

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. 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 compilasi 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 widget 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 memiliki BuildContext untuk mengatur posisi widget didalam widget tree detail mengenai widget dan bagaimana membuatnya akan dibahas pada bab selanjutnya.

1.3 Praktikum Installasi Flutter

1.3.1 Alat dan Bahan

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

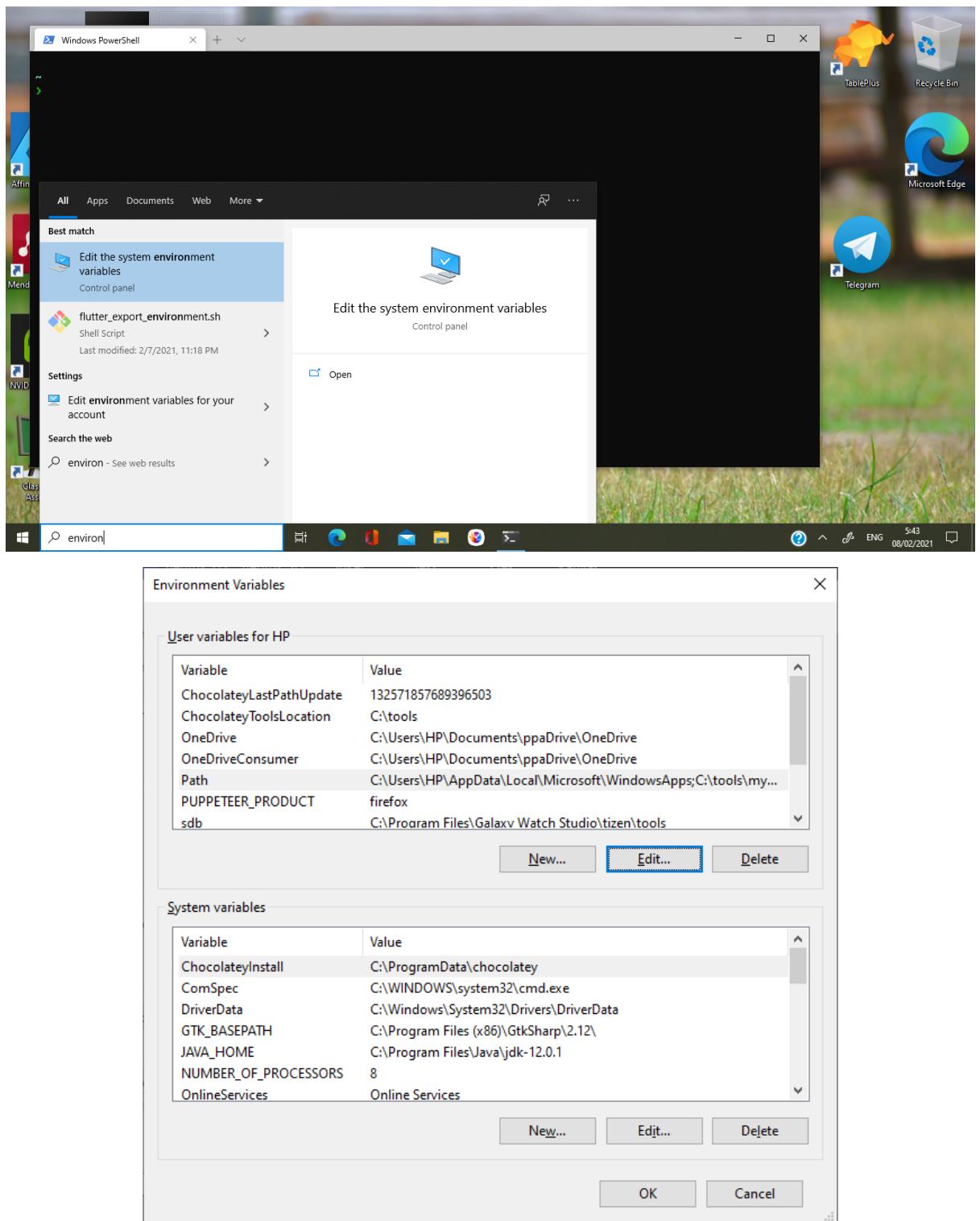
1. Sistem Operasi Windows minimal Windows 7 SP1 atau setelahnya 64 bit dan x86-64.
2. Harddisk 1,32 GB untuk Flutter saja selain IDE atau alat lain
3. Windows PowerShell 5.0 atau lebih baru <https://docs.microsoft.com/en-us/powershell/scripting/install/installing-windows-powershell>
4. Git For Windows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

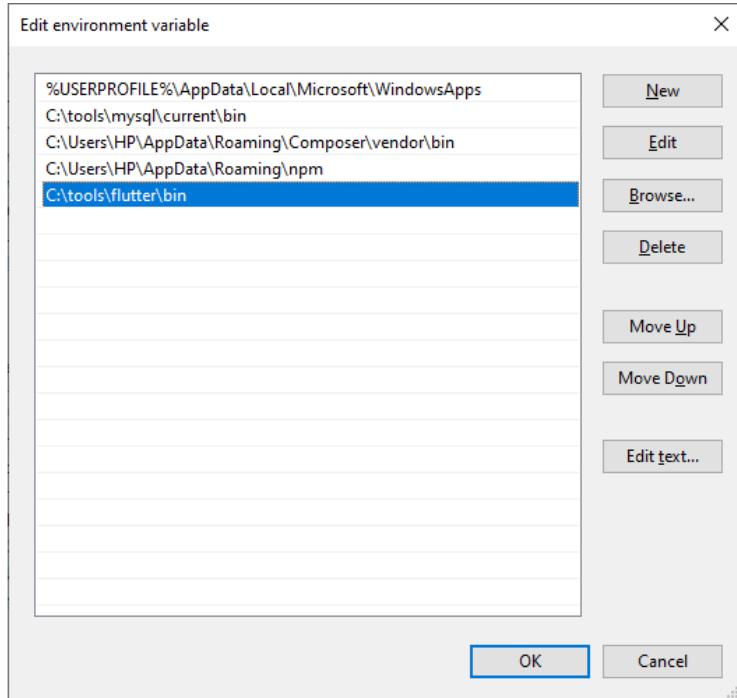
1.3.2 Langkah Langkah Praktikum

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
```

```
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)
  ✘ 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$SchemaModuleVersion.<init>(SchemaModule.java:156)
    at com.android.repository.api.SchemaModule.<init>(SchemaModule.java:75)
    at com.android.sdklib.repository.AndroidSdkHandler.<clinit>(AndroidSdkHandler.java:81)
    at com.android.sdklib.tool.sdkmanager.SdkManagerCli.main(SdkManagerCli.java:73)
    at com.android.sdklib.tool.sdkmanager.SdkManagerCli.main(SdkManagerCli.java:48)
Caused by: java.lang.ClassNotFoundException: javax.xml.bind.annotation.XmlSchema
    at java.base/jdk.internal.loader.BuiltinClassLoader.loadClass(BuiltinClassLoader.java:583)
    at java.base/jdk.internal.loader.ClassLoaders$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.

The screenshot shows the Android Studio interface. On the left, there is a sidebar listing recent projects:

- MvvmRecyclerView
- VideoViewModel
- DemoDataBinding
- MyGoldTracker
- RoomMvvmJava
- ClubRecyclerView
- EPL Club
- ClubEPL
- Quiz 2
- quiz2

The main area displays the Android Studio logo and the text "Android Studio Version 4.1.1". Below this are several options:

- + Create New Project
- Open an Existing Project
- Get from Version Control
- Profile or Debug APK
- Import Project (Gradle, Eclipse)
- Import an Android Code Sample

A context menu is open over the "Import an Android Code Sample" option, listing:

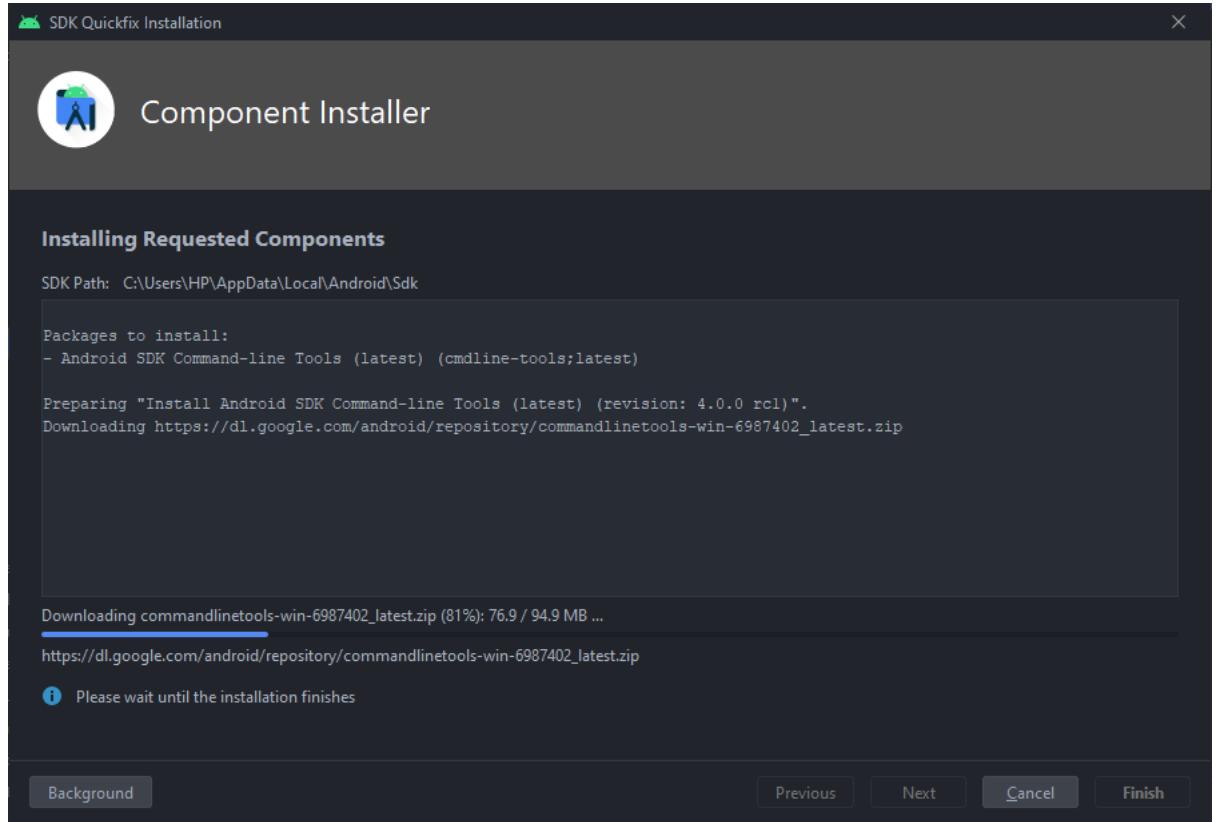
- SDK Manager
- AVD Manager
- Settings
- Plugins
- Default Project Structure...
- Run Configuration Templates...

Below the main area is the "Settings for New Projects" dialog, specifically the "Android SDK" tab under "System Settings". The "SDK Tools" tab is selected. The "SDK Platforms" tab is visible but not selected. The "SDK Update Sites" tab is also visible.

The "SDK Tools" tab shows a list of available developer tools:

Name	Version	Status
Android SDK Build-Tools	30.0.3	Update Available
LLDB		Installed
NDK (Side by side)	22.0.7026061	Update Available
Android SDK Command-line Tools (latest)		Not Installed
CMake	3.18.1	Update Available
Android Auto API Simulators	1	Not installed
Android Auto Desktop Head Unit Emulator	2.0.0 rc1	Not installed
Android Emulator	30.2.6	Update Available
Android Emulator Hypervisor Driver for AMD Processors (installer)	1.6.0	Not installed
Android SDK Platform-Tools	30.0.5	Installed
Android SDK Tools	26.1.1	Installed
Google Play APK Expansion library	1	Not installed
Google Play Instant Development SDK	1.9.0	Not installed
Google Play Licensing Library	1	Not installed
Google Play services	49	Not installed
Google USB Driver	13	Not installed
Google Web Driver	2	Not installed
Intel x86 Emulator Accelerator (HAXM installer)	7.5.6	Installed
Layout Inspector image server for API 29-30	6	Not installed

At the bottom of the dialog are buttons for "OK", "Cancel", and "Apply".



1.4 Praktikum Hello World

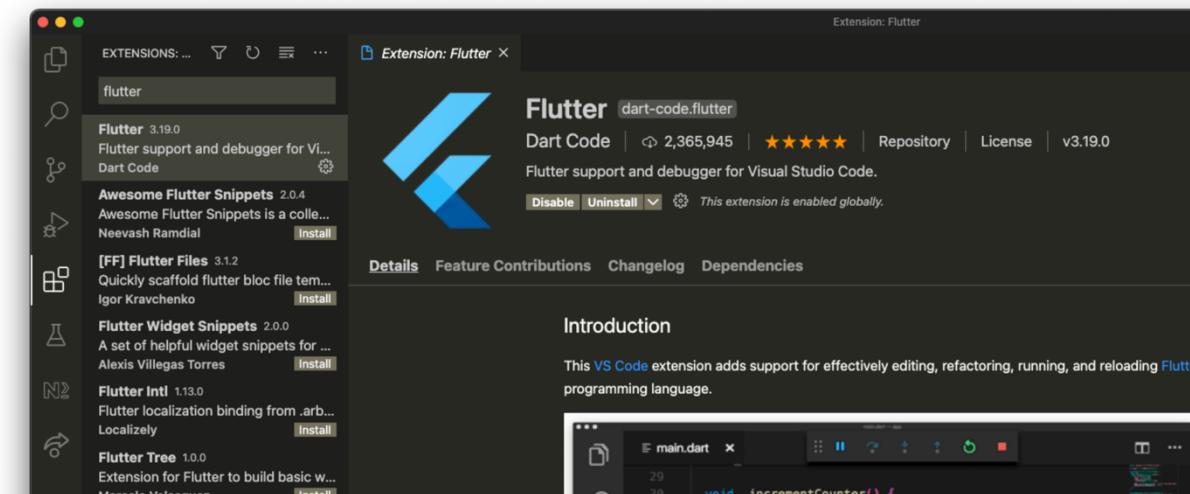
1.4.1 Alat dan Bahan

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

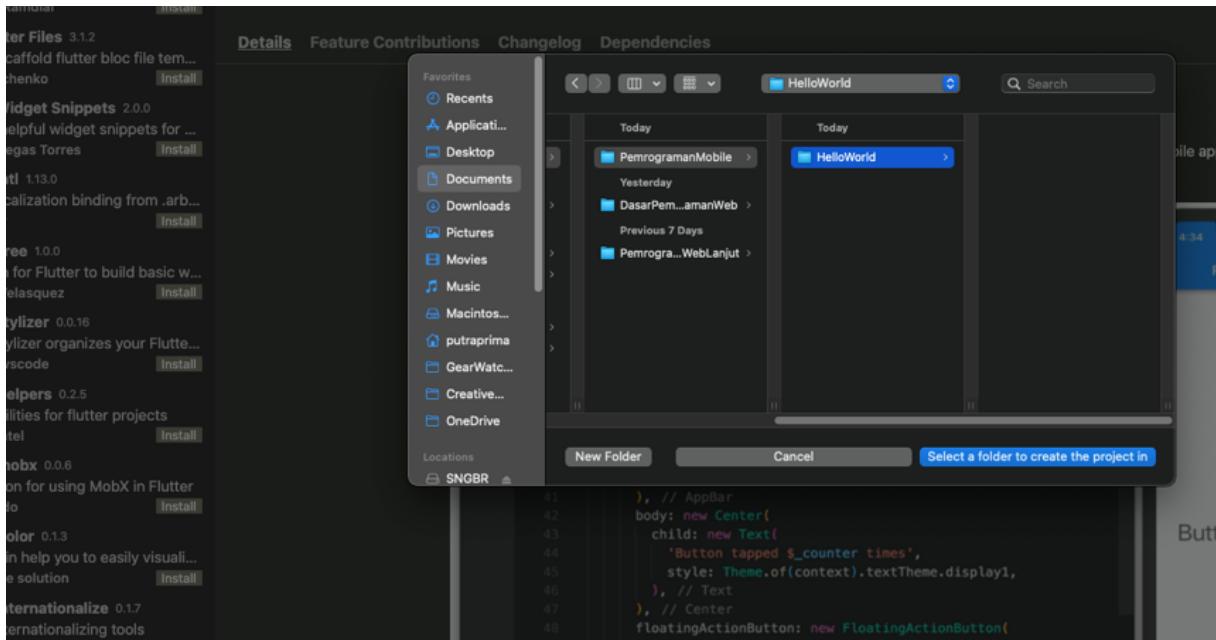
1. Install Flutter Plugin di Visual Studio Code



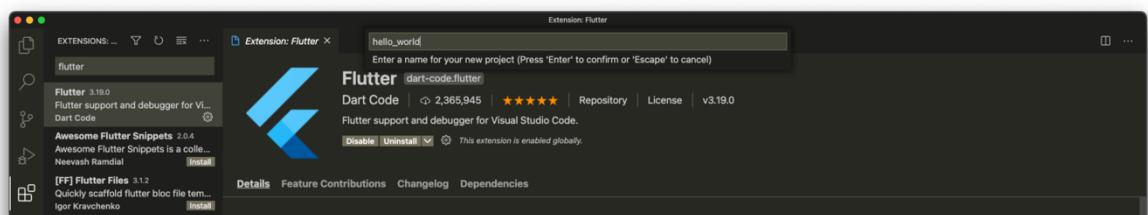
2. Tekan command Ctrl + Shift + P kemudian ketik perintah Flutter : New Application Project



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

```

main.dart — hello_world

lib > main.dart > MyApp > build
1 import 'package:flutter/material.dart';
2
3 void main() {
4   runApp(MyApp());
5 }
6
7 class MyApp extends StatelessWidget {
8   // This widget is the root of your application.
9   @override
10  Widget build(BuildContext context) {
11    return MaterialApp(
12      title: 'Flutter Demo',
13      theme: ThemeData(
14        // This is the theme of your application.
15        //
16        // Try running your application with "flutter run". You'll see the
17        // app has a blue toolbar. Then, without quitting the app, try
18        // changing the primarySwatch below to Colors.green and then invoke
19        // "hot reload" (press "r" in the console where you ran "flutter run",
20        // or simply save your changes to "hot reload" in a Flutter IDE).
21        // Notice that the counter didn't reset back to zero; the application
22        // is not restarted.
23        primarySwatch: Colors.blue,
24        // This makes the visual density adapt to the platform that you run
25        // the app on. For desktop platforms, the controls will be smaller and
26        // closer together (more dense) than on mobile platforms.
27        visualDensity: VisualDensity.adaptivePlatformDensity,
28      ), // ThemeData
29      home: MyHomePage(title: 'Flutter Demo Home Page'),
30    ); // MaterialApp
31  }
32}
33
34 class MyHomePage extends StatefulWidget {
35   MyHomePage({Key key, this.title}) : super(key: key);
36
37   // This widget is the home page of your application. It is stateful, meaning
38   // that it has a State object (defined below) that contains fields that affect
39   // how it looks.
40
41   // This class is the configuration for the state. It holds the values (in this
42   // case the title) provided by the parent (in this case the App widget) and

```

Ln 29, Col 56 Spaces: 2 UTF-8 LF Dart Flutter: 1.22.4 No Device

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.

Select a device to use

- Start iOS Simulator mobile simulator
- Start Pixel 2 API 24 mobile emulator
- Start Pixel_3a_API_30_x86 mobile emulator
- Create Android emulator

```

main.dart — hello_world

dart > MyApp
t 'package:fl
ebug
main() {
App(MyApp());

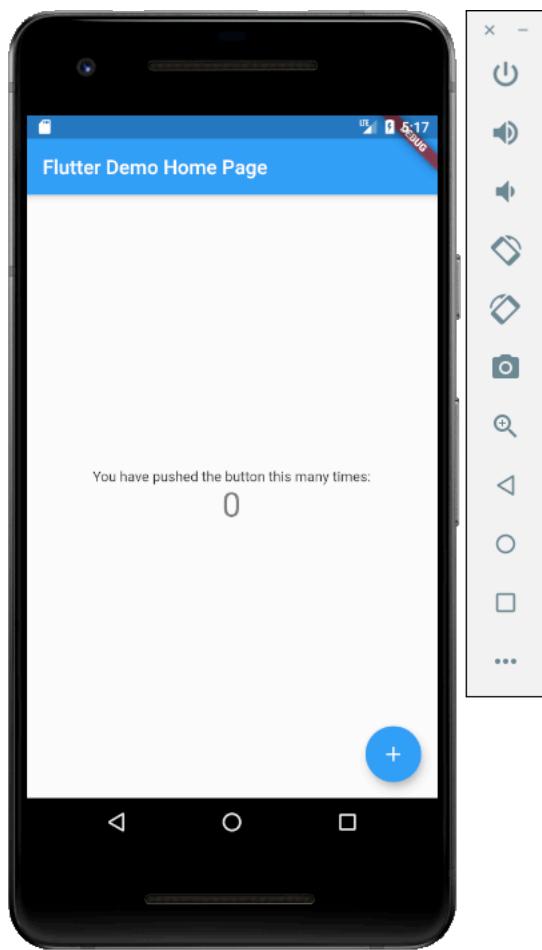
```

```

MyApp extends StatelessWidget {
This widget is the root of your application.
erride
get build(BuildContext context) {
return MaterialApp(

```

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

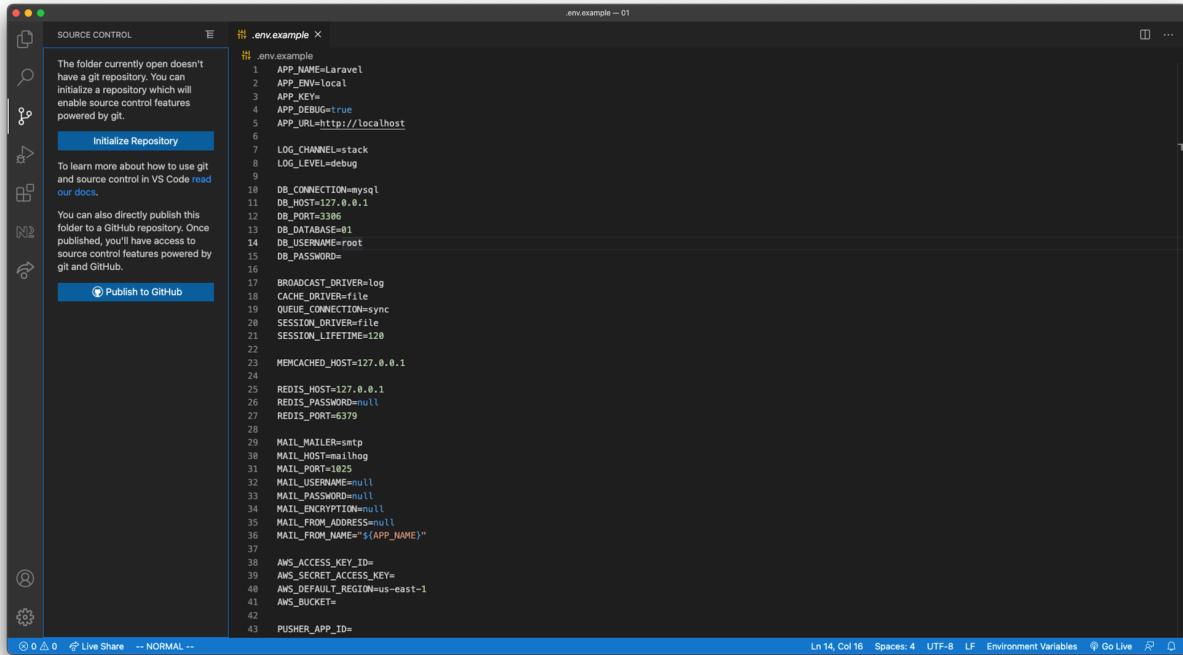
1.5.2 Langkah Langkah praktikum

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.

```
git config --global user.name = usernameAnda
```

```
git config --global user.email = emailAnda
```

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

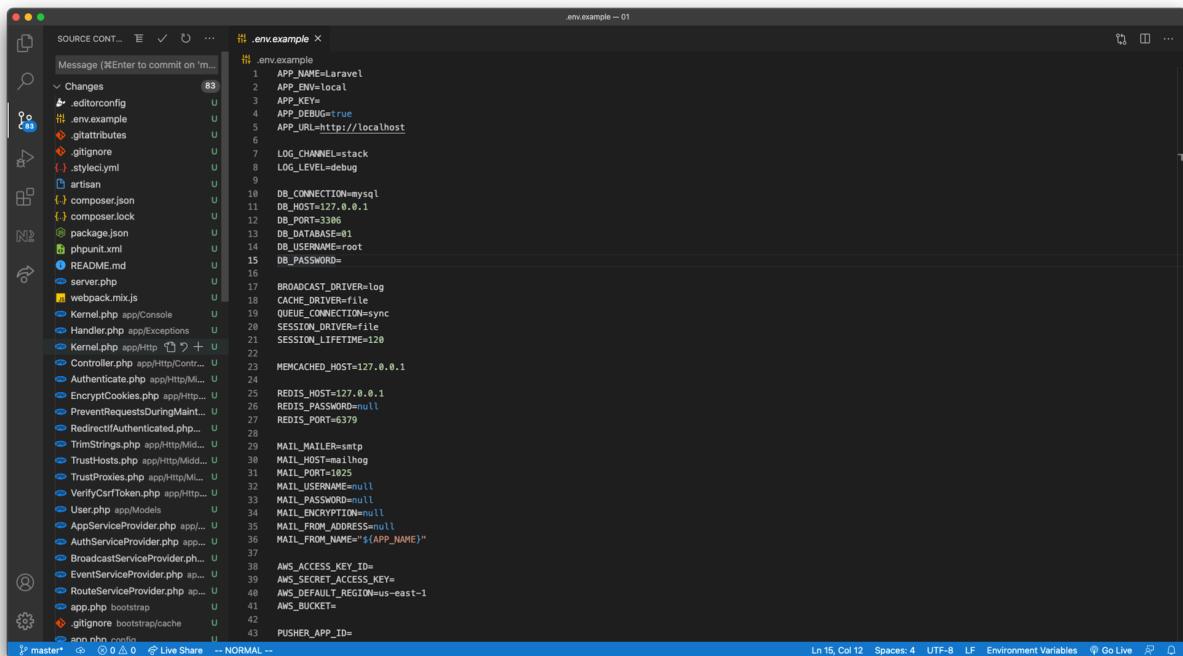


The screenshot shows the Visual Studio Code interface with the Source Control sidebar open. A message box is displayed, stating: "The folder currently open doesn't have a git repository. You can initialize a repository which will enable source control features powered by git." It includes two buttons: "Initialize Repository" and "Publish to GitHub". Below the message, there is sample .env.example configuration code. The status bar at the bottom shows "Live Share -- NORMAL --".

```
.env.example - 01
1 APP_NAME=Laravel
2 APP_ENV=local
3 APP_KEY=
4 APP_DEBUG=true
5 APP_URL=http://localhost
6
7 LOG_CHANNEL=stack
8 LOG_LEVEL=debug
9
10 DB_CONNECTION=mysql
11 DB_HOST=127.0.0.1
12 DB_PORT=3306
13 DB_DATABASE=laravel
14 DB_USERNAME=root
15 DB_PASSWORD=
16
17 BROADCAST_DRIVER=log
18 CACHE_DRIVER=file
19 QUEUE_CONNECTION=sync
20 SESSION_DRIVER=file
21 SESSION_LIFETIME=120
22
23 MEMCACHED_HOST=127.0.0.1
24
25 REDIS_HOST=127.0.0.1
26 REDIS_PASSWORD=null
27 REDIS_PORT=6379
28
29 MAIL_MAILER=smtp
30 MAIL_HOST=mailhog
31 MAIL_PORT=1025
32 MAIL_USERNAME=null
33 MAIL_PASSWORD=null
34 MAIL_ENCRYPTION=null
35 MAIL_FROM_ADDRESS=null
36 MAIL_FROM_NAME="${APP_NAME}"
37
38 AWS_ACCESS_KEY_ID=
39 AWS_SECRET_ACCESS_KEY=
40 AWS_DEFAULT_REGION=us-east-1
41 AWS_BUCKET=
42
43 PUSHER_APP_ID=
```

Ln 14, Col 16 Spaces: 4 UTF-8 LF Environment Variables Go Live

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

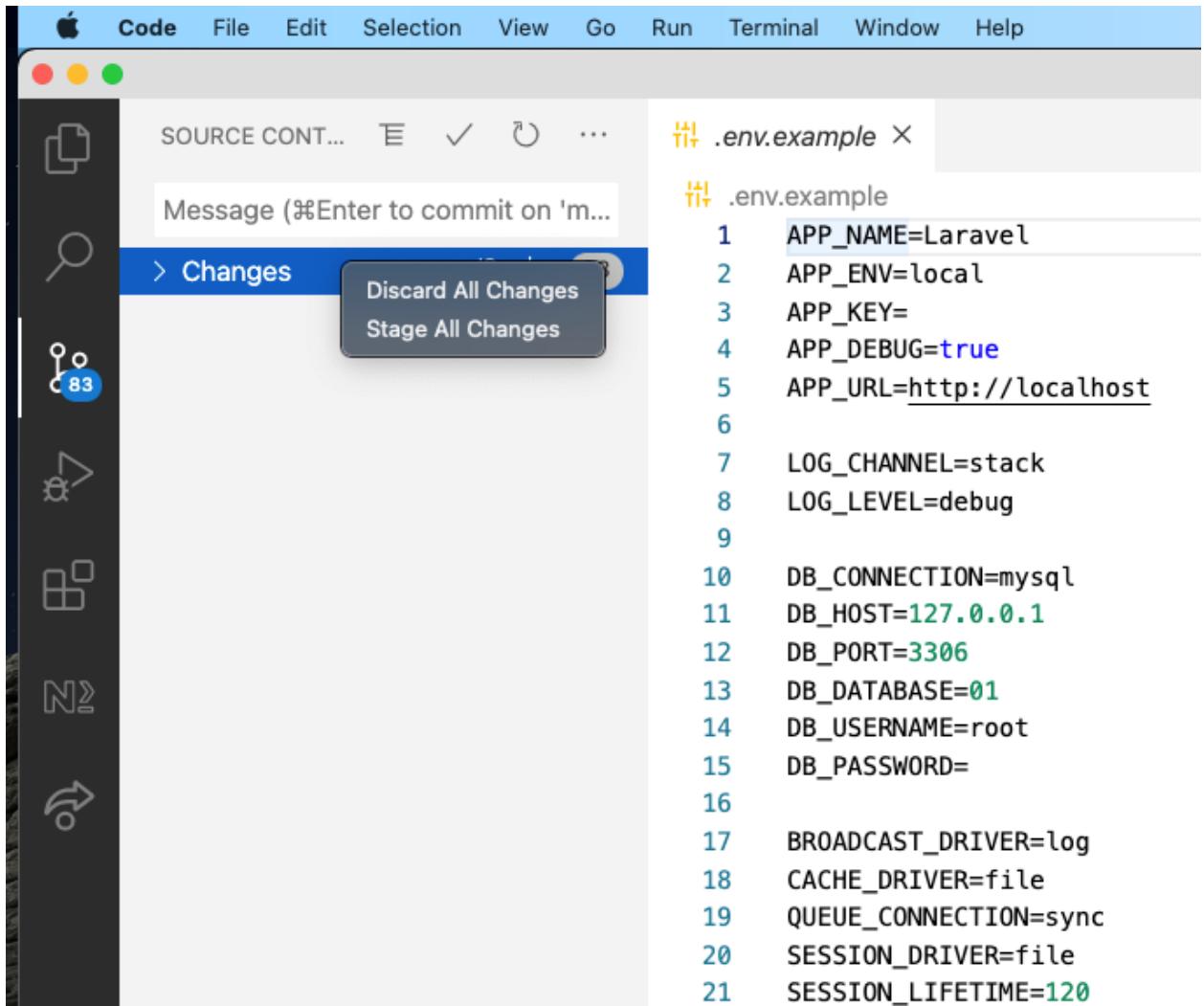


The screenshot shows the Visual Studio Code interface with the Source Control sidebar open. A commit message box is displayed: "Message (Enter to commit on 'm...')". Below it, a list of staged files is shown, including .env.example, .editorconfig, .gitattributes, .gitignore, .styleci.yml, artisan, composer.json, composer.lock, package.json, phpunit.xml, README.md, server.php, webpack.mix.js, Kernel.php, Handler.php, Controller.php, Authenticate.php, EncryptCookies.php, PreventRequestsDuringMaintenance.php, RedirectAuthenticated.php, TrimStrings.php, TrustHosts.php, TrustProxies.php, VerifyCsrfToken.php, User.php, AppServiceProvider.php, AuthServiceProvider.php, BroadcastServiceProvider.php, EventServiceProvider.php, RouteServiceProvider.php, app.php, bootstrap, and .gitignore. The status bar at the bottom shows "master* Live Share -- NORMAL --".

```
.env.example - 01
1 APP_NAME=Laravel
2 APP_ENV=local
3 APP_KEY=
4 APP_DEBUG=true
5 APP_URL=http://localhost
6
7 LOG_CHANNEL=stack
8 LOG_LEVEL=debug
9
10 DB_CONNECTION=mysql
11 DB_HOST=127.0.0.1
12 DB_PORT=3306
13 DB_DATABASE=laravel
14 DB_USERNAME=root
15 DB_PASSWORD=
16
17 BROADCAST_DRIVER=log
18 CACHE_DRIVER=file
19 QUEUE_CONNECTION=sync
20 SESSION_DRIVER=file
21 SESSION_LIFETIME=120
22
23 MEMCACHED_HOST=127.0.0.1
24
25 REDIS_HOST=127.0.0.1
26 REDIS_PASSWORD=null
27 REDIS_PORT=6379
28
29 MAIL_MAILER=smtp
30 MAIL_HOST=mailhog
31 MAIL_PORT=1025
32 MAIL_USERNAME=null
33 MAIL_PASSWORD=null
34 MAIL_ENCRYPTION=null
35 MAIL_FROM_ADDRESS=null
36 MAIL_FROM_NAME="${APP_NAME}"
37
38 AWS_ACCESS_KEY_ID=
39 AWS_SECRET_ACCESS_KEY=
40 AWS_DEFAULT_REGION=us-east-1
41 AWS_BUCKET=
42
43 PUSHER_APP_ID=
```

Ln 15, Col 12 Spaces: 4 UTF-8 LF Environment Variables Go Live

6. 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.
7. Klik dropdown changes ini sehingga daftar file yang berubah diminimize, kemudian klik kanan dan klik menu “Stage All Changes”



The screenshot shows the Visual Studio Code interface. The top bar includes the Apple logo, 'Code' (selected), File, Edit, Selection, View, Go, Run, Terminal, Window, and Help. The left sidebar has icons for Source Control, Search, Problems (with 83 items), and others. The main area shows a file named '.env.example' with the following content:

```
APP_NAME=Laravel
APP_ENV=local
APP_KEY=
APP_DEBUG=true
APP_URL=http://localhost
LOG_CHANNEL=stack
LOG_LEVEL=debug
DB_CONNECTION=mysql
DB_HOST=127.0.0.1
DB_PORT=3306
DB_DATABASE=01
DB_USERNAME=root
DB_PASSWORD=
BROADCAST_DRIVER=log
CACHE_DRIVER=file
QUEUE_CONNECTION=sync
SESSION_DRIVER=file
SESSION_LIFETIME=120
```

A context menu is open over the file list, with 'Changes' selected. The menu options 'Discard All Changes' and 'Stage All Changes' are visible. The 'Stage All Changes' option is highlighted with a blue background.

8. 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.

The screenshot shows a Mac OS X desktop environment. At the top is a blue menu bar with the Apple logo and the following menu items: Code, File, Edit, Selection, View, Go, Run, Terminal, Window, Help. Below the menu bar is a dark-themed terminal window with three colored status bars (red, yellow, green) at the top. The terminal window has a light gray background and contains the command 'git commit' followed by a message placeholder 'Message (⌘Enter to commit on 'm...')'. To the right of the terminal is a code editor window titled '.env.example'. The code editor has a light gray background and displays the following configuration file:

```
APP_NAME=Laravel
APP_ENV=local
APP_KEY=
APP_DEBUG=true
APP_URL=http://localhost
LOG_CHANNEL=stack
LOG_LEVEL=debug
DB_CONNECTION=mysql
DB_HOST=127.0.0.1
DB_PORT=3306
DB_DATABASE=01
DB_USERNAME=root
DB_PASSWORD=
BROADCAST_DRIVER=log
CACHE_DRIVER=file
QUEUE_CONNECTION=sync
SESSION_DRIVER=file
SESSION_LIFETIME=120
```

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

The screenshot shows the GitHub Codespace interface. On the left is a sidebar with various icons: a file, a magnifying glass, a gear (with a '83' notification), a play button, a square, and a refresh arrow. The main area has a header bar with 'Code' and other menu items like File, Edit, Selection, View, Go, Run, Terminal, Window, and Help. Below the header is a toolbar with icons for Source Control, Diff, Checkmark, Refresh, and More. A central text input field says 'Kode Program Pertama'. To the right is a terminal window titled '.env.example' containing the following configuration file:

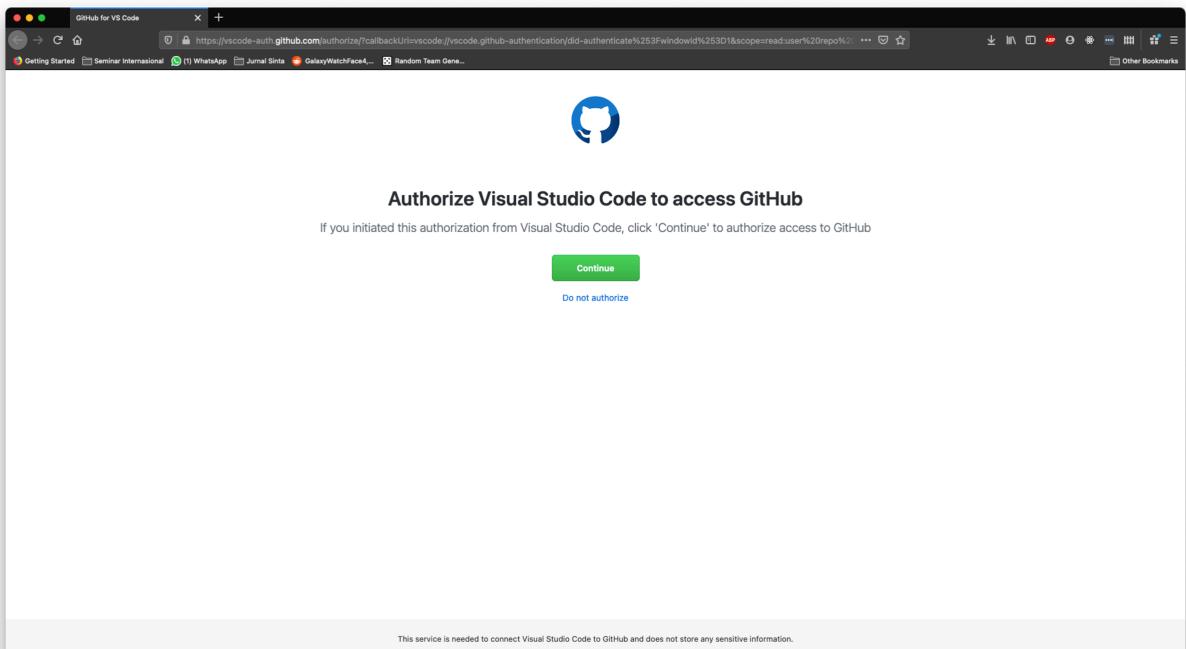
```
1 APP_NAME=Laravel
2 APP_ENV=local
3 APP_KEY=
4 APP_DEBUG=true
5 APP_URL=http://localhost
6
7 LOG_CHANNEL=stack
8 LOG_LEVEL=debug
9
10 DB_CONNECTION=mysql
11 DB_HOST=127.0.0.1
12 DB_PORT=3306
13 DB_DATABASE=01
14 DB_USERNAME=root
15 DB_PASSWORD=
16
17 BROADCAST_DRIVER=log
18 CACHE_DRIVER=file
19 QUEUE_CONNECTION=sync
20 SESSION_DRIVER=file
21 SESSION_LIFETIME=120
22
```

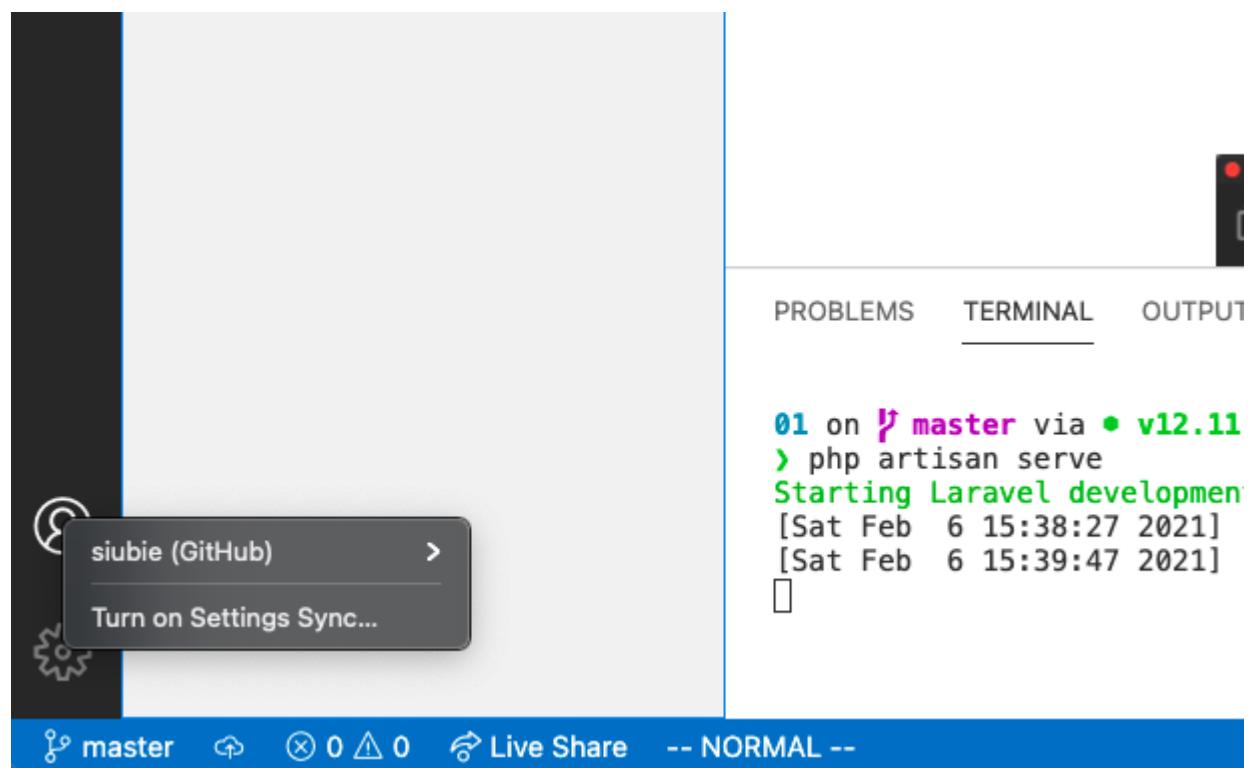
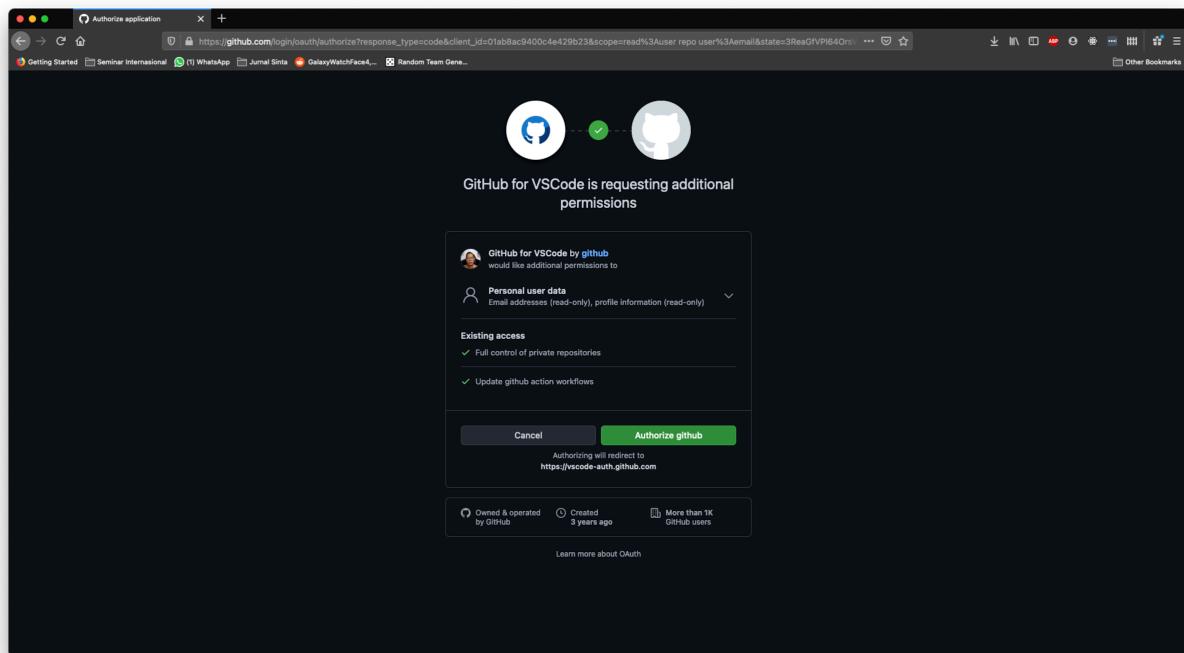
10. Jika di klik menu “Staged Change” akan hilang dan semua perubahan sudah di catat oleh git.
11. Selamat anda sudah berhasil menggunakan git pada repository lokal.
12. Selanjutnya untuk melakukan publikasi repository lokal ke github lakukan perintah berikut ini tekan Ctrl + Shift + P kemudian ketik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

welcome.blade.php — 01

```
ip X >| recently used ☰  
↳ welcor Publish to GitHub  
html> Git: Push  
={{ str_ Git: Add Remote...  
eta char Preferences: Color Theme  
eta name= Unfold All  
Fold All  
itle>Lara XML Tools: Format as XML  
-- Fonts Format Document  
ink href= Add gitignore  
Fold Recursively  
-- Styles Expand Selection  
tyle> Collapse Folders in Explorer  
/*! no WakaTime: Api Key  
style> Copy With Syntax Highlighting  
tyle>  
body {  
    font-family: 'Nunito';  
}  
=sty ht:1.:  
ht:1.:
```

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





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

```
        welcome.blade.php — 01

laravel-hello-world

elcor [ ] Publish to GitHub private repository siubie/laravel-hello-world
[ ] Publish to GitHub public repository siubie/laravel-hello-world
str_----- _ , , , _ppp.. g-----., . .

charset="utf-8">
ame="viewport" content="width=device-width, initial-scale=1">

laravel</title>
```

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 ubahlah tampilan aplikasi menjadi seperti gambar berikut dan publish ke repository github dengan nama repository polinema_mobile_flutter_hello.

