

REV 1.0

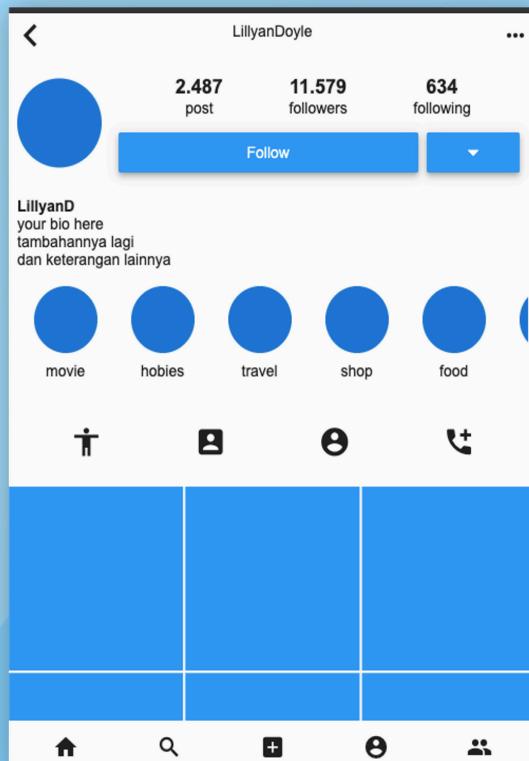


Belajar Flutter Dari Nol

**Belajar Langsung
Praktek Membuat
Layout Mirip
IG**

**Panduan Instalasi
Mac os, Windows
Dan Linux**

**Step By Step
Full Gambar**



Dart,Flutter,windows,mac os,environment,layout,widget,variable,
statefulWidget,stateLessWidget,flutter doctor,
sdk,vscode,android studio

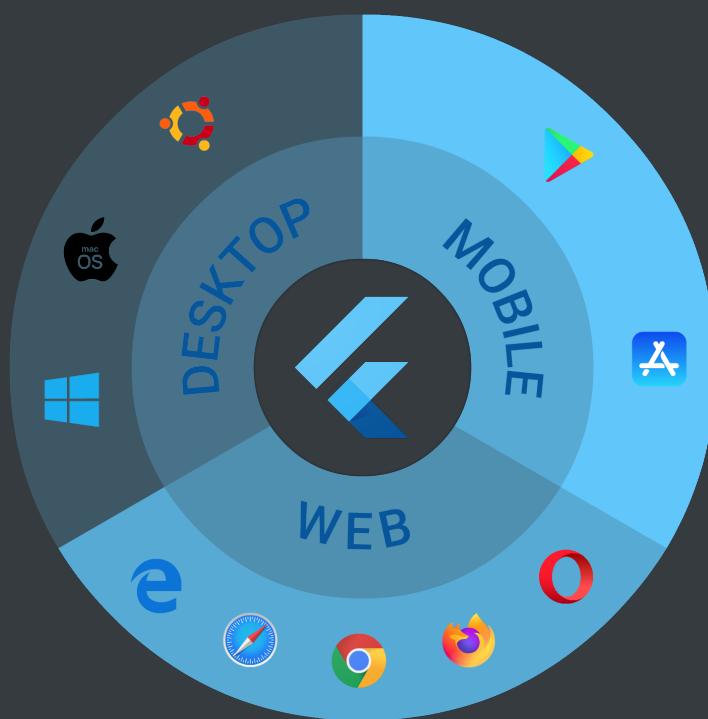
Malik Kurosaki @ 2020

Flutter Dari Nol

Bismillah.

FLUTTER

flutter adalah framework yang dipakai untuk membuat aplikasi android , eit gak hanya aplikasi android saja , bahkan ios (iphone) , web app (website) dan juga desktop app , windows , linux dan mac os, dan semua bisa dihasilkan hanya satu kali koding,lebih lengkap bisa kunjungi <https://flutter.dev/>



DART

Dart adalah bahasa pemrograman utama yang dipakai untuk mengembangkan framework flutter , dan dikembangkan langsung oleh google , mengapa google memilih dart , bukannya java atau c# atau c++ , yang sudah duluan ada, bisa kunjungi web resminya di <https://dart.dev/> secara konsep dart tidak berbeda jauh dengan bahasa pemrograman lainnya , contoh seperti variable, function,class dan semua basik bahasa pemprograman juga ada di dart, hanya saja dart sudah menggunakan beberapa metode yang sudah dikembangkan , nanti kita bahas lebih lanjut

INSTALASI

Penjelasannya segitu aja ya , pengen lengkapnya seperti sejarah, siapa pembuatnya dan mengapa , bisa kunjungi web resmsinya aja langsung , disini kali ini saya hanya akan lebih fokus aplikasi flutter nya saja ,ok langsung kunjungi <https://flutter.dev/docs/get-started/install> untuk instalasinya , pilih sesuai os kalian , untuk instalasinya sendiri tergolong susah susah gampang , disinilah ujian pertamanya ,ditahapan ini banyak yang gugur “YAKINLAH BANYAK YANG GUGUR” siapin mental

dan keseriusanmu dari sekarang sebelum ini semua jadi haanya buang buang waktu , yang terbiasa install app tinggal “next next next” harap kuatkan mental kalian , :-) seriusan karena saya sudah mengalaminya sendiri, sudah beberapa yg saya ajarkan akhirnya gugur sebelum berkembang , belum sampai pada tahap coding , hanya baru tahapan instalasi kebanyakan sudah menyerah sangat disayangka (curcol dikit)



ok lanjut ya

Select the operating system on which you are installing Flutter:

Note: Are you on Chrome OS?

If so, see the official [Chrome OS Flutter installation docs!](#)

[Windows](#) 

[macOS](#) 

[Linux](#) 

pilih os sesuai dengan os kalian

kebetulan saaya pakenya mac os , jadi saya pilihnya mac os , jika kalian pakenya windows atau linux , silahkan pilih sesuai dengan os kalian

Until kendala biasanya terjadi yaitu belum install java , belum install android sdk (android studio), dan pasang environment, sesuai dengan pengalaman saya , untuk instalasi macos dan linux tidak beda ajauh alias mirip termasuk pemasangan environmentnya , yang agak sulit biasanya di windows , dan ini butuh perhatian kusus , termasuk pengarahan sdk android dan penamaan environment, untuk java , android sdk dan ANDROID_HOME,

yang mungkin perlu dipastika sudah diinstall adalah

1. git

<https://git-scm.com/book/en/v2/Getting-Started-Installing-Git>

2. Java

<https://www.oracle.com/java/technologies/javase-jdk8-downloads.html>

3. android sdk

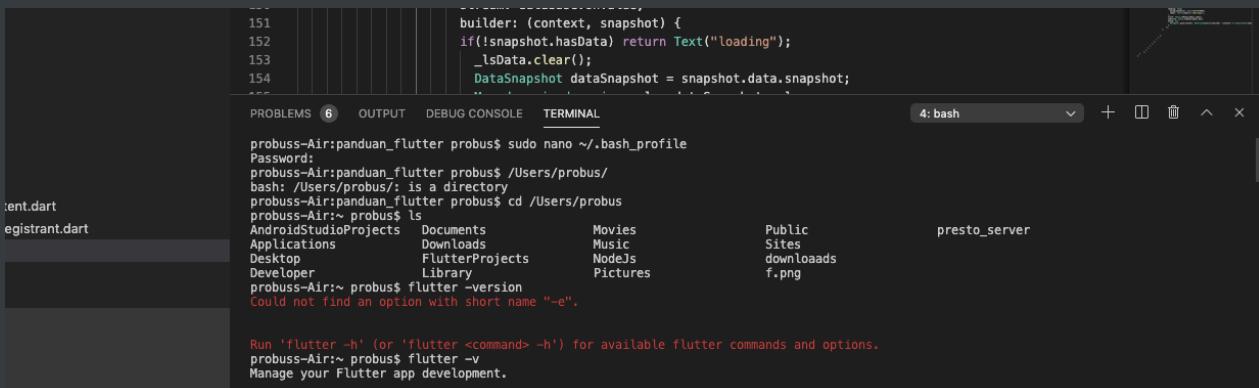
<https://developer.android.com/studio>

ini contoh environment (mac os)

secara awam , envirotmen adalah sebuah cara untuk memberitahukan os kita dimana kita meletakkan sebuah folder instalasi , yang dari situ kita bisa memerintahkannya pc kita untuk langsung eksekusi perintah tanpa harus menuju dahulu ke folder instalasi , pada kasus ini saya ingin memberitahukan pc saya bahwa saya sudah menginstall flutter / saya menempatkan file flutter di suatu tempat , jadi pc saya bisa tahu apa yang saya maksutkan ,

contoh saya ketik diterminal flutter doctor jadi pc saya sudah tahu dimana letak dari file flutter dan bisa langsung eksekusikan perintah saya, setelah iinikita akan lebis sering menggunakan perintah tersebut

setelah download dan install bahan yang diperlukan , bisa edit di bash_profile, caranya commad+space ketik terminal , didalam terminal ketik "sudo nano ~/.bash_profile" atau jika anda sudah menginstal vscode anda bisa buka terminal pada vscode ,cari menu panel atas , pilih terminal > lalu new terminal , akan membuka dendela baru dibawahnya , yaitu sebuah command line terminal



The screenshot shows a terminal window on a Mac OS system. The terminal title is '4: bash'. The command 'flutter doctor' was run, and the output is as follows:

```
151
152
153
154
builder: (context, snapshot) {
  if(!snapshot.hasData) return Text("loading");
  _lsData.clear();
  DataSnapshot dataSnapshot = snapshot.data.snapshot;
}

PROBLEMS 6 OUTPUT DEBUG CONSOLE TERMINAL
4: bash + - x
probuss-Air:panduan_flutter probus$ sudo nano ~/.bash_profile
Password:
probuss-Air:panduan_flutter probus$ /Users/probus/
bash: /Users/probus/: is a directory
probuss-Air:panduan_flutter probus$ cd /Users/probus
probuss-Air:~ probus$ ls
AndroidStudioProjects  Documents      Movies          Public
Applications         Downloads       Music           Sites
Desktop              FlutterProjects NodeJs        downloads
Developer            Library        Pictures        f.png
probuss-Air:~ probus$ flutter -version
Could not find an option with short name "-e".

Run 'flutter -h' (or 'flutter <command> -h') for available flutter commands and options.
probuss-Air:~ probus$ flutter -v
Manage your Flutter app development.
```

sudo nano ~/.bash_profile

catatan , harap berhati hati dalam pengeditan , jika ada kesalahan bisa langsung hubungi saya di wa 081338929722

```
PROBLEMS 6 OUTPUT DEBUG CONSOLE TERMINAL 4: sudo + - ×
GNU nano 2.0.6
File: /Users/probus/.bash_profile

export PATH=~/Developer/flutter/bin:$PATH
export ANDROID_HOME=/Library/Android/sdk
export PATH=$ANDROID_HOME/platform-tools:$PATH
export PATH=$ANDROID_HOME/tools:$PATH
export PATH="$PATH":$HOME/.pub-cache/bin"

# tambahan java dan gradle
export PATH="/usr/local/opt/openjdk/bin:$PATH"

# dart
export PATH=$PATH:~/Developer/flutter/bin/cache/dart-sdk/bin

#dart support path
export PATH=/usr/local/opt/dart/libexec:$PATH

#vscode
export PATH="/Applications/Visual Studio Code.app/Contents/Resources/app/bin":$PATH

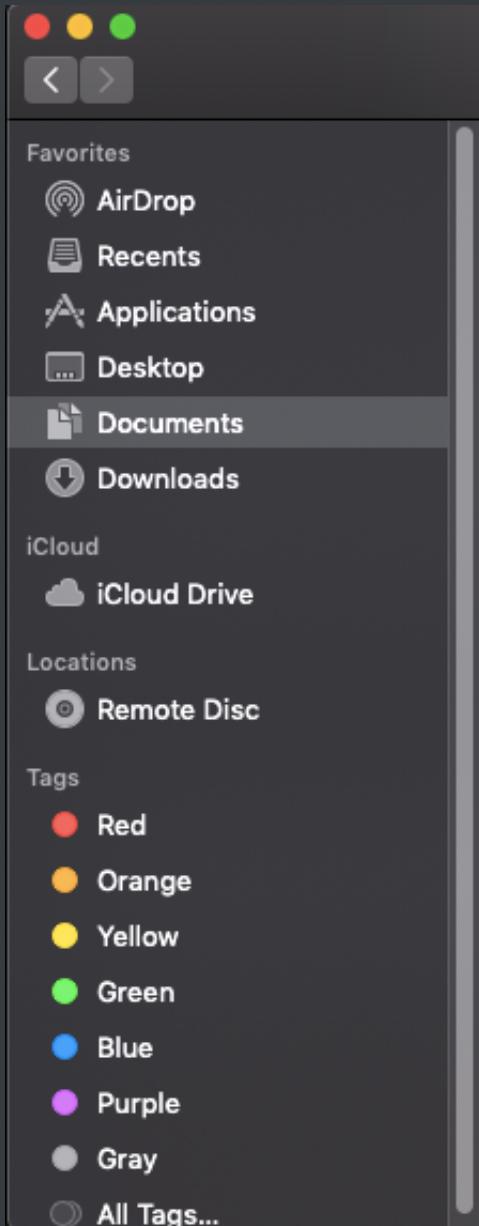
#mysql
export PATH=/usr/local/mysql/bin:$PATH

#brew
export PATH=/usr/local/sbin:$PATH

eval "$(rbenv init -)"
```

catatan "~/" itu singkatan untuk "/users/probus" , "/users/" adalah tempat folder pertama di pc mac , "/probus/" adalah nama pc saya , bisa diganti sesuai dengan nama pc anda , "bukan versi os ya " , jadi file flutter yang saya doanload tadi ada di "~/Developer/flutter" jika anda tidak mendapati folder dengan nama "Developer" anda bisa menaruhnya di folder document , saat anda download file flutter biasasnya berupa zip, lalu unzip , anda akan mendapati folder bernama flutter , lalu pindah ke folder document, jadi nantinya mengarahkannya ke export PATH=~/Documents/flutter/bin:\$PATH

yang belum terbiasa dengan terminal , ini folder yang yasa maksut



pada kasus di pc saya , saya menempatkannya di root folder di /users/probus/Developer

jika anda tidak bisa menjangkau root folder anda cukup menempatkan folder flutter yang sudah didownload dan diextract tadi di folder Documents jadi Documents/flutter folder flutter ada di dalam folder documents

contoh punya saya seperti dibawah, begitu juga untuk folder android , setelah instalasi , ternyata pc (mac os) saya mengarahkan folder instalasinya ke /Library/Android/sdk maka saya mengexportnya dan mengarakannya ke folder tersebut

catatan : tanda # adalah untuk komen, jadi semacam catatan dan tidak akan dibaca oleh mesin

```
probuss-Air:~ probus$ sudo nano ~/.bash_profile
```

GNU nano 2.0.6

File: /Users/probus/.bash_profile

```
# tambahan untuk mengarahkan flutter , folder harus sesuai dengan tempat
# folder flutter
export PATH=~/Developer/flutter/bin:$PATH

# ini properti jika sudah menginstall android studio
export ANDROID_HOME=~/Library/Android/sdk
export PATH=$ANDROID_HOME/platform-tools:$PATH
export PATH=$ANDROID_HOME/tools:$PATH

# cache flutter
export PATH="$PATH":$HOME/.pub-cache/bin"

# tambahan java dan gradle , mengarahkan ke folder instalasi java
export PATH="/usr/local/opt/openjdk/bin:$PATH"

# dart foder tujuan harus sama
export PATH=$PATH:~/Developer/flutter/bin/cache/dart-sdk/bin

#dart support path , bagi yang sudah menginstall dart
export PATH=/usr/local/opt/dart/libexec:$PATH

#vscode mengarahkan vscode , folder tujuan harus sesuai
export PATH="/Applications/Visual Studio
Code.app/Contents/Resources/app/bin":$PATH

#mysql jika sudah install mysql , jika belum tidak usah
export PATH="/usr/local/mysql/bin:$PATH"

#brew bagi yang sudah menginstall brew , bagi yang belum tidak usah
export PATH="/usr/local/sbin:$PATH"
```

edit yang diperlukan saja , jangan semuanya kecuali anda mengerti , pada kasus ini yang kita butuhkan adalah flutter , dan android , setelah selesai anda bisa menekan tombol `ctrl+x` y / yes untuk save , n / no untuk batal

lalu reload file tersebut dengan mengetik `source ~/.bash_profile`

untuk linux sendiri hampir sama dengan mac os karena emang samasama dari UNIX, untuk file bisa ada di `~/.bashrc` atau di `~/.bash_profile`

jika berhasil anda bisa mengeceknya dengan mengetik diterminal flutter --version akan menghasilkan seperti dibawah

```
probuss-Air:~ probus$ flutter --version
Flutter 1.18.0-5.0.pre.57 • channel master •
https://github.com/flutter/flutter.git
Framework • revision a5765331bc (2 days ago) • 2020-04-09 16:15:01 -0700
Engine • revision 5b4b1f33c6
Tools • Dart 2.8.0 (build 2.8.0-dev.20.0 dc当地d763)
```

jika masih belum berpengaruh , coba restart pcnya dulu

saran penempatan file flutter

Mac OS dan linux

oh iya sedikit saran , tepatkan file **flutter** dan **Android** (android sdk) di tempat yang mudah untuk dijangkau dan tidak membutuhkan ijin khusus sudo ,

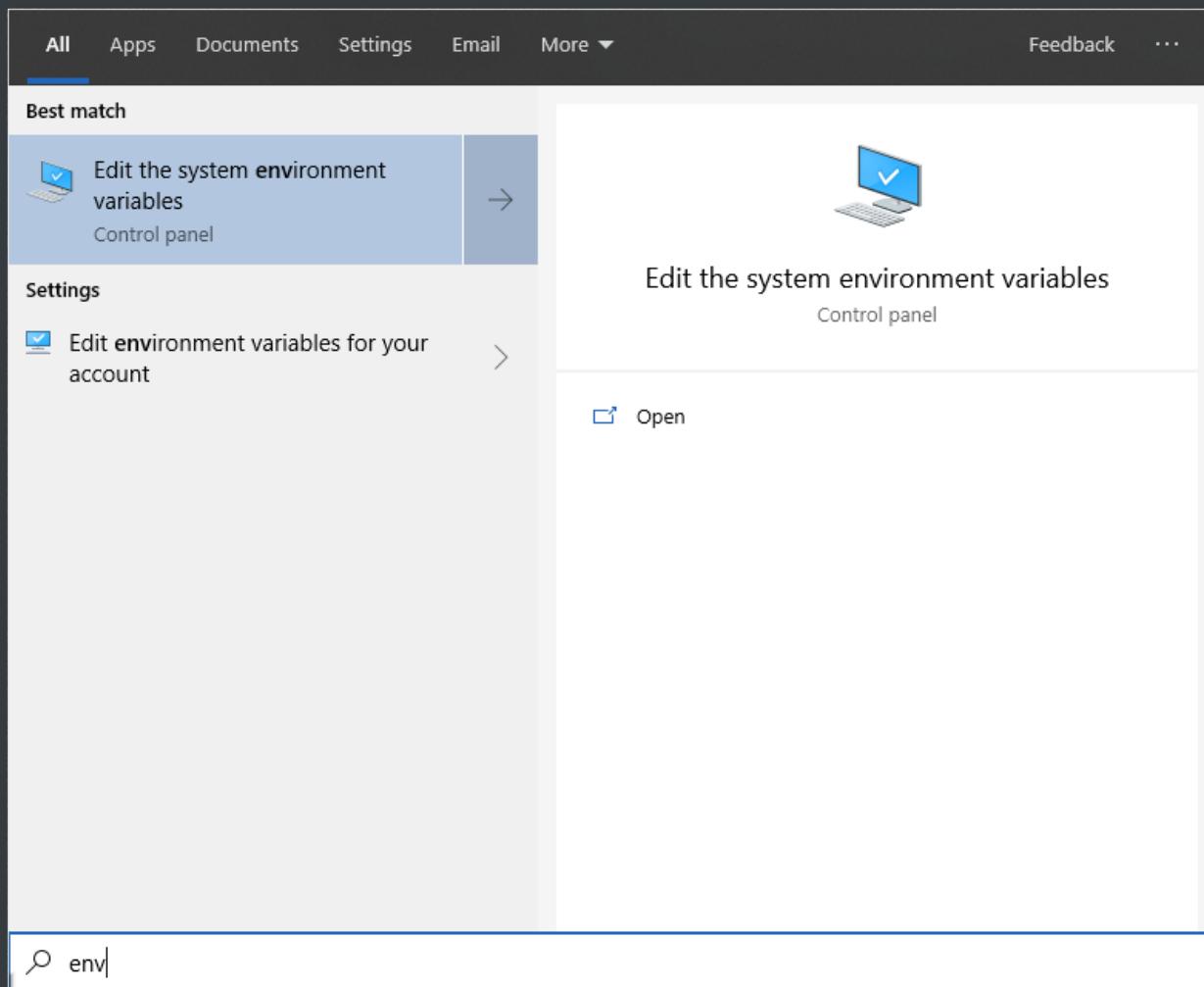
contoh seperti di /users/nama_pc/documents/flutter

untuk diwindows

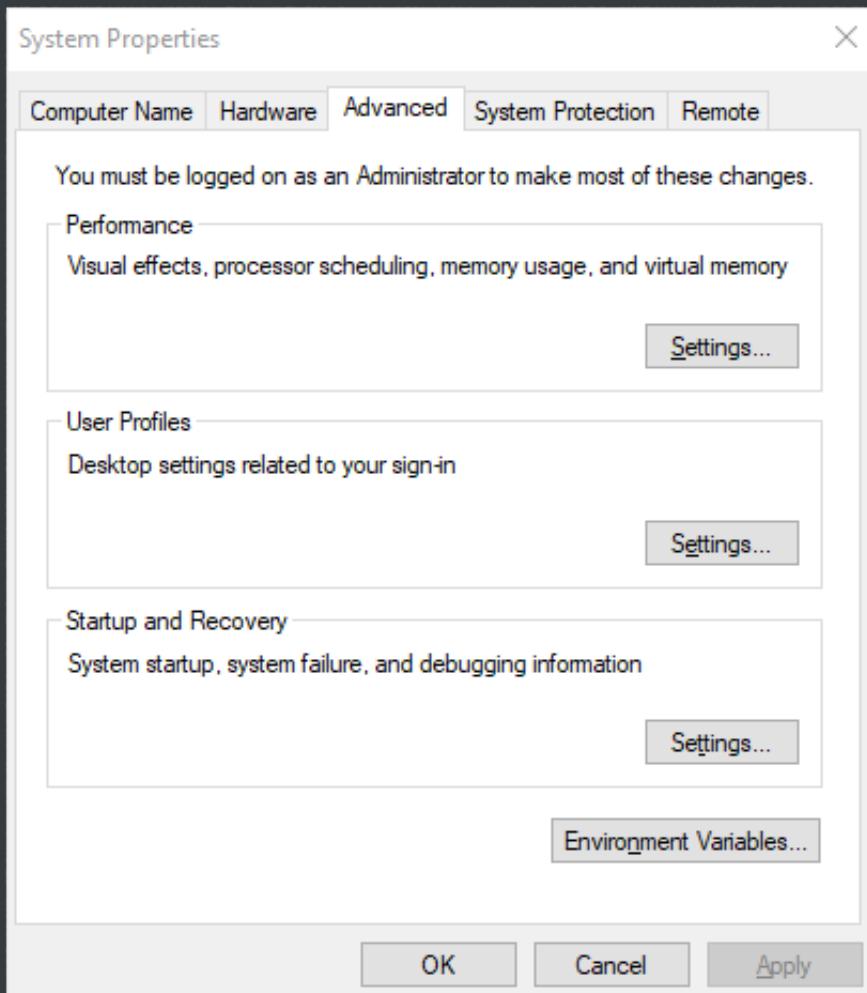
untuk windows bisa ditempatkan di C:\\\\ atau D:\\\\ jangan di program_file

contoh C:\\\\flutter atau D:\\\\flutter

contoh environment di windows



pilih environment variable



pilih path

Environment Variables

User variables for SUMIT

| Variable | Value |
|------------------|--|
| OneDrive | C:\Users\SUMIT\OneDrive |
| OneDriveConsumer | C:\Users\SUMIT\OneDrive |
| Path | C:\Users\SUMIT\AppData\Local\Microsoft\WindowsApps;C:\P... |
| TEMP | C:\Users\SUMIT\AppData\Local\Temp |
| TMP | C:\Users\SUMIT\AppData\Local\Temp |

New... Edit... Delete

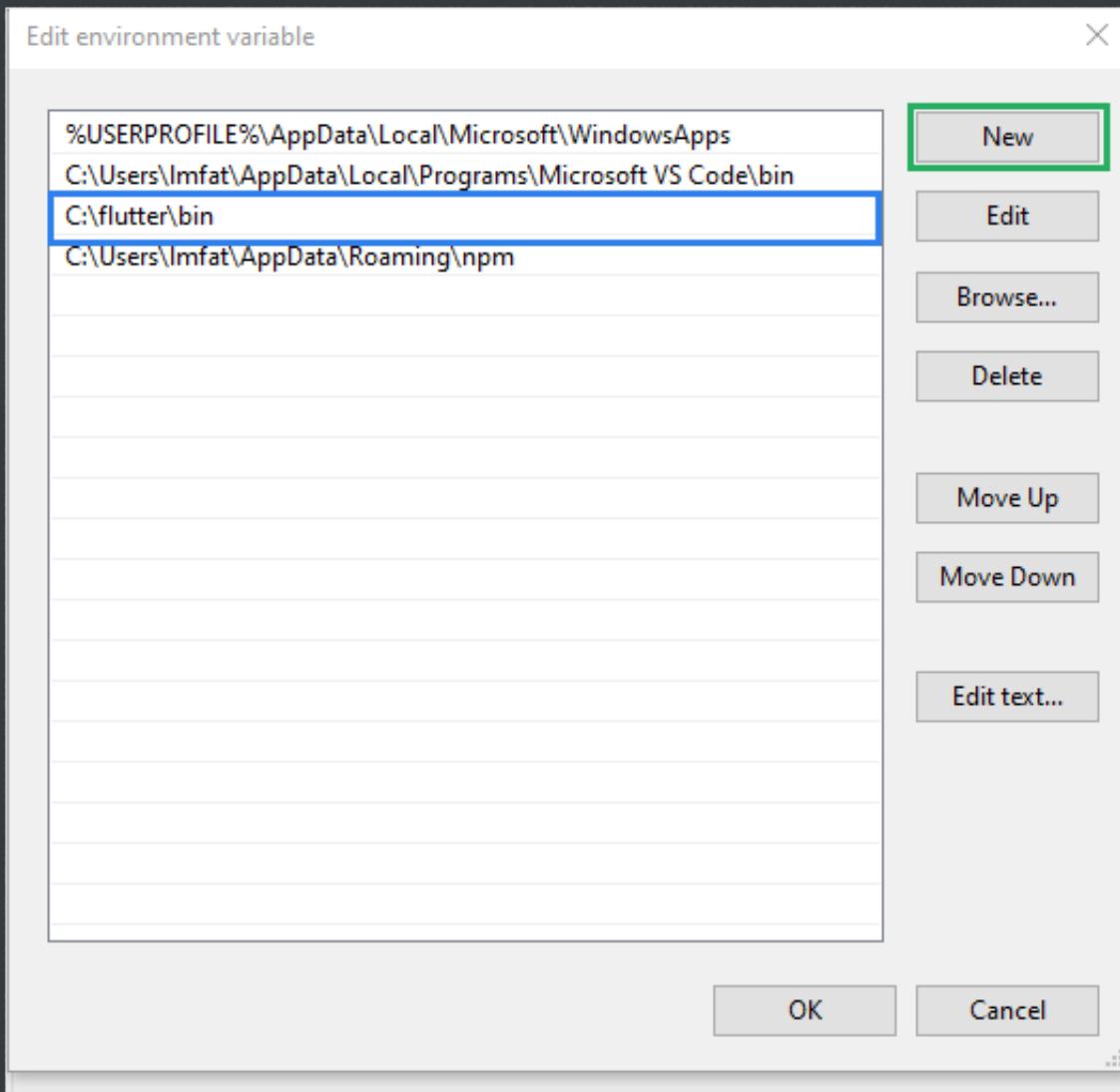
System variables

| Variable | Value |
|----------------------|---|
| ComSpec | C:\WINDOWS\system32\cmd.exe |
| DriverData | C:\Windows\System32\Drivers\DriverData |
| FSHARPISTALLDIR | C:\Program Files (x86)\Microsoft SDKs\F#\10.1\Framework\v4... |
| NUMBER_OF_PROCESSORS | 8 |
| OS | Windows_NT |
| Path | C:\Program Files (x86)\Intel\Intel(R) Management Engine Co... |
| PATHEXT | .COM;.EXE;.BAT;.CMD;.VBS;.VBE;.JS;.JSE;.WSF;.WSH;.MSC |

New... Edit... Delete

OK Cancel

new lalu arahkan ke file flutternya



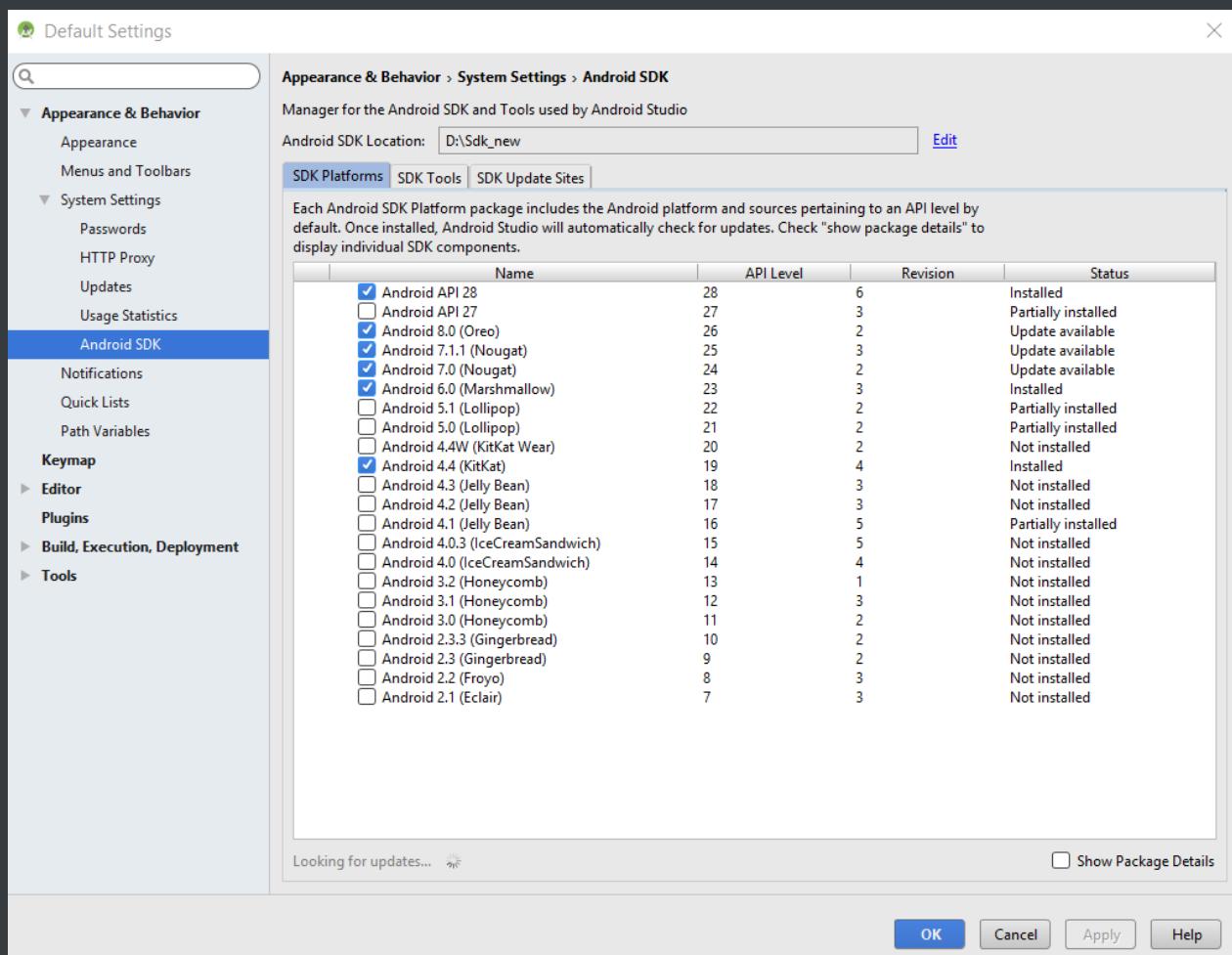
jangan lupa setting juga untuk ,git , java , android sdk , arahkan sesuai dengan folderl instalasi masing masing,

untuk windows lebih lengkapnya di : <https://flutter.dev/docs/get-started/install/windows>

untuk penggerjaan flutter sendiri aplikasi yang direkomendsikan bisa melalui android studio , dengan minimal ram adalah 8gb , bagi yang dibawahnya bisa menggunakan vscode <https://code.visualstudio.com/download>

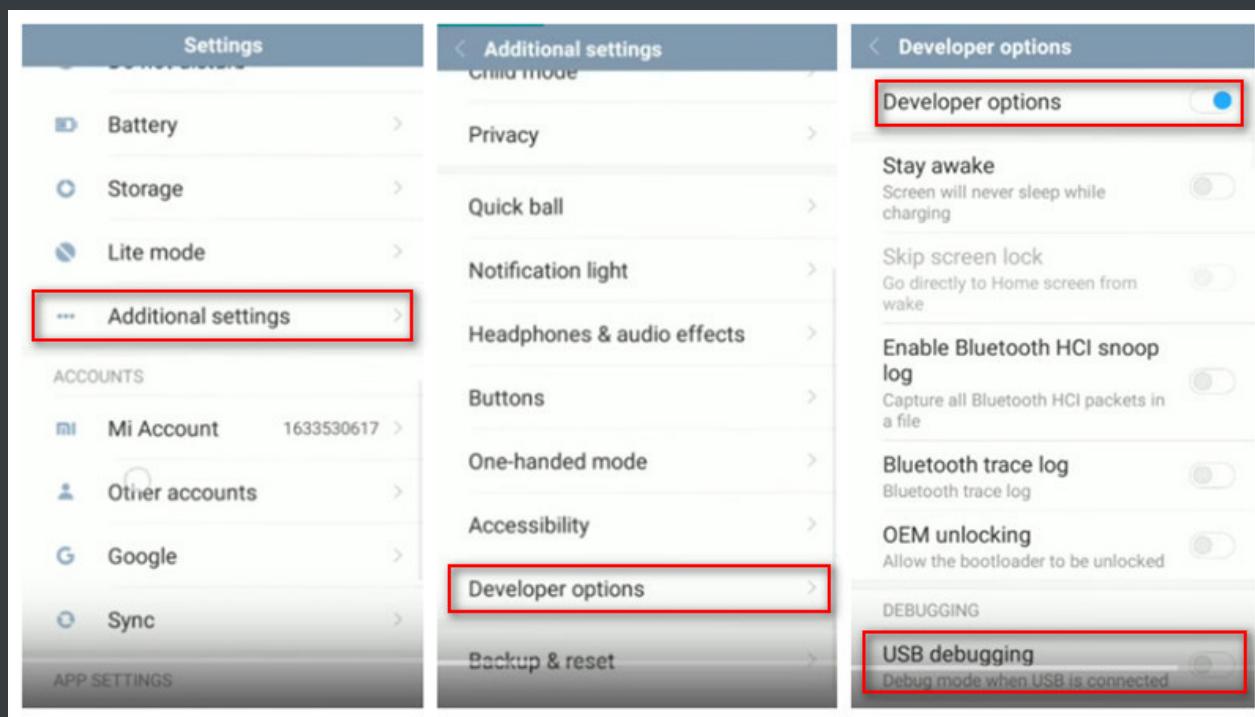
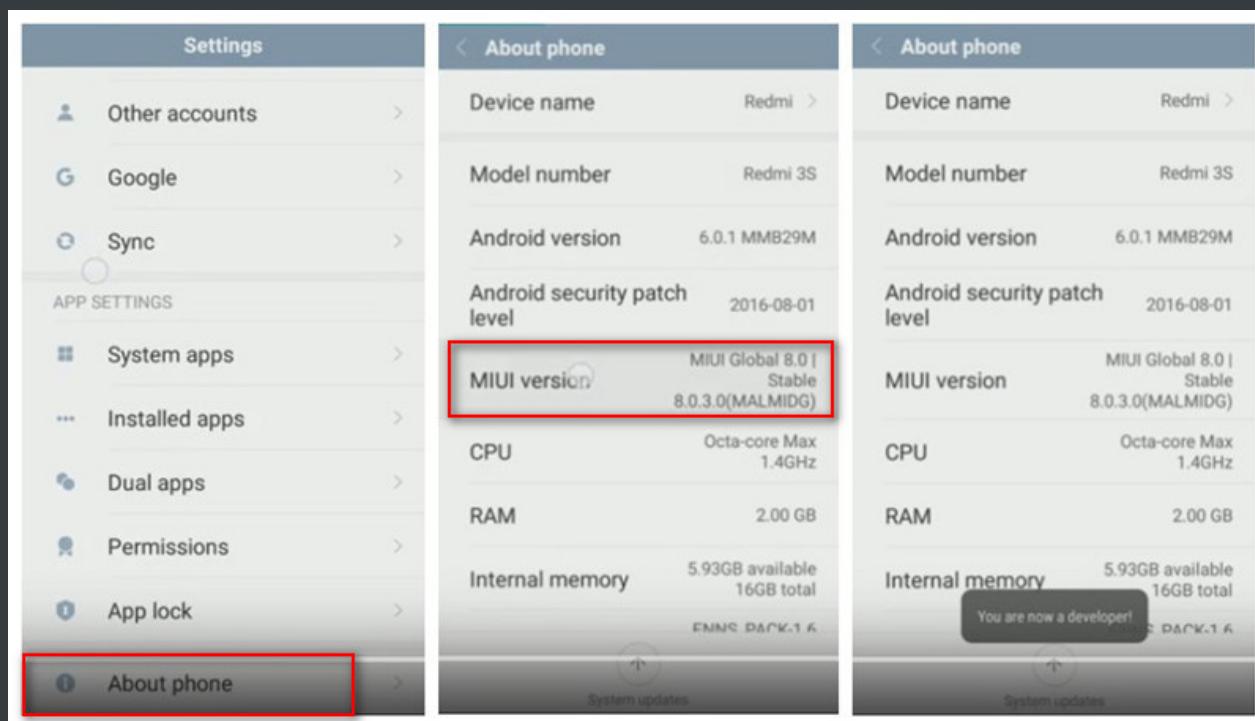
setelah itu pasang addon atau plugin flutter dan dart pada android studio ataupun vs code , kebetulan pc saya terasa berat jika menggunakan android studio maka saya menggunakan vscode,

Ada beberapa setingan pada android studio , terutama yang berhubungan dengan debugging , pastikan sdk tool telah tercentang dan android avd (untuk emulator) sudah terpasang juga



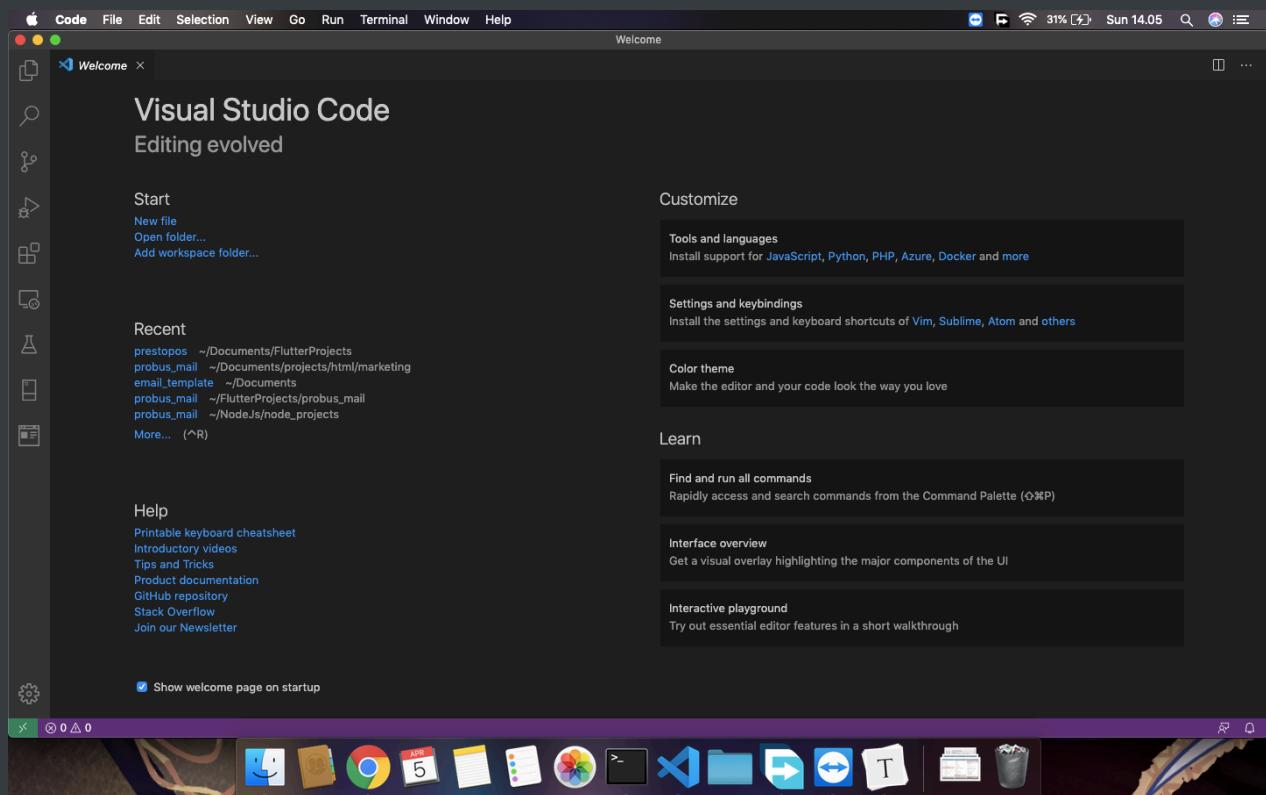
sedangkan untuk android avd (emulator) bisa langsung ke sini <https://developer.android.com/studio/run/emulator>

bagi yang terlalu berat jika harus menggunakan emulator bisa menggunakan hp langsung , dengan cara buka setting pada hanphone masung masing , pilih info , dan buka developer optionsnya , biasanya tekan 7x pada nama osnya ,



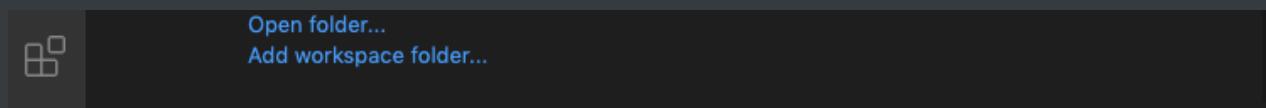
jika masih keberatan dengan metode keduanya diatas , bisa menggunakan google chrome , dan akan saya bahas next

ok lanjut , karena saya menggunakan vscode bukan android studio , pada tutorial kedepannya saya akan menunjukkan dengan menggunakan vscode , walaupun sama saja dengan android studio , cuman ya kelebihannya lebih ringan aja , ok kita buka vs code

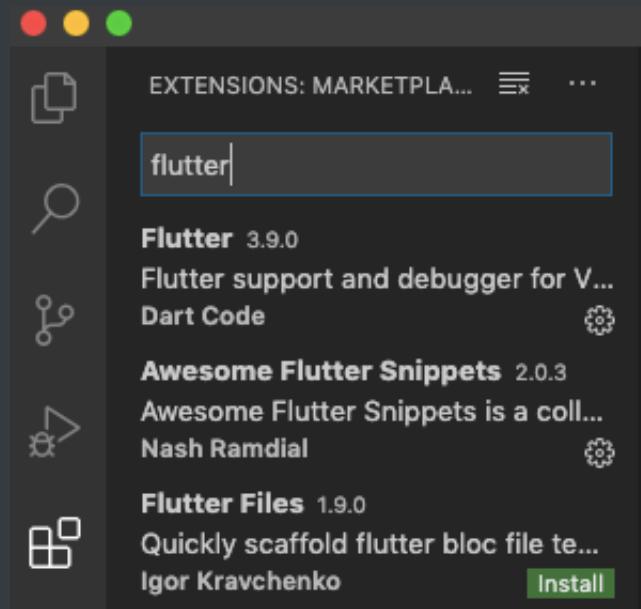


nah , seperti itu penampakannya

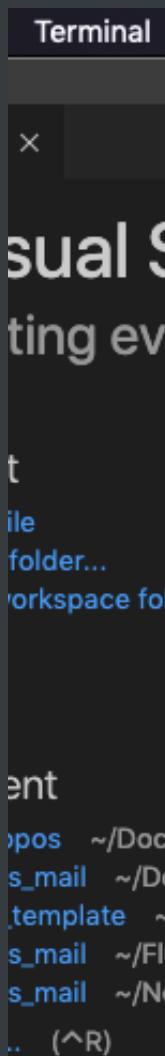
tekan tombol totak paling kiri seperti dibawah



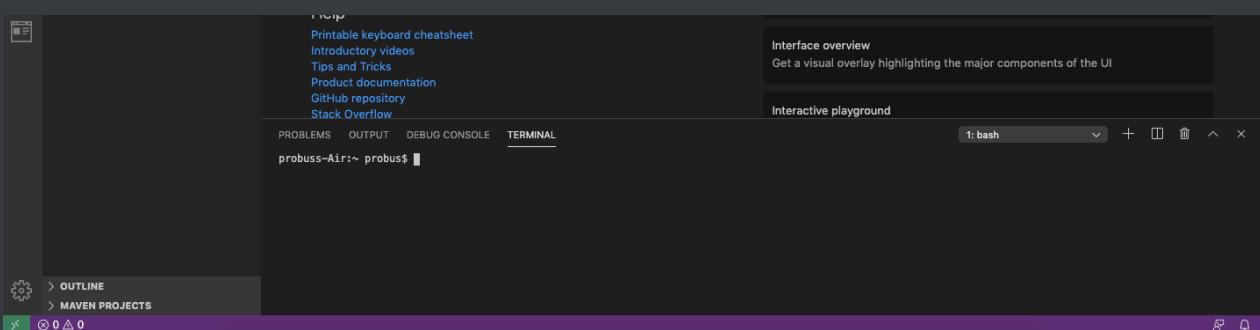
ketik flutter lalu install



lalu buka panel terminal pada panel menu atas "new terminal"



pada bagian bawah akan muncul kotak baru namanya terminal



lalu ketikkan "flutter doctor" tanpa tanda petik

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
probuss-Air:~ probus$ flutter doctor
Doctor summary (to see all details, run flutter doctor -v):
[✓] Flutter (Channel master, v1.15.19-pre.16, on Mac OS X 10.14.6 18G103, locale en-ID)
[✓] Android toolchain - develop for Android devices (Android SDK version 29.0.2)
[✓] Xcode - develop for iOS and macOS (Xcode 11.2.1)
[✓] Chrome - develop for the web
[✓] Android Studio (version 3.5)
[✓] VS Code (version 1.43.2)
[✓] Connected device (3 available)

• No issues found!
probuss-Air:~ probus$
```

pastikan no issues found! dan tidak ada error , jika ada issues atau error yang biasanya ditandai warna merah , segera perbaiki , jangan lanjutkan ke next step sebelum semua error telah diperbaiki, jika disini ada kendaala bisa hubungi saya langsung , ketik malikkurosaki pada pencarian google , atau email ke kurosaiblackangel@gmail.com tau wa 081338929722,

isssues atau error yang muncul biasanya permasalahan di environment , git,java,flutter/bil/,android android toolchain,

saya harap semua bisa berjalan lancar dan bisa melanjutkan ke next step.

selanjutnya ketik "flutter channel"

```
No issues found.
probuss-Air:~ probus$ flutter channel
Flutter channels:
* master
  dev
  beta
  stable
probuss-Air:~ probus$
```

disini ada beberapa pilihan , tergantung kebutuhan , pada instalasi pertama biasanya channet ditempatkan pada versi stable , disini karena saya ada kebutuhan untuk mendevelope versi lain , maka saya menggunakan master , bagi pemula tidak disarankan untuk versi dev dan beta namun tidak ada larangan , resiko ditanggung penumpang , hahah

untuk setingan flutter agar bisa berjalan di chroeme sebagai emulator jalankan. perintah ini

```
flutter channel master
flutter upgrade
flutter config --enable-web
```

selanjutnya ketik flutter devices untuk mengetahui device emulator apa saja yang telah terhubung, jika kalian menghidupkan android emulator maka akan muncul keterangan emulatornya , jika mencolokkan usb dengan handphone yang telah open debugginya , maka akan muncul keterangan hanphone kalian,

```
probuss-Air:~ probus$ flutter devices
3 connected devices:

macOS      • macOS      • darwin-x64      • Mac OS X 10.14.6 18G103
Chrome     • chrome     • web-javascript • Google Chrome 80.0.3987.163
Web Server • web-server • web-javascript • Flutter Tools
probuss-Air:~ probus$
```

saya harap disini semua sudah fix, dan bisa lanjut ketahap berikutnya ,

ok sekarang kita lanjut ke create project.

NEW PROJECT

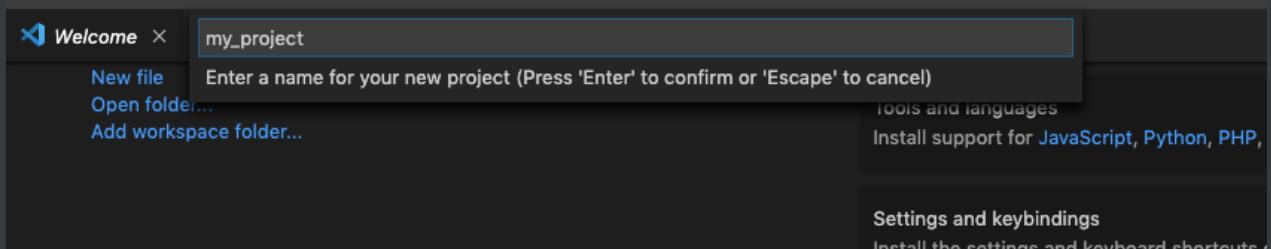
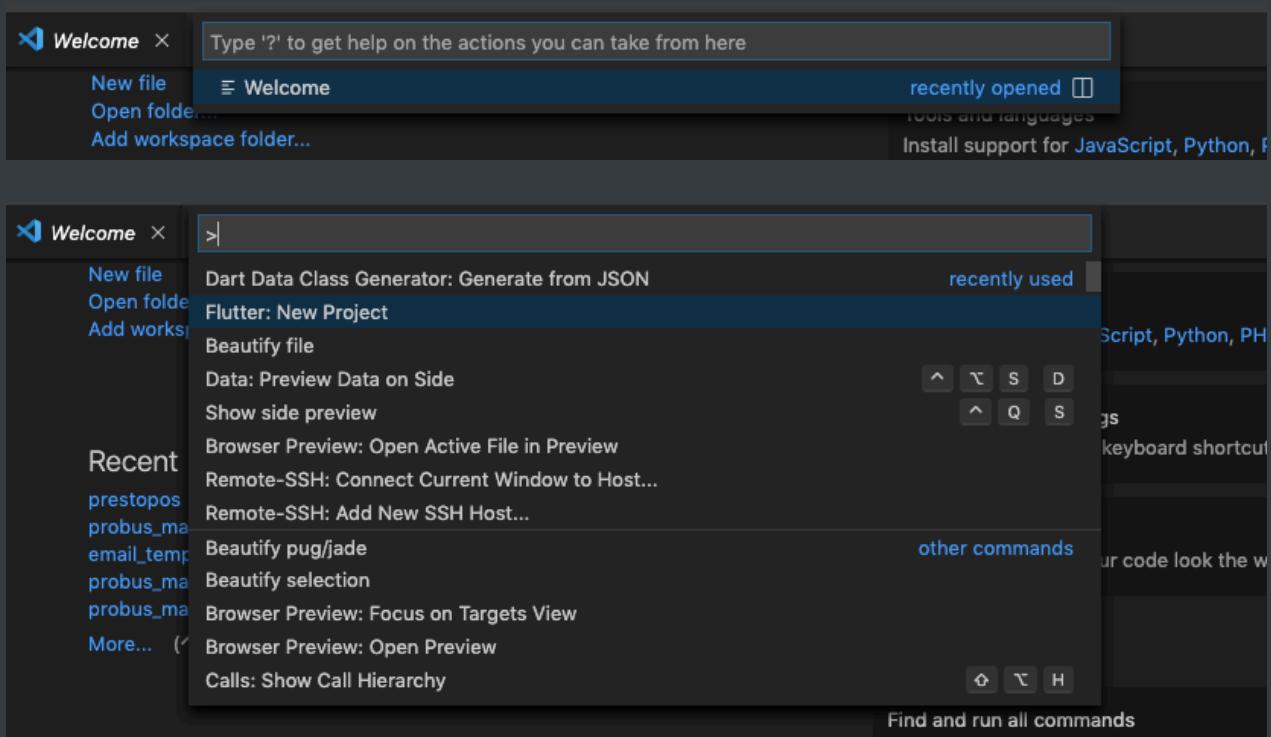
hore ... selamat saat saat mendebarkan instalasi telah dilalui , sekarang saatnya kita buat project, jika ada yang mau bikin kopi , bikin kopi aja dulu , yang mau mandi mandi aja dulu , jangan terburu buru , istirahatkan otak dulu sejenak biar gak keram ,



sekarang tekan tombol pada keyboard macos command+p+> windows ctr+p+> atau bisa juga ketik pada terminal flutter create my_project_name jika mau lengkap bisa flutter create --org com.yourdomain your_app_name super lengkapnya flutter create --androidx -t app --org com.companyname.packagename -a kotlin -i swift myapp butuh pertolongan ketik flutter create --help

Pada kasus saya , saya menggunakan command+p+> karena lebih simple

akan muncul pop baru form input, pada bagian ini kita isi dengan nama aplikasi yang akan kita buat, tidak boleh ada special karakter, spasi,ataupun huruf besar, contoh : my_project



jika bingung mau ditempatkan dimana , untuk macos dan linux bisa di documents/flutter_project
untuk windows bisa bi D:\\flutter_project hanya saran , pilihan bebas dikalian

inilah penampakannya jika kalian berhasil

A screenshot of the Visual Studio Code interface. The left sidebar shows a file tree with a 'main.dart' file selected. The main editor area displays the 'main.dart' code for a Flutter application. The code defines an 'App' widget that runs a 'HomePage'. The 'HomePage' is a stateful widget with a title 'Flutter Demo Home Page'. The bottom status bar shows 'Ln 36, Col 1' and other system information.

```
main.dart — my_project
main.dart
lib/main.dart
void main() {
  runApp(MyApp());
}

class MyApp extends StatelessWidget {
  // This widget is the root of your application.
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      title: 'Flutter Demo',
      theme: ThemeData(
        // This is the theme of your application.
        //
        // Try running your application with "flutter run". You'll see the
        // application has a blue toolbar. Then, without quitting the app, try
        // changing the primarySwatch below to Colors.green and then invoke
        // "hot reload" (press "r" in the console where you ran "flutter run",
        // or simply save your changes to "hot reload" in a Flutter IDE).
        // Notice that the counter didn't reset back to zero; the application
        // is not restarted.
        primarySwatch: Colors.blue,
      ), // ThemeData
      home: HomePage(title: 'Flutter Demo Home Page'),
    ); // MaterialApp
  }
}

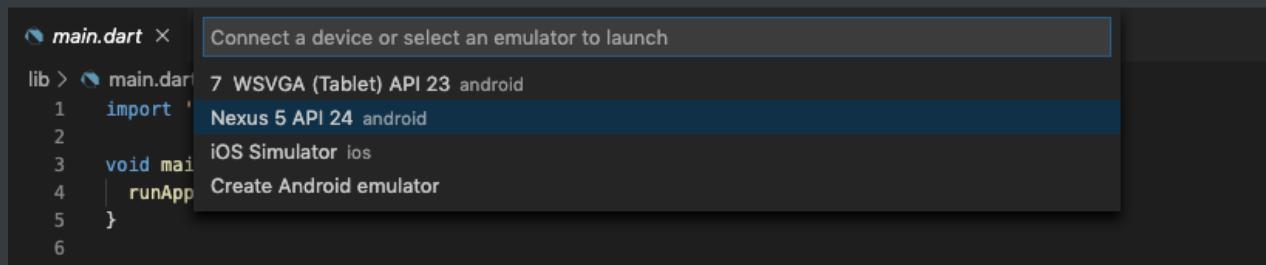
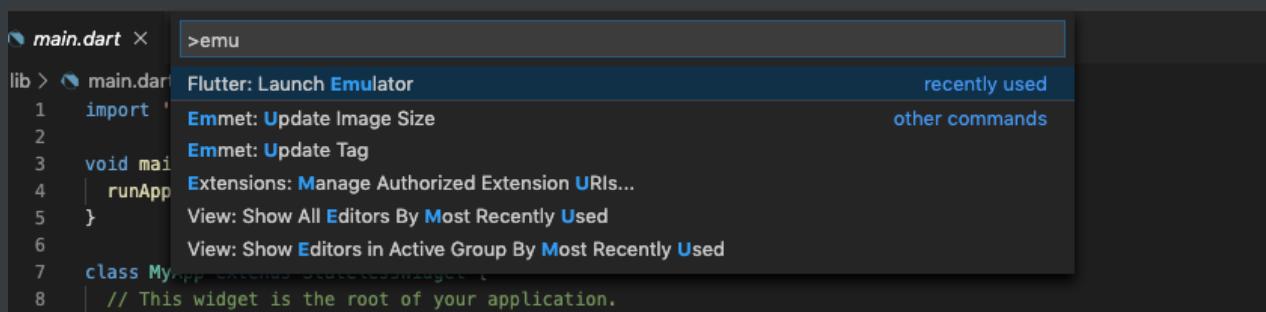
class HomePage extends StatefulWidget {
  HomePage({Key key, this.title}) : super(key: key);

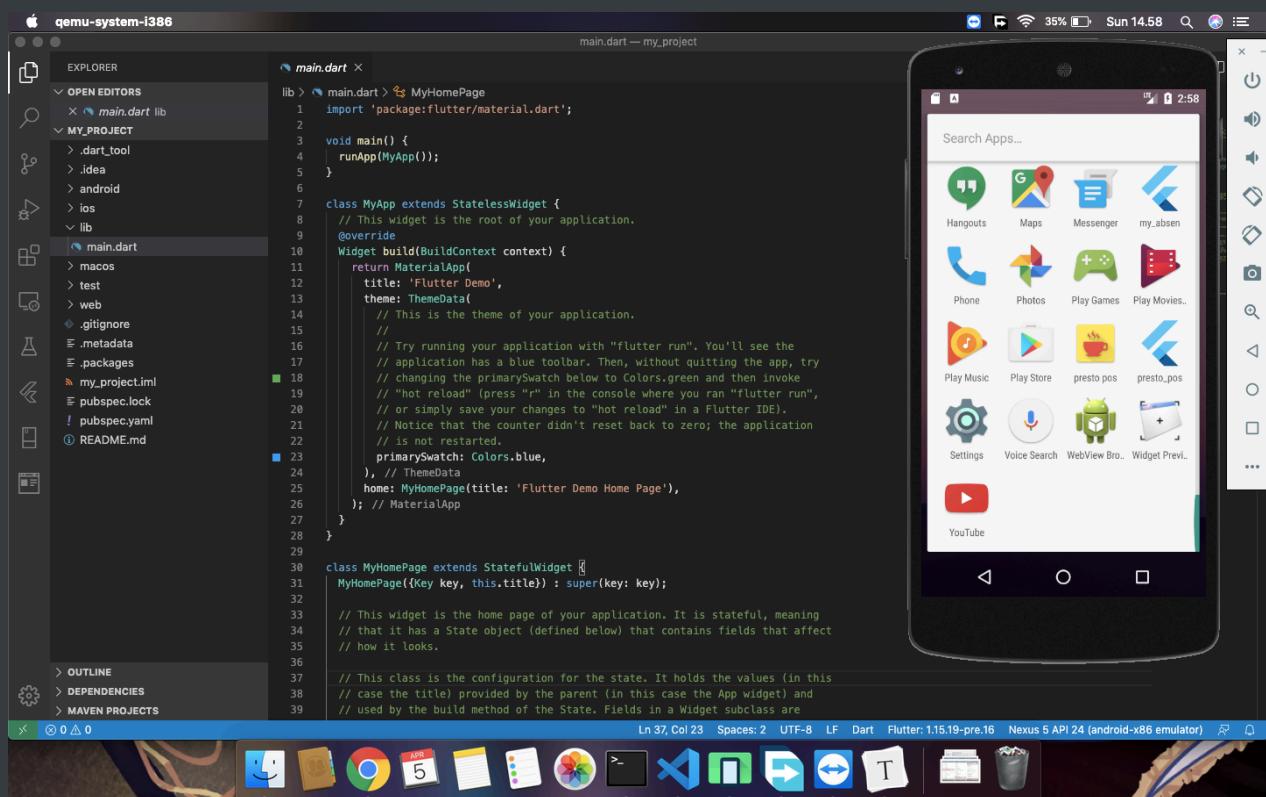
  // This widget is the home page of your application. It is stateful, meaning
  // that it has a State object (defined below) that contains fields that affect
  // how it looks.

  // This class is the configuration for the state. It holds the values (in this
  // case the title) provided by the parent (in this case the App widget) and
  // used by the build method of the State. Fields in a Widget subclass are
}

karena laptop saya tergolong lemot , kali ini saya akan menggunakan emulator di chome langsung , bagi yang spek pcnya lumayan bisa menggunakan android emulator ataupun ios emulator , gk menutup kemungkinan penggunaan emulator lainnya seperti NOX dan android emulator pihak ketiga lainnya ,
```

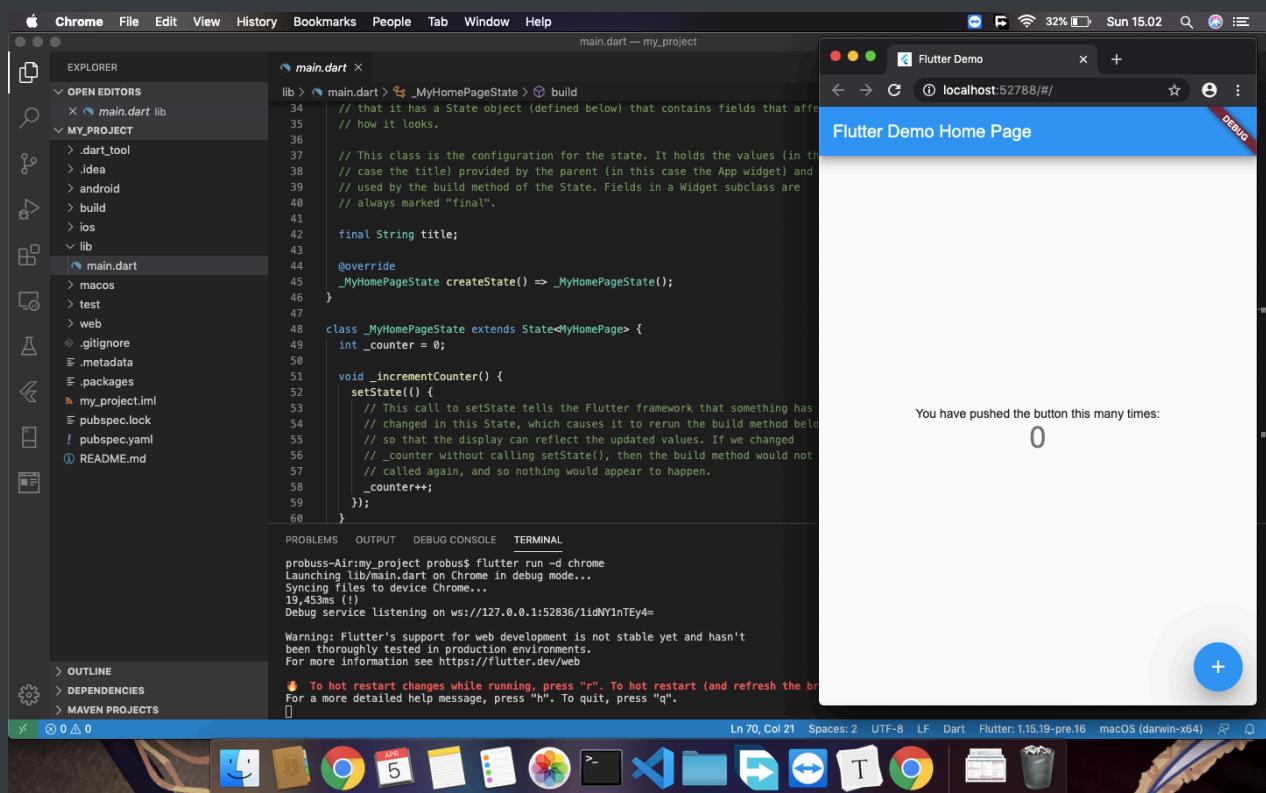
pada vscode kalian bisa membuka emulator dengan tekan tombol keyboard macos command + p+> pada kolom ketik emulator windows comman ganti dengan ctrl





sekarang coba kita jalankan , bagi yang menggunakan emulator bisa ketik di terminal flutter run maka akan otomatis mencari emulator yang aktif , bagi yang menggunakan emulator chrome bisa ketik flutter run -d chrome

bagi yang menggunakan emulator chrome , pastikan flutter channel master atau dev

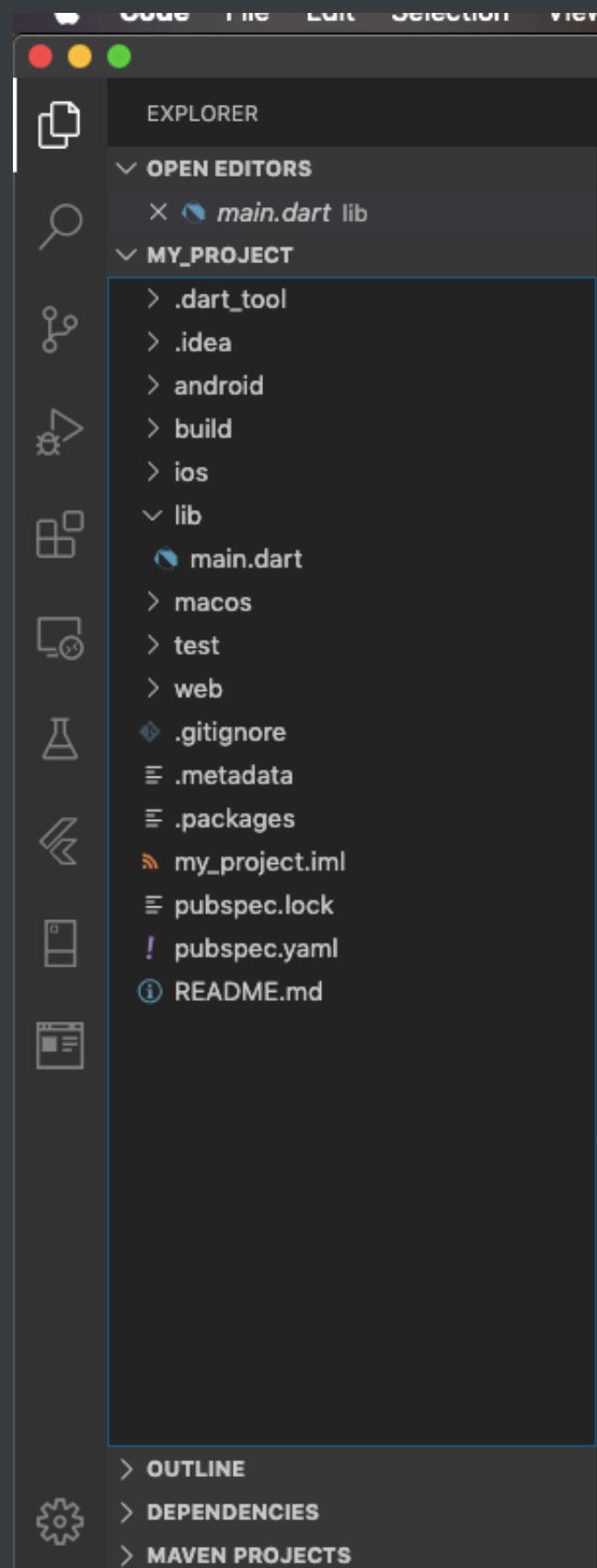


"SELAMAT ANDA SUDAH BISA BIKIN APLIKASI ANDROID"

uda sampe sini berarti anda sudah bisa bikin aplikasi android , bagi yang menggunakan emulator langsung di hanphone , bisa langsung lihat dengan nyata hahah "keren"

"bagi yang kopinya sudah habis silahkan buat lagi , bagi yang belum monggu silahkan buat kopi dulu jangan terburu buru biar gk keramotaknya hahahah"

ok selanjutnya pada kolom sebelah kiri anda bisa lihat , akalian bisa liha ada beberapa folder dan file ,



untuk penggeraan pengkodingan kita semua ada di lib/main.dart fokus dulu disitu , lainnya akan dijelaskan pada next step , biar pikirannya gk ambyar hahahah

pada main.dart kita akan jumpai kode ribet dan ruwet yang sudah dibuatkan oleh tem developer flutter

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

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

class MyApp extends StatelessWidget {
  // This widget is the root of your application.
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      title: 'Flutter Demo',
      theme: ThemeData(
        // This is the theme of your application.
        //
        // Try running your application with "flutter run". You'll see the
        // application has a blue toolbar. Then, without quitting the app,
        try
          // changing the primarySwatch below to Colors.green and then invoke
          // "hot reload" (press "r" in the console where you ran "flutter
        run",
          // or simply save your changes to "hot reload" in a Flutter IDE).
          // Notice that the counter didn't reset back to zero; the
        application
          // is not restarted.
          primarySwatch: Colors.blue,
        ),
        home: MyHomePage(title: 'Flutter Demo Home Page'),
      );
    }
  }

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

    // This widget is the home page of your application. It is stateful,
    meaning
    // that it has a State object (defined below) that contains fields that
    affect
    // how it looks.

    // This class is the configuration for the state. It holds the values (in
    this
```

```
// case the title) provided by the parent (in this case the App widget)
and
// used by the build method of the State. Fields in a Widget subclass are
// always marked "final".

final String title;

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

class _MyHomePageState extends State<MyHomePage> {
int _counter = 0;

void _incrementCounter() {
    setState(() {
        // This call to setState tells the Flutter framework that something
        // has changed in this State, which causes it to rerun the build method
        // below
        // so that the display can reflect the updated values. If we changed
        // _counter without calling setState(), then the build method would
        // not be
        // called again, and so nothing would appear to happen.
        _counter++;
    });
}

@Override
Widget build(BuildContext context) {
    // This method is rerun every time setState is called, for instance as
    // done
    // by the _incrementCounter method above.
    //
    // The Flutter framework has been optimized to make rerunning build
    // methods
    // fast, so that you can just rebuild anything that needs updating
    // rather
    // than having to individually change instances of widgets.
    return Scaffold(
        appBar: AppBar(
            // Here we take the value from the MyHomePage object that was
            // created by
            // the App.build method, and use it to set our appBar title.
            title: Text(widget.title),
    );
}
```

```
),
body: Center(
    // Center is a layout widget. It takes a single child and positions
    it
        // in the middle of the parent.
    child: Column(
        // Column is also a layout widget. It takes a list of children
        and
            // arranges them vertically. By default, it sizes itself to fit
            its
                // children horizontally, and tries to be as tall as its parent.
                //
                // Invoke "debug painting" (press "p" in the console, choose the
                // "Toggle Debug Paint" action from the Flutter Inspector in
            Android
                // Studio, or the "Toggle Debug Paint" command in Visual Studio
            Code)
                // to see the wireframe for each widget.
                //
                // Column has various properties to control how it sizes itself
            and
                // how it positions its children. Here we use mainAxisAlignment
            to
                // center the children vertically; the main axis here is the
            vertical
                // axis because Columns are vertical (the cross axis would be
                // horizontal).
            mainAxisAlignment: MainAxisAlignment.center,
            children: <Widget>[
                Text(
                    'You have pushed the button this many times:',
                ),
                Text(
                    '$_counter',
                    style: Theme.of(context).textTheme.headline4,
                ),
            ],
        ),
    floatingActionButton: FloatingActionButton(
        onPressed: _incrementCounter,
        tooltip: 'Increment',
        child: Icon(Icons.add),
    ), // This trailing comma makes auto-formatting nicer for build
methods.
```

```
    );
}
}
```

ok fokus pelan pelan , saya akan jelaskan perlahan dengan detail agar mudah untuk dimengerti

Ada pribahasa megatakan , kosongkan dulu gelasnya ok yuk kita kosongkan, klik pada kodingan lalu tekan tombol `ctrl+a` macos `command+a` lalu delete , pastikan semua kodingan terdelet dan bersih

The screenshot shows a code editor interface with a dark theme. In the top left, there's a file tree with 'main.dart' selected. Below it, a search bar has '1' in it. The main workspace is currently empty. At the bottom, there's a terminal window titled '1: bash' showing the output of running a Flutter application in debug mode on Chrome. The terminal output includes:

```
Launching lib/main.dart on Chrome in debug mode...
Syncing files to device Chrome...
19,453ms (!)
Debug service listening on ws://127.0.0.1:52836/1idNY1nTEy4=
Warning: Flutter's support for web development is not stable yet and hasn't
been thoroughly tested in production environments.
For more information see https://flutter.dev/web.

👉 To hot restart changes while running, press "r". To hot restart (and refresh the browser), press "R".
For a more detailed help message, press "h". To quit, press "q".
```

nah loh , kosong hahaha

ketik pada kolom yang kosong tadi dengan kodingan seperti dibawah

"DIKETIK JANGAN COPY PASTE!"

yang senengannya copy paste tak doain gk bisa bisa hahahah ,

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

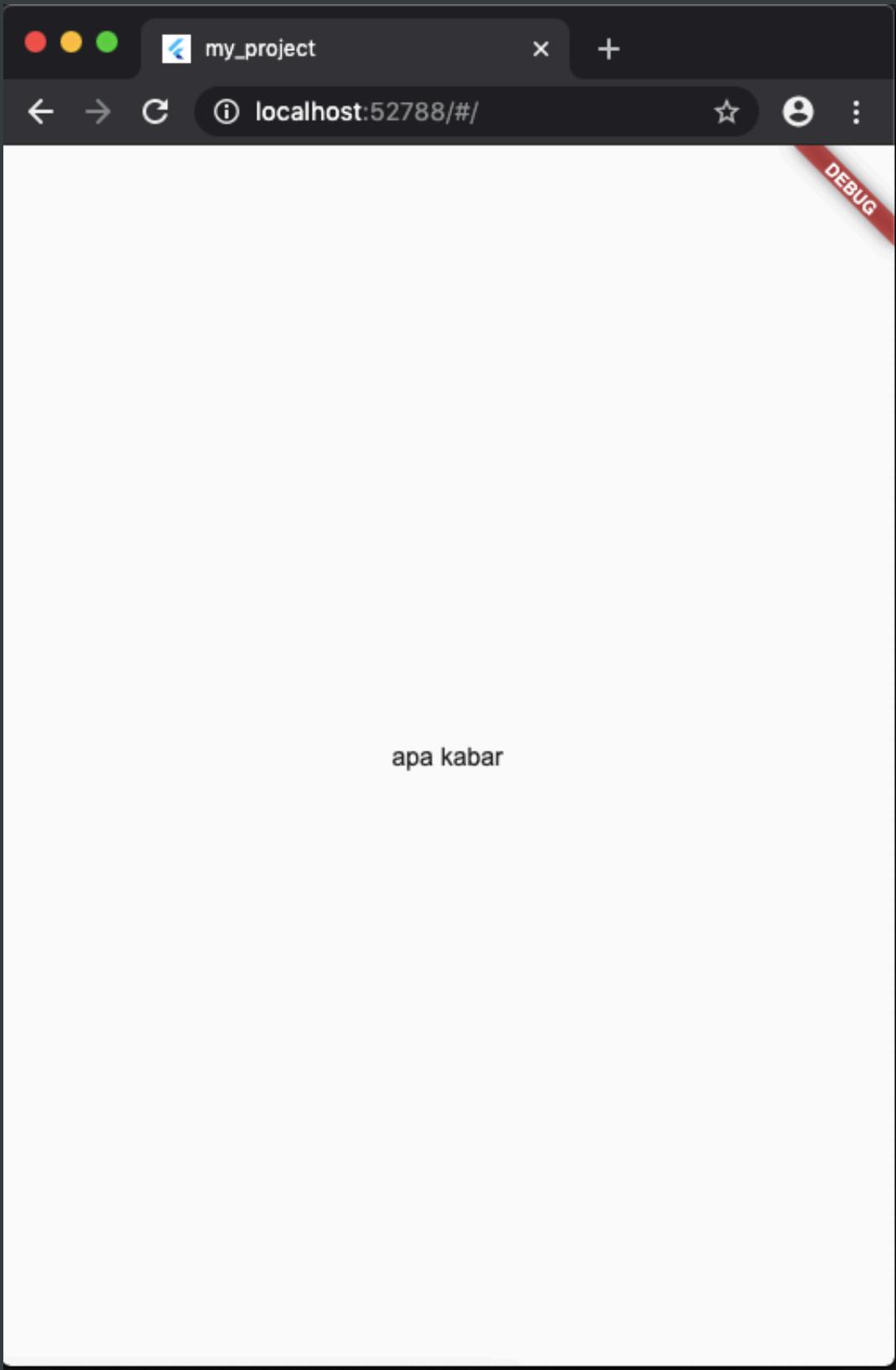
void main(){
  runApp(MyApp());
}

class MyApp extends StatelessWidget{
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      title: "my_project",
      theme: ThemeData(
        primarySwatch: Colors.blue
      ),
      home: MyHomePage(),
    );
  }
}

class MyHomePage extends StatefulWidget{
  @override
  _MyHomePageState createState() => _MyHomePageState();
}

class _MyHomePageState extends State<MyHomePage> {
  @override
  Widget build(BuildContext context) {
    // TODO: implement build
    return Scaffold(
      body: Center(
        child: Text("apa kabar"),
      ),
    );
  }
}
```

lalu jalankan dengan kerik `r` ' r ' doang ya , pada kolom terminal , nanti otomati 'reload' r artinya reload



ok sekarang saya jelaskan detailnya ,

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

ini adalah import yang dimport adalah berupa class, import artinya mengikutkan , atau mudahnya ya mengimportkan , sebenarnya diimport tapi gk keliatan, diambilkan class dari file material.dart yang sudah dibuatkan oleh team developer flutter jika penasaran isinya bisa lihat dengan arahkan mouse pada importa lalu tekan command+klik nanti otomatis diarahkan ke file tujuan , tp gk terlalu penting , sekedar tahu aja

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

void itu artinya adalah function, adalah jalur yang lelah disediakan oleh flutter untuk menjalankan class app nya , disitu ada keterangan runApp(MyApp()) jadi perintah itu untuk menjalankan class MyApp() bingung kan ?? sama dulu saya juga bingung , udah gk usah terlalu dipikir , lanjut nanti paham dengan sendirinya

```
class MyApp extends StatelessWidget{
    @override
    Widget build(BuildContext context) {
        return MaterialApp(
            title: "my_project",
            theme: ThemeData(
                primarySwatch: Colors.blue
            ),
            home: MyHomePage(),
        );
    }
}
```

class MyApp ini adalah nama class buatan kita sendiri , nama bebas gk mesti MyApp jika kalian ingin menggantinya dengan nama sendiri contoh class ProgramKu bisa tp jangan lupa ganti yg di runApp() diganti dengan runApp(ProgramKu()) udah multi ada bayangan ?? harusnya sudah ya , kl belum berarti kopinya habis hahah, selanjutnya extend yang artinya perpanjangan, lalu sebelahnya ada StatelessWidget kalo bahasa indonesianya apa ya ... mungkin bisa diartikan di google terjemah , tp saya pernah tp masih bingung untuk menjelaskannya hahaha, state = wilayah , less = kuran, widget = widget, bingung kan , lewati aja dulu sambil jalan nanti paham kok hahah , untuk state ada 2 yang sering dipakai StatelessWidget dan StatefulWidget mungkin artian mendekati yang mudah dipahami adalah tempelan ya , jadi sesuatu yang bisa ditempelkan , jaadi gini ya setiap kodingan bayangkan itu adalah objek , kl didunia nyata mungkin lebih mendekati itu benda ya , atau objek , eh puyeng hahah lanjuttttt..... kedepan akan saya jelaskan lebih detail, ok dibawahnya ada @override '@' bacanya 'at' = "pada" , override = menunggangi, jadi class StatelessWidget ini sudah punya method build dan kita bisa menungganginya , atau ngikutin kode kita , hemmm, masih susah untuk dipahami , ok lanjut nanti kita bahas mengenai class, oh iya untuk class diflutter , penunisannya

harus menggunakan hurus kecil , dan nama classnya harus dimulai dengan huruf besar dengan type camel case atau gaya punuk onta, contoh class ProgramPertamaKu untuk setiap nilai sepsi dihilangkan dan huruf pertama diganti huruf besar diikuti huruf kecil, lanjut setelah itu dibawahnya ada return atau pengembalian , dikembalikan , "catatan setiap istilah yang yang saya terangkan saya dapatkan dari otodidak mungkin tidak sesuai dengan makna sebenarnya , namun itulah istialah yg mendekati yang bisa saya pahami" ok lanjut return apapun nilai yang ada dalam metoth (metode) yang kita berikan akan dikembalikan pada method override , lanjut dibawahnya MaterialApp() ini adalah class yang sudah disediaakan oleh fluter yang ,bisa kita gunakan

jika kita arahkan mouse pada class MaterialApp() akan muncup popup turunan metodnya yang bisa kita pakai

```
class MyApp extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      title: (new) MaterialApp MaterialApp({Key key, GlobalKey<NavigatorState> navigatorKey, Widget home, Map<String theme g, Widget Function(BuildContext)> routes, String initialRoute, Route<dynamic> Function(RouteSettings) pri onGenerateRoute, List<Route<dynamic>> Function(String) onGenerateInitialRoutes, Route<dynamic> Functi ), // on(RouteSettings) onUnknownRoute, List<NavigatorObserver> navigatorObservers, Widget Function(BuildCon home: text, Widget) builder, String title, String Function(BuildContext) onGenerateTitle, Color color, Theme Data theme, ThemeData darkTheme, ThemeMode themeMode, Locale locale, Iterable<LocalizationsDelegate<dy ), ); // M l
    }
  }
}
class MyHom @override _MyHomePa
{
  package:flutter/src/material/app.dart
}
Creates a MaterialApp.
class _MyHomePageState extends State<MyHomePage> {
  @override
```

termasuk pada kodingan ini adalah title , theme, dan home tidak hanya pada itu , tapi banyak seperti pada keterangan pada gambar, title untuk menamai aplikasi kita theme biasanya berisi pengaturan warna dan lainnya , next kita bahas, lalu ada home untuk mengarahkan ke class home, hemmm ... ok sebelum lanjut mungkin ada baiknya saya jelaskan mengenai **class , method, variable dan OOP** agar di next step tidak mengalami kendala

CLASS !

jadi di tutorial pembuatan aplikasi android ini kita menggunakan flutter flutter itu adalah framework yang artinya adalah kumpulan dari beberapa class dan method yang dibuat untuk mempermudah kita sebagai developer , sedangkan bahas pemrograman sendiri menggunakan dart

jadi didunia ini banyak sekali bahasa pemrograman , contoh : C , C++ , C## , JAVA , JAVA SCRIPT , PHP dan banyak lainnya , bisa mencari refensi di google , nah bahasa pemrograman itu fungsinya adalah perintah , mesin atau computer , atau handphone, hanya mengerti 0 nol dan 1 , dan tidak mengerti bahasa manusia , untuk lebih lanjut memahami 0 dan 1 , bisa langsung ke mbh google , disini saha hanya akan menjelaskan yg perlu saja agar ada gambaran yang mudah dipahami,sesuai judos tutorial ini , dari noll , mungkin bangi yang sudah mengerti akan terasa

membosankan , jika itu bisa di skip aja ke next step, tapi kali ini saya ingin memberikan wawasan bagi yang benar benar ingin memulai dari nol , namun masih ada semangat kuat untuk memahaminya , ok lanjut, jadi mesin tdk bisa mengerti perintah kita , disitulah diadakan bahasa pemprograman agar mesin atau komputer bisa mengerti apa yang kita maksut, walau setelah itu masih adala lagi pemprosesan yang tidak perlu kita ketahui secara detailnya , nah dalam bahas pemprograman itu telah diatur sedemikian rupa agar mesin bisa mengerti kita , dan kita bisa mengerti kebiasaan mesin apa saja yang sisa dilakukan olles mesin dan apa saja yang tidak bisa dilakukan oleh mesing , dengan aturan yang telah dibuat oleh pengembang sedemikian rupa , agar manusia lebih mudah memerintah mesin, jadi bahasa pemprograman saat ini yang ada adalah teknikatau cara terbaik untuk cmanusia bisa memerintah mesin , waalau ada cara yg lebih ribet lagi.

nah dari keterangan diatas kita bisa paham sebenarnya ada beberapa aturan agar kita bisa memerintah mesin, seperti halnya manusia bahasa pemprograman juga mempunyai karakteristiknya sendiri , walau semuanya sebenarnya mirip , contoh jika dibahasa manusia , "bahasa inggris" eat, indonesia makan, jawa mangan, dan lainnya , nah ini ada lang lput , seperti halnya cabang bahasa yang lainnya , bahasa pemprograman juga seperti itu , jika kita ingin menguasainya , kita harus membiasakan diri, beda dengan mate matika ya , yang kita benar benar harus menggunakan yang namanya IQ , pemberian dari lahir , bahasa pemprograman tidak seperti itu, bahasa pemprograman sama seperti bahasa manusia yang lain cara menguasainya , kita hanya butuh pembiasaan diri , jadi yg merasa IQ nya normal cendrung dibawah rata rata , jangan kuatir , anda cukup mebiasakan diri , anda tidak membutuhkan IQterlalu banyak untuk menguasainya, contoh sederhana , jika anda dilepas keamirika 2tahun dan disana tidak ada yg bisa berbahasa indonesia , dan hanya berbahasa inggris , dalamkurung waktu dua tahun apakan anda masih kesulitan berbahasa inggris , ya tentu tidak, anak anak diamerika itu tidak perlu sekolah jika hanya ingin menguasai bahasa ingris,

sudah ya basa basinya , kita lanjut

peraturan dalam bahasa pemprograman yang umum itu ada

1. class
2. variable
3. method

itu yang umum ya , walau tidak semuanya seperti itu, biasanya yang seperti itu akan sering kita jumpai pada bahasa pemprograman yang bertipe object orientation proramming atau pemprograman yang berfokus pada objek (dalam artian sebenarnya)

nah jika kita ingin memberika perintah pada mesin atau lebih mudahnya dalam kontex kali ini adalah hp android, jadi jika kita ingin memerintah hp android halpertama yang harus kita fikirkan adalah nama perintahnya , dan itu tersebut, atau terdeklarasi dalam class, (mungkin ini bukan pengertian sebenarnya , tp saya berusaha agar mudah untuk dipahami, dengan pendekatan logikan dan pengandaian),

```
class MunculkanKalimat{  
}
```

untuk aturan penulisannya harus seperti itu ya ,jadi kali ini saya ingin memerintah hanphone saya agar mau memunculkan huruf atau kalimat sesuai keinginan saya

selanjutnya adalah deklarasi variabel atau tempat, kembali seperti penjelasan sebelumnya , hanphone saya tidak mengeti apa apa , kita lah sebagai manusia yg memerintah diharuskan untuk tahu jika hanphone saya tidak mengeti jadi kita membuat cara agar handphone kita mengeti apa yang kita maksut,

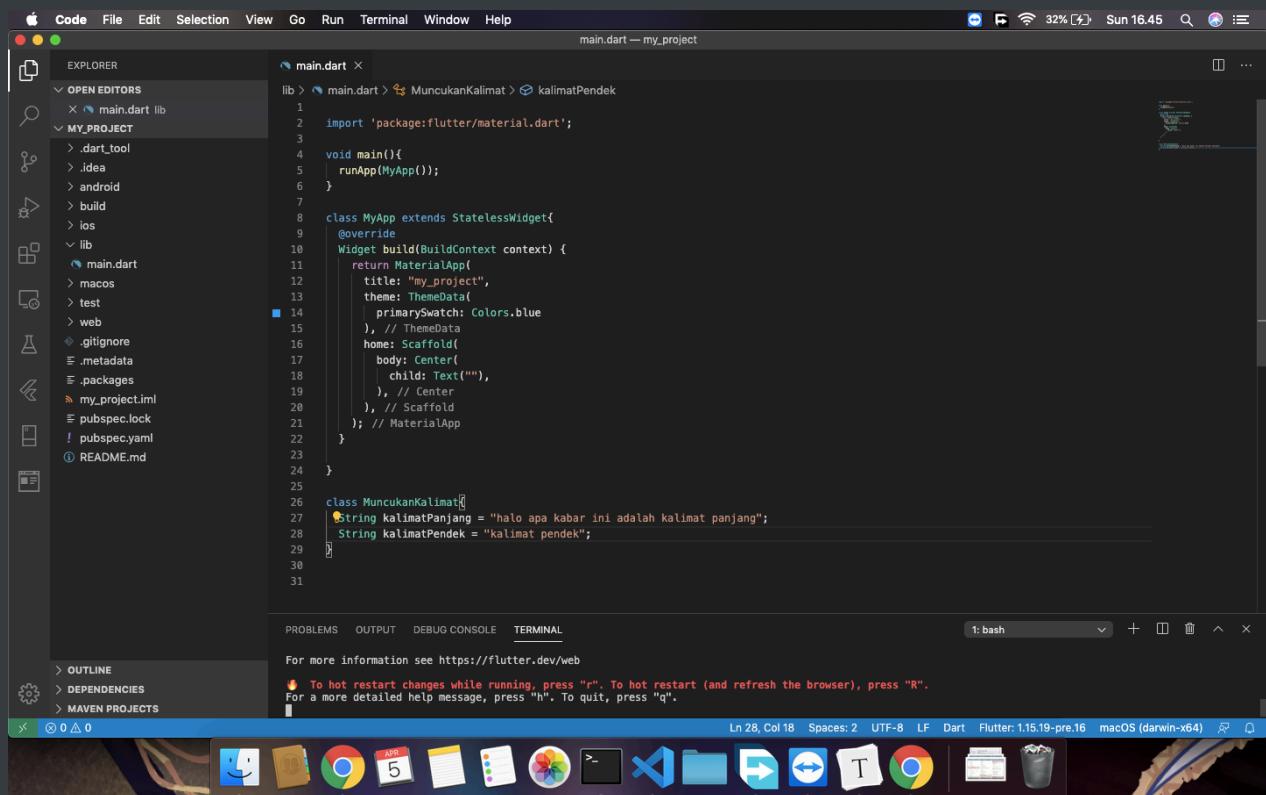
kali ini kita akan menuliskan sebuah pengertian , atau artian yang terkandung dalam variabel

```
class MunculkanKalimat{  
    String kalimatPanjang;  
    String kalimat Pendek;  
}
```

diatas kita sudah membuat pengertian , `kalimatPanjang` dan `kalimatPendek` ok sebagai manusia kita pasti tahu apa itu kalimat panjang dan kalimat pendek , tp ingat handphone kita tidak tahu, selanjutnya kita menjelaskan ke handphone kita apa itu kalimat panjang dan apa itu kalimat pendek agar sesuai dengan keinginan kita

```
class MunculkanKalimat{  
    String kalimatPanjang = "halo apa kabar ini adalah kalimat panjang";  
    String kalimatPendek = "kalimat pendek";  
}
```

ok , untuk lebih mudah memahaminya kita kembali ke codingan flutter kita yang di vscode dan langsung kita terapkan



buat kodingan seperti diatas

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

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

class MyApp extends StatelessWidget{
    @override
    Widget build(BuildContext context) {
        return MaterialApp(
            title: "my_project",
            theme: ThemeData(
                primarySwatch: Colors.blue
            ),
            home: Scaffold(
                body: Center(
                    // catatan : penerapan class yang kita buat tadi
                    child: Text(MuncukanKalimat().kalimatPanjang),
                ),
            ),
        );
}
```

```
}

}

// catatan : kalian bisa membuat catatan dengan memberikan garis miring
// dua kali // <-
// ini adalah catatan untuk kita sebagai programer dan tidak akan dibaca
oleh mesin

class MunculkanKalimat{
    String kalimatPanjang = "halo apa kabar ini adalah kalimat panjang";
    String kalimatPendek = "kalimat pendek";
}
```

dan ketikkan kodingan class yang kita buat tadi lalu jalankan dengan ketik `r` lalu enter pada terminal
untuk perintah pada terminal

`r` untuk reload biasa

`R` untuk reload semuanya dari awal

`h` untuk help

`q` untuk quit alias keluar, atau berhenti,

untuk menjalankan perintah jangan lupa `flutter run` untuk emulator chrome `flutter run -d chrome`

```
main.dart
lib > main.dart > MyApp > build
10  Widget build(BuildContext context) {
11    return MaterialApp(
12      title: "my_project",
13      theme: ThemeData(
14        primarySwatch: Colors.blue
15      ), // ThemeData
16      home: Scaffold(
17        body: Center([
18          // catatan : penerapan class yang kita buat tadi
19          child: Text(MuncukanKalimat().kalimatPanjang),
20        ], // Center
21      ), // Scaffold
22    ); // MaterialApp
23  }
24
25 }
26
27 // kalian bisa membuat catatan dengan memberikan garis miring dua kali // <-
28 // ini adalah catatan untuk kita sebagai programer dan tidak akan dibaca oleh
29
30 class MuncukanKalimat{
31   String kalimatPanjang = "halo apa kabar ini adalah kalimat panjang";
32   String kalimatPendek = "kalimat pendek";
33 }
34
35

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
been thoroughly tested in production environments.
For more information see https://flutter.dev/web

🔥 To hot restart changes while running, press "r". To hot restart (and refresh the browser), press "F5".
For a more detailed help message, press "h". To quit, press "q".

Performing hot restart...
4,476ms (1)
Restarted application in 4,478ms.

Performing hot restart...
31ms
Restarted application in 32ms.
```

kalimat panjang yang kita perintahkan tadi sudah muncul di hanphone atau di emulator kita , ginama ... sudah mulai ada bayangan tipis tipis ??

ok lanjut kita ganti dengan kalimat pendek , apa yang akan muncul

```
home: Scaffold(
  body: Center(
    // catatan : penerapan class yang kita buat tadi
    child: Text(MuncukanKalimat().kalimatPendek),
  ),
),
```

ok lanjut ke penjelasan `class` selanjutnya , disini kita sudah bisa membuat perintah sederhana untuk hanphone kita agar bisa memunculkan kalimat sesuai dengan kemauan kita,

lanjut ke tingkatan `class` yang lebih kompleks,

```
class MuncukanKalimat{
  String kalimatPanjang;
  String kalimatPendek;

  // ini namanya constructor atau pembentuk
```

```
MuncukanKalimat({this.kalimatPanjang,this.kalimatPendek});  
  
// ini namanya metode  
String getKalimatPanjang() => kalimatPanjang;  
setKalimatPanjang(String nilai)=> kalimatPanjang = nilai;  
  
// ini namanya metode  
String geKalimatPendek() => kalimatPendek;  
setKalimatPendek(String nilai) => kalimatPendek = nilai;  
}
```

kita modifikasi clas yang kita buat tadi lalu terapkan pada vscode lalu reload r

```
import 'package:flutter/material.dart';  
  
void main(){  
  runApp(MyApp());  
}  
  
class MyApp extends StatelessWidget{  
  @override  
  Widget build(BuildContext context) {  
    return MaterialApp(  
      title: "my_project",  
      theme: ThemeData(  
        primarySwatch: Colors.blue  
>,  
      home: Scaffold(  
        body: Center(  
          // catatan : penerapan class yang kita buat tadi  
          child: Text(MuncukanKalimat(kalimatPanjang: "ini adalah kalimat  
panjang yang bisa saya rubah langsung ditempat").kalimatPanjang ),  
        ),  
      ),  
    );  
  }  
  
  // kalian bisa membuat catatan dengan memberikan garis miring dua kali //  
  <-
```

```
// ini adalah catatan untuk kita sebagai programer dan tidak akan dibaca oleh mesin

class MuncukanKalimat{
    String kalimatPanjang;
    String kalimatPendek;

    // ini namanya constructor atau pembentuk
    MuncukanKalimat({this.kalimatPanjang,this.kalimatPendek});

    // ini namanya metode
    String getKalimatPanjang() => kalimatPanjang;
    setKalimatPanjang(String nilai)=> kalimatPanjang = nilai;

    // ini namanya metode
    String geKalimatPendek() => kalimatPendek;
    setKalimatPendek(String nilai) => kalimatPendek = nilai;
}
```

dengan method yang kita buat tadi kita sudah bisa memodifikasi apa yang akan kita munculkan secara langsung dan kita terapkan pada codingan,

gimana uda mulai sakit kepala ?? heheh pelanpelan , nanti ngerti dengan sendirinya, lanjut ya ..

tambahkan kodingan ini pada akhir class

```
String jadiHurufBesarSemua(){
    return kalimatPanjang.toUpperCase();
}
```

lalu terapkan pad vscode

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

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

```
}

class MyApp extends StatelessWidget{
    @override
    Widget build(BuildContext context) {
        return MaterialApp(
            title: "my_project",
            theme: ThemeData(
                primarySwatch: Colors.blue
            ),
            home: Scaffold(
                body: Center(
                    // catatan : penerapan class yang kita buat tadi
                    child: Text(MuncukanKalimat(kalimatPanjang: "ini adalah kalimat
panjang yang bisa saya rubah langsung ditempat").jadiHurufBesarSemua() ),
                ),
            ),
        );
    }
}

// kalian bisa membuat catatan dengan memberikan garis miring dua kali //
<-
// ini adalah catatan untuk kita sebagai programer dan tidak akan dibaca
oleh mesin

class MuncukanKalimat{
    String kalimatPanjang;
    String kalimatPendek;

    // ini namanya constructor atau pembentuk
    MuncukanKalimat({this.kalimatPanjang,this.kalimatPendek});

    // ini namanya metode
    String getKalimatPanjang() => kalimatPanjang;
    setKalimatPanjang(String nilai)=> kalimatPanjang = nilai;

    // ini namanya metode
    String geKalimatPendek() => kalimatPendek;
    setKalimatPendek(String nilai) => kalimatPendek = nilai;

    String jadiHurufBesarSemua(){
        return kalimatPanjang.toUpperCase();
    }
}
```

```
}
```

perubahan disini juga

```
home: Scaffold(  
    body: Center(  
        // catatan : penerapan class yang kita buat tadi  
        child: Text(MunculkanKalimat(kalimatPanjang: "ini adalah kalimat  
panjang yang bisa saya rubah langsung ditempat").jadiHurufBesarSemua() ),  
    ),  
,
```

lalu reload r

TARAAAAAAA

hurufnya jadi huruf besar semua ... kok bisa ???

nah itulah fungsinya method

gimana ?? uda ada bayangan dikit dikit ??



terapkan kodingan berikut pada vscode lalu rubah pada `jadiBesar` menjadi iya atau tidak lalu reload dan lihat hasilnya

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

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

class MyApp extends StatelessWidget{
    @override
    Widget build(BuildContext context) {
        return MaterialApp(
            title: "my_project",
            theme: ThemeData(
                primarySwatch: Colors.blue
            ),
            home: Scaffold(
                body: Center(
                    // catatan : penerapan class yang kita buat tadi
                    child: Text(MuncukanKalimat(
                        kalimatPanjang: "ini adalah kalimat panjang yang bisa saya
                        rubah langsung ditempat",
                        jadiBesar: "iya"
                    ).harusJadiBesarAtauTidak()),
                ),
            ),
        );
    }
}

// kalian bisa membuat catatan dengan memberikan garis miring dua kali //
<-
// ini adalah catatan untuk kita sebagai programer dan tidak akan dibaca
oleh mesin

class MuncukanKalimat{
    String kalimatPanjang;
    String kalimatPendek;
    String jadiBesar;

    // ini namanya constructor atau pembentuk
    MuncukanKalimat({this.kalimatPanjang,this.kalimatPendek,this.jadiBesar});

    // ini namanya metode
    String getKalimatPanjang() => kalimatPanjang;
    setKalimatPanjang(String nilai)=> kalimatPanjang = nilai;
```

```
// ini namanya metode
String geKalimatPendek() => kalimatPendek;
setKalimatPendek(String nilai) => kalimatPendek = nilai;

String getJadiBesar()=> jadiBesar;
setJadiBesar(String nilai)=> jadiBesar = nilai;

String jadiHurufBesarSemua(){
    return kalimatPanjang.toUpperCase();
}

String harusJadiBesarAtauTidak(){
    // catatan : jika apakah mengandung adalah iya maka kalimat panjang
    // akan dirubah ke huruf besar
    if(jadiBesar.contains("iya")){
        return kalimatPanjang.toUpperCase();
        // jika tidak mengandung iya maka kalimatpanjang
        // akan dirubah ke huruf kecil
    }else{
        return kalimatPanjang.toLowerCase();
    }
}
```

jika berhasil maka tulisan bisa menjadi besar dan kecil sesuai perintah kita , ok ini adalah kalimat paling sederhana untuk memahami perintah pada bahasa pemrograman , teliti perlahan , lihat pada perubahan , bagian mana yang berubah dan efeknya apa, sebelum saya jelaskan lebih lanjut saya harap sampai sini sudah dipahami atau paling tidak ada gambaran walau samar samar

bagi yang masih bingung saya jelaskan secara perlahan ,

jadi nama perintah kita adalah `class MunculkanKalimat()`

nama objek atau variabelnya adalah `String kalimatPanjang` dan `String kalimatPendek`

untuk geter dan setternya adalah `getKalimatPanjang()` dan `setKalimatPanjang(String nilai)` dan yang kalimatPendek juga, ini berfungsi untuk mendapatkan isi dari kalimat, dan memberi nilai pada kalimat, ingat ya hanphone kita tidak tahu cara cara mendapatkannya , kita harus memberikan caranya , bagaimana mendapatkan nilainya , dan memberikan nilainya ,

dan untuk method atau metodenya adalah `jadiHurufBesarSemua()` dan `harusJadiBesarAtauTidak()`

nah pada vscode yang berwarna hijab dan dimulai dengan huruf besar itu adalah `class` perintah jika kita lihat pada vscode

```
class MyApp extends StatelessWidget{
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      title: "my_project",
      theme: ThemeData(
        primarySwatch: Colors.blue
      ), // ThemeData
      home: Scaffold(
        body: Center(
          // catatan : penerapan class yang kita buat tadi
          child: Text(MunculkanKalimat(
            kalimatPanjang: "ini adalah kalimat panjang yang bisa saya rubah langsung ditempat",
            jadiBesar: "iya"
          ).harusJadiBesarAtauTidak() ), // MunculkanKalimat // Text
        ), // Center
      ), // Scaffold
    ); // MaterialApp
}
```

Nama classnya adalah `MyApp` didalamnya ada `Widget, MaterialApp, ThemeData, Colors, Scafold, Center, Text`

jaadi bisa diambil kesimpulan ada class didalam class , disitulah penerapan classnya , bisa turun temurun didalamnya sesuai kebutuhan.

uda mulai pusing kepala barby ...??? hahah sabar gk ada sesuatu yg instan, pelajari dan resapi secara perlahan , biarkan waktu berkerja , intinya janganberhenti berfikir dan mencoba, untuk next step saya harap masalah class sudah bisa clear atau paling tidak ada bayangan yang bisa jadi acuan untuk diingat

SEKILAS DART

flutter adalah framework sedangkan dart adalah bahasa pemprogramannya , jadi flutter adalah kumpulan beberapa class yang dibuat menggunakan bahasa dart yang dibuat sedemikian rupa agar memudahkan kita untuk membuat aplikasi ,

untuk mempelajari bahasa dart lebih lanjut kalian bisa mengunjungi web resminya <https://dart.dev/>

dalam bahasa pemrograman dart ada beberapa komponen dasar yang sangat berguna jika diketahui seperti dibawah ini

```
abstract dynamic implements show
as else import static
assert enum in super
async export interface switch
await extends is sync
break external library this
case factory mixin throw
catch false new true
class final null try
const finally on typedef
continue for operator var
covariant Function part void
default get rethrow while
deferred hide return with
do if set yield
```

seperti bahasa pemrograman lainnya dart juga mempunyai perintah operator aritmatik

| no | nama | keterangan | contoh |
|----|------|---|---|
| 1 | + | penambahan | var berapa = 1+1; |
| 2 | - | pengurangan | Var berapa = 2-1; |
| 3 | * | pengali | var berapa = 2*2; |
| 4 | / | pembagi | var berapa = 4/2; |
| 5 | ~/ | Pembagi integer, mengembalikan hasil pembagian kedalam bentuk integer bulat | var berapa = 5~/3; hasilnya bukan decimal tapi bilangan bulat |
| 6 | ++ | increment /penambahan terus menerus | berapa ++; |
| 7 | -- | Decrement / pengurangan terus menerus | berapa --; |
| 8 | % | Modulo | berapa = 100%1.5; menghasilkan |

persamaan perbandingan operator

| no | nama | keterangan | contoh |
|----|------|------------------------------|-----------------------------|
| 1 | > | lebih besar dari | if(4>3) menghasilkan true |
| 2 | < | kurang dari | if(2<4) menghasilkan true |
| 3 | <= | kurang atau sama dengan | if(3<=3) menghasilkan true |
| 4 | >= | lebih besar atau sama dengan | if(5>=5) menghasilkan true |
| 5 | == | sama dengan | if(5==5) menghasilkan true |
| 6 | != | tidak sama dengan | if("kucing"!="anjing") true |

Ada masih banyak operator lainnya contoh seperti ? , ?? , | , && dan lain sebagainya akan dibahas sambil jalan , hemat space otak dulu biar gak terlalu penuh hahahah

untuk jenis type deklarasi

```
var = "dynamic (bebas)"
String = "untuk hurruf"
int = "untuk integer atau bilangan bulat 1 2 3 dan seterusnya"
double = "decimal 0.1 0.4" menggunakan titik bukan koma
bool = "pilihan yang menghasilkan true atau false, benar atau salah"
```

null adalah kosong / hampa untuk sesuatu yang belum terdeklarasi dan belum mempunyai artian

contoh

```
var air = "sirup" jenis airnya adalah sirup
var air; jika digunakan akan menghasilkan null , karena belum diberi artiannya kita tahu typenya air tapi tidak tahu dia itu sirup atau airputih biasa , jika dipertanyakan maka akan menghasilkan ketidak tahuhan , dan ini sering menghasilkan error, untuk menghindari error kita bisa mendeklarasikannya pada aliran berjalan
```

contoh :

```
var air;
...
...
```

```
air = "coca cola"  
  
dan lainnya , yang diatas adalah yang paling sering dipakai
```

disini saya hanya akan menjelaskan tentang dart secara singkat saja, walau sebenarnya ada banyak sekali yang bisa untuk dijelaskan , jadi sesuatu yang sepetutnya perlu diketahui, jika pun harus dipaksakan di terangkan semuanya , malah akan hanya menimbulkan kebingungan, karena kita akan lebih berfokus dengan metode learning by doing , belajar dengan langsung mempraktekkannya

jangan terlalu dipikirkan , nikmati saja prosesnya



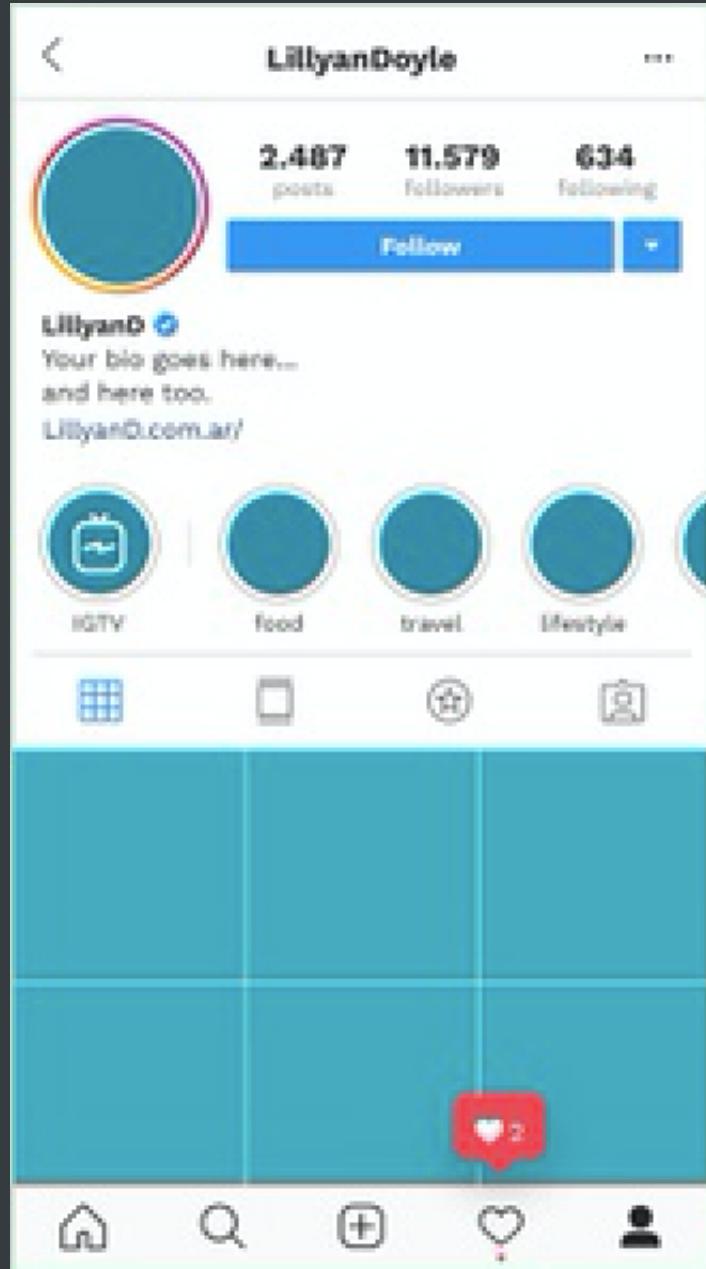
balik kelaptop

kalo suntuk , mumet santai aja dulu , bikin kopi , nonton anime , jalan jalan , freshkan pikiran dulu baru lanjut,karena perjalanan masih jauh jendral.



ngomong ngomong learning by doing , yuk kita belajar dengan langsung praktik,jadi teori diselingi praktik , teori lagi dan langsung praktik , jadi biar gk ambyar , dan menurut saya jaadi lebih efektif, karena mudah diingat

kita akan belajar membuat layout mirip instagram



ok , kita pake layout tersebut sebagai bahan acuan , untuk membuat layout , dan hapus dulu semua kodingan kita mulai dari nol ya , kaya pertamina

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

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

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

```
body: SafeArea(  
    child: Container(  
        child: Column(  
            children: <Widget>[  
                Text("aplikasi pertamaku mirip instagram")  
            ],  
        ),  
    ),  
),  
,  
);  
};  
}  
}
```

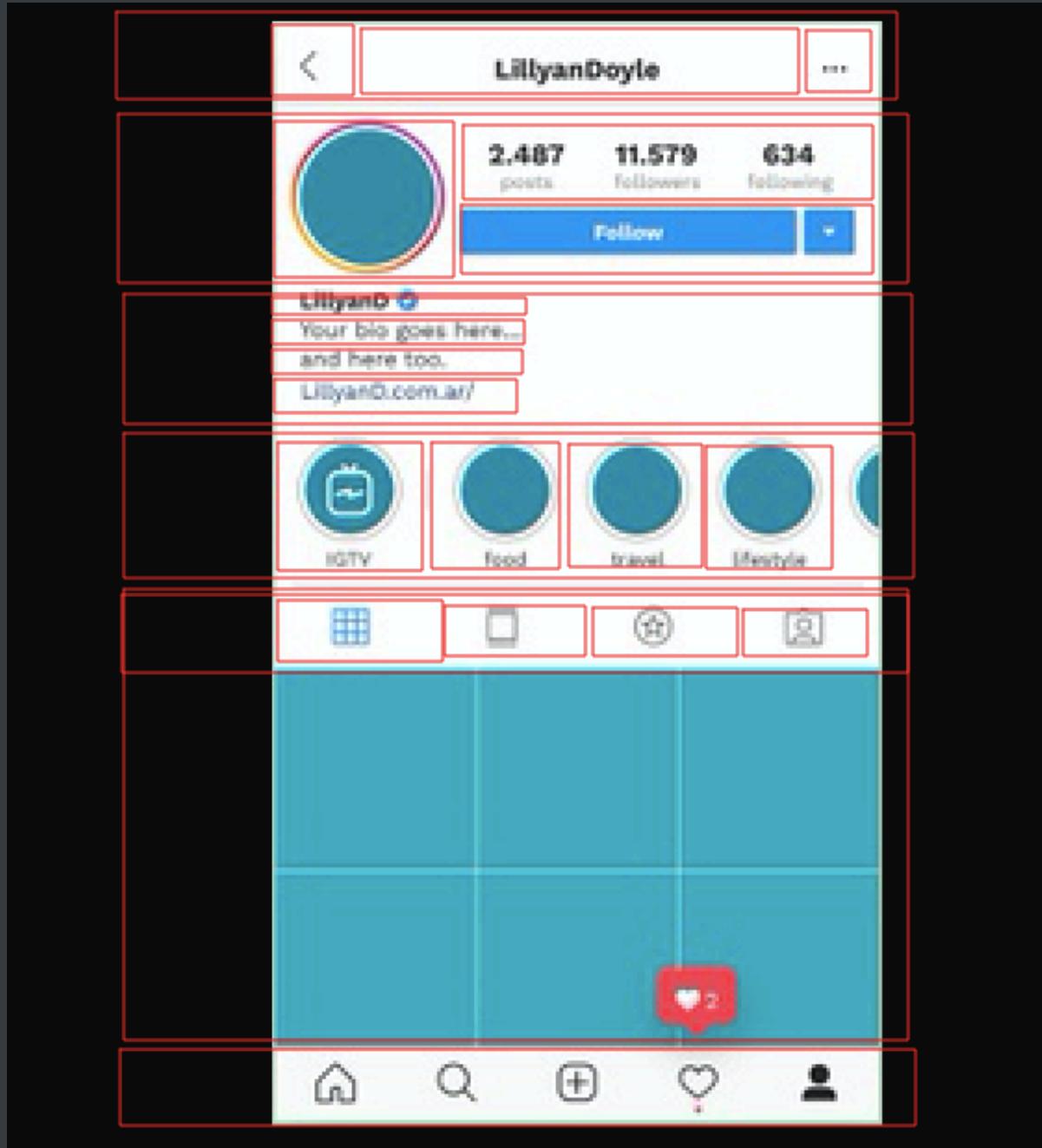
ketik koding pada vscode seperti diatas , lalu run atau reload,

pada kodingan diatas ita bisa lihat turunan classnya , ada yang baru yaitu `SafeArea` dan `Column` bahasa indonesianya kolom.

`class SafeArea()` , pada umumnya semua layar hanphone itu kotak uda lempeng aja , tp tidak dijamin sekarang karena banyak hanphone ada ponnya , nah untuk mengantisipasi itu dibuatlah class `SafeArea()` oleh developer flutter tp itu optional alias pilihan si pengkoding , mau menerapkan atau tidak dan bisa diakali dengan beberapa trik,

`class Column()`, atau kolom , seperti pengertiannya kolom , clas ini dibuat untuk membuat kolom menurun, pada dasarnya layout itu adalah tabel, jadi ada column,ada row, pengertian mudahnya yaitu kotak kebawah , dan kotak kesamping

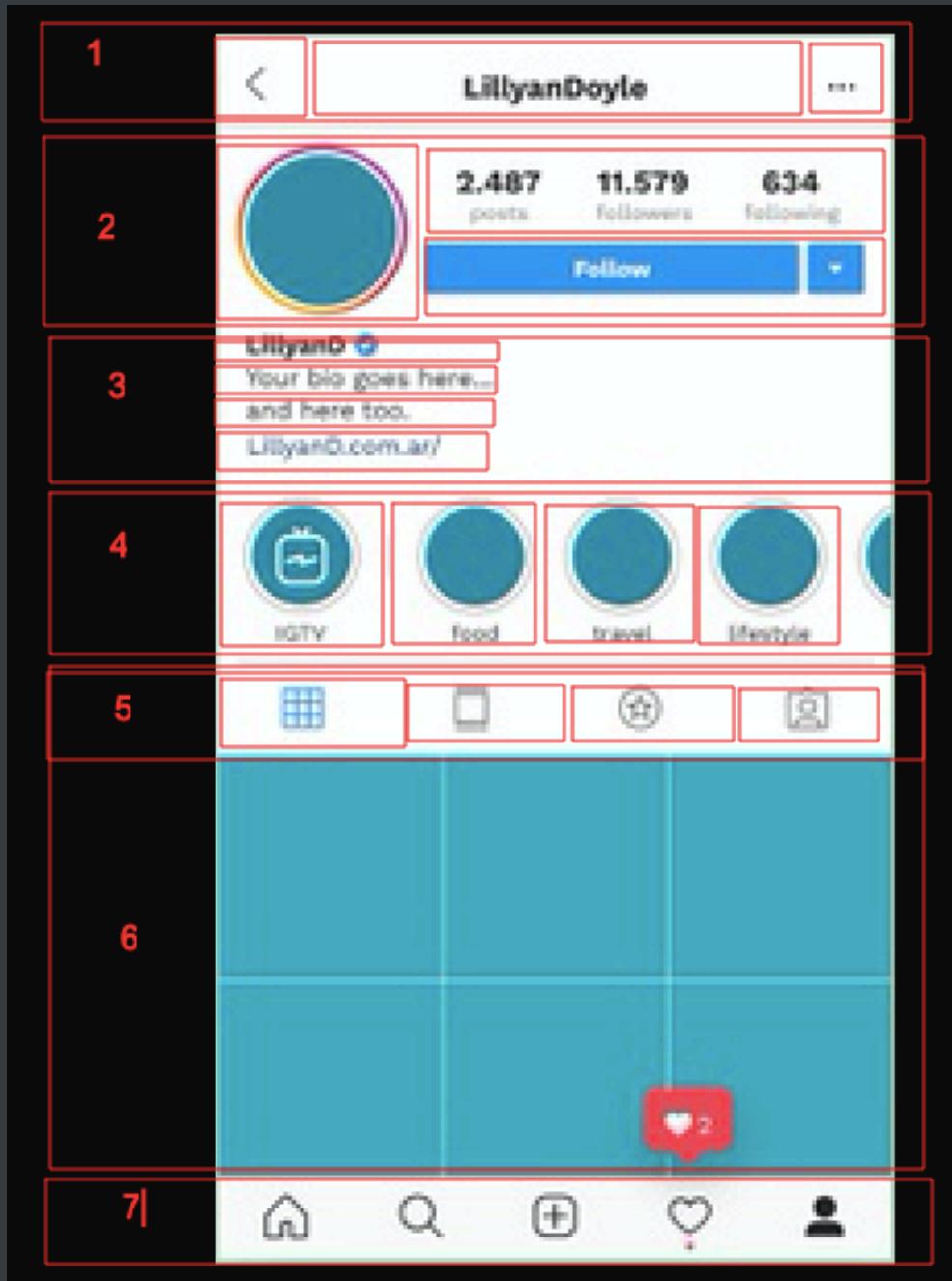




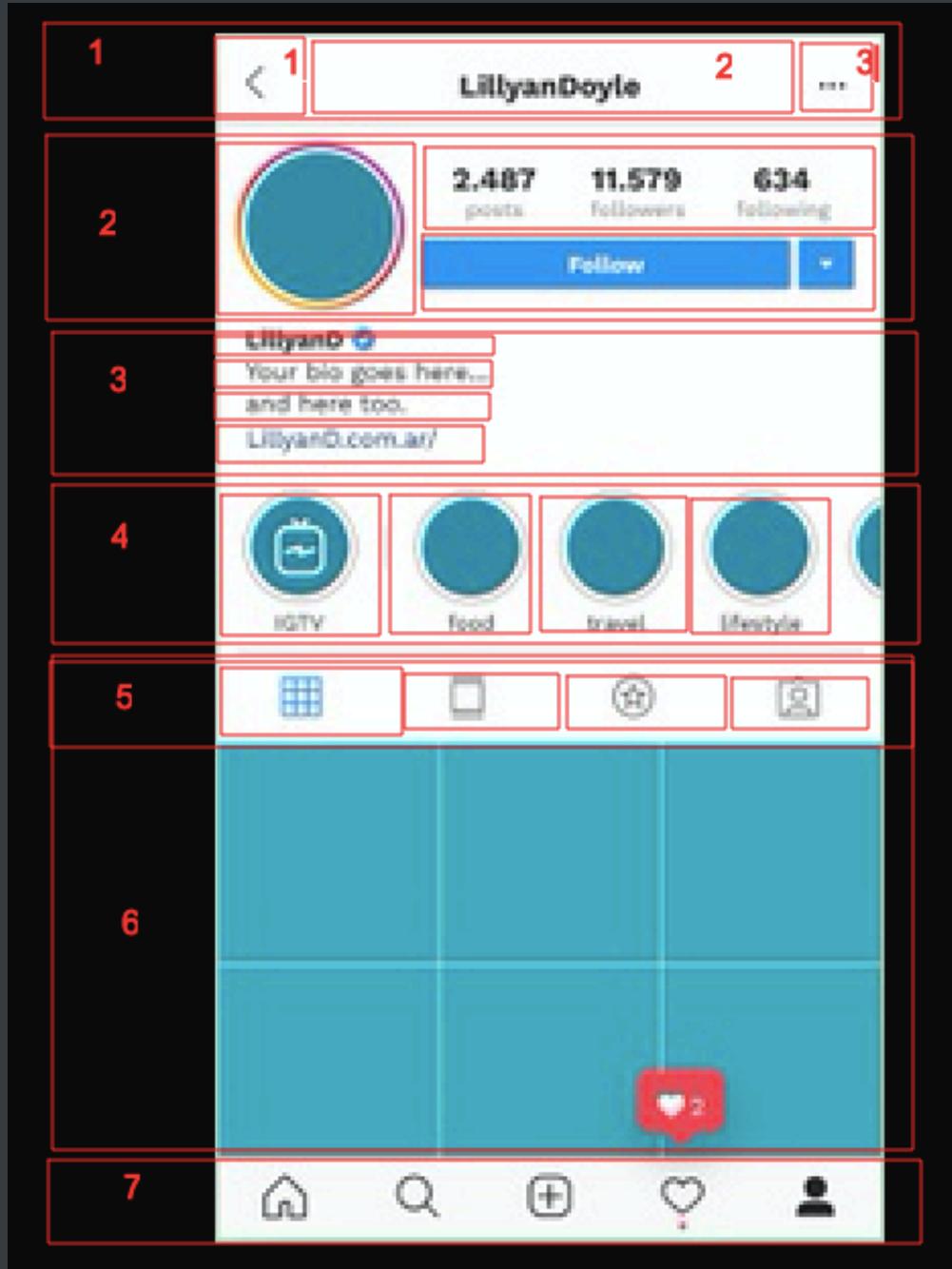
untuk mempermudah bayangan bisa dilihaht pada gambar diatas , kotak panjang panjang kebawah itu namanya colum dan kotak kecil didalam kotak panjang itu namanya row

dadn sedangkan class `Container()` adalah sebagai pembungkusnya, masih bingung?? ok langsng ke prakteknya,

pada contoh gambar diatas kita memiliki 7 kolom ... darimana ?? lihat contoh gambar dibawah



dan pada kolom pertama (1) , kita memiliki 3 row, seperti gambar dibawah



jadi kita butuh 7 kolom pada kodingan, ok langsung ko vs code

edit codingan pada vscode , dan buat seperti contoh diabawah ,lalu reload

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

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

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

```
return MaterialApp(
    home: Scaffold(
        body: SafeArea(
            child: Container(
                child: Column(
                    children: <Widget>[
                        Container(
                            child: Text("kolom 1"),
                        ),
                        Container(
                            child: Text("kolom 2"),
                        ),
                        Container(
                            child: Text("kolom 3"),
                        ),
                        Container(
                            child: Text("kolom 4"),
                        ),
                        Container(
                            child: Text("kolom 5"),
                        ),
                        Container(
                            child: Text("kolom 6"),
                        ),
                        Container(
                            child: Text("kolom 7"),
                        )
                    ],
                )
            ),
        );
    });
}
```

jika benar maka akan seperti dibawah ini



ok pada prosses ini kita sudah berhasil membuat column, next selanjutnya kita butuh row ,didalam kolom pertama,

edit pada vscode lalu reload

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

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

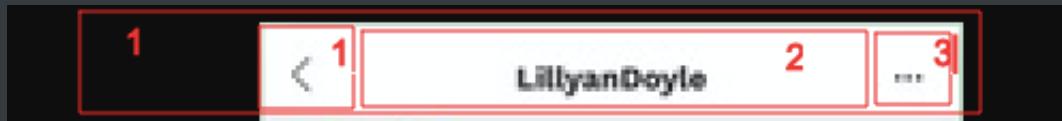
class MyApp extends StatelessWidget{
    @override
    Widget build(BuildContext context) {
        return MaterialApp(
            home: Scaffold(
                body: SafeArea(
                    child: Container(
                        child: Column(
                            children: <Widget>[
                                Container(
                                    child: Row(
                                        mainAxisAlignment: MainAxisAlignment.max,
                                        crossAxisAlignment: CrossAxisAlignment.start,
                                        children: <Widget>[
                                            Container(
                                                child: Text("kolom pertama, row pertama"),
                                            ),
                                            Expanded(
                                                child: Container(
                                                    child: Text("kolom pertama, row kedua"),
                                                ),
                                            ),
                                            child: Container(
                                                child: Text("kolom pertama, row ketiga"),
                                            ),
                                        ],
                                    ),
                                ),
                                Container(
                                    child: Text("kolom 2"),
                                ),
                                Container(
                                    child: Text("kolom 3"),
                                ),
                                Container(
                                    child: Text("kolom 4"),
                                ),
                            ],
                        ),
                    ),
                ),
            ),
        );
    }
}
```




nah dari sini uda mulai ada banyangan ??apa itu colum dan row??

untuk menghilangkan tulisan debug yang mengganggu bisa tambahkan kode berikut pada materian app

```
class MyApp extends StatelessWidget{
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      // yg dibawah ini untuk menghilangkan tulisan debug
      debugShowCheckedModeBanner: false,
      home: Scaffold(
        body: SafeArea(
          child: Container(
            child: Column(
```



kembali ke vscode edit code seperti dibawah lalu reload

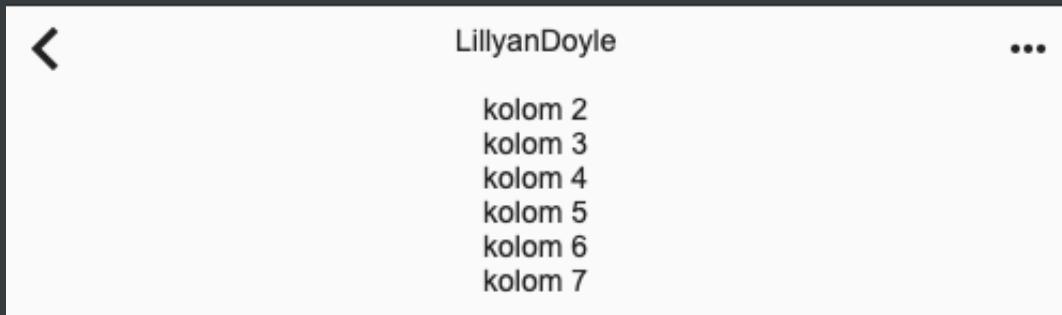
```
import 'package:flutter/material.dart';

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

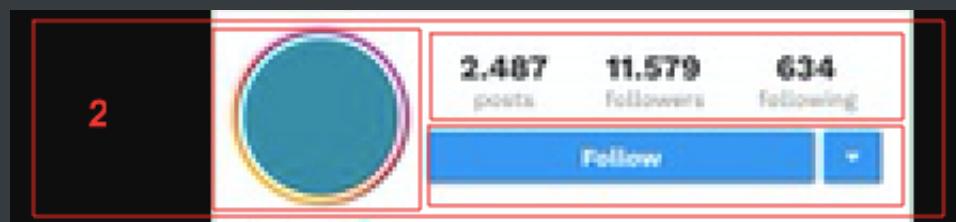
class MyApp extends StatelessWidget{
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      debugShowCheckedModeBanner: false,
      home: Scaffold(
        body: SafeArea(
          child: Container(
            child: Column(
              children: <Widget>[
                Container(
                  padding: EdgeInsets.all(8),
                  child: Row(
                    mainAxisAlignment: MainAxisAlignment.spaceEvenly,
                    children: <Widget>[
                      Container(
                        width: 100,
                        height: 100,
                        color: Colors.red,
                      ),
                      Container(
                        width: 100,
                        height: 100,
                        color: Colors.green,
                      ),
                      Container(
                        width: 100,
                        height: 100,
                        color: Colors.blue,
                      ),
                    ],
                  ),
                ),
              ],
            ),
          ),
        ),
      ),
    );
  }
}
```

```
        crossAxisAlignment: CrossAxisAlignment.start,
        children: <Widget>[
            Container(
                child: Icon(Icons.arrow_back_ios),
            ),
            Expanded(
                child: Container(
                    alignment: Alignment.center,
                    child: Text("LillyanDoyle"),
                ),
            ),
            Container(
                child: Icon(Icons.more_horiz),
            )
        ],
    ),
    Container(
        child: Text("kolom 2"),
    ),
    Container(
        child: Text("kolom 3"),
    ),
    Container(
        child: Text("kolom 4"),
    ),
    Container(
        child: Text("kolom 5"),
    ),
    Container(
        child: Text("kolom 6"),
    ),
    Container(
        child: Text("kolom 7"),
    )
],
)
),
),
),
),
);
}
}
```

hasilnya seperti dibawah



ok sekarang kita sudah berhasil mengerjakan layout pada kolom pertama,mari kita lanjut ke kolom kedua



ganti kodingan kolom kedua dengan code berikut

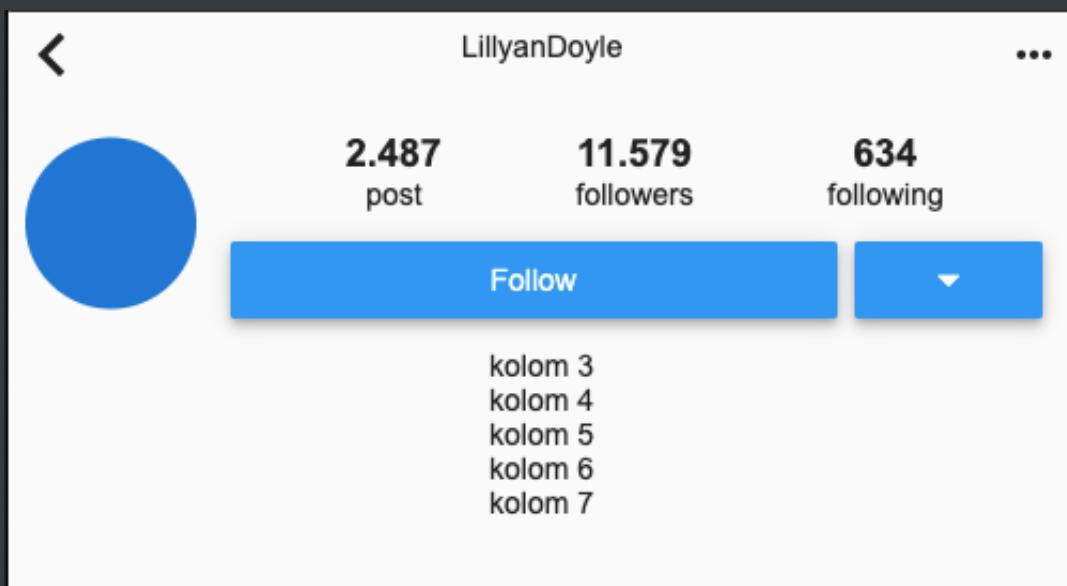
```
Container(
    child: Row(
        mainAxisAlignment: MainAxisAlignment.max,
        children: <Widget>[
            Container(
                padding: EdgeInsets.all(8),
                child: CircleAvatar(
                    radius: 40,
                ),
            ),
            Expanded(
                child: Container(
                    padding: EdgeInsets.all(8),
                    child: Column(
                        children: <Widget>[
                            Container(
                                child: Row(
                                    mainAxisSize:
```

```
        children: <Widget>[
            Text("2.487",style:
                TextStyle(fontSize: 18,fontWeight: FontWeight.w700),),
                Text("post")
            ],
        ),
    ),
    Container(
        padding: EdgeInsets.all(8),
        child: Column(
            children: <Widget>[
                Text("11.579",style:
                    TextStyle(fontSize: 18,fontWeight: FontWeight.w700),),
                    Text("followers")
                ],
            ),
        ),
    ),
    Container(
        padding: EdgeInsets.all(8),
        child: Column(
            children: <Widget>[
                Text("634",style:
                    TextStyle(fontSize: 18,fontWeight: FontWeight.w700),),
                    Text("following")
                ],
            ),
        ),
    ),
    ],
),
),
Container(
    child: Row(
        children: <Widget>[
            Expanded(
                child: RaisedButton(
                    color: Colors.blue,
                    child: Text("Follow",style:
                        TextStyle(color: Colors.white),),
                    onPressed: () => print("follow"),
                ),
            ),
            Container(
                margin:
                    EdgeInsets.symmetric(horizontal: 8),
                child: RaisedButton(

```

```
        child:  
        Icon(Icons.arrow_drop_down,color: Colors.white),  
                color: Colors.blue,  
                onPressed: () => print("down"),  
            ),  
        )  
    ],  
),  
)  
],  
),  
),  
),  
),  
),  
)
```

jika berhasil akan terbentuk seperti gambar dibawah ini



dan untuk kolom ke 3 dan seterusnya , edit kode seperti berikut

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

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

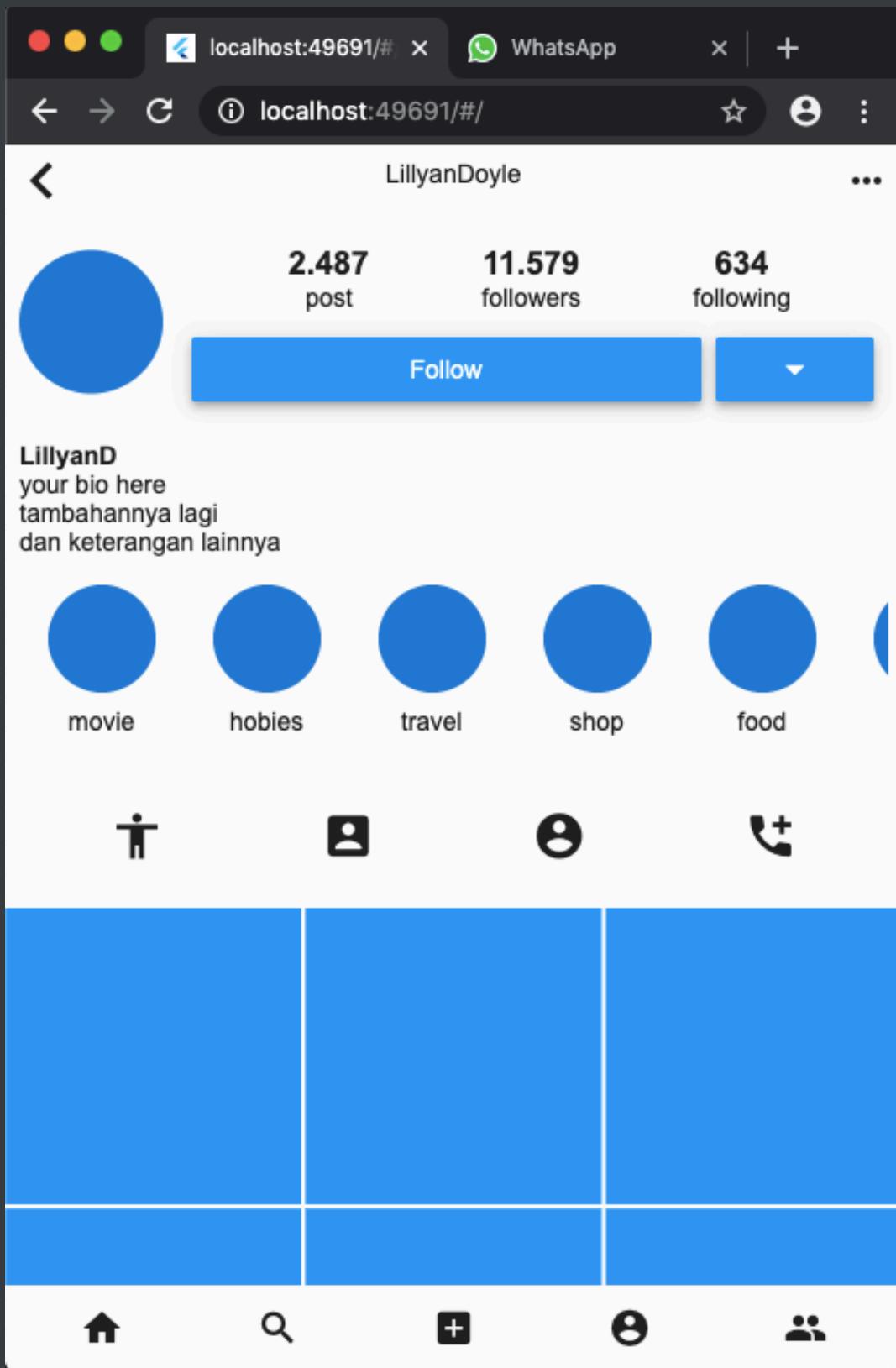
class MyApp extends StatelessWidget{
  final _list =
  ["movie", "hobbies", "travel", "shop", "food", "igTv", "jobs", "home"];
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      debugShowCheckedModeBanner: false,
      home: Scaffold(
        body: SafeArea(
          child: Container(
            child: Column(
              children: <Widget>[
                Container(
                  padding: EdgeInsets.all(8),
                  child: Row(
                    mainAxisAlignment: MainAxisAlignment.max,
                    crossAxisAlignment: CrossAxisAlignment.start,
                    children: <Widget>[
                      Container(
                        child: Icon(Icons.arrow_back_ios),
                      ),
                      Expanded(
                        child: Container(
                          alignment: Alignment.center,
                          child: Text("LillyanDoyle"),
                        ),
                      ),
                      Container(
                        child: Icon(Icons.more_horiz),
                      )
                    ],
                  ),
                ),
                Container(
                  child: Row(
                    mainAxisAlignment: MainAxisAlignment.max,
                    children: <Widget>[
                      Container(
                        padding: EdgeInsets.all(8),
                        child: CircleAvatar(
                          radius: 40,
```

```
        ),
    ),
    Expanded(
        child: Container(
            padding: EdgeInsets.all(8),
            child: Column(
                children: <Widget>[
                    Container(
                        child: Row(
                            mainAxisAlignment:
MainAxisAlignment.spaceEvenly,
                            mainAxisSize: MainAxisSize.max,
                            children: <Widget>[
                                Container(
                                    padding: EdgeInsets.all(8),
                                    child: Column(
                                        children: <Widget>[
                                            Text("2.487",style:
TextStyle(fontSize: 18,fontWeight: FontWeight.w700),),
                                            Text("post")
                                        ],
                                    ),
                                ),
                                Container(
                                    padding: EdgeInsets.all(8),
                                    child: Column(
                                        children: <Widget>[
                                            Text("11.579",style:
TextStyle(fontSize: 18,fontWeight: FontWeight.w700),),
                                            Text("followers")
                                        ],
                                    ),
                                ),
                                Container(
                                    padding: EdgeInsets.all(8),
                                    child: Column(
                                        children: <Widget>[
                                            Text("634",style:
TextStyle(fontSize: 18,fontWeight: FontWeight.w700),),
                                            Text("following")
                                        ],
                                    ),
                                ),
                            ],
                        ),
                    )
                ],
            ),
        ),
    ),
)
```



```
),
Container(
  padding: EdgeInsets.all(8),
  child: SingleChildScrollView(
    scrollDirection: Axis.horizontal,
    child: Row(
      children: _list.map((e) => Container(
        padding: EdgeInsets.symmetric(horizontal: 16),
        child: Column(
          children: <Widget>[
            CircleAvatar(
              radius: 30,
            ),
            Padding(
              padding: const EdgeInsets.all(8.0),
              child: Text(e),
            )
          ],
        ),
      )),
    ),
  ),
),
Container(
  padding: EdgeInsets.symmetric(vertical: 16),
  child: Row(
    mainAxisAlignment: MainAxisAlignment.spaceEvenly,
    children: <Widget>[
      FlatButton(
        child: Icon(Icons.accessibility,size: 30,),
        onPressed: () {
          },
      ),
      FlatButton(
        child: Icon(Icons.account_box,size: 30,),
        onPressed: () {
          },
      ),
      FlatButton(
        child: Icon(Icons.account_circle,size: 30,),
        onPressed: () {
          },
      ),
    ],
  ),
),
```

```
        ),
        FlatButton(
            child: Icon(Icons.add_call, size: 30,),
            onPressed: () {
                ...
            },
        )
    ],
),
),
),
Expanded(
    child: Container(
        child: GridView.count(
            crossAxisCount: 3,
            crossAxisSpacing: 2,
            mainAxisSpacing: 2,
            children: List.generate(6, (index) => Container(
                color: Colors.blue,
            )),),
),
),
),
Container(
    child: Row(
        mainAxisAlignment: MainAxisAlignment.spaceEvenly,
        children: <Widget>[
            FlatButton(
                child: Icon(Icons.home),
                onPressed: () => print("home"),
            ),
            FlatButton(
                child: Icon(Icons.search),
                onPressed: () => print("home"),
            ),
            FlatButton(
                child: Icon(Icons.add_box),
                onPressed: () => print("home"),
            ),
            FlatButton(
                child: Icon(Icons.account_circle),
                onPressed: () => print("home"),
            ),
            FlatButton(
                child: Icon(Icons.people),
                onPressed: () => print("home"),
            ),
        ],
    ),
),
```

BOOMMM !

selamat kita sudah bisa membuat layout mirip instagram

tutorial diatas adalah tutorial paling sederhana mendekati termudah untuk dipahami dalam pembentukan layout, dan belum untuk penerapan secara functional , akan dibahas pada tutorial berikutnya , pada tutorial dasar ini saya berusaha sedetail mungkin dan sederhana mmungkin agar mudah diserap dan dipahaami ,dan belum pada tahapan produksi , saya harap inilah penjelasan yang mudah untuk diterima , untuk penerapan secara functional logic kita akan membedahnya di tutorial selanjutnya , jika ada pertanyaan , atau penjelasan saya kurang detail, jangan sungkan untuk menanyakannya langsung kepada saya , bisa melalui WA, atau email atau sosial media lainnya yang saya miliki , atau anda tinggal ketik kata kunci "malikkurosaki" pada pencarian google , disitu akan langung ditujukan pada akun media sosial saya , semua menggunakan nama yang sama yaitu "malik kurosaki"

Ada beberapa aplikasi yang telah saya buat diplaystore , anda bisa mengunjunginya langsung di alamat url : <https://play.google.com/store/apps/developer?id=malikkurosaki>

walau tkidak semua aplikasi yang telah saya buat menggunakan flutter , ada beberapa aplikasi yang masih menggunakan java , namun saya telah berkomitmen kedepan saya akan terusmenggunakan flutter , Karena dari segi efesiensi, dan kecepatan flutter sangatlain bisa diandalkan,

akhir kata dari saya , jangan mudah menyerah ,sabar dan nikmati prossernya , dan tetap fokus pada tujuan ,next lanjut ke buku kedua ,yang akan lebih membahas ke kodingan lebih dalam