog(
  title: Text("My title"),
  content: Text("This is my message."),
  actions: [
    okButton,
  ],
);

// show the dialog
showDialog(
  context: context,
  builder: (BuildContext context) {
    return alert;
  },
);
}
```

2.5.7 Input dan Selection Widget

Flutter menyediakan widget yang dapat menerima input dari pengguna aplikasi yaitu antara lain Checkbox, Date and Time Pickers, Radio Button, Slider, Switch, TextField. Contoh penggunaan TextField widget pada source code dan ouputnya adalah sebagai berikut:

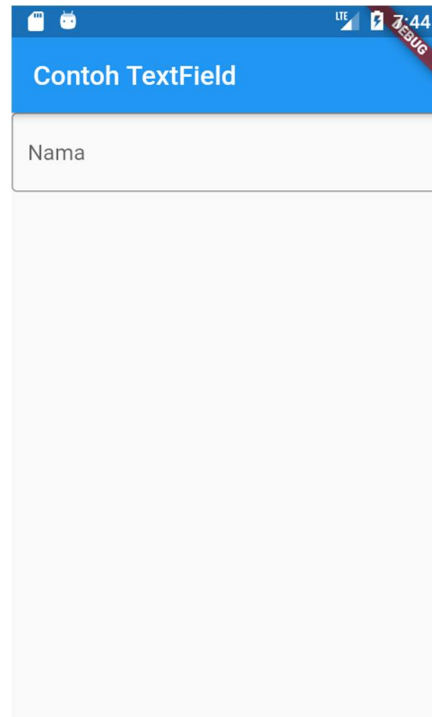
```
class MyApp extends StatelessWidget {  
  @override  
  Widget build(BuildContext context) {  
    return MaterialApp(  
  
      home: Scaffold(  
        appBar: AppBar(  
          title: Text("Contoh TextField")  
        ),  
        body: TextField(  
          obscureText: false,  
          decoration: InputDecoration(  
            border: OutlineInputBorder(),  

```

```

        labelText: 'Nama',
      ),
    ),
  ),
);
}
}

```



2.5.8 Date and Time Pickers

Date and Time Pickers termasuk pada kategori input dan selection widget, berikut adalah contoh penggunaan Date and Time Pickers.

```

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

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

class MyApp extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      title: 'Contoh Date Picker',
      home: MyHomePage(title: 'Contoh Date Picker'),
    );
  }
}

```

```

    }
}

class MyHomePage extends StatefulWidget {
  MyHomePage({Key key, this.title}) : super(key: key);

  final String title;

  @override
  _MyHomePageState createState() => _MyHomePageState();
}

class _MyHomePageState extends State<MyHomePage> {
  // Variable/State untuk mengambil tanggal
  DateTime selectedDate = DateTime.now();

  // Initial SelectDate Flutter
  Future<Null> _selectDate(BuildContext context) async {
    // Initial DateTime Final Picked
    final DateTime picked = await showDatePicker(
      context: context,
      initialDate: selectedDate,
      firstDate: DateTime(2015, 8),
      lastDate: DateTime(2101));
    if (picked != null && picked != selectedDate)
      setState(() {
        selectedDate = picked;
      });
  }

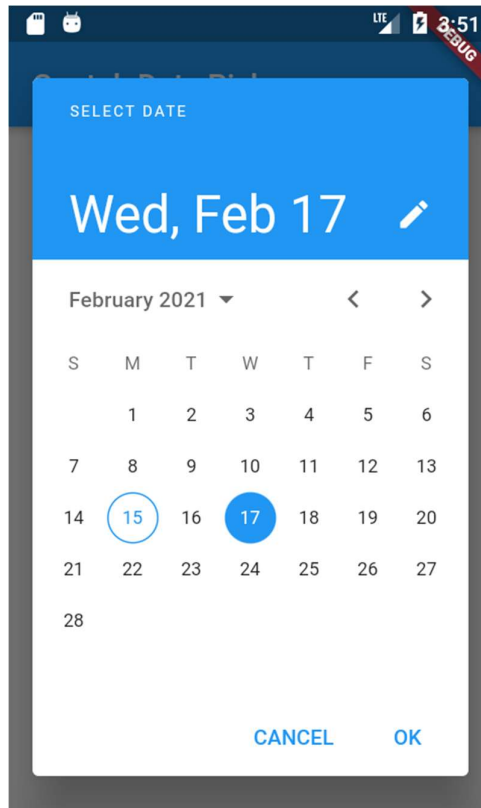
  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: Text(widget.title),
      ),
      body: Center(
        child: Column(
          mainAxisAlignment: MainAxisAlignment.min,
          children: <Widget>[
            Text("${selectedDate.toLocal()}".split(' ')[0]),
            SizedBox(height: 20.0),
            RaisedButton(
              onPressed: () => {
                _selectDate(context),
                print(selectedDate.day + selectedDate.month +
selectedDate.year )
              },
              child: Text('Pilih Tanggal'),
            ),
          ],
        ),
      ),
    );
  }
}

```

```

    ],
  ),
),
);
}
}

```



2.6 Build in Layout Widget

2.6.1 Container

Container widget berguna untuk menyimpan berbagai macam attribute dan menampung berbagai macam fungsi objek. Container merupakan **single child objek** yang artinya hanya dapat memiliki satu buah child widget, akan tetapi dalam sebuah container kita dapat menempatkan row, column, text dan container lain. Container memiliki beberapa property yaitu:

- A. property child : digunakan untuk membuat menampung widget didalam container. Widget yang ditampung antara lain Text, Column, ListView, Buton dan lain sebagainya.

```

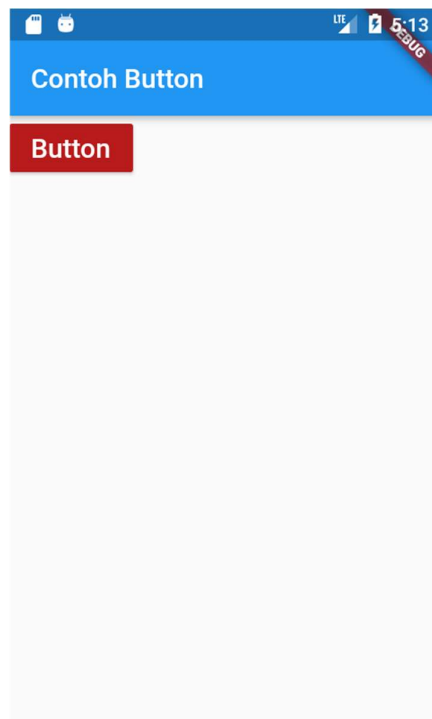
class MyApp extends StatelessWidget {
  @override
  Widget build(BuildContext context) {

```

```

return MaterialApp(
  home: Scaffold(
    appBar: AppBar(title: Text("Contoh Button")),
    body: Container(
      child: RaisedButton(
        textColor: Colors.white,
        onPressed: () {},
        color: Colors.red[900],
        child: Text(
          "Button",
          style: TextStyle(fontSize: 20)
        ),
      ),
    ),
  ),
);
}
}

```



B. property alignment : mengatur posisi child widget menggunakan property Alignment.

```

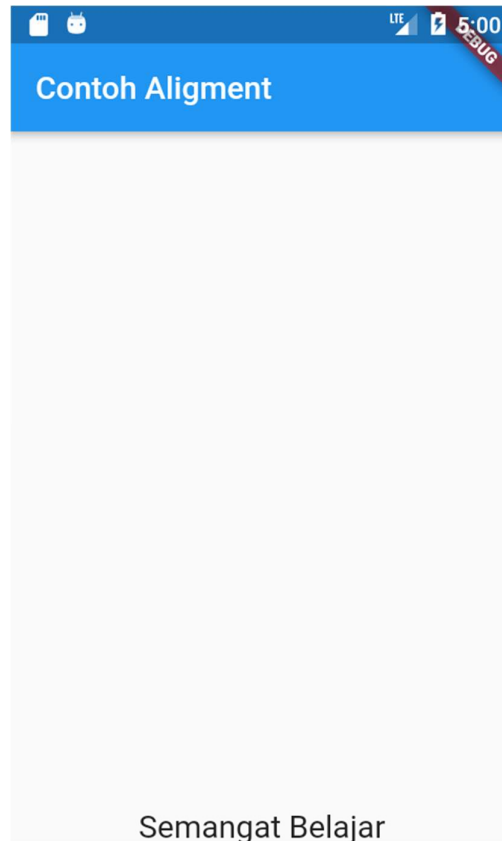
class MyApp extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      home: Scaffold(
        appBar: AppBar(title: Text("Contoh Aligment")),
        body: Container(
          alignment: Alignment.bottomCenter,
          child: Text(
            'Semangat Belajar',

```

```

        style: TextStyle(
          fontSize: 20,
        ),
      )),
    );
  }
}

```



- C. property color : digunakan untuk mengubah wara latar belakang container. Untuk memilih warna dapat menghover warna maka akan muncul pilihan warna yang dapat kita gunakan seperti berikut:

Colors.amber[50]	0xFFFFF8E1
Colors.amber[100]	0xFFFFECB3
Colors.amber[200]	0xFFFFE082
Colors.amber[300]	0xFFFFD54F

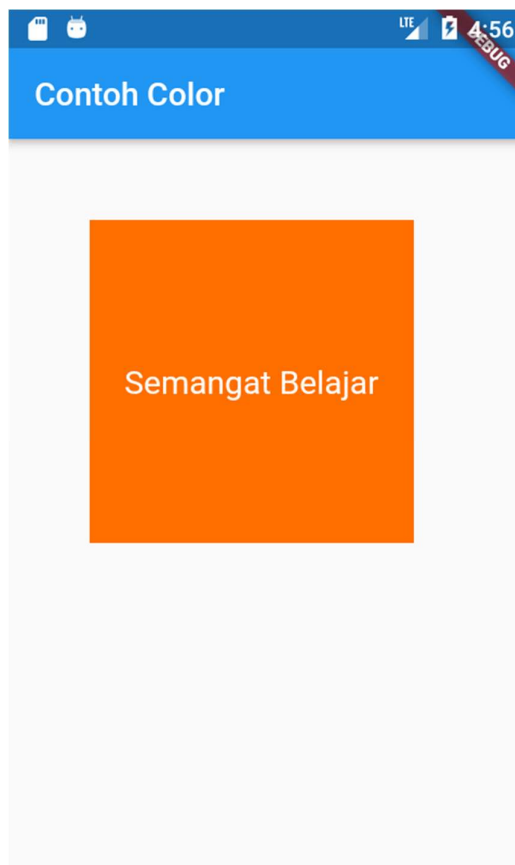
```

color: Colors.amber,

```

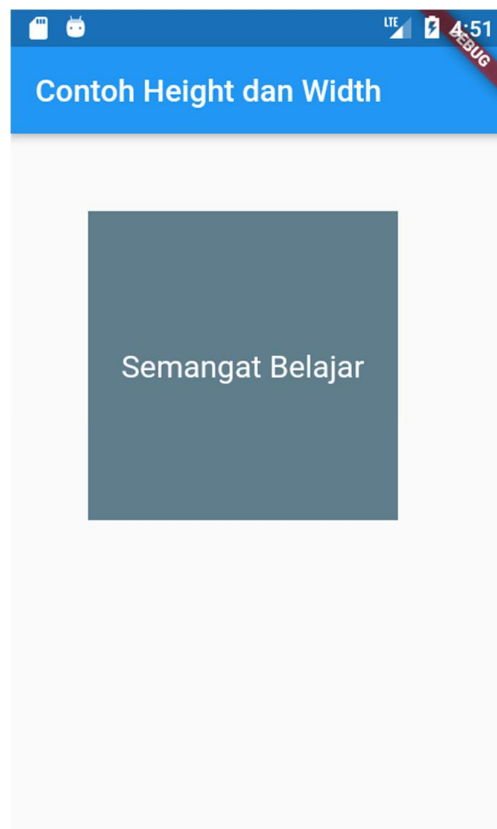
Contoh penggunaan color pada container adalah sebagai berikut:

```
class MyApp extends StatelessWidget {  
  @override  
  Widget build(BuildContext context) {  
    return MaterialApp(  
      home: Scaffold(  
        appBar: AppBar(title: Text("Contoh Color")),  
        body: Container(  
          margin: EdgeInsets.all(50),  
          height: 200,  
          width: 200,  
          alignment: Alignment.center,  
          color: Colors.amber[900],  
          child: Text(  
            'Semangat Belajar',  
            style: TextStyle(fontSize: 20, color: Colors.white),  
          )),  
      );  
    }  
  }  
}
```



D. Property height dan width : Secara default ukuran container menyesuaikan dengan ukuran body layar, maka untuk mengatur layoutnya dapat menggunakan property height dan width.

```
class MyApp extends StatelessWidget {  
  @override  
  Widget build(BuildContext context) {  
    return MaterialApp(  
      home: Scaffold(  
        appBar: AppBar(title: Text("Contoh Height dan Width")),  
        body: Container(  
          margin: EdgeInsets.all(50),  
          height: 200,  
          width: 200,  
          alignment: Alignment.center,  
          color: Colors.blueGrey,  
          child: Text(  
            'Semangat Belajar',  
            style: TextStyle(fontSize: 20, color: Colors.white),  
          )),  
      ),  
    );  
  }  
}
```

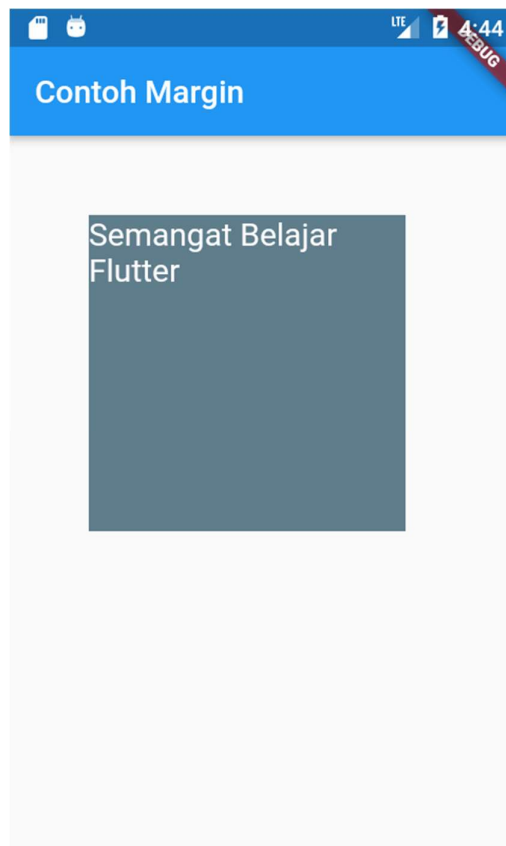


E. property margin : membuat jarak container dengan dengan widget lainnya.


```

class MyApp extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      home: Scaffold(
        appBar: AppBar(title: Text("Contoh Margin")),
        body: Container(
          margin: EdgeInsets.all(50),
          height: 200,
          width: 200,
          alignment: Alignment.topLeft,
          color: Colors.blueGrey,
          child: Text(
            'Semangat Belajar Flutter',
            style: TextStyle(fontSize: 20, color: Colors.white),
          )),
      ),
    );
  }
}

```

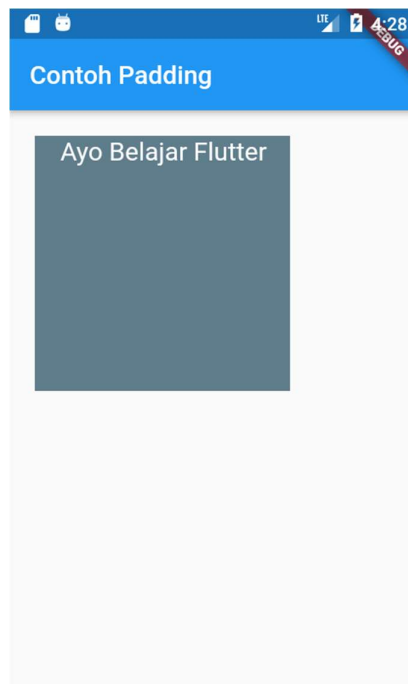


- F. property padding : digunakan untuk menambahkan jarak antara container dengan widget yang ada didalam container tersebut.

```

class MyApp extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      home: Scaffold(
        appBar: AppBar(title: Text("Contoh Padding")),
        body: Container(
          padding: EdgeInsets.only(left: 20),
          margin: EdgeInsets.all(20),
          height: 200,
          width: 200,
          alignment: Alignment.topLeft,
          color: Colors.blueGrey,
          child: Text(
            'Ayo Belajar Flutter',
            style: TextStyle(fontSize: 20, color: Colors.white),
          )),
      ),
    );
  }
}

```



G. property transform : berfungsi untuk melakukan rotasi pada container dengan melakukan dari berbagai sumbu putar misalnya X,Y, dan Z.

```

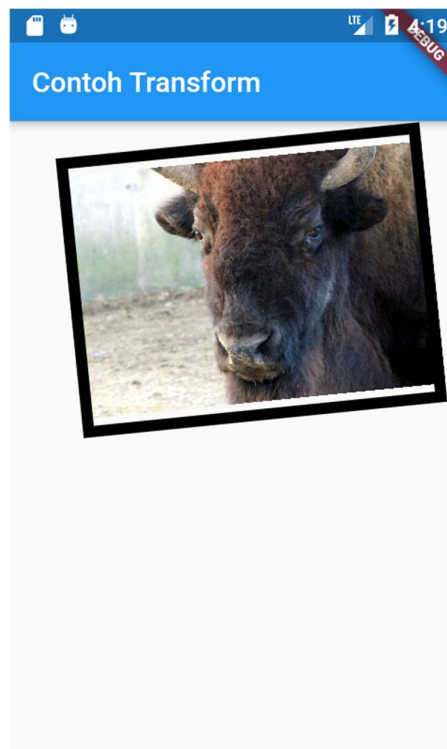
class MyApp extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      home: Scaffold(
        appBar: AppBar(title: Text("Contoh Transform")),

```

```

        body: Container(
          decoration: BoxDecoration(
            image: const DecorationImage(
              image: NetworkImage(
                'https://pixnio.com/free-
images/2017/03/07/2017-03-07-10-59-39-900x600.jpg'),
              fit: BoxFit.fitWidth,
            ),
            border: Border.all(
              color: Colors.black,
              width: 8,
            ),
          ),
          height: 200,
          width: 300,
          margin: const EdgeInsets.only(left: 30.0, right: 30.0,
top: 30),
          transform: Matrix4.rotationZ(-0.1),
        ),
      ),
    );
  }
}

```

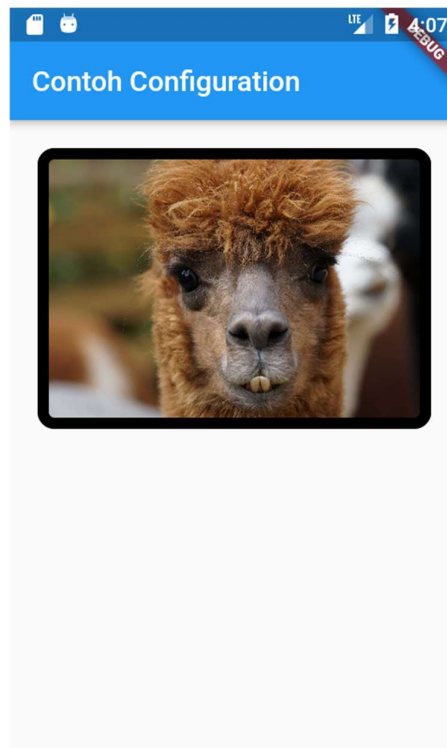


H. property decoration : untuk mencustom container dengan berbagai macam efek misalnya dengan mengubah warna border, memberikan gambar, dan membuat efek bayangan.

```

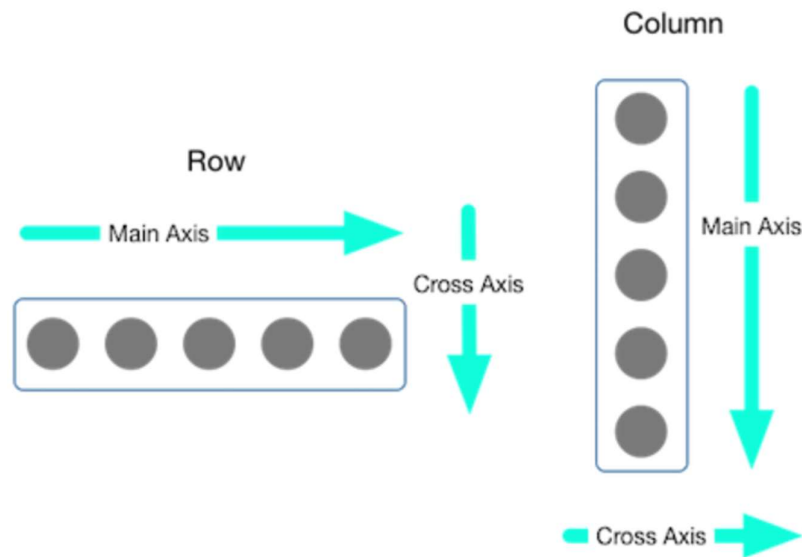
class MyApp extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      home: Scaffold(
        appBar: AppBar(title: Text("Contoh Configuration")),
        body: Container(
          decoration: BoxDecoration(
            color: const Color(0xff7c94b6),
            image: const DecorationImage(
              image: NetworkImage(
                'https://pixnio.com/free-
images/2018/12/02/2018-12-02-19-17-12.jpg'),
              fit: BoxFit.fitWidth,
            ),
            border: Border.all(
              color: Colors.black,
              width: 8,
            ),
            borderRadius: BorderRadius.circular(12),
          ),
          height: 200,
          width: 300,
          margin: EdgeInsets.all(20))),
    );
  }
}

```



2.6.2 Row dan Column

Column widget digunakan untuk mengatur tata letak widget secara vertikal. Sedangkan row digunakan untuk mengatur tata letak widget secara horizontal. Berikut adalah gambaran perbedaan row dan widget adalah sebagai berikut:



(sumber : <https://flutter.dev/docs>)

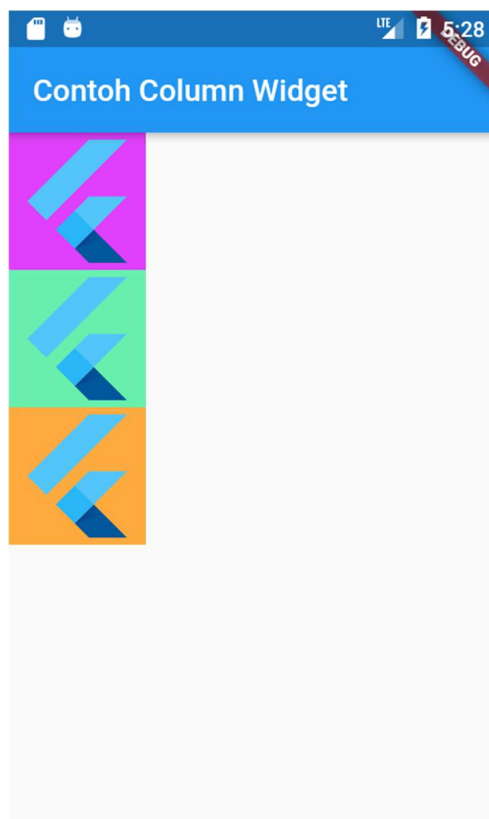
- Berikut adalah contoh penggunaan Column Widget

```
class MyApp extends StatelessWidget {  
  @override  
  Widget build(BuildContext context) {  
    return MaterialApp(  
      home: Scaffold(  
        appBar: AppBar(title: Text("Contoh Column Widget")),  
        body: Column(  
          children: [  
            Container(  
              color: Colors.purpleAccent,  
              child: FlutterLogo(  
                size: 90.0,  
              ),  
            ),  
            Container(  
              color: Colors.greenAccent,  
              child: FlutterLogo(  
                size: 90.0,  
              ),  
            ),  
          ],  
        ),  
      ),  
    );  
  }  
}
```

```

    ),
    Container(
      color: Colors.orangeAccent,
      child: FlutterLogo(
        size: 90.0,
      ),
    ),
  ],
),
),
);
}
}

```



- Berikut adalah contoh penggunaan Row Widget

```

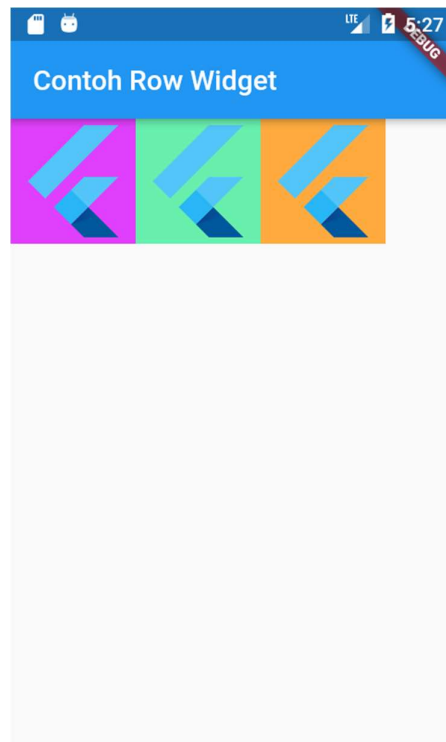
class MyApp extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      home: Scaffold(
        appBar: AppBar(title: Text("Contoh Row Widget")),
        body: Row(
          children: [
            Container(
              color: Colors.purpleAccent,

```

```

        child: FlutterLogo(
          size: 90.0,
        ),
      ),
      Container(
        color: Colors.greenAccent,
        child: FlutterLogo(
          size: 90.0,
        ),
      ),
      Container(
        color: Colors.orangeAccent,
        child: FlutterLogo(
          size: 90.0,
        ),
      ),
    ],
  ),
);
}
}

```



2.6.3 Stack

Stack Widget digunakan untuk menumpuk beberapa widget pada beberapa lapisan.

```

class MyApp extends StatelessWidget {
  @override

```

```

Widget build(BuildContext context) {
  return MaterialApp(
    debugShowCheckedModeBanner: false,
    home: Scaffold(
      appBar: AppBar(
        title: Text("Contoh Stack Widget"),
      ),
      body: Stack(
        children: <Widget>[
          Container(
            color: Colors.green,
            alignment: Alignment.bottomCenter,
            child: Text("Satu", style: TextStyle(fontSize:30,color:
Colors.white)),
          ),
          Container(
            color: Colors.red,
            alignment: Alignment.bottomCenter,
            child: Text("Dua", style: TextStyle(fontSize:30,color:
Colors.white)),
            height: 400.0,
            width: 300.0,
          ),
          Container(
            color: Colors.deepPurple,
            alignment: Alignment.bottomCenter,
            child: Text("Tiga", style: TextStyle(fontSize:30,color:
Colors.white)),
            height: 200.0,
            width: 200.0,
          ),
        ],
      ),
    ),
  );
}

```




2.6.4 ListView

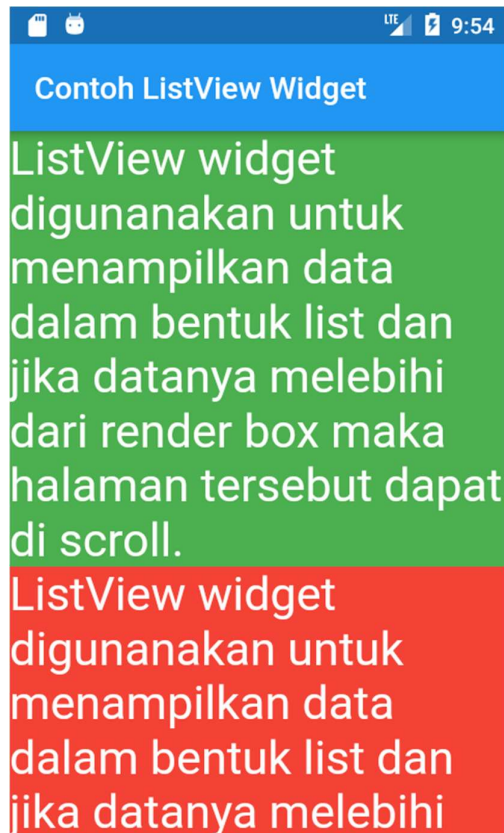
ListView widget digunakan untuk menampilkan data dalam bentuk list dan jika datanya melebihi dari render box maka halaman tersebut dapat di scroll.

```
class MyApp extends StatelessWidget {  
  @override  
  Widget build(BuildContext context) {  
    return MaterialApp(  
      debugShowCheckedModeBanner: false,  
      home: Scaffold(  
        appBar: AppBar(  
          title: Text("Contoh ListView Widget"),  
        ),  
        body: ListView(  
          children: <Widget>[  
            Container(  
              color: Colors.green,  
              alignment: Alignment.topLeft,  
              child: Text("ListView widget digunakan untuk menampilkan  
data dalam bentuk list dan jika datanya melebihi dari render box maka  
halaman tersebut dapat di scroll.", style: TextStyle(fontSize:30,color:  
Colors.white)),  
          ],  
        ),  
      ),  
    );  
  }  
}
```

```

        Container(
          color: Colors.red,
          alignment: Alignment.topLeft,
          child: Text("ListView widget digunakan untuk menampilkan
data dalam bentuk list dan jika datanya melebihi dari render box maka
halaman tersebut dapat di scroll.", style: TextStyle(fontSize:30,color:
Colors.white)),
          height: 400.0,
          width: 300.0,
        ),
        Container(
          color: Colors.deepPurple,
          alignment: Alignment.topLeft,
          child: Text("ListView widget digunakan untuk menampilkan
data dalam bentuk list dan jika datanya melebihi dari render box maka
halaman tersebut dapat di scroll.", style: TextStyle(fontSize:30,color:
Colors.white)),
          height: 200.0,
          width: 200.0,
        ),
      ],
    ),
  ),
);
}
}

```



2.6.5 GridView

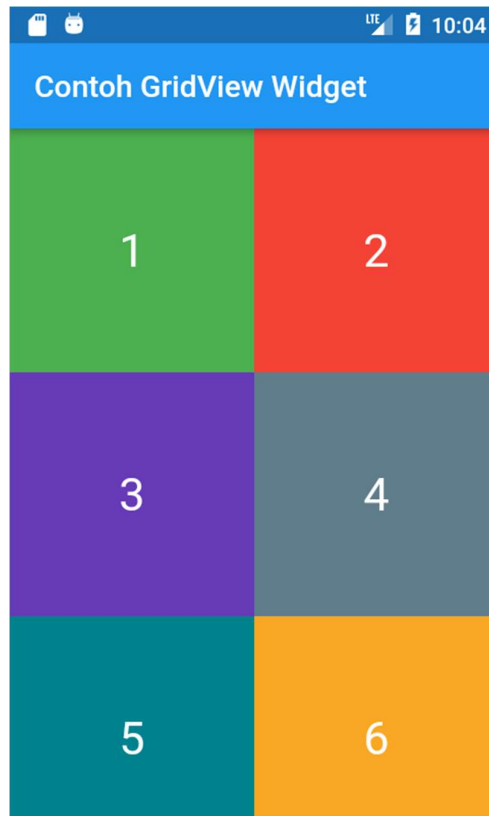
Gridview digunakan untuk menata tata letak widget pada list 2 dimensi. GridView juga secara otomatis menyediakan scrolling ketika konten melebihi render box.

```
class MyApp extends StatelessWidget {  
  @override  
  Widget build(BuildContext context) {  
    return MaterialApp(  
      debugShowCheckedModeBanner: false,  
      home: Scaffold(  
        appBar: AppBar(  
          title: Text("Contoh GridView Widget"),  
        ),  
        body: GridView.count(  
          crossAxisCount: 2,  
          children: <Widget>[  
            Container(  
              color: Colors.green,  
              alignment: Alignment.center,  
              child: Text("1", style: TextStyle(fontSize: 30, color:  
Colors.white)),  
            ),  
          ],  
        ),  
      ),  
    );  
  }  
}
```

```

        Container(
          color: Colors.red,
          alignment: Alignment.center,
          child: Text("2", style: TextStyle(fontSize: 30, color:
Colors.white)),
          height: 400.0,
          width: 300.0,
        ),
        Container(
          color: Colors.deepPurple,
          alignment: Alignment.center,
          child: Text("3", style: TextStyle(fontSize: 30, color:
Colors.white)),
          height: 200.0,
          width: 200.0,
        ),
        Container(
          color: Colors.blueGrey,
          alignment: Alignment.center,
          child: Text("4", style: TextStyle(fontSize: 30, color:
Colors.white)),
          height: 200.0,
          width: 200.0,
        ),
        Container(
          color: Colors.cyan[800],
          alignment: Alignment.center,
          child: Text("5", style: TextStyle(fontSize: 30, color:
Colors.white)),
          height: 200.0,
          width: 200.0,
        ),
        Container(
          color: Colors.yellow[800],
          alignment: Alignment.center,
          child: Text("6", style: TextStyle(fontSize: 30, color:
Colors.white)),
          height: 200.0,
          width: 200.0,
        ),
      ],
    ),
  ),
);
}
}

```



2.7 Praktikum 1 Membuat UI Sederhana

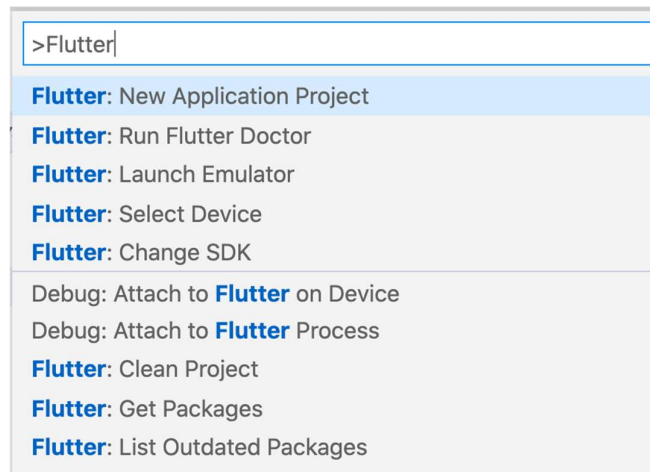
2.7.1 Alat dan Bahan

Pada praktikum pertama anda akan membuat sebuah project memanfaatkan widget flutter.

1. Visual Studio Code
2. Flutter SDK
3. Emulator atau device android

2.7.2 Langkah Praktikum

1. Buat project baru dengan menggunakan vscode misalnya dengan nama flutter_basic. Pada vscode dapat dilakukan dengan cara memilih menu '**View => Command Pallete**' selanjutnya klik Flutter akan muncul tampilan seperti pada gambar di bawah ini. Selanjutnya pilih **Flutter: New Application Project** , anda akan diminta untuk menentukan path dimana project akan dibuat dan menentukan nama project di folder tersebut.



Untuk membuat project flutter dapat juga dilakukan pada terminal atau cmd dengan menggunakan perintah seperti berikut:

```
flutter create nama_project
```

2. Dengan informasi yang anda dapatkan dari teori diatas buatlah UI sederhana seperti tampilan dibawah ini menggunakan flutter.



LTE 8:11

MyApp

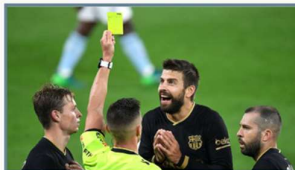
BERITA TERBARU

PERTANDINGAN HARI INI



Costa Mendekat Ke Palmeiras

Transfer



Pique Bilang Wasit Untungkan Madrid, Koeman Tepok Jidat

Barcelona Feb 13, 2021



Pique Bilang Wasit Untungkan Madrid, Koeman Tepok Jidat

