



MODUL

MOBILE Programming



Disusun Oleh:
UNIT PENGEMBANG AKADEMIK

UNIVERSITAS BINA SARANA INFORMATIKA
FAKULTAS TEKNIK DAN INFORMATIKA
PROGRAM STUDI ILMU KOMPUTER

2021

KATA PENGANTAR

Puja dan juga puji syukur selalu kami panjatkan kehadirat Allah Swt yang telah memberikan semua nikmatnya sehingga penulis berhasil menyelesaikan modul ajar yang berjudul Mobile Programming ini tanpa adanya kendala yang berarti. Tujuan dari penyusunan modul ini adalah untuk memudahkan para mahasiswa Universitas Bina Sarana Informatika dalam memahami bagaimana teori dan sedikit praktik mengenai Mobile Programming yang kesannya cukup rumit sehingga menjadi lebih mudah.

Keberhasilan penyusunan modul ini tentunya bukan atas usaha penulis saja namun ada banyak pihak yang turut membantu dan memberikan dukungan untuk suksesnya penulisan buku ini. Untuk itu, penulis mengucapkan terima kasih yang sebesar-besarnya kepada semua pihak yang telah memberikan dukungan baik secara moril ataupun material sehingga buku ini berhasil disusun.

Modul yang ada ini tentu tidak luput dari kekurangan. Selalu ada celah untuk perbaikan. Kritik, saran serta masukkan dari pembaca sangat kami harapkan untuk penyempurnaan kedepannya.

Pontianak, Maret 2021

Penulis

DAFTAR ISI

KATA PENGANTAR	i
DAFTAR PUSTAKA	ii
MINIMUM SYSTEM REQUIREMENTS	iv
PENGARAHAN TUGAS BESAR	iv
PERTEMUAN 1	1
A. API SPEC.....	1
1. Registrasi.....	1
2. Login.....	1
3. Produk.....	2
a) List Produk.....	2
b) Create Produk.....	3
c) Update Produk	3
d) Show Produk.....	4
e) Delete Produk.....	4
B. Installasi Apache, MySql dan PHP (XAMPP)	5
C. Install Composer	10
D. Install Postman	12
E. Pembuatan Database	13
1. SQL membuat tabel member	13
2. SQL membuat tabel member_token	14
3. SQL membuat tabel produk	14
F. Installasi Lumen sebagai Restful API.....	15
PERTEMUAN 2	16
G. Konfigurasi Projek	16
1. Koneksi ke database.....	16
2. Membuat hasil response.....	17
H. Membuat Toko API	18
1. Registrasi.....	18
a) Membuat model Registrasi	18
b) Membuat controller Registrasi.....	19
c) Menambah route Registrasi	20
2. Login.....	22
a) Membuat model Member.....	22
b) Membuat model Login.....	22
c) Membuat controller Login	23
d) Menambahkan route Login	25
e) Mencoba Rest.....	25
PERTEMUAN 3	26
3. CRUD Produk.....	26
a) Membuat model Produk.....	26
b) Membuat controller produk	26
c) Mencoba Rest.....	31
PERTEMUAN 4	34
I. Pembuatan Aplikasi Mobile dengan Flutter	34

1.	Menyiapkan perangkat.....	34
a)	Install Git	34
b)	Install Android Studio.....	34
c)	Install Flutter.....	36
d)	Konfigurasi Android Studio dengan Flutter	39
2.	Membuat dan menjalankan projek.....	40
3.	Struktur Folder Flutter	44
	PERTEMUAN 5	46
4.	Membuat Halaman.....	46
a)	Registrasi.....	47
b)	Login	50
c)	Form Produk	53
d)	Detail Produk	55
e)	Tampil List Produk	57
	PERTEMUAN 6	61
5.	Membuat Model.....	61
a)	Login	61
b)	Registrasi.....	62
c)	Produk	62
	PERTEMUAN 7	63
6.	Membuat Helper Modul.....	63
a)	Menambahkan depedencies	63
b)	Membuat Class Token	64
c)	Http request.....	65
	PERTEMUAN 8	68
	PERTEMUAN 9	69
7.	Membuat Bloc.....	69
a)	Registrasi.....	69
b)	Login	70
c)	Logout.....	70
d)	Produk	70
	PERTEMUAN 10 - 12	72
8.	Menyatukan Fungsionalitas	72
a)	Membuat Common Dialog Widget.....	72
b)	Modifikasi main.dart.....	75
c)	Modifikasi registrasi_page.dart.....	76
d)	Modifikasi login_page.dart (fungsi login)	80
e)	Modifikasi produk_page.dart	84
f)	Memodifikasi Form Produk (produk_form.dart)	89
	PERTEMUAN 13-15	100
	PERTEMUAN 16	101

MINIMUM SYSTEM REQUIREMENTS

1. CPU: Core i5
2. CPU SPEED: 2.0 GHz
3. RAM: 8 GB
4. OS: Windows 7 64-bit SP1
5. FREE DISK SPACE: 350 MB

PENGARAHAN TUGAS BESAR

Penjelasan Kontrak Kuliah dan Penegasan Tugas Project

1. Tugas Projek (Kelompok)

Tugas project diadakan untuk memperoleh nilai UTS dan UAS, dengan kata lain tugas projek ini sebagai pengganti UTS dan UAS. Tugas ini dikerjakan secara kelompok dengan maksimal 1 kelompok sebanyak 5 mahasiswa atau lebih disesuaikan dengan jumlah mahasiswa dalam satu kelas.

2. Pembagian Kelompok

Untuk penbagian kelompok ditentukan oleh dosen pengampu matakuliah, disesuaikan dengan kelompok tugas pada matakuliah yang lain. Seperti matakuliah Web Programming III, APSI, IMK, dan MPSI

3. Bentuk Tugas (Kelompok)

Bentuk Tugas Project :

- a. Project merupakan program aplikasi mobile dengan tema project harus sama dengan tema yang diambil pada mata kuliah Web Programming III
- b. Setelah pertemuan ke 11, masing-masing kelompok wajib demo program yang telah dikerjakan (Presentasi)

PERTEMUAN 1

A. API SPEC

API SPEC ini bermaksud untuk membuat standar API sebagai dokumentasi kepada pengembang baik itu frontend maupun backend

1. Registrasi

EndPoint	/registrasi
Method	POST
Header	<ul style="list-style-type: none">Content-Type: application/json
Body	{ "nama" : "string", "email" : "string, unique", "password" : "string" }
Response	{ "code" : "integer", "status" : "boolean", "data" : "string" }

2. Login

EndPoint	/login
Method	POST
Header	<ul style="list-style-type: none">Content-Type: application/json
Body	{ "email" : "string", "password" : "string" }
Response	{ "code" : "integer", "status" : "boolean", "data" : "string" }

```

    "data"  : {
        "token" : "string",
        "user"  : {
            "id"   : "integer",
            "email": "string",
        }
    }
}

```

3. Produk

a) List Produk

EndPoint	/produk
Method	GET
Header	<ul style="list-style-type: none"> Content-Type: application/json
Response	<pre> { "code" : "integer", "status" : "boolean", "data" : [{ "id" : "integer", "kode_produk" : "string", "nama_produk" : "string", "harga" : "integer", }, { "id" : "integer", "kode_produk" : "string", "nama_produk" : "string", "harga" : "integer", }] } </pre>

b) Create Produk

EndPoint	/produk
Method	POST
Header	<ul style="list-style-type: none"> Content-Type: application/json
Body	<pre>{ "kode_produk" : "string", "nama_produk" : "string", "harga" : "integer" }</pre>
Response	<pre>{ "code" : "integer", "status" : "boolean", "data" : { "id" : "integer", "kode_produk" : "string", "nama_produk" : "string", "harga" : "integer", } }</pre>

c) Update Produk

EndPoint	/produk/{id}/update
Method	POST
Header	<ul style="list-style-type: none"> Content-Type: application/json
Body	<pre>{ "kode_produk" : "string", "nama_produk" : "string", "harga" : "integer" }</pre>
Response	<pre>{ "code" : "integer", "status" : "boolean", }</pre>

	<pre> "data" : "boolean" } </pre>
--	---------------------------------------

d) Show Produk

EndPoint	/produk/{id}
Method	GET
Header	<ul style="list-style-type: none"> Content-Type: application/json
Response	<pre> { "code" : "integer", "status" : "boolean", "data" : { "id" : "integer", "kode_produk" : "string", "nama_produk" : "string", "harga" : "integer", } } </pre>

e) Delete Produk

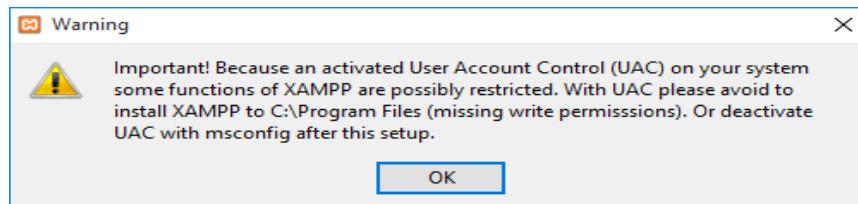
EndPoint	/produk/{id}
Method	DELETE
Header	<ul style="list-style-type: none"> Content-Type: application/json
Response	<pre> { "code" : "integer", "status" : "boolean", "data" : "boolean" } </pre>

B. Installasi Apache, MySql dan PHP (XAMPP)

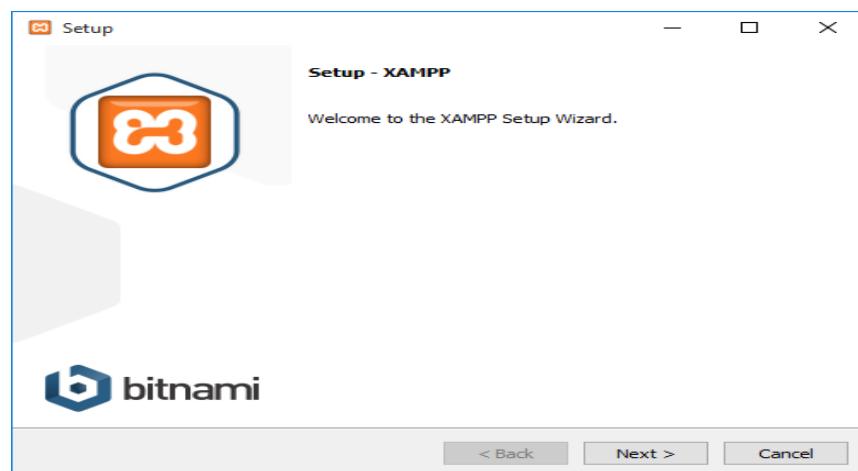
XAMPP adalah perangkat lunak untuk membuat komputer menjadi web server. XAMPP dapat di download melalui link url:

<https://www.apachefriends.org/download.html>.

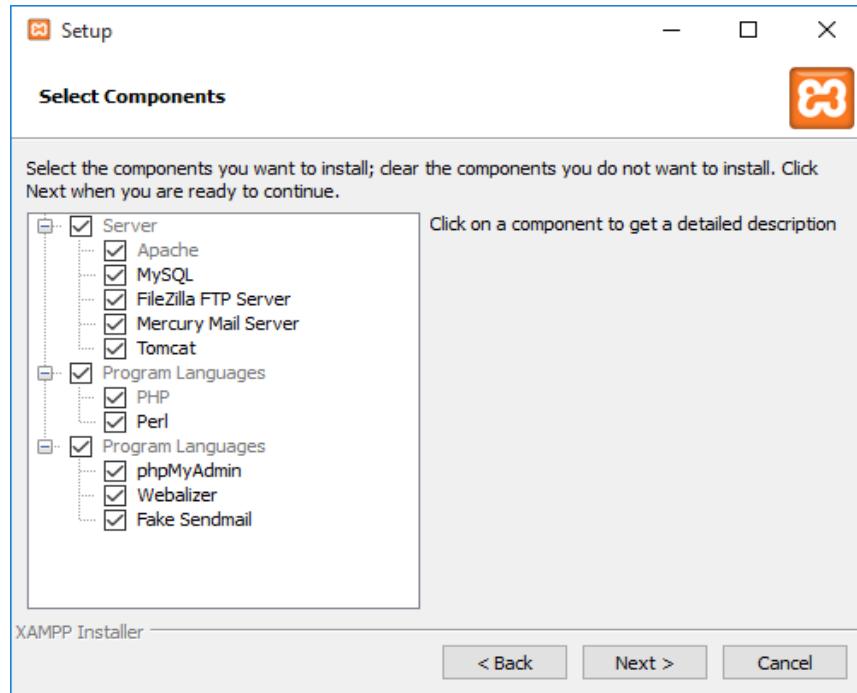
Setelah itu, double klik file xampp yang baru di download. Jika muncul pesan error seperti gambar dibawah, abaikan saja dan lanjutkan klik tombol OK.



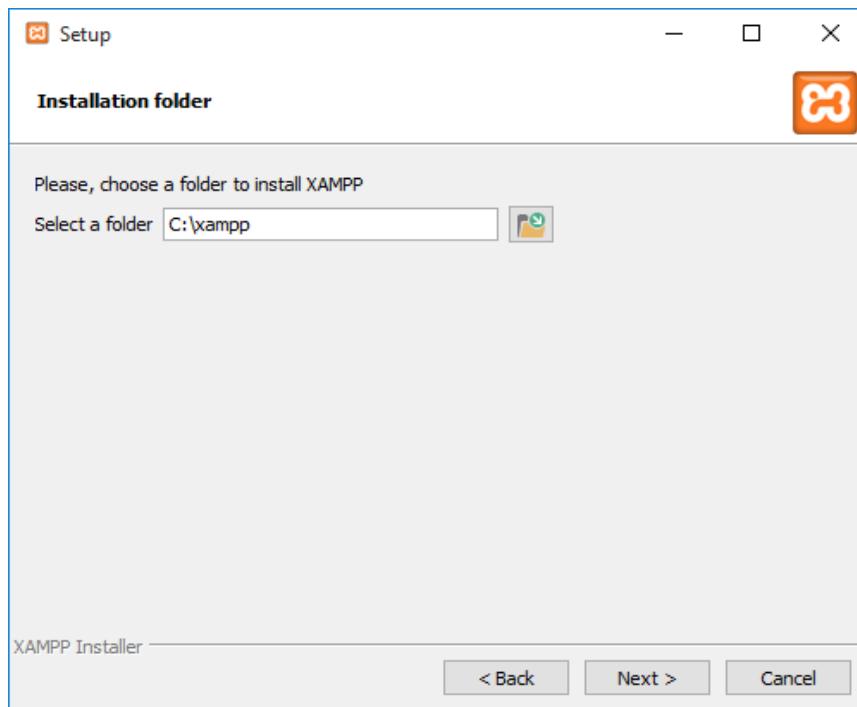
Berikutnya akan muncul jendela **Setup-XAMPP**. Klik tombol **Next** untuk melanjutkan proses berikutnya.



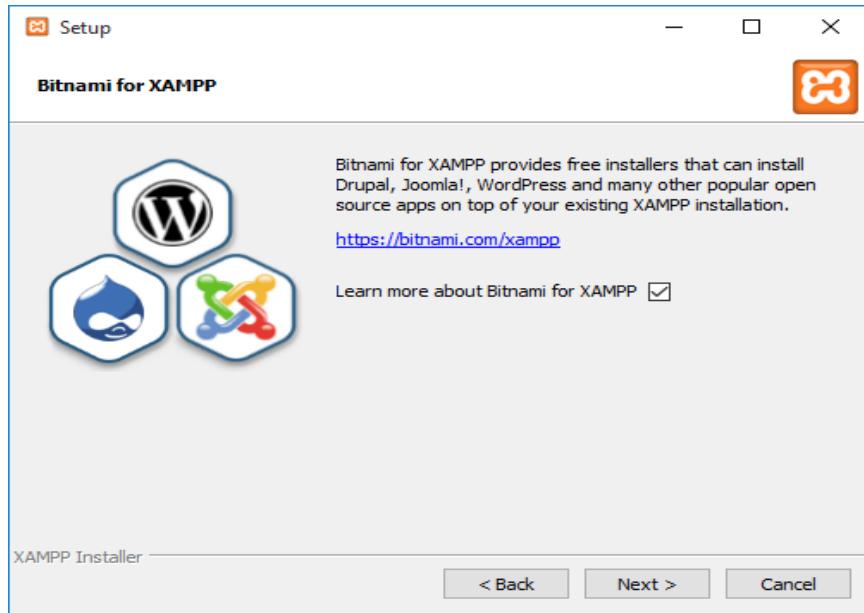
Selanjutnya akan muncul jendela **Select Components**, yang meminta untuk memilih aplikasi yang akan diinstall. Centang saja semua kemudian klik tombol **Next**.



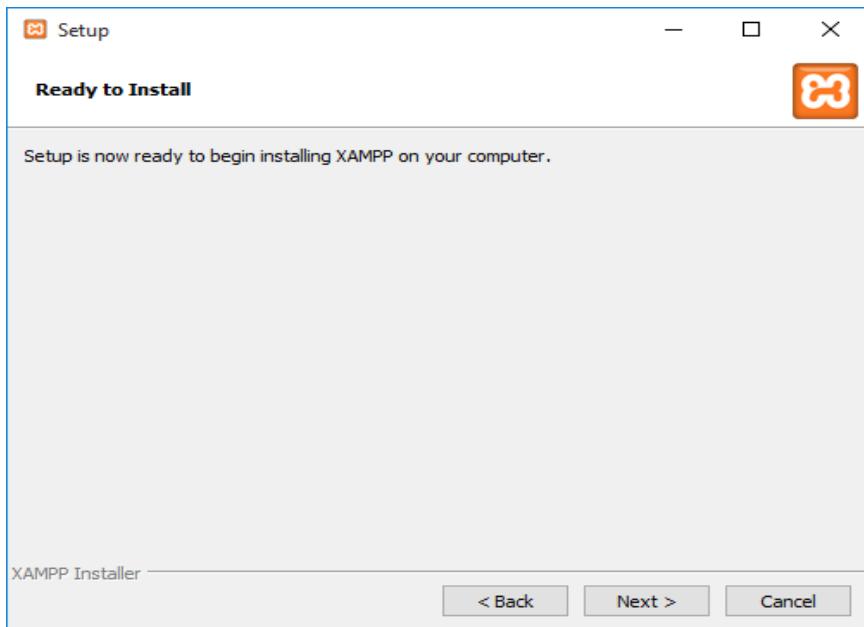
Kemudian akan muncul jendela **Installation Folder**, dimana anda diminta untuk menentukan lokasi penyimpanan folder xampp, secara bawaan akan diarahkan ke lokasi **c:\xampp**. Jika anda ingin menyimpannya di folder lain, anda dapat menekan tombol bergambar folder (**Browse**), kemudian klik tombol **Next**.



Kita akan menjumpai jendela tawaran untuk mempelajari lebih lanjut tentang Bitnami. Silahkan checklist jika ingin mempelajari lebih lanjut. Bitnami adalah pustaka dari aplikasi client server yang populer seperti misalnya CMS WordPress atau Drupal, dengan penawaran kemudahan dalam installasi hanya dengan satu klik. Kemudian klik tombol **Next**.



Kemudian muncul jendela **Ready To Install** yang menunjukkan xampp sudah siap di install. Kemudian klik tombol **Next**.



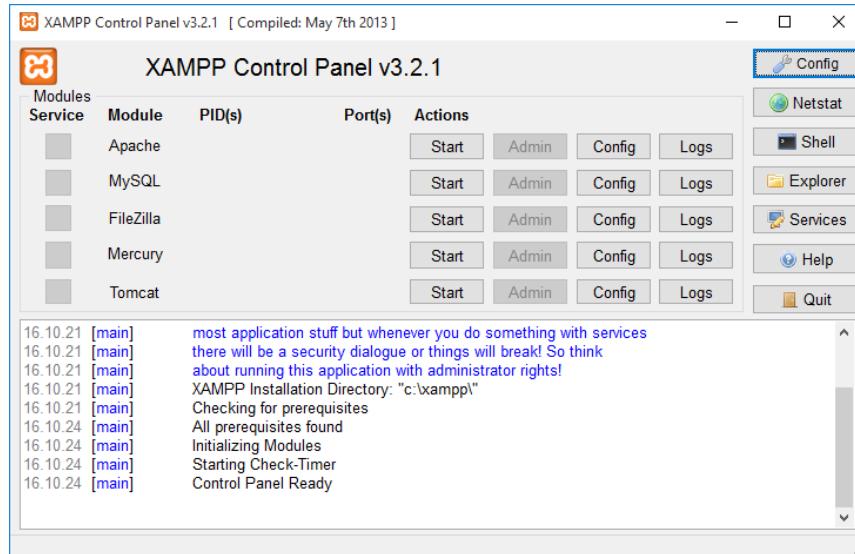
Dan proses installasi pun berjalan.



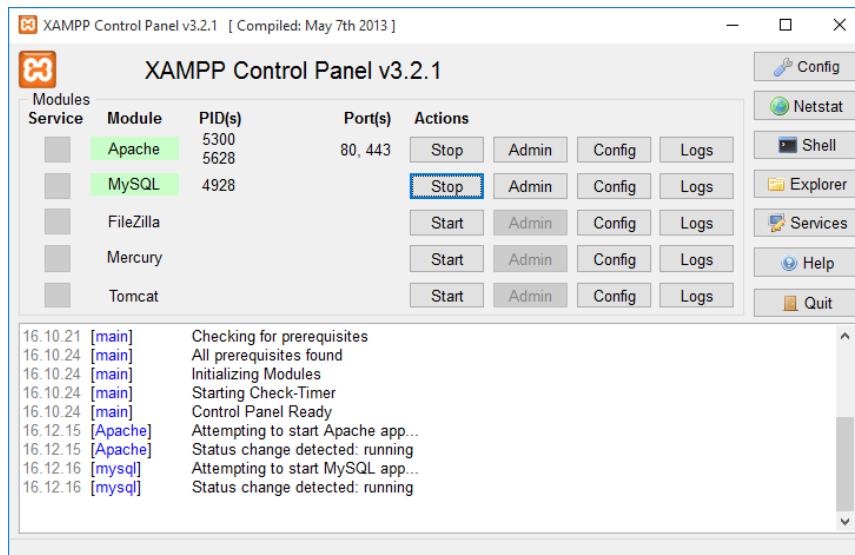
Tunggu hingga proses install selesai dan muncul jendela sebagai berikut.



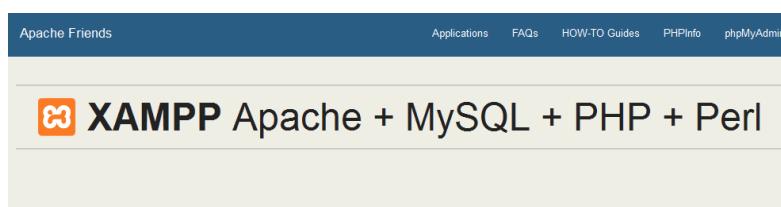
Klik tombol **Finish** setelah itu akan muncul jendela **Xampp Control Panel** yang berguna untuk menjalankan server.



Klik tombol **Start** pada kolom **Actions** untuk module **Apache** dan **MySQL**.



Untuk mengetahui apakah installasi telah berhasil atau tidak, ketikkan ‘localhost/’ pada browser, jika berhasil akan muncul halaman seperti gambar dibawah.



Welcome to XAMPP for Windows 5.6.12

You have successfully installed XAMPP on this system! Now you can start using Apache, MySQL, PHP and other components. You can find more info in the FAQs section or check the HOW-TO Guides for getting started with PHP applications.

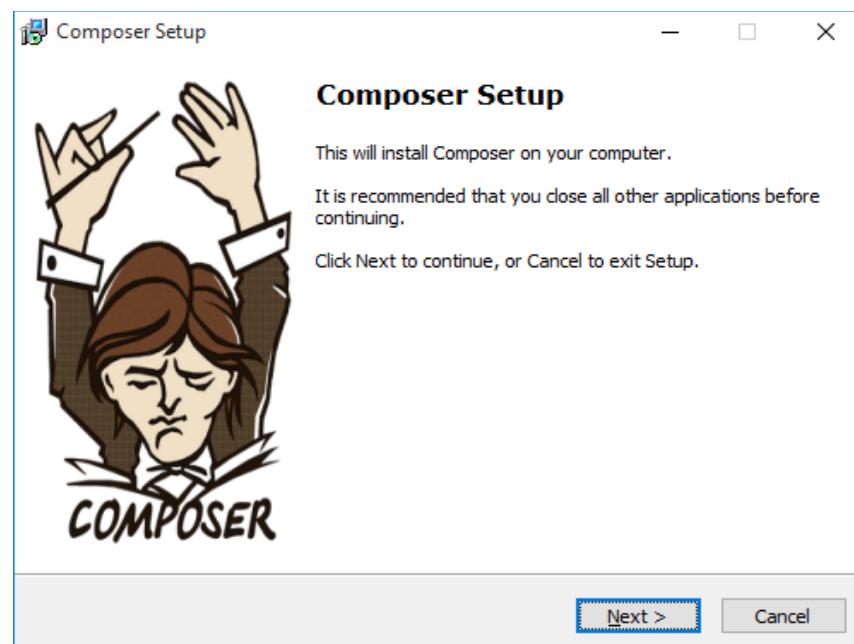
Start the XAMPP Control Panel to check the server status.

C. Install Composer

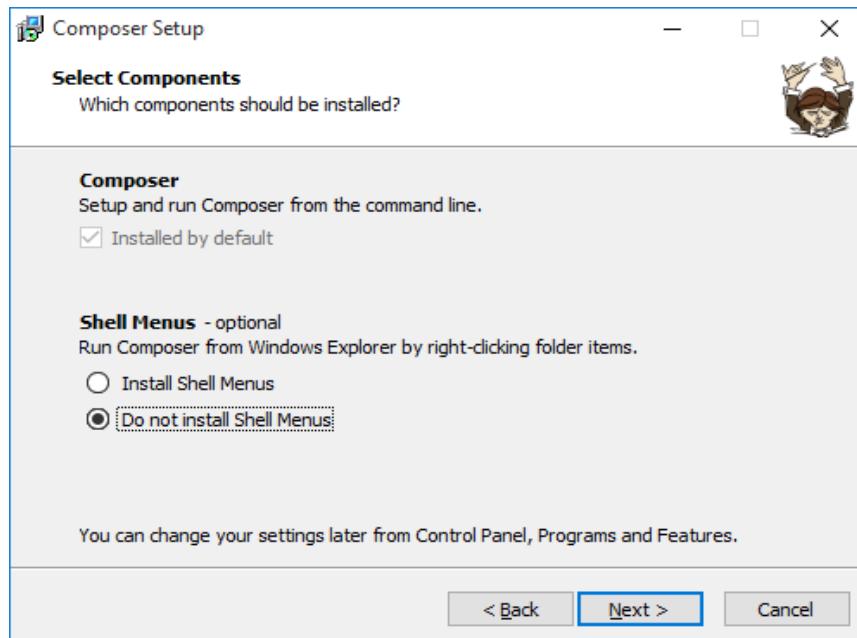
Composer adalah Dependency Management Tools untuk PHP. Sederhananya Composer akan membantu kita mencari dan mendownload file-file yang diperlukan oleh sebuah aplikasi yang akan install yang akan tersimpan pada folder **vendor**. Composer untuk windows dapat didownload di <https://getcomposer.org/Composer-Setup.exe>. Install composer sangat mudah, cukup double klik file composer yang telah di download.



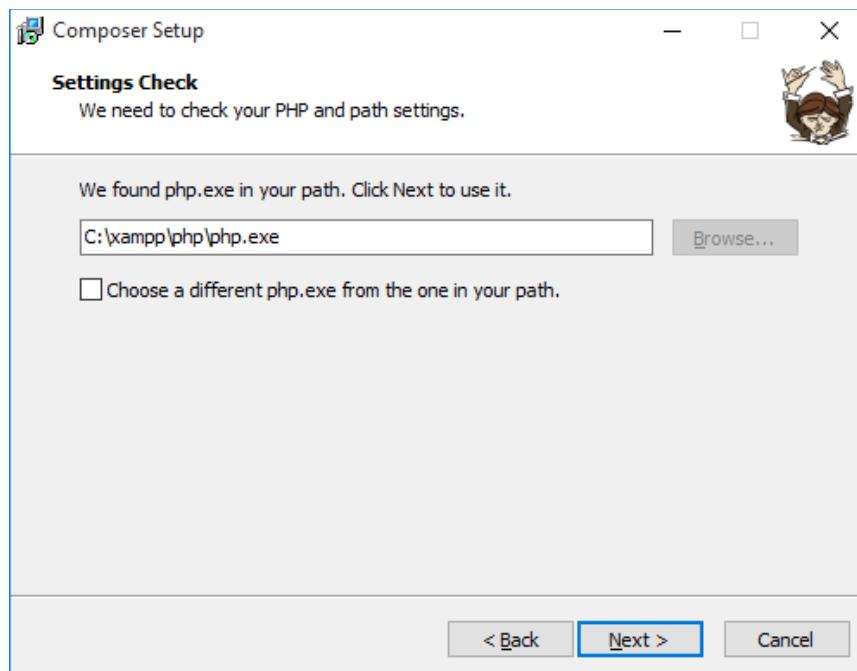
Kemudian klik tombol **Next**.

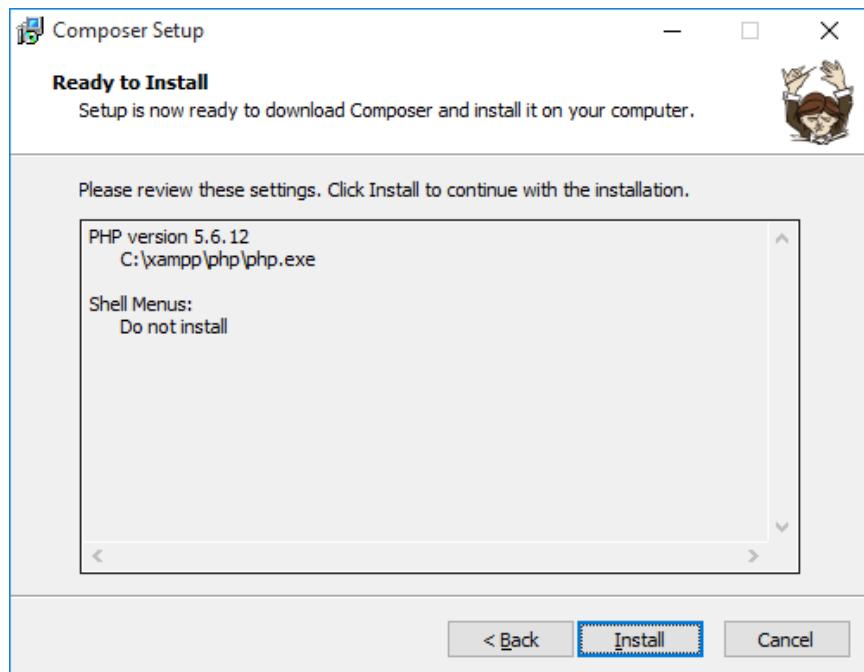


Pada jendela **Selects Components** kita dapat lanjutkan dengan menekan tombol **Next**.



Pada umumnya secara otomatis composer akan mendapatkan file php.exe yang telah kita install sebelumnya (Installasi XAMPP). Lanjutkan dengan menekan tombol **Next** dan **Install**.



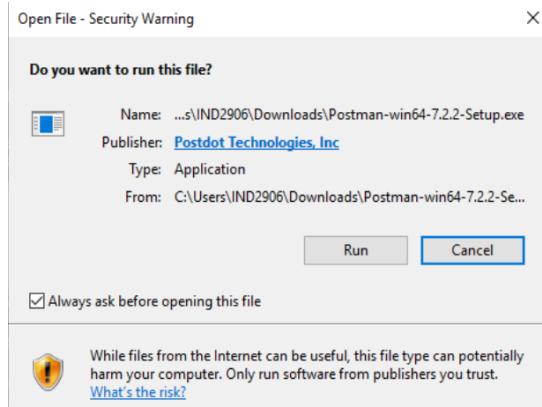


D. Install Postman

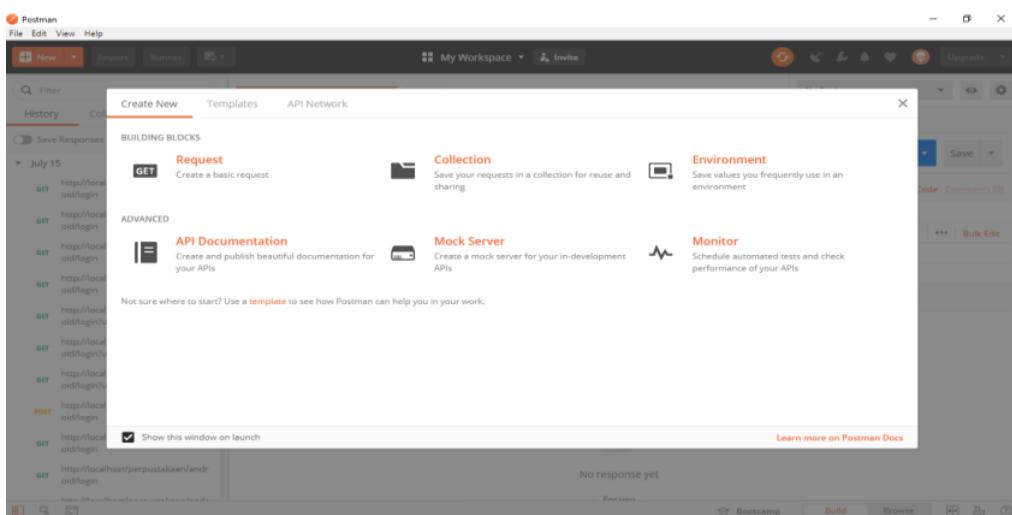
Postman adalah sebuah aplikasi fungsinya adalah sebagai REST Client atau istilahnya adalah aplikasi yang digunakan untuk melakukan uji coba REST API yang telah kita buat. Postman ini merupakan tools wajib bagi para developer yang bergerak pada pembuatan API, fungsi utama postman ini adalah sebagai GUI API Caller Pemanggil. namun sekarang postman juga menyediakan fitur lain yaitu Sharing Collection API for Documentation (free), Testing API (free), Realtime Collaboration Team (paid), Monitoring API (paid), Integration (paid).

Postman tersedia sebagai aplikasi asli untuk sistem operasi macOS, Windows (32-bit dan 64-bit), dan Linux (32-bit dan 64-bit). Untuk mendapatkan aplikasi Postman, dapat diunduh pada website resminya yaitu getpostman.com atau dapat diunduh pada halaman <https://www.postman.com/downloads/>

Setelah berhasil mengunduh paket instalasi postman, kemudian jalankan dengan cara klik dua kali. Pilih run jika muncul pop up seperti berikut :



Kemudian tunggu hingga proses instalasi selesai dan muncul seperti gambar berikut



E. Pembuatan Database

Buat database dengan nama : **toko_api**

Kemudian buat tabel-tabel dengan perintah sebagai berikut :

1. SQL membuat tabel member

```
CREATE table member (
    id INT NOT NULL AUTO_INCREMENT,
    nama VARCHAR(255) NOT NULL,
    email VARCHAR(255) NOT NULL,
    password VARCHAR(255) NOT NULL,
    PRIMARY KEY(id)
);
```

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	id 	int(11)			No	None		AUTO_INCREMENT
2	nama	varchar(255)	utf8mb4_general_ci		No	None		
3	email	varchar(255)	utf8mb4_general_ci		No	None		
4	password	varchar(255)	utf8mb4_general_ci		No	None		

2. SQL membuat tabel member_token

```
CREATE table member_token (
    id INT NOT NULL AUTO_INCREMENT,
    member_id INT NOT NULL,
    auth_key VARCHAR(255) NOT NULL,
    FOREIGN KEY (member_id) REFERENCES member(id) on update cascade on delete no action,
    PRIMARY KEY(id)
);
```

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	id 	int(11)			No	None		AUTO_INCREMENT
2	member_id 	int(11)			No	None		
3	auth_key	varchar(255)	utf8mb4_general_ci		No	None		

3. SQL membuat tabel produk

```
CREATE table produk (
    id INT NOT NULL AUTO_INCREMENT,
    kode_produk VARCHAR(255) NOT NULL,
    nama_produk VARCHAR(255) NOT NULL,
    harga INT NOT NULL,
    PRIMARY KEY(id)
);
```

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	id 	int(11)			No	None		AUTO_INCREMENT
2	kode_produk	varchar(255)	utf8mb4_general_ci		No	None		
3	nama_produk	varchar(255)	utf8mb4_general_ci		No	None		
4	harga	int(11)			No	None		

F. Installasi Lumen sebagai Restful API

Selanjutnya install projek lumen pada web server (pada folder C:\xampp\htdocs)

Untuk membuat projek lumen dapat menggunakan composer pada command prompt dengan menjalankan perintah

```
composer create-project --prefer-dist laravel/lumen <nama_projek>
```

Sekarang kita akan membuat projek Restful API dengan nama **toko-api**. Silahkan buka Command Prompt kemudian ketikkan perintah

```
C:\Users\BSI> cd c:\xampp\htdocs
```

```
C:\xampp\htdocs> composer create-project --prefer-dist laravel/lumen toko-api
```

PERTEMUAN 2

G. Konfigurasi Projek

1. Koneksi ke database

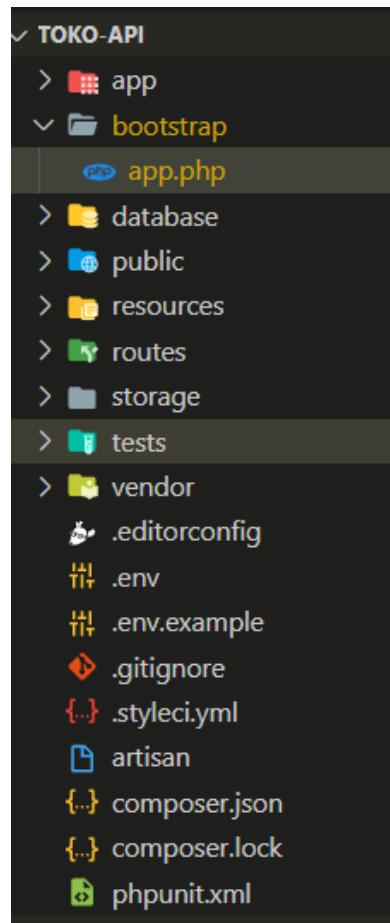
Buka file **.env** kemudian ubah kode berikut

```
DB_DATABASE=homestead  
DB_USERNAME=homestead  
DB_PASSWORD=secret
```

Menjadi sebagai berikut

```
DB_DATABASE=toko_api  
DB_USERNAME=root  
DB_PASSWORD=
```

Kemudian buka file **app.php** pada folder “bootstrap”



Kemudian hilangkan komentar pada kode berikut (pada baris 26 dan 28)

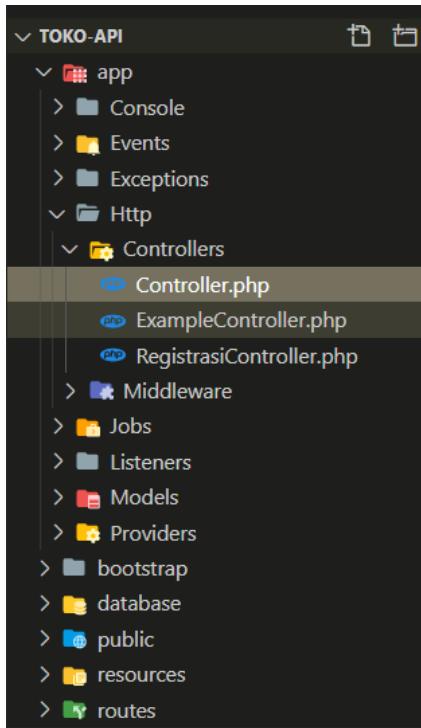
```
// $app->withFacades();  
  
// $app->withEloquent();
```

Menjadi

```
$app->withFacades();  
  
$app->withEloquent();
```

2. Membuat hasil response

Buka file **Controller.php** pada folder “app\Http\Controllers”



Kemudian tambahkan kode pada file tersebut sehingga menjadi

```
<?php  
  
namespace App\Http\Controllers;  
  
use Laravel\Lumen\Routing\Controller as BaseController;  
  
class Controller extends BaseController  
{
```

```

protected function responseHasil($code, $status, $data)
{
    return \response()->json([
        'code' => $code,
        'status' => $status,
        'data' => $data
    ]);
}
}

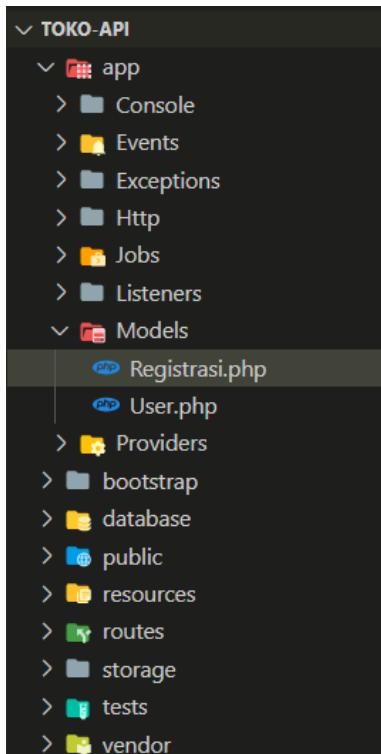
```

H. Membuat Toko API

1. Registrasi

a) Membuat model Registrasi

Buat sebuah file dengan nama **Registrasi.php** pada folder “app/Models”



Kemudian pada file **Registrasi.php** ketikkan kode berikut

```

<?php
namespace App\Models;
use Illuminate\Database\Eloquent\Model;

```

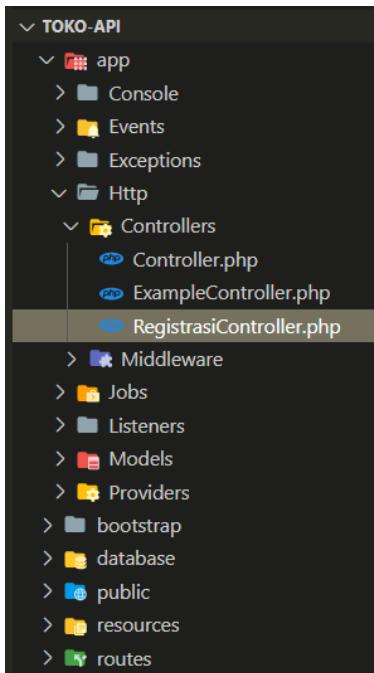
```

class Registrasi extends Model
{
    protected $table = 'member';
    protected $fillable = ['nama', 'email', 'password'];
    public $timestamps = false;
}

```

b) Membuat controller Registrasi

Buat sebuah file dengan nama **RegistrasiController.php** pada folder “app\Http\Controllers”



Kemudian pada file **RegistrasiController.php** tersebut masukkan kode berikut

```

<?php

namespace App\Http\Controllers;

use App\Models\Registrasi;
use Illuminate\Http\Request;
use Illuminate\Support\Facades\Hash;
class RegistrasiController extends Controller

```

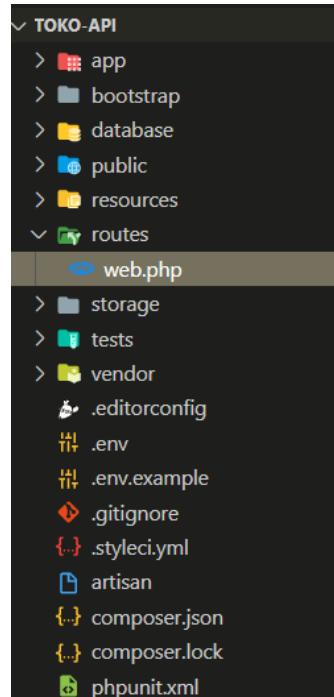
```
{
    public function registrasi(Request $request)
    {
        $nama = $request->input('nama');
        $email = $request->input('email');
        $password = Hash::make($request->input('password'));

        Registrasi::create([
            'nama' => $nama,
            'email' => $email,
            'password' => $password,
        ]);

        return $this->responseHasil(200, true, "Registrasi Berhasil");
    }
}
```

c) Menambah route Registrasi

Buka file **web.php** pada folder “routes”



Kemudian tambahkan kode berikut

```

<?php
/** @var \Laravel\Lumen\Routing\Router $router */
/*
-----
/ Application Routes
-----
/
| Here is where you can register all of the routes for an application.
| It is a breeze. Simply tell Lumen the URIs it should respond to
| and give it the Closure to call when that URI is requested.
|
*/
$router->get('/', function () use ($router) {
    return $router->app->version();
});

$router->post('/registrasi', ['uses' => 'RegistrasiController@registrasi']);

```

Pada baris terakhir kita menambahkan routing registrasi agar dapat diakses. Untuk mengakses registrasi, kita gunakan Postman dengan alamat url **localhost/toko-api/public/registrasi** dengan method **POST**

Adapun langkah menggunakan postman untuk menguji Rest API yang telah dibuat adalah sebagai berikut:

1. Buka aplikasi Postman yang telah terinstall
2. Masukkan alamat url untuk melakukan registrasi toko-api yang telah kita buat yaitu <http://localhost/toko-api/public/registrasi>
3. Selanjutnya pilih pengujian dengan type POST
4. Kemudian klik **Body** yang berada pada bagian bawah inputan url, kemudian pilih **x-www-form-urlencoded**
5. Kemudian isi key sesuai dengan field request pada **RegistrasiController** pada fungsi registrasi yaitu *nama, email, password* kemudian klik tombol **Send**

```

$nama = $request->input('nama');
$email = $request->input('email');
$password = Hash::make($request->input('password'));

```

The screenshot shows a Postman interface. On the left, a 'Body' section is set to 'form-data'. It contains three fields: 'nama' with value 'Administrator', 'email' with value 'admin@admin.com', and 'password' with value 'admin'. On the right, the response is displayed in a JSON viewer. The JSON object has three properties: 'code' (value: 200), 'status' (value: true), and 'data' (value: 'Registrasi Berhasil').

2. Login

a) Membuat model Member

Buat sebuah file dengan nama **Member.php** pada folder *Models* dan ketikkan kode berikut

```

<?php
namespace App\Models;
use Illuminate\Database\Eloquent\Model;
class Member extends Model
{
    protected $table = 'member';
    public $timestamps = false;
}

```

b) Membuat model Login

Buat sebuah file dengan nama **Login.php** pada folder “app/Models” dan ketikkan kode berikut

```
<?php
```

```

namespace App\Models;
use Illuminate\Database\Eloquent\Model;
class Login extends Model
{
    protected $table = 'member_token';
    protected $fillable = ['member_id', 'auth_key'];
    public $timestamps = false;
}

```

c) Membuat controller Login

Buat sebuah file dengan nama **LoginController** pada folder “app/Http/Controllers” dan ketikkan kode berikut

```

<?php
namespace App\Http\Controllers;

use App\Models\Login;
use App\Models\Member;
use Illuminate\Http\Request;
use Illuminate\Support\Facades\Hash;

class LoginController extends Controller
{
    public function login(Request $request)
    {
        $email = $request->input('email');
        $password = $request->input('password');

        $member = Member::query()->firstWhere(['email' => $email]);
        if ($member == null) {
            return $this->responseHasil(400, false, 'Email tidak ditemukan');
        }
        if (!Hash::check($password, $member->password)) {

```

```

        return $this->responseHasil(400, false, 'Password tidak valid');

    }

$login = Login::create([
    'member_id' => $member->id,
    'auth_key' => $this->RandomString(),
]);

if (!$login) {
    return $this->responseHasil(401, false, 'Unauthorized');
}

$data = [
    'token' => $login->auth_key,
    'user' => [
        'id' => $member->id,
        'email' => $member->email,
    ]
];

return $this->responseHasil(200, true, $data);
}

private function RandomString($length = 100)
{
    $karakkter =
'012345678dssd9abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ';
    $panjang_karakter = strlen($karakkter);
    $str = '';
    for ($i = 0; $i < $length; $i++) {
        $str .= $karakkter[rand(0, $panjang_karakter - 1)];
    }
    return $str;
}

```

d) Menambahkan route Login

Pada route tambahkan kode untuk mengakses login sehingga menjadi sebagai berikut

```
$router->post('/registrasi', ['uses' => 'RegistrasiController@registerasi']);  
$router->post('/login', ['uses' => 'LoginController@login']);
```

e) Mencoba Rest

Silahkan coba Rest API dengan memasukkan url <http://localhost/toko-api/public/login> dengan method POST

The screenshot shows a Postman interface with a POST request to `localhost/toko-api/public/login`. The request body is set to `form-data` with two fields: `email` (value: `admin@admin.com`) and `password` (value: `admin`). The response is displayed in JSON format, showing a success status (code 200, status true) and a data object containing a token and a user object with id 6 and email `admin@admin.com`. The token is a long string of characters.

```
POST localhost/toko-api/public/login  
Body form-data  
Key Value  
email admin@admin.com  
password admin  
Body  
Pretty Raw Preview Visualize JSON  
1 "code": 200,  
2 "status": true,  
3 "data": {  
4     "token":  
5         "hjFgKg9Qe4TQRuUb1zQgc8yE0wGQ2SzzedONYo6C8R2fqvVtc45wcweKcCmYUzTXT54smeVRfk0R  
6         XLDjuYKLfd2ZYddqYrJtfgq",  
7     "user": {  
8         "id": 6,  
9         "email": "admin@admin.com"  
10    }  
11 }
```

PERTEMUAN 3

3. CRUD Produk
 - a) Membuat model Produk

Buat sebuah file dengan nama **Produk.php** pada folder “app/Models” dan ketikkan kode berikut

```
<?php

namespace App\Models;
use Illuminate\Database\Eloquent\Model;

class Produk extends Model
{
    protected $table = 'produk';
    protected $fillable = ['kode_produk', 'nama_produk', 'harga'];
    public $timestamps = false;
}
```

- b) Membuat controller produk

Buat sebuah file dengan nama **ProdukController** pada folder “app/Http/Controllers” dan ketikkan kode berikut

```
<?php

namespace App\Http\Controllers;

use App\Models\Produk;
use Illuminate\Http\Request;

class ProdukController extends Controller
{



}
```

Selanjutnya pada class **ProdukController** kita akan menambahkan fungsi (function) CRUD produk, yaitu:

(1) Membuat fungsi create produk

```
public function create(Request $request)
{
    $kodeProduk = $request->input('kode_produk');
    $namaProduk = $request->input('nama_produk');
    $harga = $request->input('harga');

    $produk = Produk::create([
        'kode_produk' => $kodeProduk,
        'nama_produk' => $namaProduk,
        'harga' => $harga,
    ]);
    return $this->responseHasil(200, true, $produk);
}
```

(2) Membuat fungsi list produk

```
public function list()
{
    $produk = Produk::all();
    return $this->responseHasil(200, true, $produk);
}
```

(3) Membuat fungsi tampil produk

```
public function show($id)
{
    $produk = Produk::findOrFail($id);
    return $this->responseHasil(200, true, $produk);
}
```

(4) Membuat fungsi update produk

```
public function update(Request $request, $id)
{
    $kodeProduk = $request->input('kode_produk');
    $namaProduk = $request->input('nama_produk');
    $harga = $request->input('harga');

    $produk = Produk::findOrFail($id);
    $result = $produk->update([
        'kode_produk' => $kodeProduk,
        'nama_produk' => $namaProduk,
        'harga' => $harga,
    ]);
}
```

```

    ]);
    return $this->responseHasil(200, true, $result);
}

```

(5) Membuat fungsi delete produk

```

public function delete($id)
{
    $produk = Produk::findOrFail($id);
    $delete = $produk->delete();
    return $this->responseHasil(200, true, $delete);
}

```

Sehingga adapun keseluruhan kode **ProdukController.php** adalah sebagai berikut

```

<?php

namespace App\Http\Controllers;

use App\Models\Produk;
use Illuminate\Http\Request;

class ProdukController extends Controller
{
    public function create(Request $request)
    {
        $kodeProduk = $request->input('kode_produkt');
        $namaProduk = $request->input('nama_produkt');
        $harga = $request->input('harga');

        $produk = Produk::create([
            'kode_produkt' => $kodeProduk,
            'nama_produkt' => $namaProduk,
            'harga' => $harga,
        ]);
        return $this->responseHasil(200, true, $produk);
    }

    public function list()
    {
        $produk = Produk::all();
        return $this->responseHasil(200, true, $produk);
    }
}

```

```

public function show($id)
{
    $produk = Produk::findOrFail($id);
    return $this->responseHasil(200, true, $produk);
}

public function update(Request $request, $id)
{
    $kodeProduk = $request->input('kode_produk');
    $namaProduk = $request->input('nama_produk');
    $harga = $request->input('harga');

    $produk = Produk::findOrFail($id);
    $result = $produk->update([
        'kode_produk' => $kodeProduk,
        'nama_produk' => $namaProduk,
        'harga' => $harga,
    ]);
    return $this->responseHasil(200, true, $result);
}

public function delete($id)
{
    $produk = Produk::findOrFail($id);
    $delete = $produk->delete();
    return $this->responseHasil(200, true, $delete);
}

```

(6) Menambahkan route produk

Agar ProdukController dapat diakses, selanjutnya tambah route untuk mengakses produk pada file “routes/web.php”

```

$router->group(['prefix' => 'produk'], function ($router) {
    $router->post('/', ['uses' => 'ProdukController@create']);
    $router->get('/', ['uses' => 'ProdukController@list']);
    $router->get('/{id}', ['uses' => 'ProdukController@show']);
    $router->post('/{id}/update', ['uses' => 'ProdukController@update']);
    $router->delete('/{id}', ['uses' => 'ProdukController@delete']);
});

```

Adapun keseluruhan kode pada “routes/web.php” adalah sebagai berikut

```
<?php
```

```

/** @var \Laravel\Lumen\Routing\Router $router */

/*
-----
| Application Routes
|-----
|
| Here is where you can register all of the routes for an application.
| It is a breeze. Simply tell Lumen the URLs it should respond to
| and give it the Closure to call when that URI is requested.
|
*/
$router->get('/', function () use ($router) {
    return $router->app->version();
});

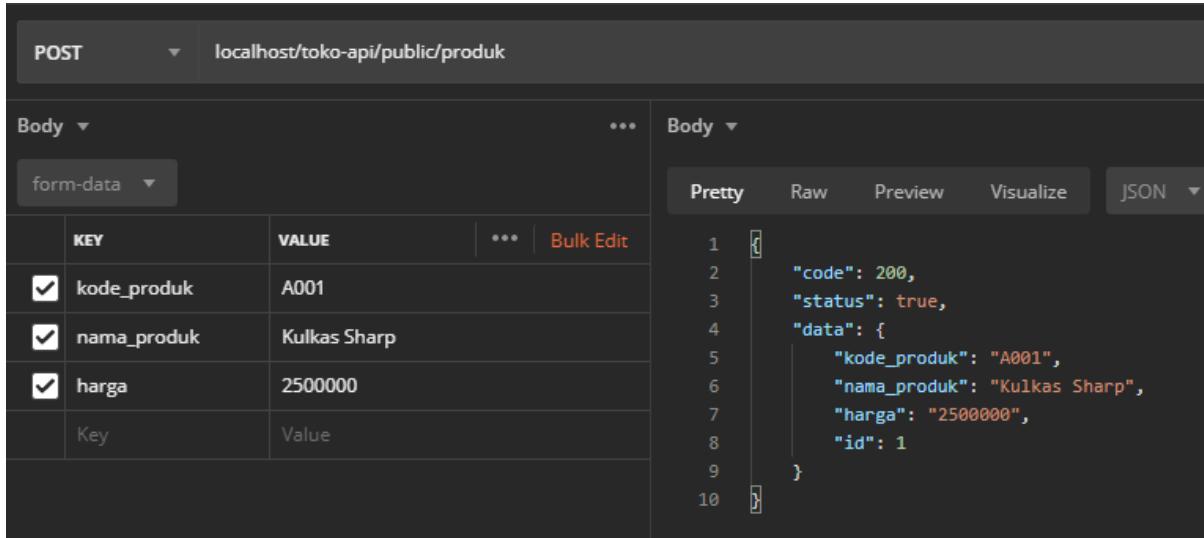
$router->post('/registrasi', ['uses' => 'RegistrasiController@register']);
$router->post('/login', ['uses' => 'LoginController@login']);

$router->group(['prefix' => 'produk'], function ($router) {
    $router->post('/', ['uses' => 'ProdukController@create']);
    $router->get('/', ['uses' => 'ProdukController@list']);
    $router->get('/{id}', ['uses' => 'ProdukController@show']);
    $router->post('/{id}/update', ['uses' => 'ProdukController@update']);
    $router->delete('/{id}', ['uses' => 'ProdukController@delete']);
});

```

c) Mencoba Rest

- (1) Create Produk (localhost/toko-api/public/produk) dengan method POST



POST <localhost/toko-api/public/produk>

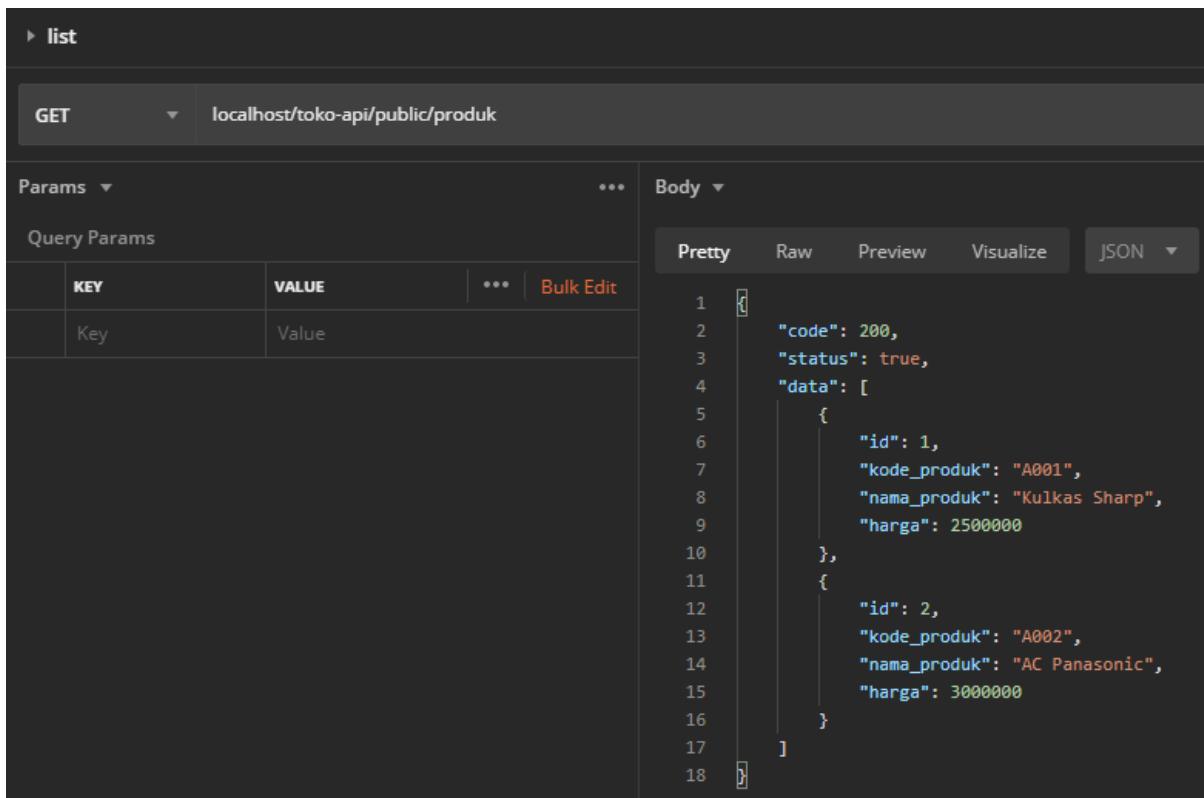
Body [form-data](#)

KEY	VALUE
<input checked="" type="checkbox"/> kode_produk	A001
<input checked="" type="checkbox"/> nama_produk	Kulkas Sharp
<input checked="" type="checkbox"/> harga	2500000
Key	Value

Body [Pretty](#) [Raw](#) [Preview](#) [Visualize](#) [JSON](#)

```
1 [  
2   "code": 200,  
3   "status": true,  
4   "data": {  
5     "kode_produk": "A001",  
6     "nama_produk": "Kulkas Sharp",  
7     "harga": "2500000",  
8     "id": 1  
9   }  
10 ]
```

- (2) List Produk (localhost/toko-api/public/produk) dengan method GET



list

GET <localhost/toko-api/public/produk>

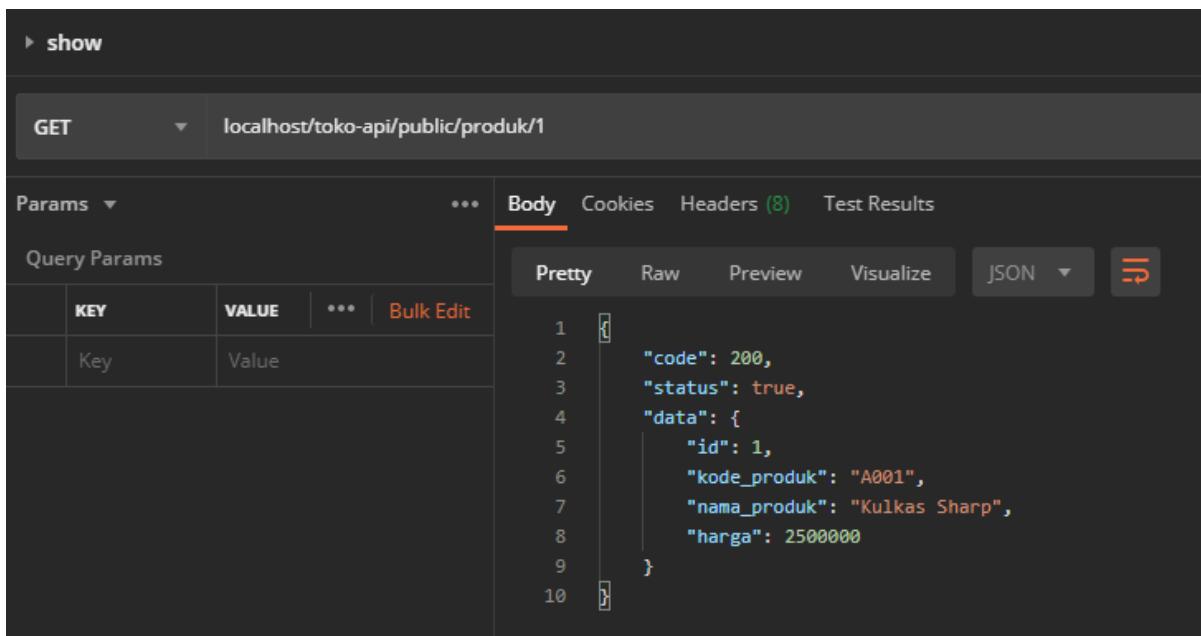
Params [Query Params](#)

KEY	VALUE
Key	Value

Body [Pretty](#) [Raw](#) [Preview](#) [Visualize](#) [JSON](#)

```
1 [  
2   "code": 200,  
3   "status": true,  
4   "data": [  
5     {  
6       "id": 1,  
7       "kode_produk": "A001",  
8       "nama_produk": "Kulkas Sharp",  
9       "harga": 2500000  
10      },  
11      {  
12        "id": 2,  
13        "kode_produk": "A002",  
14        "nama_produk": "AC Panasonic",  
15        "harga": 3000000  
16      }  
17    ]  
18 ]
```

(3) Show Produk (localhost/toko-api/public/produk/{id}) dengan method GET



show

GET localhost/toko-api/public/produk/1

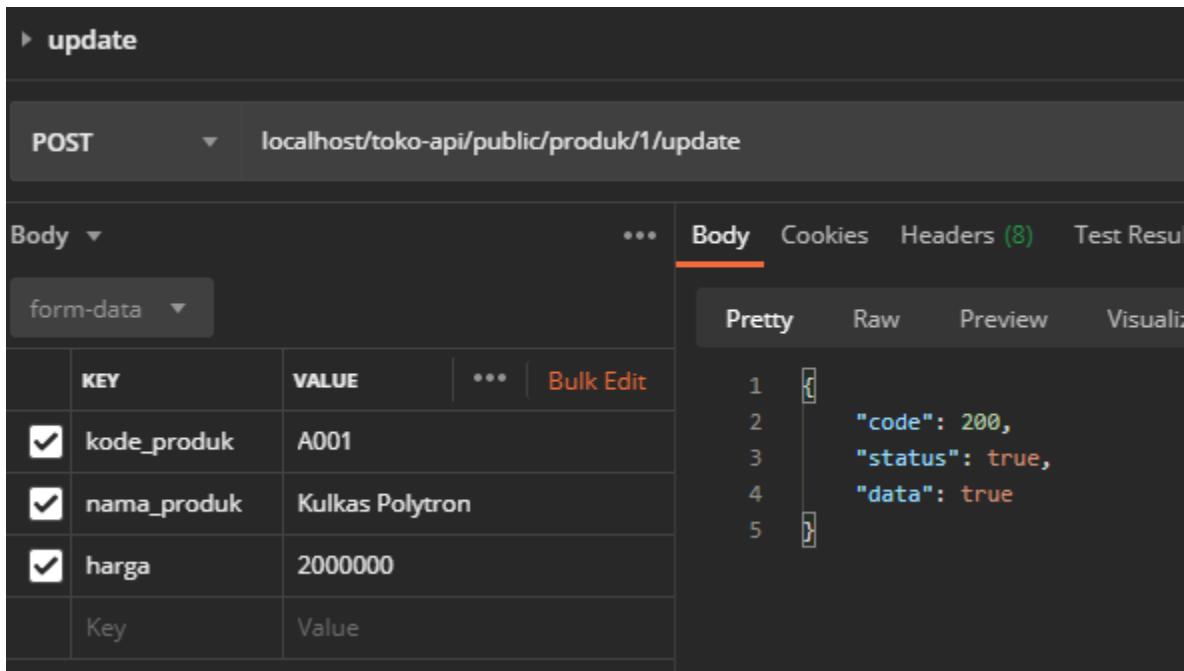
Params **Query Params**

KEY	VALUE
Key	Value

Body

```
1  {
2      "code": 200,
3      "status": true,
4      "data": {
5          "id": 1,
6          "kode_produk": "A001",
7          "nama_produk": "Kulkas Sharp",
8          "harga": 2500000
9      }
10 }
```

(4) Update Produk (localhost/toko-api/public/produk/{id}/update) dengan method POST



update

POST localhost/toko-api/public/produk/1/update

Body **form-data**

KEY	VALUE
<input checked="" type="checkbox"/> kode_produk	A001
<input checked="" type="checkbox"/> nama_produk	Kulkas Polytron
<input checked="" type="checkbox"/> harga	2000000
Key	Value

Body

```
1  {
2      "code": 200,
3      "status": true,
4      "data": true
5 }
```

- (5) Delete Produk (localhost/toko-api/public/produk) dengan method DELETE

The screenshot shows the Postman interface for a DELETE request. The URL is set to `localhost/toko-api/public/produk/4`. The response body is displayed in a pretty-printed JSON format:

```
1 {  
2   "code": 200,  
3   "status": true,  
4   "data": true  
5 }
```

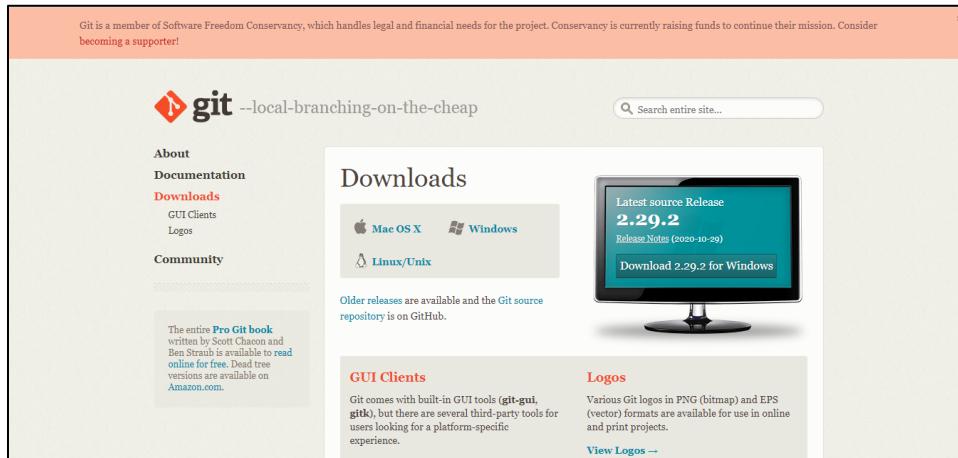
PERTEMUAN 4

I. Pembuatan Aplikasi Mobile dengan Flutter

1. Menyiapkan perangkat

a) Install Git

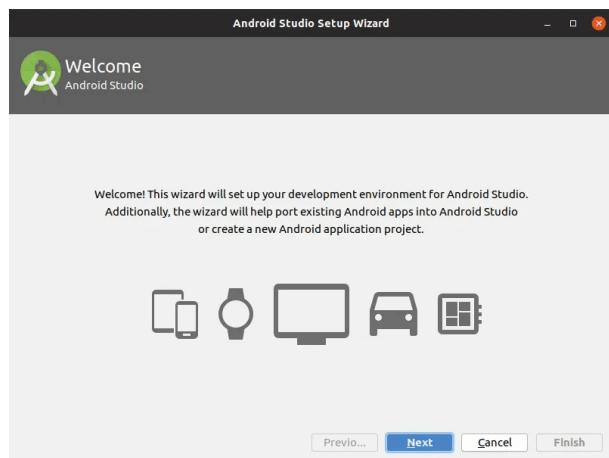
Buka laman <https://git-scm.com/downloads>, kemudian klik tombol download



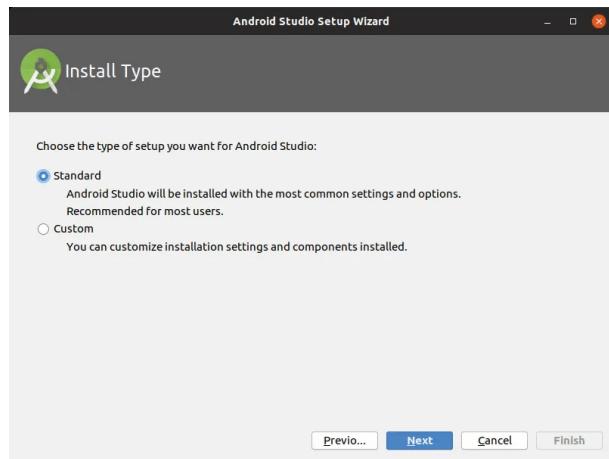
Kemudian lakukan installasi git dari file yang telah diunduh.

b) Install Android Studio

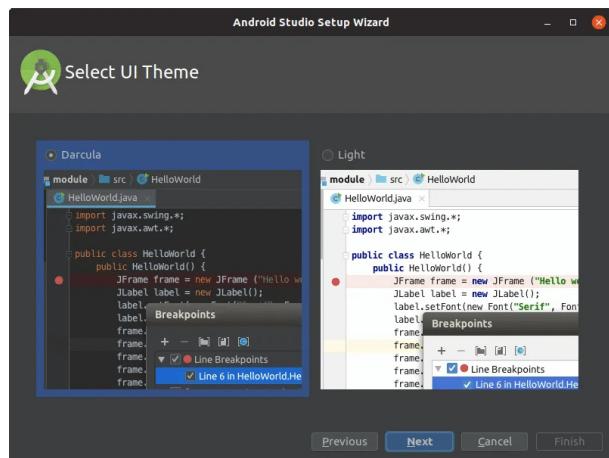
Android studio dapat diunduh pada laman <https://developer.android.com/studio>. Setelah diunduh klik dua kali pada file yang telah diunduh tersebut, kemudian lakukan installasi dengan mengikuti langkah-langkah yang telah disediakan



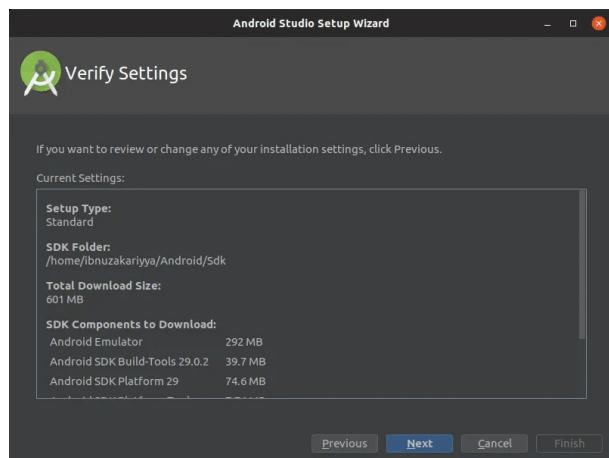
Kemudian pilih tipe standar dan klik next



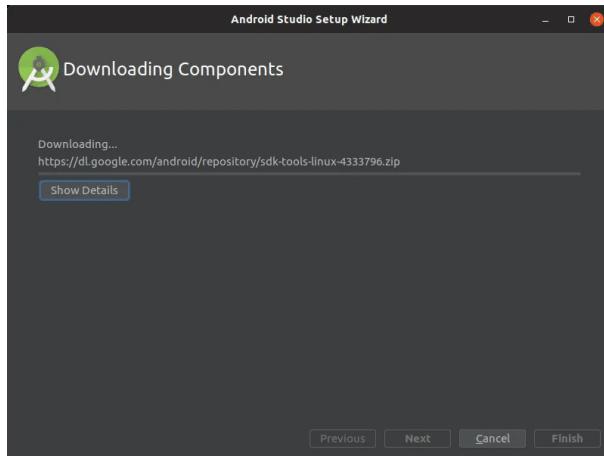
Pilih tema tampilan kemudian klik next



Kemudian klik tombol next



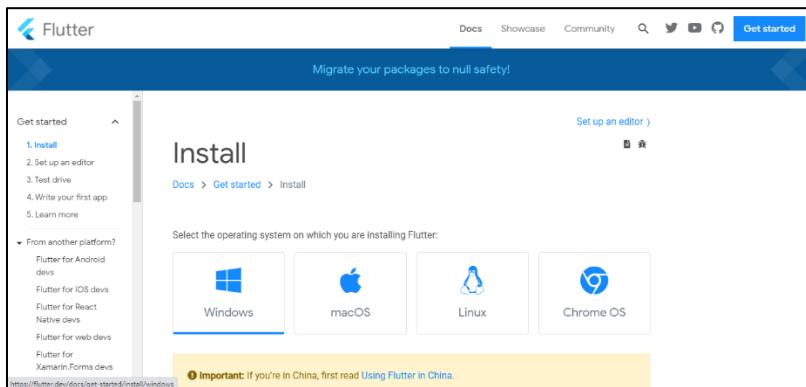
Pastikan komputer terhubung dengan internet yang stabil, karena android studio akan mengunduh komponen-komponen yang diperlukan



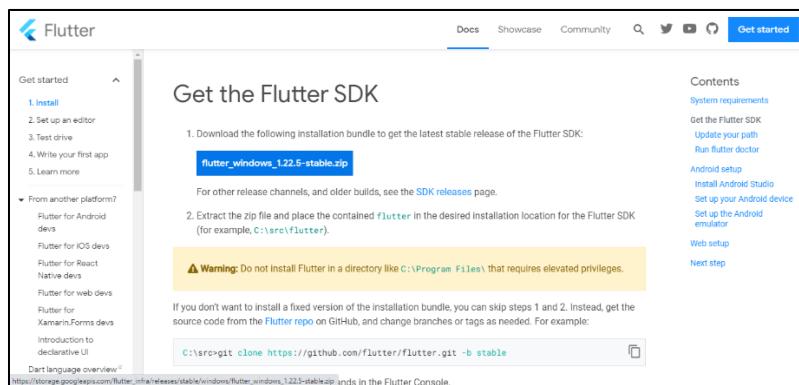
Setelah selesai klik finish

c) Install Flutter

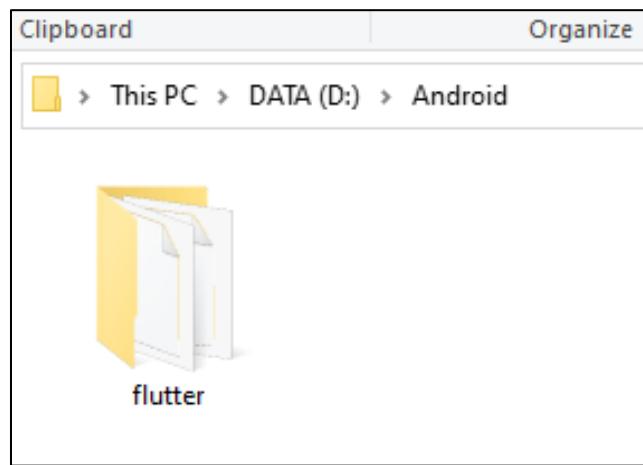
Buka laman <https://flutter.dev/docs/get-started/install> Kemudian pilih “Windows”



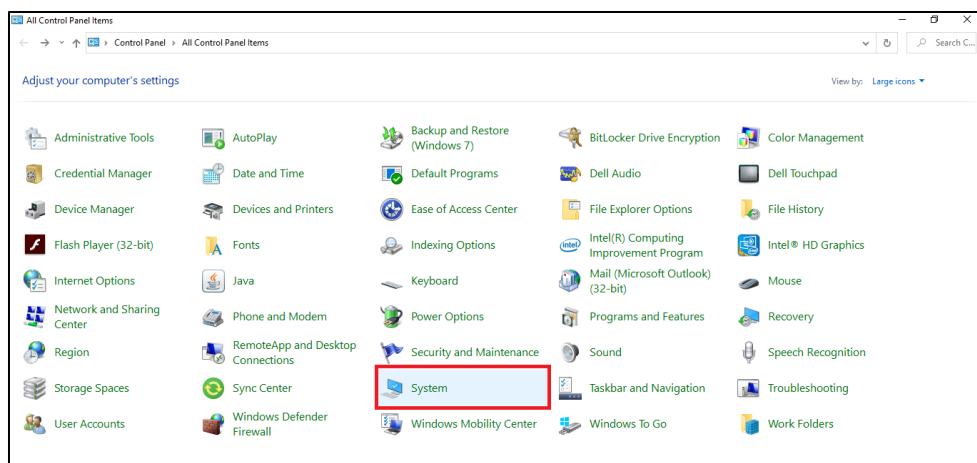
Kemudian pilih **flutter_windows_** untuk mengunduh file flutter



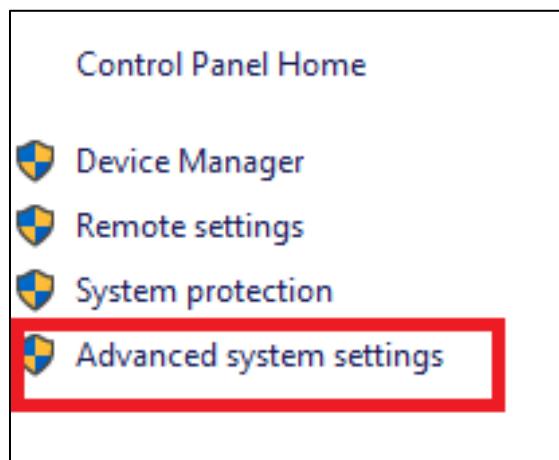
Kemudian ekstrak file zip flutter misalnya di “D:/Android”



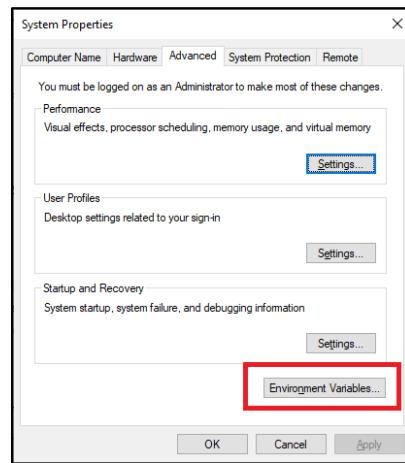
Kemudian buka **Control Panel**, pilih “System”



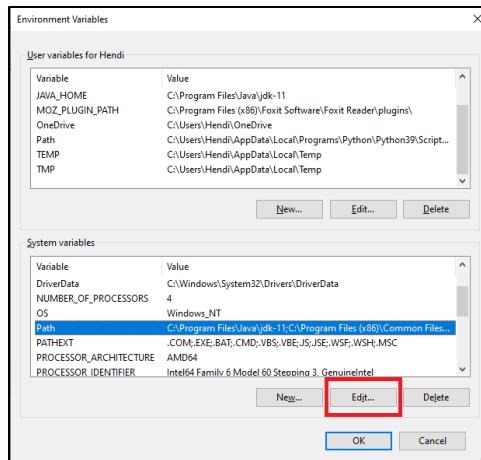
Kemudian pilih “Advanced System Setting”



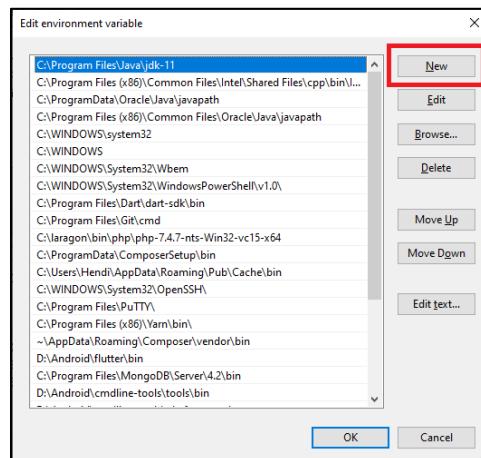
Kemudian klik tombol “Environtment Variables”



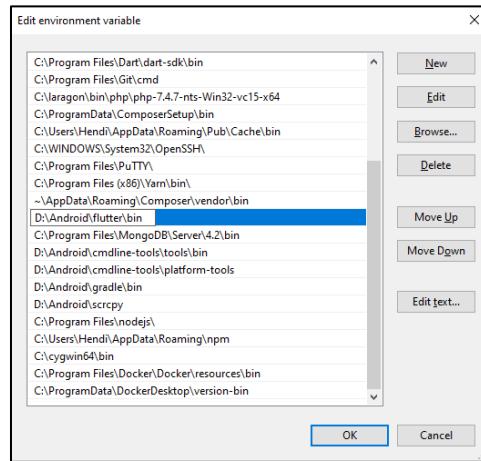
Kemudian pilih pada “System Variables” pilih “Path” dan klik tombol “Edit”



Kemudian klik tombol “New”

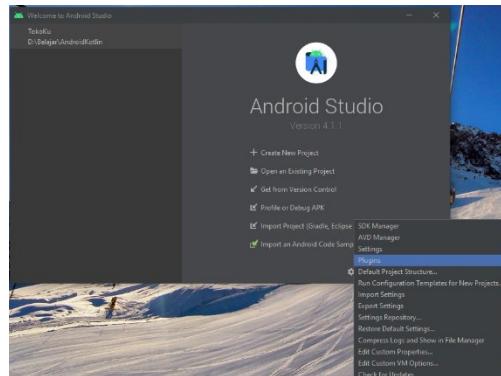


Kemudian masukkan alamat folder bin pada flutter yang telah kita ekstrak dalam hal ini misalnya “D:\Android\flutter\bin”

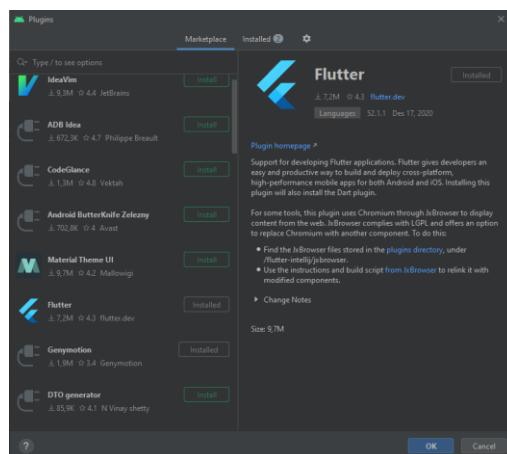


d) Konfigurasi Android Studio dengan Flutter

Jalankan Android Studio kemudian pada menu “Configure” pilih “Plugins”



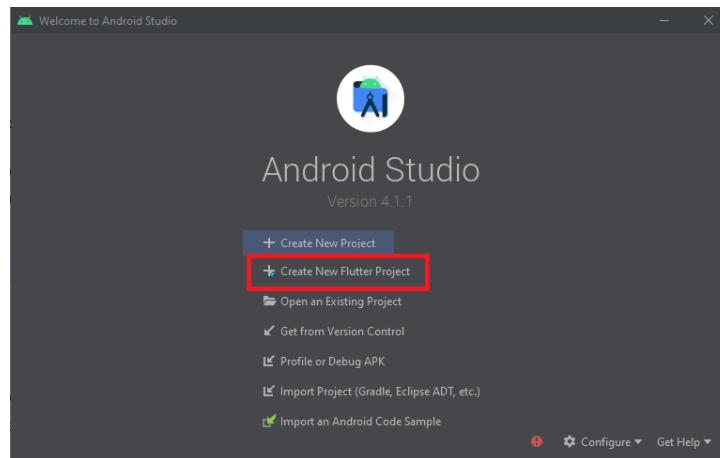
Pilih “Flutter” kemudian klik tombol “Install”



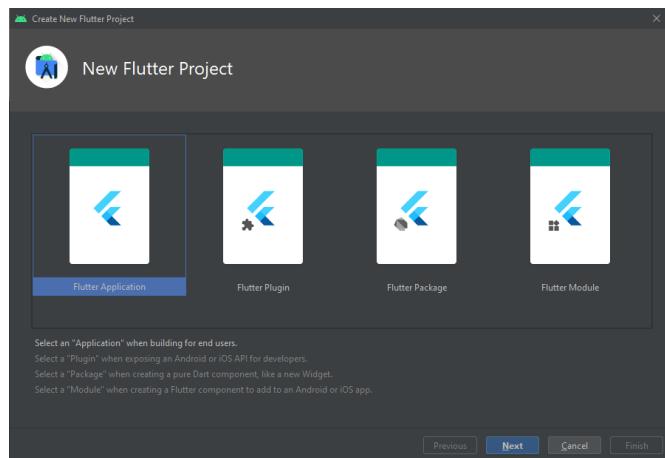
Setelah selesai, restart/tutup Android Studio

2. Membuat dan menjalankan projek

Jalankan Android Studio kemudian pilih “Create New Flutter Project”



Kemudian pilih “Flutter Application” dan klik “Next”



Kemudian tentukan :

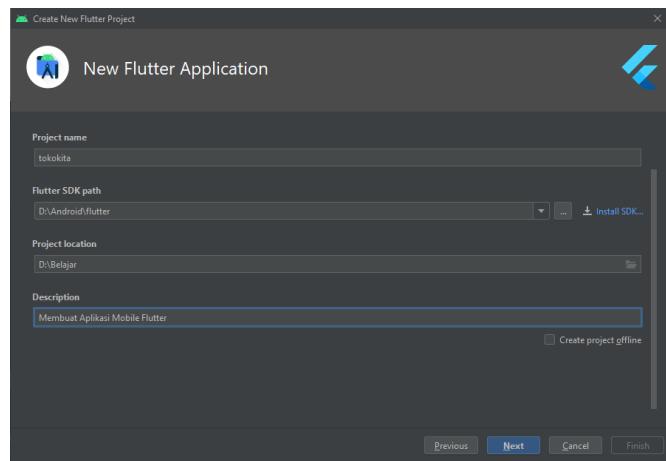
Project Name : tokokita

Flutter SDK path : D:\Android\flutter (masukkan sesuai alamat folder flutter diletakkan)

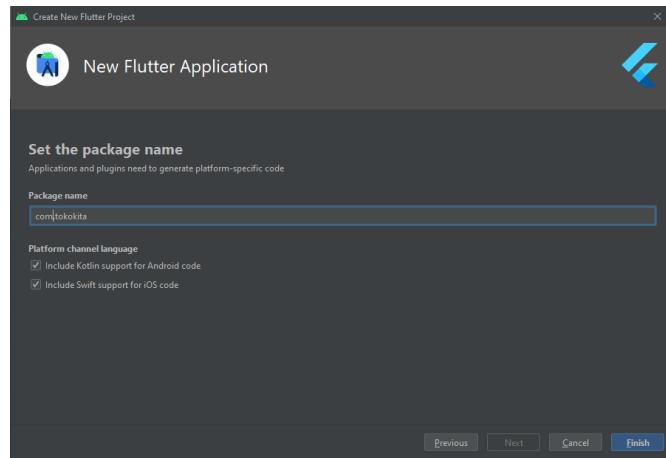
Project location : D:\Belajar (bebas memasukkan dimanapun)

Description : Membuat Aplikasi Mobile Flutter (bebas)

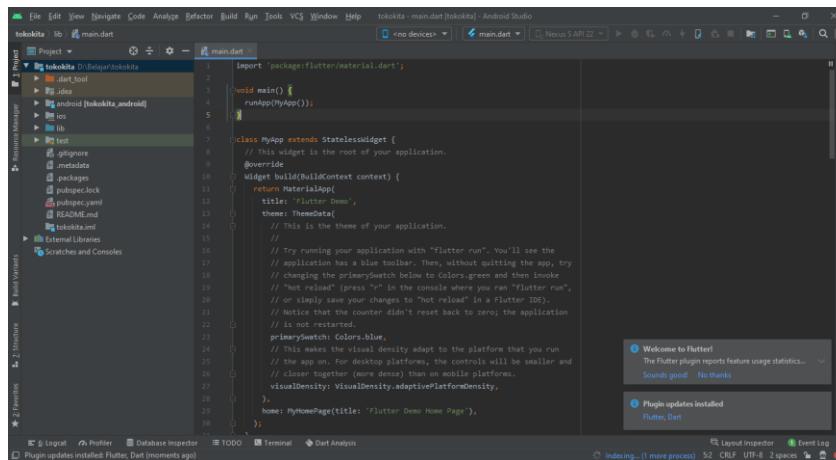
Kemudian klik tombol “Next”



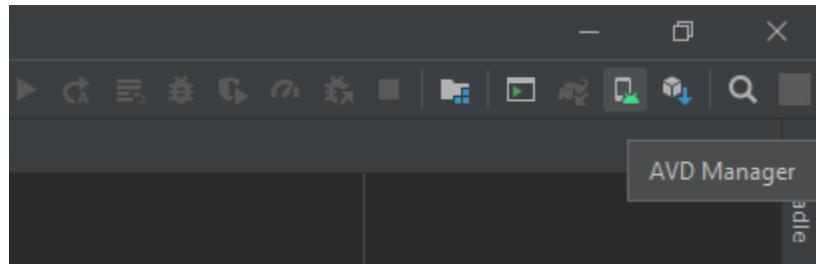
Kemudian klik tombol “Finish”. Pastikan perangkat terhubung ke internet, karena diperlukan untuk mengunduh projek yang dibuat



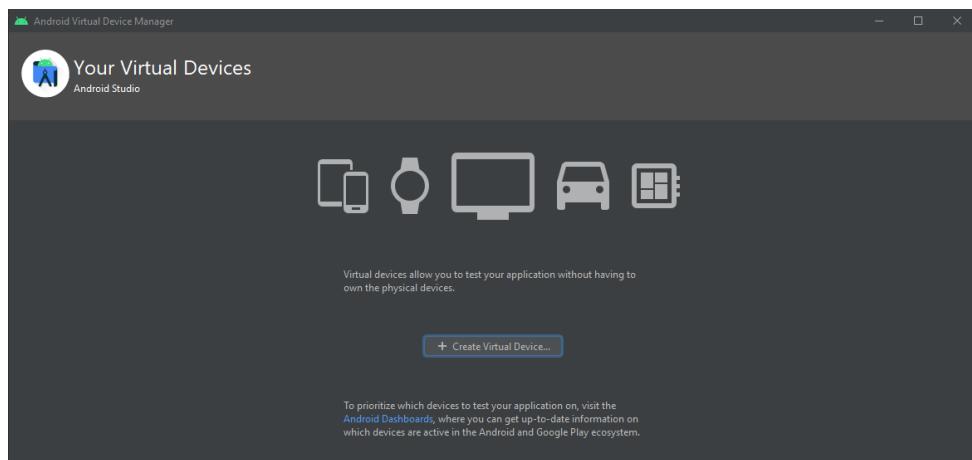
Setelah selesai akan muncul projek yang telah dibuat



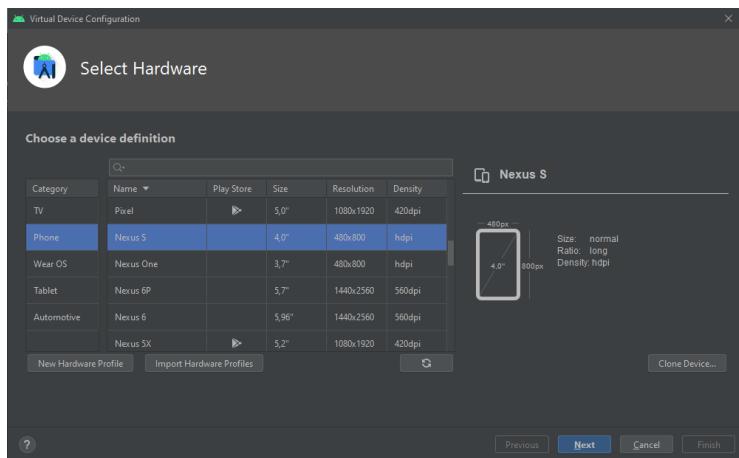
Untuk menjalankan projek, kita memerlukan emulator android. Pertama kita akan membuat emulator android dengan cara klik “AVD Manager” pada pojok kiri atas



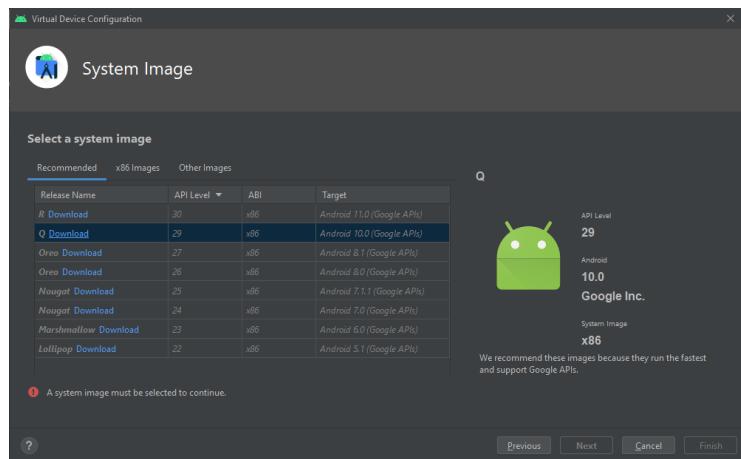
Kemudian klik tombol “Create Virtual Device”



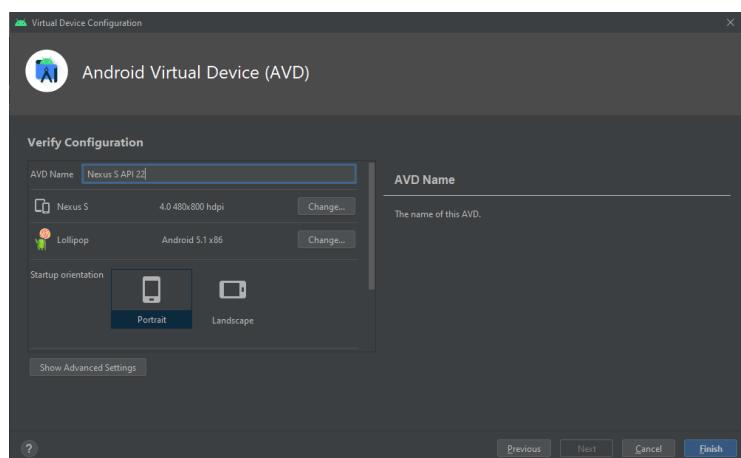
Pilih salah satu device yang dinginkan, misalnya “Nexus S” kemudian klik tombol “Next”



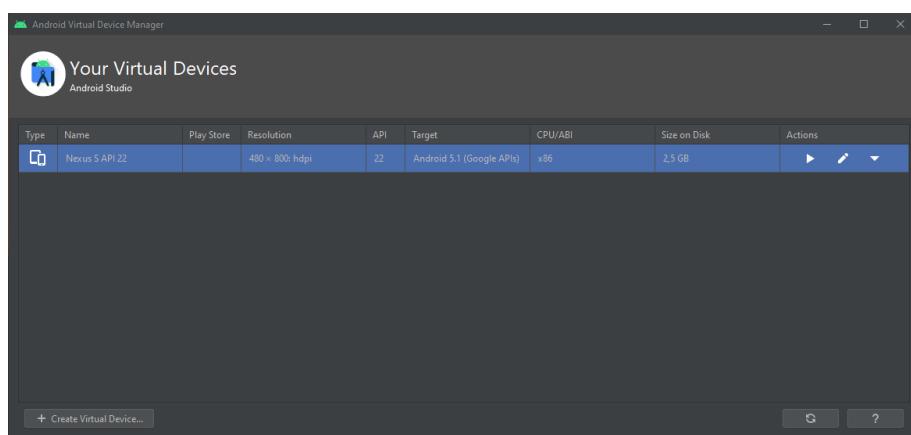
Kemudian kita akan memilih Versi Sistem Operasi Android, dalam hal ini misalnya “Q”, jika belum ada maka kita akan diminta untuk mengunduh terlebih dahulu.



Setelah itu klik Finish



Untuk Menjalankan Emulator, klik logo play



Kemudian akan muncul emulator android seperti berikut:



Jika emulator sudah berjalan, kita dapat mengeksekusi projek dengan cara klik tombol play (berwarna hijau) pada Android Studio

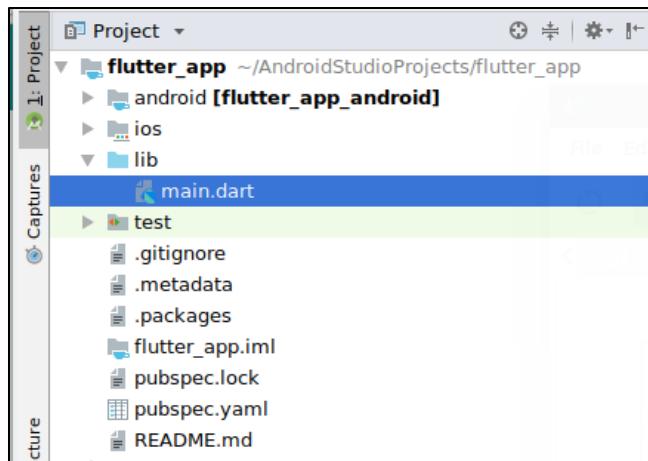


Pada emulator akan tampil hasil projek seperti berikut



3. Struktur Folder Flutter

Adapun struktur folder Projek flutter adalah sebagai berikut:

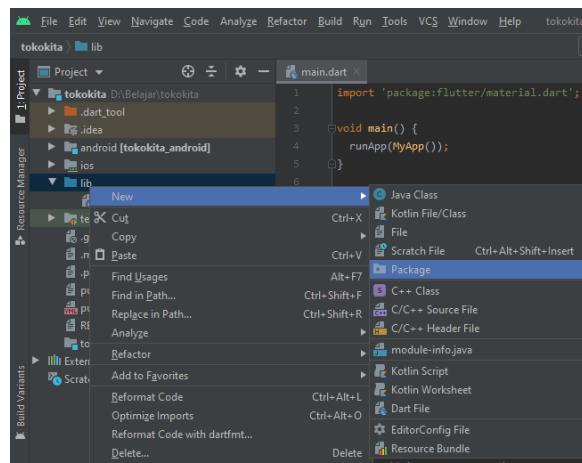


- android** berisi source code untuk aplikasi android;
- ios** berisi source code untuk aplikasi iOS;
- lib** berisi source code Dart, di sini kita akan menulis kode aplikasi;
- test** berisi source code Dart untuk testing aplikasi;
- .gitignore** adalah file [Git](#);
- .metadata** merupakan file yang berisi metadata project yang di-generate otomatis;
- .packages** merupakan file yang berisi alamat path package yang dibuat oleh pub;
- flutter_app.iml** merupakan file XML yang berisi keterangan project;
- pubspec.lock** merupakan file yang berisi versi-versi library atau package. File ini dibuat oleh pub. Fungsinya untuk mengunci versi package.
- pubspec.yaml** merupakan file yang berisi informasi tentang project dan libraray yang dibutuhkan;
- README.md** merupakan file markdown yang berisi penjelasan tentang source code.

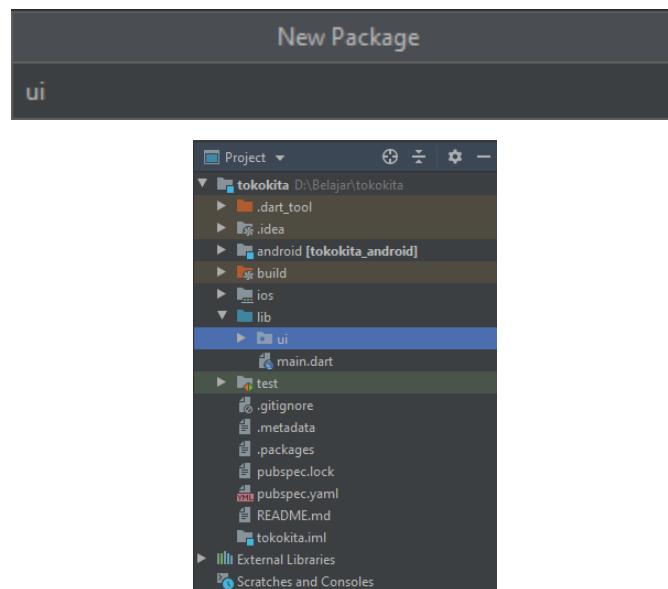
PERTEMUAN 5

4. Membuat Halaman

Pertama kita akan memecah bagian-bagian kode menjadi beberapa bagian, adapun untuk tampilan, dikelompokkan kedalam folder atau package **ui**. Buat folder/package dengan cara klik kanan folder **lib** kemudian pilih **New -> package**

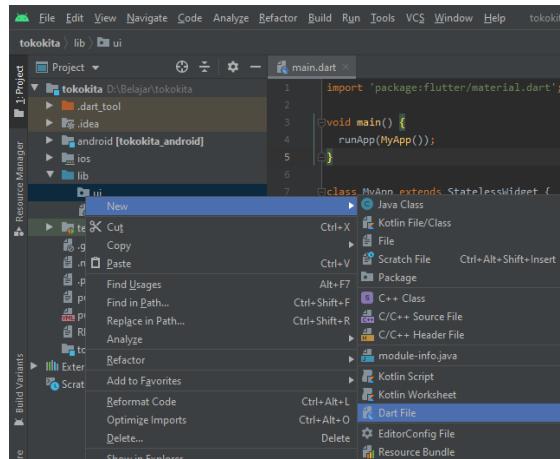


Setelah itu masukkan nama package yaitu **ui**

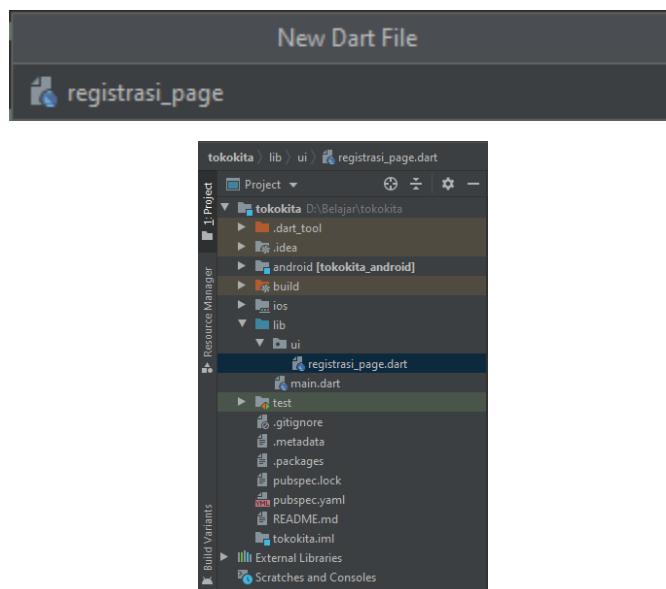


a) Registrasi

Buat sebuah file dengan nama **registrasi_page.dart** pada folder **ui** dengan cara klik kanan folder **ui** kemudian pilih **New -> Dart File**



Kemudian ketikkan **registrasi_page**



Pada file **registrasi_page.dart** ketikkan kode berikut

```
1. import 'package:flutter/material.dart';
2.
3. class RegistrasiPage extends StatefulWidget {
4.   @override
5.   _RegistrasiPageState createState() => _RegistrasiPageState();
6. }
7.
```

```

8. class _RegistrasiPageState extends State<RegistrasiPage> {
9.   final GlobalKey<FormState> _formKey = GlobalKey<FormState>();
10.  bool _isLoading = false;
11.
12.  final TextEditingController _namaTextboxController = TextEditingController();
13.  final TextEditingController _emailTextboxController = TextEditingController();
14.  final TextEditingController _passwordTextboxController = TextEditingController();
15.
16. @override
17. Widget build(BuildContext context) {
18.   return Scaffold(
19.     appBar: AppBar(title: Text("Registrasi")),
20.     body: SingleChildScrollView(
21.       child: Container(
22.         child: Padding(
23.           padding: const EdgeInsets.all(8.0),
24.           child: Form(
25.             key: _formKey,
26.             child: Column(
27.               mainAxisAlignment: MainAxisAlignment.center,
28.               children: [
29.                 _namaTextField(),
30.                 _emailTextField(),
31.                 _passwordTextField(),
32.                 _passwordKonfirmasiTextField(),
33.                 _buttonRegistrasi()
34.               ],
35.             ),
36.           ),
37.         ),
38.       ),
39.     ),
40.   );
41. }
42.
43. //Membuat Textbox Nama
44. Widget _namaTextField() {
45.   return TextFormField(
46.     decoration: InputDecoration(labelText: "Nama"),
47.     keyboardType: TextInputType.text,
48.     controller: _namaTextboxController,
49.     validator: (value){
50.       if(value.length < 3){
51.         return "Nama harus diisi minimal 3 karakter";
52.       }
53.       return null;
54.     },
55.   );
56. }
57.
58. //Membuat Textbox email
59. Widget _emailTextField() {
60.   return TextFormField(
61.     decoration: InputDecoration(labelText: "Email"),
62.     keyboardType: TextInputType.emailAddress,
63.     controller: _emailTextboxController,
64.     validator: (value){
65.       //validasi harus diisi
66.       if(value.isEmpty){
67.         return 'Email harus diisi';

```

```

68.        }
69.        //validasi email
70.        Pattern pattern = r'^((([^\>()[]\\.,;:\\s@\\"]+\\.([^\>()[]\\.,;:\\s@\\"]+)*|(\".+\")|((\\[[0-9]{1,3}\\.[0-9]{1,3}\\.[0-9]{1,3}\\.[0-9]{1,3}\\])|(([a-zA-Z\\-0-9]+\\.){2,}))$';
71.        RegExp regex = new RegExp(pattern);
72.        if (!regex.hasMatch(value)) {
73.            return "Email tidak valid";
74.        }
75.        return null;
76.    },
77. );
78. }
79.
80. //Membuat Textbox password
81. Widget _passwordTextField() {
82.     return TextFormField(
83.         decoration: InputDecoration(labelText: "Password"),
84.         keyboardType: TextInputType.text,
85.         obscureText: true,
86.         controller: _passwordTextboxController,
87.         validator: (value){
88.             //jika karakter yang dimasukkan kurang dari 6 karakter
89.             if(value.length < 6){
90.                 return "Password harus diisi minimal 6 karakter";
91.             }
92.             return null;
93.         },
94.     );
95. }
96.
97. //membuat textbox Konfirmasi Password
98. Widget _passwordKonfirmasiTextField() {
99.     return TextFormField(
100.         decoration: InputDecoration(labelText: "Konfirmasi Password"),
101.         keyboardType: TextInputType.text,
102.         obscureText: true,
103.         validator: (value){
104.             //jika inputan tidak sama dengan password
105.             if(value != _passwordTextboxController.text) {
106.                 return "Konfirmasi Password tidak sama";
107.             }
108.             return null;
109.         },
110.     );
111. }
112.
113. //Membuat Tombol Registrasi
114. Widget _buttonRegistrasi() {
115.     return RaisedButton(
116.         child: Text("Registrasi"),
117.         onPressed: (){
118.             var validate = _formKey.currentState.validate();
119.         });
120.     }
121. }

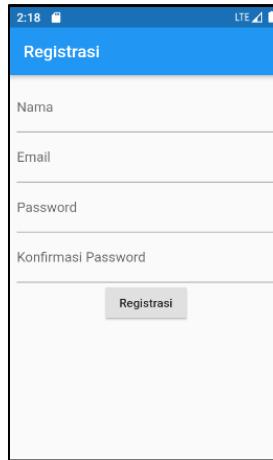
```

Untuk mencoba halaman **registrasi_page**, buka file **main.dart** kemudian ubah kode menjadi seperti berikut ini

```

1. import 'package:flutter/material.dart';
2. import 'package:tokokita/ui/registrasi_page.dart';
3.
4. void main() {
5.   runApp(MyApp());
6. }
7.
8. class MyApp extends StatefulWidget {
9.   @override
10.  _MyAppState createState() => _MyAppState();
11. }
12.
13. class _MyAppState extends State<MyApp> {
14.
15.   @override
16.   Widget build(BuildContext context) {
17.     return MaterialApp(
18.       title: 'Toko Kita',
19.       debugShowCheckedModeBanner: false,
20.       home: RegistrasiPage(),
21.     );
22.   }
23. }
```

Dan hasilnya seperti berikut



b) Login

Buat sebuah file dengan nama **login_page.dart** pada folder **ui** dengan kode berikut

```

1. import 'package:flutter/material.dart';
2. import 'package:tokokita/ui/registrasi_page.dart';
3.
4. class LoginPage extends StatefulWidget {
5.   @override
6.   _LoginPageState createState() => _LoginPageState();
7. }
8.
```

```

9. class _LoginPageState extends State<LoginPage> {
10.   final _formKey = GlobalKey<FormState>();
11.   bool _isLoading = false;
12.
13.   final _emailTextboxController = TextEditingController();
14.   final _passwordTextboxController = TextEditingController();
15.
16.   @override
17.   Widget build(BuildContext context) {
18.     return Scaffold(
19.       appBar: AppBar(title: Text("Login")),
20.       body: SingleChildScrollView(
21.         child: Container(
22.           child: Padding(
23.             padding: const EdgeInsets.all(8.0),
24.             child: Form(
25.               key: _formKey,
26.               child: Column(
27.                 children: [
28.                   _emailTextField(),
29.                   _passwordTextField(),
30.                   _buttonLogin(),
31.                   SizedBox(height: 30, ),
32.                   _menuRegistrasi()
33.                 ],
34.               ),
35.             ),
36.           ),
37.         ),
38.       ),
39.     );
40.   }
41.
42.   //Membuat Textbox email
43.   Widget _emailTextField() {
44.     return TextFormField(
45.       decoration: InputDecoration(labelText: "Email"),
46.       keyboardType: TextInputType.emailAddress,
47.       controller: _emailTextboxController,
48.       validator: (value){
49.         //validasi harus diisi
50.         if(value.isEmpty){
51.           return 'Email harus diisi';
52.         }
53.         return null;
54.       },
55.     );
56.   }
57.
58.   //Membuat Textbox password
59.   Widget _passwordTextField() {
60.     return TextFormField(
61.       decoration: InputDecoration(labelText: "Password"),
62.       keyboardType: TextInputType.text,
63.       obscureText: true,
64.       controller: _passwordTextboxController,
65.       validator: (value){
66.         //jika karakter yang dimasukkan kurang dari 6 karakter
67.         if(value.isEmpty){
68.           return "Password harus diisi";

```

```

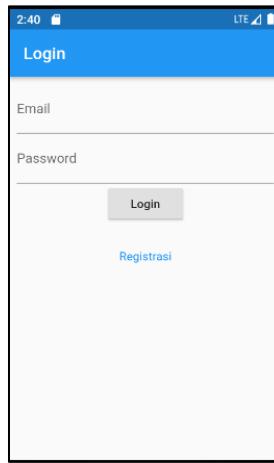
69.         }
70.         return null;
71.     },
72. );
73. }
74.
75. //Membuat Tombol Login
76. Widget _buttonLogin() {
77.     return RaisedButton(
78.         child: Text("Login"),
79.         onPressed: (){
80.             var validate = _formKey.currentState.validate();
81.         });
82. }
83.
84. // Membuat menu untuk membuka halaman registrasi
85. Widget _menuRegistrasi() {
86.     return Container(
87.         child: Center(
88.             child: InkWell(
89.                 child: Text("Registrasi",style: TextStyle(color: Colors.blue),),
90.                 onTap: () {
91.                     Navigator.push(context, new MaterialPageRoute(builder: (context)=> RegistrasiPage()));
92.                 },
93.             ),
94.         ),
95.     );
96. }
97. }
```

Untuk mencobanya modifikasi file **main.dart** dimana pada bagian home akan memanggil **LoginPage()**

```

1. import 'package:flutter/material.dart';
2. import 'package:tokokita/ui/login_page.dart';
3.
4. void main() {
5.   runApp(MyApp());
6. }
7.
8. class MyApp extends StatefulWidget {
9.   @override
10.  _MyAppState createState() => _MyAppState();
11. }
12.
13. class _MyAppState extends State<MyApp> {
14.
15.   @override
16.   Widget build(BuildContext context) {
17.     return MaterialApp(
18.       title: 'Toko Kita',
19.       debugShowCheckedModeBanner: false,
20.       home: LoginPage(),
21.     );
22.   }
23. }
```

Pada saat dijalankan akan terdapat link untuk membuka halaman registrasi pada bagian bawah form



c) Form Produk

Buat sebuah file dengan nama **produk_form.dart** pada folder **ui** dengan kode berikut

```
1. import 'package:flutter/material.dart';
2. import 'package:tokokita/model/produk.dart';
3.
4. class ProdukForm extends StatefulWidget {
5.   Produk produk;
6.   ProdukForm({this.produk});
7.   @override
8.   _ProdukFormState createState() => _ProdukFormState();
9. }
10.
11. class _ProdukFormState extends State<ProdukForm> {
12.   final GlobalKey<FormState> _formKey = GlobalKey<FormState>();
13.   bool _isLoading = false;
14.   String judul = "TAMBAH PRODUK";
15.   String tombolSubmit = "SIMPAN";
16.
17.   final TextEditingController _kodeProdukTextboxController = TextEditingController();
18.   final TextEditingController _namaProdukTextboxController = TextEditingController();
19.   final TextEditingController _hargaProdukTextboxController = TextEditingController();
20.
21.   @override
22.   void initState() {
23.     // TODO: implement initState
24.     super.initState();
25.     isUpdate();
26.   }
27.
28.   isUpdate(){
29.     if(widget.produk!=null){
30.       setState(() {
31.         judul = "UBAH PRODUK";
32.         tombolSubmit = "UBAH";
```

```

33.         _kodeProdukTextboxController.text = widget.produk.kodeProduk;
34.         _namaProdukTextboxController.text = widget.produk.namaProduk;
35.         _hargaProdukTextboxController.text = widget.produk.hargaProduk.toString();
36.     });
37. }else{
38.     judul = "TAMBAH PRODUK";
39.     tombolSubmit = "SIMPAN";
40. }
41. }
42.
43. @override
44. Widget build(BuildContext context) {
45.     return Scaffold(
46.         appBar: AppBar(title: Text(judul)),
47.         body: SingleChildScrollView(
48.             child: Container(
49.                 child: Padding(
50.                     padding: const EdgeInsets.all(8.0),
51.                     child: Form(
52.                         key: _formKey,
53.                         child: Column(
54.                             children: [
55.                                 _kodeProdukTextField(),
56.                                 _namaProdukTextField(),
57.                                 _hargaProdukTextField(),
58.                                 _buttonSubmit()
59.                             ],
60.                         ),
61.                     ),
62.                 ),
63.             ),
64.         ),
65.     );
66. }
67.
68. //Membuat Textbox Kode Produk
69. Widget _kodeProdukTextField() {
70.     return TextFormField(
71.         decoration: InputDecoration(labelText: "Kode Produk"),
72.         keyboardType: TextInputType.text,
73.         controller: _kodeProdukTextboxController,
74.         validator: (value) {
75.             if (value.isEmpty) {
76.                 return "Kode Produk harus diisi";
77.             }
78.             return null;
79.         },
80.     );
81. }
82.
83. //Membuat Textbox Nama Produk
84. Widget _namaProdukTextField() {
85.     return TextFormField(
86.         decoration: InputDecoration(labelText: "Nama Produk"),
87.         keyboardType: TextInputType.text,
88.         controller: _namaProdukTextboxController,
89.         validator: (value) {
90.             if (value.isEmpty) {
91.                 return "Nama Produk harus diisi";
92.             }

```

```

93.         return null;
94.     },
95. );
96. }
97.
98. //Membuat Textbox Harga Produk
99. Widget _hargaProdukTextField() {
100.     return TextFormField(
101.         decoration: InputDecoration(labelText: "Harga"),
102.         keyboardType: TextInputType.number,
103.         controller: _hargaProdukTextboxController,
104.         validator: (value) {
105.             if (value.isEmpty) {
106.                 return "Harga harus diisi";
107.             }
108.             return null;
109.         },
110.     );
111. }
112.
113. //Membuat Tombol Simpan/Ubah
114. Widget _buttonSubmit() {
115.     return RaisedButton(
116.         child: Text(tombolSubmit),
117.         onPressed: () {
118.             var validate = _formKey.currentState.validate();
119.         });
120.     }
121. }

```

d) Detail Produk

Buat sebuah file dengan nama **produk_detail.dart** pada folder **ui** dengan kode berikut

```

1. import 'package:flutter/material.dart';
2. import 'package:tokokita/model/produk.dart';
3. import 'package:tokokita/ui/produk_form.dart';
4.
5. class ProdukDetail extends StatefulWidget {
6.     Produk produk;
7.     ProdukDetail({this.produk});
8.     @override
9.     _ProdukDetailState createState() => _ProdukDetailState();
10. }
11.
12. class _ProdukDetailState extends State<ProdukDetail> {
13.     @override
14.     Widget build(BuildContext context) {
15.         return Scaffold(
16.             appBar: AppBar(
17.                 title: Text('Detail Produk'),
18.             ),
19.             body: Center(
20.                 child: Column(
21.                     children: [
22.                         Text(
23.                             "Kode : ${widget.produk.kodeProduk}",

```

```

24.             style: TextStyle(fontSize: 20.0),
25.         ),
26.         Text(
27.             "Nama : ${widget.produk.namaProduk}",
28.             style: TextStyle(fontSize: 18.0),
29.         ),
30.         Text(
31.             "Harga : Rp. ${widget.produk.hargaProduk.toString()}",
32.             style: TextStyle(fontSize: 18.0),
33.         ),
34.         _tombolHapusEdit()
35.     ],
36. ),
37. ),
38. );
39. }
40.
41. Widget _tombolHapusEdit() {
42.     return Row(
43.         mainAxisAlignment: MainAxisAlignment.min,
44.         children: [
45.             //Tombol Edit
46.             RaisedButton(
47.                 child: Text("EDIT"), color: Colors.green, onPressed: () {
48.                     Navigator.push(
49.                         context,
50.                         new MaterialPageRoute(
51.                             builder: (context) => ProdukForm(produk: widget.produk,)));
52.                 }),
53.             //Tombol Hapus
54.             RaisedButton(
55.                 child: Text("DELETE"), color: Colors.red, onPressed: ()=>confirmHapus(),
56.             ],
57.         );
58.     }
59.
60.     void confirmHapus() {
61.         AlertDialog alertDialog = new AlertDialog(
62.             content: Text("Yakin ingin menghapus data ini?"),
63.             actions: [
64.                 //tombol hapus
65.                 RaisedButton(
66.                     child: Text("Ya"),
67.                     color: Colors.green,
68.                     onPressed: (){},),
69.                 ),
70.                 //tombol batal
71.                 RaisedButton(
72.                     child: Text("Batal"),
73.                     color: Colors.red,
74.                     onPressed: ()=>Navigator.pop(context),
75.                 )
76.             ],
77.         );
78.
79.         showDialog(context: context, child: alertDialog);
80.     }
81. }
```

e) Tampil List Produk

Buat sebuah file dengan nama **produk_page.dart** pada folder **ui** dengan kode berikut

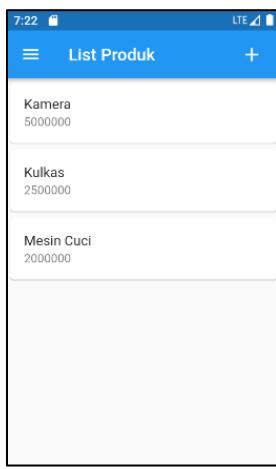
```
1. import 'package:flutter/cupertino.dart';
2. import 'package:flutter/material.dart';
3. import 'package:tokokita/model/produk.dart';
4. import 'package:tokokita/ui/produk_detail.dart';
5. import 'package:tokokita/ui/produk_form.dart';
6.
7. class ProdukPage extends StatefulWidget {
8.   @override
9.   _ProdukPageState createState() => _ProdukPageState();
10. }
11.
12. class _ProdukPageState extends State<ProdukPage> {
13.   @override
14.   Widget build(BuildContext context) {
15.     return Scaffold(
16.       appBar: AppBar(
17.         title: Text('List Produk'),
18.         actions: [
19.           Padding(
20.             padding: EdgeInsets.only(right: 20.0),
21.             child: GestureDetector(
22.               child: Icon(Icons.add, size: 26.0),
23.               onTap: () async {
24.                 Navigator.push(
25.                   context,
26.                   new MaterialPageRoute(
27.                     builder: (context) => ProdukForm()));
28.               },
29.             )),
30.           ],
31.         ),
32.         drawer: Drawer(
33.           child: ListView(
34.             children: [
35.               ListTile(
36.                 title: Text('Logout'),
37.                 trailing: Icon(Icons.logout),
38.                 onTap: () async {},
39.               )
40.             ],
41.           ),
42.         ),
43.         body: ListView(
44.           children: [
45.             ItemProduk(produk: Produk(id: 1, kodeProduk: 'A001', namaProduk: 'Kamera', hargaProduk: 5000000)),
46.             ItemProduk(produk: Produk(id: 2, kodeProduk: 'A002', namaProduk: 'Kulkas', hargaProduk: 2500000)),
47.             ItemProduk(produk: Produk(id: 3, kodeProduk: 'A003', namaProduk: 'Mesin Cuci', hargaProduk: 2000000)),
48.           ],
49.         ),
50.       );
51.     }
52. }
```

```
53.  
54. class ItemProduk extends StatelessWidget {  
55.   final Produk produk;  
56.  
57.   ItemProduk({this.produk});  
58.  
59.   @override  
60.   Widget build(BuildContext context) {  
61.     return Container(  
62.       child: GestureDetector(  
63.         onTap: () {  
64.           Navigator.push(  
65.             context,  
66.             new MaterialPageRoute(  
67.               builder: (context) => ProdukDetail(produk: produk,));  
68.         },  
69.         child: Card(  
70.           child: ListTile(  
71.             title: Text(produk.namaProduk),  
72.             subtitle: Text(produk.hargaProduk.toString()),  
73.           ),  
74.           ),  
75.         ),  
76.       );  
77.   }  
78. }
```

Untuk mencobanya modifikasi file **main.dart** menjadi seperti berikut

```
1. import 'package:flutter/material.dart';  
2. import 'package:tokokita/ui/produk_page.dart';  
3.  
4. void main() {  
5.   runApp(MyApp());  
6. }  
7.  
8. class MyApp extends StatefulWidget {  
9.   @override  
10.  _MyAppState createState() => _MyAppState();  
11. }  
12.  
13. class _MyAppState extends State<MyApp> {  
14.  
15.   @override  
16.   Widget build(BuildContext context) {  
17.     return MaterialApp(  
18.       title: 'Toko Kita',  
19.       debugShowCheckedModeBanner: false,  
20.       home: ProdukPage(),  
21.     );  
22.   }  
23. }
```

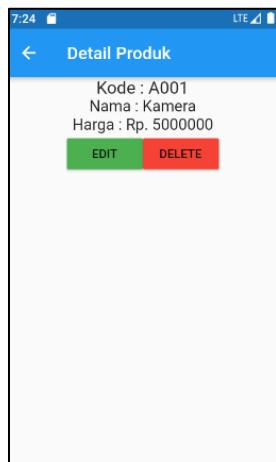
Maka tampilannya akan menjadi seperti berikut



Pada saat tombol tambah diklik maka akan muncul form produk seperti berikut



Jika salah satu data produk diklik maka akan muncul detail produk



Ketika tombol **EDIT** diklik maka akan muncul form produk untuk mengubah data produk

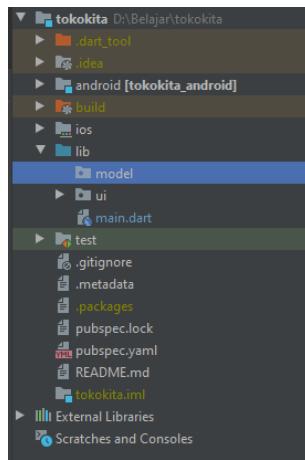


Pada materi selanjutnya akan ada modifikasi pada tampil produk agar dapat menampilkan data dari Rest API serta modifikasi pada Form Produk sehingga dapat berfungsi untuk menyimpan ataupun mengubah data pada Rest API

PERTEMUAN 6

5. Membuat Model

Buat folder/package dengan nama **model** pada folder **lib**



a) Login

Buat sebuah file dengan nama **login.dart** pada folder **model**. Kemudian masukkan kode berikut

```
1. class Login{
2.   int code;
3.   bool status;
4.   String token;
5.   int userID;
6.   String userEmail;
7.
8.   Login({this.code, this.status, this.token, this.userID, this.userEmail});
9.
10. factory Login.fromJson(Map<String, dynamic> obj) {
11.   return Login(
12.     code: obj['code'],
13.     status: obj['status'],
14.     token: obj['data']['token'],
15.     userID: obj['data']['user']['id'],
16.     userEmail: obj['data']['user']['email']
17.   );
18. }
19. }
```

b) Registrasi

Buat sebuah file dengan nama **registrasi.dart** pada folder **model**. Kemudian masukkan kode berikut

```
1. class Registrasi {
2.   int code;
3.   bool status;
4.   String data;
5.
6.   Registrasi({this.code, this.status, this.data});
7.
8.   factory Registrasi.fromJson(Map<String, dynamic> obj) {
9.     return Registrasi(
10.       code: obj['code'],
11.       status: obj['status'],
12.       data: obj['data']
13.     );
14.   }
15. }
```

c) Produk

Buat sebuah file dengan nama **produk.dart** pada folder **model**. Kemudian ketikkan kode berikut

```
1. class Produk{
2.   int id;
3.   String kodeProduk;
4.   String namaProduk;
5.   int hargaProduk;
6.
7.   Produk({this.id, this.kodeProduk, this.namaProduk, this.hargaProduk});
8.
9.   factory Produk.fromJson(Map<String, dynamic> obj) {
10.     return Produk(
11.       id: obj['id'],
12.       kodeProduk: obj['kode_produk'],
13.       namaProduk: obj['nama_produk'],
14.       hargaProduk: obj['harga']
15.     );
16.   }
17. }
```

PERTEMUAN 7

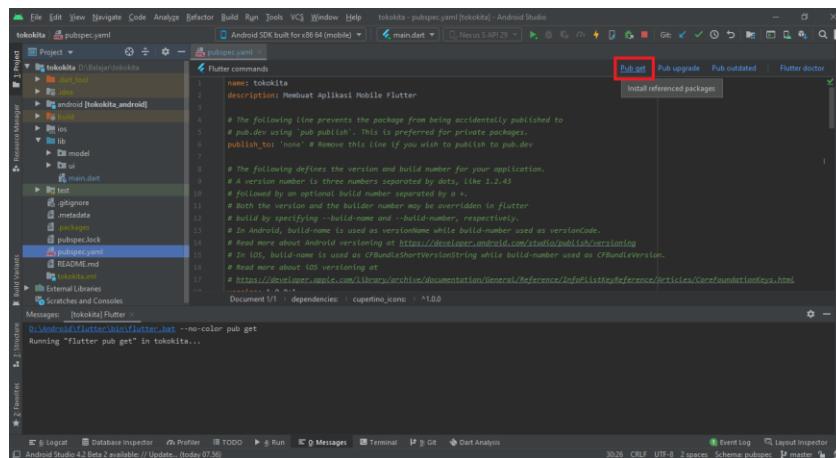
6. Membuat Helper Modul

a) Menambahkan dependencies

Dalam pembuatan aplikasi ini dibutuhkan dependensi untuk menerima dan mengirim request ke Rest API (**http**) dan dependensi untuk menyimpan token (**shared_preferences**). Pastikan komputer atau laptop terhubung ke internet, buka file **pubspec.yaml** kemudian tambahkan kode berikut

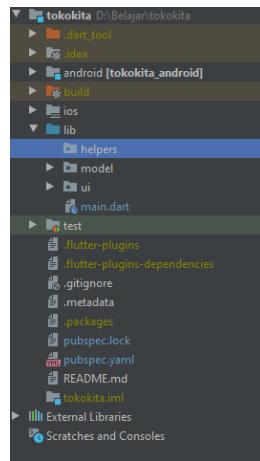
```
18. version: 1.0.0+1
19.
20. environment:
21.   sdk: ">=2.7.0 <3.0.0"
22.
23. dependencies:
24.   flutter:
25.     sdk: flutter
26.
27.
28.   # The following adds the Cupertino Icons font to your application.
29.   # Use with the CupertinoIcons class for iOS style icons.
30.   cupertino_icons: ^1.0.0
31.   http: ^0.12.2
32.   shared_preferences: ^0.5.12+2
33.
34. dev_dependencies:
35.   flutter_test:
36.     sdk: flutter
37.
```

Kemudian klik **Pub get** dan tunggu hingga proses selesai. Projek akan mengunduh dependensi **http** dan **shared_preferences**



b) Membuat Class Token

Buat sebuah folder/packages dengan nama **helpers**



Kemudian pada folder **helpers** buat sebuah file dengan nama **user_info.dart** dan masukkan kode berikut

```
1. import 'package:shared_preferences/shared_preferences.dart';
2.
3. class UserInfo {
4.   Future setToken(String value) async {
5.     final SharedPreferences pref = await SharedPreferences.getInstance();
6.     return pref.setString("token", value);
7.   }
8.
9.   Future<String> getToken() async {
10.    final SharedPreferences pref = await SharedPreferences.getInstance();
11.    return pref.getString("token");
12.  }
13.
14.  Future setUserID(int value) async {
15.    final SharedPreferences pref = await SharedPreferences.getInstance();
16.    return pref.setInt("userID", value);
17.  }
18.
19.  Future<int> getUserID() async {
20.    final SharedPreferences pref = await SharedPreferences.getInstance();
21.    return pref.getInt("userID");
22.  }
23.
24.  Future logout() async {
25.    final SharedPreferences pref = await SharedPreferences.getInstance();
26.    pref.clear();
27.  }
28. }
```

c) Http request

(1) Membuat Modul Error Handling

Buat sebuah file pada folder **helpers** dengan nama **app_exception.dart** kemudian ketikkan kode berikut

```
1. class AppException implements Exception {
2.   final _message;
3.   final _prefix;
4.
5.   AppException([this._message, this._prefix]);
6.
7.   String toString() {
8.     return "${_prefix}${_message}";
9.   }
10. }
11.
12. class FetchDataException extends AppException {
13.   FetchDataException([String message])
14.     : super(message, "Error During Communication: ");
15. }
16.
17. class BadRequestException extends AppException {
18.   BadRequestException([message]) : super(message, "Invalid Request: ");
19. }
20.
21. class UnauthorisedException extends AppException {
22.   UnauthorisedException([message]) : super(message, "Unauthorised: ");
23. }
24.
25. class UnprocessableEntityException extends AppException {
26.   UnprocessableEntityException([message])
27.     : super(message, "Unprocessable Entity: ");
28. }
29.
30. class InvalidInputException extends AppException {
31.   InvalidInputException([String message]) : super(message, "Invalid Input: ");
32. }
```

Pada file ini berfungsi sebagai penanganan jika terjadi error saat melakukan permintaan atau pengiriman ke Rest API

(2) Membuat Modul Http Request

Agar dapat mengirim atau menerima permintaan ke Rest API, dibuat sebuah fungsi baik itu method POST, GET dan DELETE.

Buat sebuah file di dalam folder **helpers** dengan nama **api.dart** dan masukkan kode berikut

```
1. import 'dart:io';
2.
3. import 'package:http/http.dart' as http;
```

```

4. import 'package:tokokita/helpers/user_info.dart';
5. import 'app_exception.dart';
6.
7. class Api{
8.   Future<dynamic> post(String url, dynamic data) async {
9.     var token = await UserInfo().getToken();
10.    var responseJson;
11.    try {
12.      final response = await http.post(url, body: data, headers: {
13.        HttpHeaders.authorizationHeader: "Bearer $token"
14.      });
15.      responseJson = _returnResponse(response);
16.    } on SocketException {
17.      throw FetchDataException('No Internet connection');
18.    }
19.    return responseJson;
20.  }
21.
22. Future<dynamic> get(String url) async {
23.   var token = await UserInfo().getToken();
24.   var responseJson;
25.   try {
26.     final response = await http.get(url, headers: {
27.       HttpHeaders.authorizationHeader: "Bearer $token"
28.     });
29.     responseJson = _returnResponse(response);
30.   } on SocketException {
31.     throw FetchDataException('No Internet connection');
32.   }
33.   return responseJson;
34. }
35.
36. Future<dynamic> delete(String url) async {
37.   var token = await UserInfo().getToken();
38.   var responseJson;
39.   try {
40.     final response = await http.delete(url, headers: {
41.       HttpHeaders.authorizationHeader: "Bearer $token"
42.     });
43.     responseJson = _returnResponse(response);
44.   } on SocketException {
45.     throw FetchDataException('No Internet connection');
46.   }
47.   return responseJson;
48. }
49.
50. dynamic _returnResponse(http.Response response) {
51.   switch (response.statusCode) {
52.     case 200:
53.       return response;
54.     case 400:
55.       throw BadRequestException(response.body.toString());
56.     case 401:
57.     case 403:
58.       throw UnauthorisedException(response.body.toString());
59.     case 422:
60.       throw InvalidInputException(response.body.toString());
61.     case 500:
62.     default:
63.       throw FetchDataException(

```

```

64.          'Error occured while Communication with Server with StatusCode : ${response
65.              .statusCode}');
66.      }
67.  }

```

(3) Membuat List API url

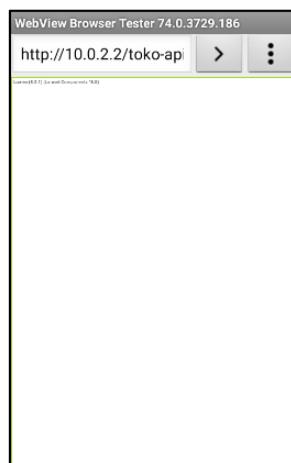
Buat sebuah file di dalam folder **helpers** dengan nama **api_url.dart**, dimana fungsi dari file ini adalah menyimpan alamat-alamat API yang telah dibuat sebelumnya (Endpoint dari API Spec)

```

1. class ApiUrl {
2.   static const String baseUrl = 'http://10.0.2.2/toko-api/public';
3.
4.   static const String registrasi = baseUrl + '/registrasi';
5.   static const String login = baseUrl + '/login';
6.   static const String listProduk = baseUrl + '/produk';
7.   static const String createProduk = baseUrl + '/produk';
8.
9.   static String updateProduk(int id) {
10.     return baseUrl + '/produk/' + id.toString() + '/update';
11.   }
12.
13. static String showProduk(int id) {
14.   return baseUrl + '/produk/' + id.toString();
15. }
16.
17. static String deleteProduk(int id) {
18.   return baseUrl + '/produk/' + id.toString();
19. }
20. }

```

Pada baris kedua (baseUrl) merupakan alamat IP dari Rest API, untuk memeriksa apakah terhubung dengan emulator atau tidak, dapat dilakukan dengan memasukkan alamat tersebut pada browser yang ada pada emulator



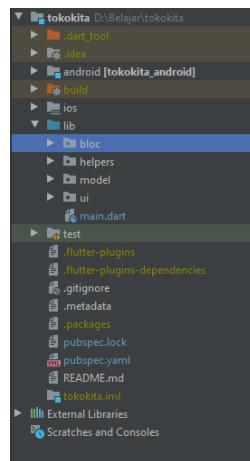
PERTEMUAN 8

UJIAN TENGAH SEMESTER

PERTEMUAN 9

7. Membuat Bloc

Buat sebuah folder/package bernama **bloc**. Di dalam folder ini berisisi file-file yang berfungsi sebagai controller baik itu untuk melakukan proses login, registrasi dan lain-lain.



a) Registrasi

Buat sebuah file dengan nama **registrasi_bloc.dart** pada folder **bloc**. Kemudian masukkan kode berikut

```
1. import 'dart:convert';
2.
3. import 'package:tokokita/helpers/api.dart';
4. import 'package:tokokita/helpers/api_url.dart';
5. import 'package:tokokita/model/registrasi.dart';
6.
7. class RegistrasiBloc{
8.     static Future<Registrasi> registrasi({String nama, String email, String password}) as
9.     sync {
10.         String apiUrl = ApiUrl.registrasi;
11.
12.         var body = {
13.             "nama": nama,
14.             "email":email,
15.             "password":password
16.         };
17.
18.         var response = await Api().post(apiUrl, body);
19.         var jsonObj = json.decode(response.body);
20.         return Registrasi.fromJson(jsonObj);
21.     }
22. }
```

b) Login

Buat sebuah file dengan nama **login_bloc.dart** pada folder **bloc**. Kemudian masukkan kode berikut

```
1. import 'dart:convert';
2.
3. import 'package:tokokita/helpers/api.dart';
4. import 'package:tokokita/helpers/api_url.dart';
5. import 'package:tokokita/model/login.dart';
6.
7. class LoginBloc{
8.     static Future<Login> login({String email, String password}) async {
9.         String apiUrl = ApiUrl.login;
10.        var body = {"email": email, "password": password};
11.        var response = await Api().post(apiUrl, body);
12.        var jsonObj = json.decode(response.body);
13.        return Login.fromJson(jsonObj);
14.    }
15. }
```

c) Logout

Buat sebuah file dengan nama **logout_bloc.dart** pada folder **bloc**. Kemudian masukkan kode berikut

```
1. import 'package:tokokita/helpers/user_info.dart';
2.
3. class LogoutBloc{
4.     static Future logout() async {
5.         await UserInfo().logout();
6.     }
7. }
```

d) Produk

Pada bagian ini akan dibuat beberapa fungsi untuk mengambil, mengubah dan menghapus data produk dari Rest API. Buat sebuah file dengan nama **produk_bloc.dart** pada folder **bloc** kemudian masukkan kode berikut

```
1. import 'dart:convert';
2.
3. import 'package:tokokita/helpers/api.dart';
4. import 'package:tokokita/helpers/api_url.dart';
5. import 'package:tokokita/model/produk.dart';
6.
7. class ProdukBloc{
8.     static Future<List<Produk>> getProduks() async {
9.         String apiUrl = ApiUrl.listProduk;
10.        var response = await Api().get(apiUrl);
```

```

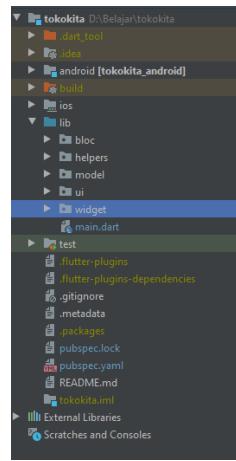
11.     var jsonObj = json.decode(response.body);
12.     List<dynamic> listProduk = (jsonObj as Map<String, dynamic>)['data'];
13.     List<Produk> produks = [];
14.     for(int i=0; i < listProduk.length; i++){
15.         produks.add(Produk.fromJson(listProduk[i]));
16.     }
17.     return produks;
18. }
19.
20. static Future addProduk({Produk produk}) async {
21.     String apiUrl = ApiUrl.createProduk;
22.
23.     var body = {
24.         "kode_produk": produk.kodeProduk,
25.         "nama_produk":produk.namaProduk,
26.         "harga":produk.hargaProduk.toString()
27.     };
28.
29.     var response = await Api().post(apiUrl, body);
30.     var jsonObj = json.decode(response.body);
31.     return jsonObj['status'];
32. }
33.
34. static Future<bool> updateProduk({Produk produk}) async {
35.     String apiUrl = ApiUrl.updateProduk(produk.id);
36.
37.     var body = {
38.         "kode_produk": produk.kodeProduk,
39.         "nama_produk":produk.namaProduk,
40.         "harga":produk.hargaProduk.toString()
41.     };
42.     print("Body : $body");
43.     var response = await Api().post(apiUrl, body);
44.     var jsonObj = json.decode(response.body);
45.     return jsonObj['data'];
46. }
47.
48. static Future<bool> deleteProduk({int id}) async {
49.     String apiUrl = ApiUrl.deleteProduk(id);
50.
51.     var response = await Api().delete(apiUrl);
52.     var jsonObj = json.decode(response.body);
53.     return (jsonObj as Map<String, dynamic>)['data'];
54. }
55. }
```

PERTEMUAN 10 - 12

8. Menyatukan Fungsionalitas

a) Membuat Common Dialog Widget

Pada bagian ini akan dibuat dua buah dialog yang nantinya akan digunakan pada tampilan. Buat sebuah folder/package dengan nama **widget**



Kemudian buat sebuah file dengan nama **success_dialog.dart** dengan kode

```
1. import 'package:flutter/material.dart';
2.
3. class Consts {
4.   Consts._();
5.
6.   static const double padding = 16.0;
7.   static const double avatarRadius = 66.0;
8. }
9.
10. class SuccessDialog extends StatelessWidget {
11.   final String description;
12.   final VoidCallback onClick;
13.
14.   SuccessDialog({this.description, this.onClick});
15.
16.   @override
17.   Widget build(BuildContext context) {
18.     return Dialog(
19.       shape: RoundedRectangleBorder(
20.         borderRadius: BorderRadius.circular(Consts.padding)),
21.       elevation: 0.0,
22.       backgroundColor: Colors.transparent,
23.       child: dialogContent(context),
24.     );
25.   }
26.
27.   dialogContent(BuildContext context) {
28.     return Container(
```

```

29.     padding: EdgeInsets.only(
30.         top: Consts.padding,
31.         bottom: Consts.padding,
32.         left: Consts.padding,
33.         right: Consts.padding,
34.     ),
35.     margin: EdgeInsets.only(top: Consts.avatarRadius),
36.     decoration: new BoxDecoration(
37.         color: Colors.white,
38.         shape: BoxShape.rectangle,
39.         borderRadius: BorderRadius.circular(Consts.padding),
40.         boxShadow: [
41.             BoxShadow(
42.                 color: Colors.black26,
43.                 blurRadius: 10.0,
44.                 offset: const Offset(0.0, 10.0),
45.             ),
46.         ],
47.     ),
48.     child: Column(
49.         mainAxisAlignment: MainAxisAlignment.min,
50.         children: [
51.             Text(
52.                 "SUKSES",
53.                 style: TextStyle(
54.                     fontSize: 24.0,
55.                     fontWeight: FontWeight.w700,
56.                     color: Colors.green),
57.             ),
58.             SizedBox(height: 16.0),
59.             Text(
60.                 description,
61.                 textAlign: TextAlign.center,
62.                 style: TextStyle(
63.                     fontSize: 16.0,
64.                 ),
65.             ),
66.             SizedBox(height: 24.0),
67.             Align(
68.                 alignment: Alignment.bottomRight,
69.                 child: FlatButton(
70.                     onPressed: () {
71.                         Navigator.of(context).pop(); // To close the dialog
72.                         okClick();
73.                     },
74.                     child: Text("OK"),
75.                 ),
76.             ),
77.         ],
78.     ),
79. );
80. }
81. }

```

Kemudian buat file dengan nama **warning_dialog.dart** dengan kode

```

1. import 'package:flutter/material.dart';
2.
3. class Consts {

```

```

4.     Consts._();
5.
6.     static const double padding = 16.0;
7.     static const double avatarRadius = 66.0;
8.   }
9.
10.  class WarningDialog extends StatelessWidget {
11.    final String description;
12.    final VoidCallback okClick;
13.
14.    WarningDialog({this.description, this.okClick});
15.
16.    @override
17.    Widget build(BuildContext context) {
18.      return Dialog(
19.        shape: RoundedRectangleBorder(
20.          borderRadius: BorderRadius.circular(Consts.padding)),
21.        elevation: 0.0,
22.        backgroundColor: Colors.transparent,
23.        child: dialogContent(context),
24.      );
25.    }
26.
27.    dialogContent(BuildContext context) {
28.      return Container(
29.        padding: EdgeInsets.only(
30.          top: Consts.padding,
31.          bottom: Consts.padding,
32.          left: Consts.padding,
33.          right: Consts.padding,
34.        ),
35.        margin: EdgeInsets.only(top: Consts.avatarRadius),
36.        decoration: new BoxDecoration(
37.          color: Colors.white,
38.          shape: BoxShape.rectangle,
39.          borderRadius: BorderRadius.circular(Consts.padding),
40.          boxShadow: [
41.            BoxShadow(
42.              color: Colors.black26,
43.              blurRadius: 10.0,
44.              offset: const Offset(0.0, 10.0),
45.            ),
46.          ],
47.        ),
48.        child: Column(
49.          mainAxisSize: MainAxisSize.min,
50.          children: [
51.            Text(
52.              "GAGAL",
53.              style: TextStyle(
54.                fontSize: 24.0, fontWeight: FontWeight.w700, color: Colors.red),
55.            ),
56.            SizedBox(height: 16.0),
57.            Text(
58.              description,
59.              textAlign: TextAlign.center,
60.              style: TextStyle(
61.                fontSize: 16.0,
62.              ),
63.            ),

```

```

64.         SizedBox(height: 24.0),
65.         Align(
66.             alignment: Alignment.bottomRight,
67.             child: FlatButton(
68.                 onPressed: () {
69.                     Navigator.of(context).pop(); // To close the dialog
70.                     okClick();
71.                 },
72.                 child: Text("OK"),
73.             ),
74.         ),
75.     ],
76. ),
77. );
78. }
79. 
```

b) Modifikasi main.dart

Buka kembali file **main.dart** kita akan memodifikasi file tersebut dengan kondisi jika belum login maka akan membuka halaman login, namun jika sudah login maka akan membuka halaman list produk

```

1. import 'package:flutter/material.dart';
2. import 'package:tokokita/helpers/user_info.dart';
3. import 'package:tokokita/ui/login_page.dart';
4. import 'package:tokokita/ui/produk_page.dart';
5.
6. void main() {
7.   runApp(MyApp());
8. }
9.
10. class MyApp extends StatefulWidget {
11.   @override
12.   _MyAppState createState() => _MyAppState();
13. }
14.
15. class _MyAppState extends State<MyApp> {
16.   Widget page = CircularProgressIndicator();
17.
18.   @override
19.   void initState() {
20.     super.initState();
21.     isLogin();
22.   }
23.
24.   void isLogin() async {
25.     var token = await UserInfo().getToken();
26.     if(token!=null){
27.       setState(() {
28.         page = ProdukPage();
29.       });
30.     }else{
31.       setState(() {
32.         page = LoginPage();
33.       });
34.     }
35.   }
36.
37.   void okClick() {
38.     setState(() {
39.       page = ProdukPage();
40.     });
41.   }
42.
43.   void cancelClick() {
44.     Navigator.pop(context);
45.   }
46.
47.   void showOkDialog() {
48.     showDialog(
49.       context: context,
50.       builder: (context) {
51.         return AlertDialog(
52.           title: Text("Success"),
53.           content: Text("Data berhasil diupdate"),
54.           actions: [
55.             TextButton(
56.               onPressed: cancelClick,
57.               child: Text("Cancel"),
58.             ),
59.             TextButton(
60.               onPressed: okClick,
61.               child: Text("OK"),
62.             ),
63.           ],
64.         );
65.       },
66.     );
67.   }
68.
69.   void showCancelDialog() {
70.     showDialog(
71.       context: context,
72.       builder: (context) {
73.         return AlertDialog(
74.           title: Text("Cancel"),
75.           content: Text("Data gagal diupdate"),
76.           actions: [
77.             TextButton(
78.               onPressed: cancelClick,
79.               child: Text("OK"),
80.             ),
81.           ],
82.         );
83.       },
84.     );
85.   }
86.
87.   void showLogoutDialog() {
88.     showDialog(
89.       context: context,
90.       builder: (context) {
91.         return AlertDialog(
92.           title: Text("Logout"),
93.           content: Text("Anda yakin ingin keluar?"),
94.           actions: [
95.             TextButton(
96.               onPressed: cancelClick,
97.               child: Text("Cancel"),
98.             ),
99.             TextButton(
100.               onPressed: () {
101.                 Navigator.of(context).pop();
102.                 Navigator.pushReplacementNamed(context, '/login');
103.               },
104.               child: Text("Logout"),
105.             ),
106.           ],
107.         );
108.       },
109.     );
110.   }
111.
112.   void okClick() {
113.     setState(() {
114.       page = ProdukPage();
115.     });
116.   }
117.
118.   void cancelClick() {
119.     Navigator.pop(context);
120.   }
121.
122.   void showOkDialog() {
123.     showDialog(
124.       context: context,
125.       builder: (context) {
126.         return AlertDialog(
127.           title: Text("Success"),
128.           content: Text("Data berhasil diupdate"),
129.           actions: [
130.             TextButton(
131.               onPressed: cancelClick,
132.               child: Text("Cancel"),
133.             ),
134.             TextButton(
135.               onPressed: okClick,
136.               child: Text("OK"),
137.             ),
138.           ],
139.         );
140.       },
141.     );
142.   }
143.
144.   void showCancelDialog() {
145.     showDialog(
146.       context: context,
147.       builder: (context) {
148.         return AlertDialog(
149.           title: Text("Cancel"),
150.           content: Text("Data gagal diupdate"),
151.           actions: [
152.             TextButton(
153.               onPressed: cancelClick,
154.               child: Text("OK"),
155.             ),
156.           ],
157.         );
158.       },
159.     );
160.   }
161.
162.   void showLogoutDialog() {
163.     showDialog(
164.       context: context,
165.       builder: (context) {
166.         return AlertDialog(
167.           title: Text("Logout"),
168.           content: Text("Anda yakin ingin keluar?"),
169.           actions: [
170.             TextButton(
171.               onPressed: cancelClick,
172.               child: Text("Cancel"),
173.             ),
174.             TextButton(
175.               onPressed: () {
176.                 Navigator.of(context).pop();
177.                 Navigator.pushReplacementNamed(context, '/login');
178.               },
179.               child: Text("Logout"),
180.             ),
181.           ],
182.         );
183.       },
184.     );
185.   }
186.
187.   void okClick() {
188.     setState(() {
189.       page = ProdukPage();
190.     });
191.   }
192.
193.   void cancelClick() {
194.     Navigator.pop(context);
195.   }
196.
197.   void showOkDialog() {
198.     showDialog(
199.       context: context,
200.       builder: (context) {
201.         return AlertDialog(
202.           title: Text("Success"),
203.           content: Text("Data berhasil diupdate"),
204.           actions: [
205.             TextButton(
206.               onPressed: cancelClick,
207.               child: Text("Cancel"),
208.             ),
209.             TextButton(
210.               onPressed: okClick,
211.               child: Text("OK"),
212.             ),
213.           ],
214.         );
215.       },
216.     );
217.   }
218.
219.   void showCancelDialog() {
220.     showDialog(
221.       context: context,
222.       builder: (context) {
223.         return AlertDialog(
224.           title: Text("Cancel"),
225.           content: Text("Data gagal diupdate"),
226.           actions: [
227.             TextButton(
228.               onPressed: cancelClick,
229.               child: Text("OK"),
230.             ),
231.           ],
232.         );
233.       },
234.     );
235.   }
236.
237.   void showLogoutDialog() {
238.     showDialog(
239.       context: context,
240.       builder: (context) {
241.         return AlertDialog(
242.           title: Text("Logout"),
243.           content: Text("Anda yakin ingin keluar?"),
244.           actions: [
245.             TextButton(
246.               onPressed: cancelClick,
247.               child: Text("Cancel"),
248.             ),
249.             TextButton(
250.               onPressed: () {
251.                 Navigator.of(context).pop();
252.                 Navigator.pushReplacementNamed(context, '/login');
253.               },
254.               child: Text("Logout"),
255.             ),
256.           ],
257.         );
258.       },
259.     );
260.   }
261.
262.   void okClick() {
263.     setState(() {
264.       page = ProdukPage();
265.     });
266.   }
267.
268.   void cancelClick() {
269.     Navigator.pop(context);
270.   }
271.
272.   void showOkDialog() {
273.     showDialog(
274.       context: context,
275.       builder: (context) {
276.         return AlertDialog(
277.           title: Text("Success"),
278.           content: Text("Data berhasil diupdate"),
279.           actions: [
280.             TextButton(
281.               onPressed: cancelClick,
282.               child: Text("Cancel"),
283.             ),
284.             TextButton(
285.               onPressed: okClick,
286.               child: Text("OK"),
287.             ),
288.           ],
289.         );
290.       },
291.     );
292.   }
293.
294.   void showCancelDialog() {
295.     showDialog(
296.       context: context,
297.       builder: (context) {
298.         return AlertDialog(
299.           title: Text("Cancel"),
300.           content: Text("Data gagal diupdate"),
301.           actions: [
302.             TextButton(
303.               onPressed: cancelClick,
304.               child: Text("OK"),
305.             ),
306.           ],
307.         );
308.       },
309.     );
310.   }
311.
312.   void showLogoutDialog() {
313.     showDialog(
314.       context: context,
315.       builder: (context) {
316.         return AlertDialog(
317.           title: Text("Logout"),
318.           content: Text("Anda yakin ingin keluar?"),
319.           actions: [
320.             TextButton(
321.               onPressed: cancelClick,
322.               child: Text("Cancel"),
323.             ),
324.             TextButton(
325.               onPressed: () {
326.                 Navigator.of(context).pop();
327.                 Navigator.pushReplacementNamed(context, '/login');
328.               },
329.               child: Text("Logout"),
330.             ),
331.           ],
332.         );
333.       },
334.     );
335.   }
336.
337.   void okClick() {
338.     setState(() {
339.       page = ProdukPage();
340.     });
341.   }
342.
343.   void cancelClick() {
344.     Navigator.pop(context);
345.   }
346.
347.   void showOkDialog() {
348.     showDialog(
349.       context: context,
350.       builder: (context) {
351.         return AlertDialog(
352.           title: Text("Success"),
353.           content: Text("Data berhasil diupdate"),
354.           actions: [
355.             TextButton(
356.               onPressed: cancelClick,
357.               child: Text("Cancel"),
358.             ),
359.             TextButton(
360.               onPressed: okClick,
361.               child: Text("OK"),
362.             ),
363.           ],
364.         );
365.       },
366.     );
367.   }
368.
369.   void showCancelDialog() {
370.     showDialog(
371.       context: context,
372.       builder: (context) {
373.         return AlertDialog(
374.           title: Text("Cancel"),
375.           content: Text("Data gagal diupdate"),
376.           actions: [
377.             TextButton(
378.               onPressed: cancelClick,
379.               child: Text("OK"),
380.             ),
381.           ],
382.         );
383.       },
384.     );
385.   }
386.
387.   void showLogoutDialog() {
388.     showDialog(
389.       context: context,
390.       builder: (context) {
391.         return AlertDialog(
392.           title: Text("Logout"),
393.           content: Text("Anda yakin ingin keluar?"),
394.           actions: [
395.             TextButton(
396.               onPressed: cancelClick,
397.               child: Text("Cancel"),
398.             ),
399.             TextButton(
400.               onPressed: () {
401.                 Navigator.of(context).pop();
402.                 Navigator.pushReplacementNamed(context, '/login');
403.               },
404.               child: Text("Logout"),
405.             ),
406.           ],
407.         );
408.       },
409.     );
410.   }
411.
412.   void okClick() {
413.     setState(() {
414.       page = ProdukPage();
415.     });
416.   }
417.
418.   void cancelClick() {
419.     Navigator.pop(context);
420.   }
421.
422.   void showOkDialog() {
423.     showDialog(
424.       context: context,
425.       builder: (context) {
426.         return AlertDialog(
427.           title: Text("Success"),
428.           content: Text("Data berhasil diupdate"),
429.           actions: [
430.             TextButton(
431.               onPressed: cancelClick,
432.               child: Text("Cancel"),
433.             ),
434.             TextButton(
435.               onPressed: okClick,
436.               child: Text("OK"),
437.             ),
438.           ],
439.         );
440.       },
441.     );
442.   }
443.
444.   void showCancelDialog() {
445.     showDialog(
446.       context: context,
447.       builder: (context) {
448.         return AlertDialog(
449.           title: Text("Cancel"),
450.           content: Text("Data gagal diupdate"),
451.           actions: [
452.             TextButton(
453.               onPressed: cancelClick,
454.               child: Text("OK"),
455.             ),
456.           ],
457.         );
458.       },
459.     );
460.   }
461.
462.   void showLogoutDialog() {
463.     showDialog(
464.       context: context,
465.       builder: (context) {
466.         return AlertDialog(
467.           title: Text("Logout"),
468.           content: Text("Anda yakin ingin keluar?"),
469.           actions: [
470.             TextButton(
471.               onPressed: cancelClick,
472.               child: Text("Cancel"),
473.             ),
474.             TextButton(
475.               onPressed: () {
476.                 Navigator.of(context).pop();
477.                 Navigator.pushReplacementNamed(context, '/login');
478.               },
479.               child: Text("Logout"),
480.             ),
481.           ],
482.         );
483.       },
484.     );
485.   }
486.
487.   void okClick() {
488.     setState(() {
489.       page = ProdukPage();
490.     });
491.   }
492.
493.   void cancelClick() {
494.     Navigator.pop(context);
495.   }
496.
497.   void showOkDialog() {
498.     showDialog(
499.       context: context,
500.       builder: (context) {
501.         return AlertDialog(
502.           title: Text("Success"),
503.           content: Text("Data berhasil diupdate"),
504.           actions: [
505.             TextButton(
506.               onPressed: cancelClick,
507.               child: Text("Cancel"),
508.             ),
509.             TextButton(
510.               onPressed: okClick,
511.               child: Text("OK"),
512.             ),
513.           ],
514.         );
515.       },
516.     );
517.   }
518.
519.   void showCancelDialog() {
520.     showDialog(
521.       context: context,
522.       builder: (context) {
523.         return AlertDialog(
524.           title: Text("Cancel"),
525.           content: Text("Data gagal diupdate"),
526.           actions: [
527.             TextButton(
528.               onPressed: cancelClick,
529.               child: Text("OK"),
530.             ),
531.           ],
532.         );
533.       },
534.     );
535.   }
536.
537.   void showLogoutDialog() {
538.     showDialog(
539.       context: context,
540.       builder: (context) {
541.         return AlertDialog(
542.           title: Text("Logout"),
543.           content: Text("Anda yakin ingin keluar?"),
544.           actions: [
545.             TextButton(
546.               onPressed: cancelClick,
547.               child: Text("Cancel"),
548.             ),
549.             TextButton(
550.               onPressed: () {
551.                 Navigator.of(context).pop();
552.                 Navigator.pushReplacementNamed(context, '/login');
553.               },
554.               child: Text("Logout"),
555.             ),
556.           ],
557.         );
558.       },
559.     );
560.   }
561.
562.   void okClick() {
563.     setState(() {
564.       page = ProdukPage();
565.     });
566.   }
567.
568.   void cancelClick() {
569.     Navigator.pop(context);
570.   }
571.
572.   void showOkDialog() {
573.     showDialog(
574.       context: context,
575.       builder: (context) {
576.         return AlertDialog(
577.           title: Text("Success"),
578.           content: Text("Data berhasil diupdate"),
579.           actions: [
580.             TextButton(
581.               onPressed: cancelClick,
582.               child: Text("Cancel"),
583.             ),
584.             TextButton(
585.               onPressed: okClick,
586.               child: Text("OK"),
587.             ),
588.           ],
589.         );
590.       },
591.     );
592.   }
593.
594.   void showCancelDialog() {
595.     showDialog(
596.       context: context,
597.       builder: (context) {
598.         return AlertDialog(
599.           title: Text("Cancel"),
600.           content: Text("Data gagal diupdate"),
601.           actions: [
602.             TextButton(
603.               onPressed: cancelClick,
604.               child: Text("OK"),
605.             ),
606.           ],
607.         );
608.       },
609.     );
610.   }
611.
612.   void showLogoutDialog() {
613.     showDialog(
614.       context: context,
615.       builder: (context) {
616.         return AlertDialog(
617.           title: Text("Logout"),
618.           content: Text("Anda yakin ingin keluar?"),
619.           actions: [
620.             TextButton(
621.               onPressed: cancelClick,
622.               child: Text("Cancel"),
623.             ),
624.             TextButton(
625.               onPressed: () {
626.                 Navigator.of(context).pop();
627.                 Navigator.pushReplacementNamed(context, '/login');
628.               },
629.               child: Text("Logout"),
630.             ),
631.           ],
632.         );
633.       },
634.     );
635.   }
636.
637.   void okClick() {
638.     setState(() {
639.       page = ProdukPage();
640.     });
641.   }
642.
643.   void cancelClick() {
644.     Navigator.pop(context);
645.   }
646.
647.   void showOkDialog() {
648.     showDialog(
649.       context: context,
650.       builder: (context) {
651.         return AlertDialog(
652.           title: Text("Success"),
653.           content: Text("Data berhasil diupdate"),
654.           actions: [
655.             TextButton(
656.               onPressed: cancelClick,
657.               child: Text("Cancel"),
658.             ),
659.             TextButton(
660.               onPressed: okClick,
661.               child: Text("OK"),
662.             ),
663.           ],
664.         );
665.       },
666.     );
667.   }
668.
669.   void showCancelDialog() {
670.     showDialog(
671.       context: context,
672.       builder: (context) {
673.         return AlertDialog(
674.           title: Text("Cancel"),
675.           content: Text("Data gagal diupdate"),
676.           actions: [
677.             TextButton(
678.               onPressed: cancelClick,
679.               child: Text("OK"),
680.             ),
681.           ],
682.         );
683.       },
684.     );
685.   }
686.
687.   void showLogoutDialog() {
688.     showDialog(
689.       context: context,
690.       builder: (context) {
691.         return AlertDialog(
692.           title: Text("Logout"),
693.           content: Text("Anda yakin ingin keluar?"),
694.           actions: [
695.             TextButton(
696.               onPressed: cancelClick,
697.               child: Text("Cancel"),
698.             ),
699.             TextButton(
700.               onPressed: () {
701.                 Navigator.of(context).pop();
702.                 Navigator.pushReplacementNamed(context, '/login');
703.               },
704.               child: Text("Logout"),
705.             ),
706.           ],
707.         );
708.       },
709.     );
710.   }
711.
712.   void okClick() {
713.     setState(() {
714.       page = ProdukPage();
715.     });
716.   }
717.
718.   void cancelClick() {
719.     Navigator.pop(context);
720.   }
721.
722.   void showOkDialog() {
723.     showDialog(
724.       context: context,
725.       builder: (context) {
726.         return AlertDialog(
727.           title: Text("Success"),
728.           content: Text("Data berhasil diupdate"),
729.           actions: [
730.             TextButton(
731.               onPressed: cancelClick,
732.               child: Text("Cancel"),
733.             ),
734.             TextButton(
735.               onPressed: okClick,
736.               child: Text("OK"),
737.             ),
738.           ],
739.         );
740.       },
741.     );
742.   }
743.
744.   void showCancelDialog() {
745.     showDialog(
746.       context: context,
747.       builder: (context) {
748.         return AlertDialog(
749.           title: Text("Cancel"),
750.           content: Text("Data gagal diupdate"),
751.           actions: [
752.             TextButton(
753.               onPressed: cancelClick,
754.               child: Text("OK"),
755.             ),
756.           ],
757.         );
758.       },
759.     );
760.   }
761.
762.   void showLogoutDialog() {
763.     showDialog(
764.       context: context,
765.       builder: (context) {
766.         return AlertDialog(
767.           title: Text("Logout"),
768.           content: Text("Anda yakin ingin keluar?"),
769.           actions: [
770.             TextButton(
771.               onPressed: cancelClick,
772.               child: Text("Cancel"),
773.             ),
774.             TextButton(
775.               onPressed: () {
776.                 Navigator.of(context).pop();
777.                 Navigator.pushReplacementNamed(context, '/login');
778.               },
779.               child: Text("Logout"),
780.             ),
781.           ],
782.         );
783.       },
784.     );
785.   }
786.
787.   void okClick() {
788.     setState(() {
789.       page = ProdukPage();
790.     });
791.   }
792.
793.   void cancelClick() {
794.     Navigator.pop(context);
795.   }
796.
797.   void showOkDialog() {
798.     showDialog(
799.       context: context,
800.       builder: (context) {
801.         return AlertDialog(
802.           title: Text("Success"),
803.           content: Text("Data berhasil diupdate"),
804.           actions: [
805.             TextButton(
806.               onPressed: cancelClick,
807.               child: Text("Cancel"),
808.             ),
809.             TextButton(
810.               onPressed: okClick,
811.               child: Text("OK"),
812.             ),
813.           ],
814.         );
815.       },
816.     );
817.   }
818.
819.   void showCancelDialog() {
820.     showDialog(
821.       context: context,
822.       builder: (context) {
823.         return AlertDialog(
824.           title: Text("Cancel"),
825.           content: Text("Data gagal diupdate"),
826.           actions: [
827.             TextButton(
828.               onPressed: cancelClick,
829.               child: Text("OK"),
830.             ),
831.           ],
832.         );
833.       },
834.     );
835.   }
836.
837.   void showLogoutDialog() {
838.     showDialog(
839.       context: context,
840.       builder: (context) {
841.         return AlertDialog(
842.           title: Text("Logout"),
843.           content: Text("Anda yakin ingin keluar?"),
844.           actions: [
845.             TextButton(
846.               onPressed: cancelClick,
847.               child: Text("Cancel"),
848.             ),
849.             TextButton(
850.               onPressed: () {
851.                 Navigator.of(context).pop();
852.                 Navigator.pushReplacementNamed(context, '/login');
853.               },
854.               child: Text("Logout"),
855.             ),
856.           ],
857.         );
858.       },
859.     );
860.   }
861.
862.   void okClick() {
863.     setState(() {
864.       page = ProdukPage();
865.     });
866.   }
867.
868.   void cancelClick() {
869.     Navigator.pop(context);
870.   }
871.
872.   void showOkDialog() {
873.     showDialog(
874.       context: context,
875.       builder: (context) {
876.         return AlertDialog(
877.           title: Text("Success"),
878.           content: Text("Data berhasil diupdate"),
879.           actions: [
880.             TextButton(
881.               onPressed: cancelClick,
882.               child: Text("Cancel"),
883.             ),
884.             TextButton(
885.               onPressed: okClick,
886.               child: Text("OK"),
887.             ),
888.           ],
889.         );
890.       },
891.     );
892.   }
893.
894.   void showCancelDialog() {
895.     showDialog(
896.       context: context,
897.       builder: (context) {
898.         return AlertDialog(
899.           title: Text("Cancel"),
900.           content: Text("Data gagal diupdate"),
901.           actions: [
902.             TextButton(
903.               onPressed: cancelClick,
904.               child: Text("OK"),
905.             ),
906.           ],
907.         );
908.       },
909.     );
910.   }
911.
912.   void showLogoutDialog() {
913.     showDialog(
914.       context: context,
915.       builder: (context) {
916.         return AlertDialog(
917.           title: Text("Logout"),
918.           content: Text("Anda yakin ingin keluar?"),
919.           actions: [
920.             TextButton(
921.               onPressed: cancelClick,
922.               child: Text("Cancel"),
923.             ),
924.             TextButton(
925.               onPressed: () {
926.                 Navigator.of(context).pop();
927.                 Navigator.pushReplacementNamed(context, '/login');
928.               },
929.               child: Text("Logout"),
930.             ),
931.           ],
932.         );
933.       },
934.     );
935.   }
936.
937.   void okClick() {
938.     setState(() {
939.       page = ProdukPage();
940.     });
941.   }
942.
943.   void cancelClick() {
944.     Navigator.pop(context);
945.   }
946.
947.   void showOkDialog() {
948.     showDialog(
949.       context: context,
950.       builder: (context) {
951.         return AlertDialog(
952.           title: Text("Success"),
953.           content: Text("Data berhasil diupdate"),
954.           actions: [
955.             TextButton(
956.               onPressed: cancelClick,
957.               child: Text("Cancel"),
958.             ),
959.             TextButton(
960.               onPressed: okClick,
961.               child: Text("OK"),
962.             ),
963.           ],
964.         );
965.       },
966.     );
967.   }
968.
969.   void showCancelDialog() {
970.     showDialog(
971.       context: context,
972.       builder: (context) {
973.         return AlertDialog(
974.           title: Text("Cancel"),
975.           content: Text("Data gagal diupdate"),
976.           actions: [
977.             TextButton(
978.               onPressed: cancelClick,
979.               child: Text("OK"),
980.             ),
981.           ],
982.         );
983.       },
984.     );
985.   }
986.
987.   void showLogoutDialog() {
988.     showDialog(
989.       context: context,
990.       builder: (context) {
991.         return AlertDialog(
992.           title: Text("Logout"),
993.           content: Text("Anda yakin ingin keluar?"),
994.           actions: [
995.             TextButton(
996.               onPressed: cancelClick,
997.               child: Text("Cancel"),
998.             ),
999.             TextButton(
1000.               onPressed: () {
1001.                 Navigator.of(context).pop();
1002.                 Navigator.pushReplacementNamed(context, '/login');
1003.               },
1004.               child: Text("Logout"),
1005.             ),
1006.           ],
1007.         );
1008.       },
1009.     );
1010.   }
1011.
1012.   void okClick() {
1013.     setState(() {
1014.       page = ProdukPage();
1015.     });
1016.   }
1017.
1018.   void cancelClick() {
1019.     Navigator.pop(context);
1020.   }
1021.
1022.   void showOkDialog() {
1023.     showDialog(
1024.       context: context,
1025.       builder: (context) {
1026.         return AlertDialog(
1027.           title: Text("Success"),
1028.           content: Text("Data berhasil diupdate"),
1029.           actions: [
1030.             TextButton(
1031.               onPressed: cancelClick,
1032.               child: Text("Cancel"),
1033.             ),
1034.             TextButton(
1035.               onPressed: okClick,
1036.               child: Text("OK"),
1037.             ),
1038.           ],
1039.         );
1040.       },
1041.     );
1042.   }
1043.
1044.   void showCancelDialog() {
1045.     showDialog(
1046.       context: context,
1047.       builder: (context) {
1048.         return AlertDialog(
1049.           title: Text("Cancel"),
1050.           content: Text("Data gagal diupdate"),
1051.           actions: [
1052.             TextButton(
1053.               onPressed: cancelClick,
1054.               child: Text("OK"),
1055.             ),
1056.           ],
1057.         );
1058.       },
1059.     );
1060.   }
1061.
1062.   void showLogoutDialog() {
1063.     showDialog(
1064.       context: context,
1065.       builder: (context) {
1066.         return AlertDialog(
1067.           title: Text("Logout"),
1068.           content: Text("Anda yakin ingin keluar?"),
1069.           actions: [
1070.             TextButton(
1071.               onPressed: cancelClick,
1072.               child: Text("Cancel"),
1073.             ),
1074.             TextButton(
1075.               onPressed: () {
1076.                 Navigator.of(context).pop();
1077.                 Navigator.pushReplacementNamed(context, '/login');
1078.               },
1079.               child: Text("Logout"),
1080.             ),
1081.           ],
1082.         );
1083.       },
1084.     );
1085.   }
1086.
1087.   void okClick() {
1088.     setState(() {
1089.       page = ProdukPage();
1090.     });
1091.   }
1092.
1093.   void cancelClick() {
1094.     Navigator.pop(context);
1095.   }
1096.
1097.   void showOkDialog() {
1098.     showDialog(
1099.       context: context,
1100.      builder: (context) {
1101.        return AlertDialog(
1102.          title: Text("Success"),
1103.          content: Text("Data berhasil diupdate"),
1104.          actions: [
1105.            TextButton(
1106.              onPressed: cancelClick,
1107.              child: Text("Cancel"),
1108.            ),
1109.            TextButton(
1110.              onPressed: okClick,
1111.              child: Text("OK"),
1112.            ),
1113.          ],
1114.        );
1115.      },
1116.    );
1117.  }
1118.
1119.  void showCancelDialog() {
1120.    showDialog(
1121.      context: context,
1122.      builder: (context) {
1123.        return AlertDialog(
1124.          title: Text("Cancel"),
1125.          content: Text("Data gagal diupdate"),
1126.          actions: [
1127.            TextButton(
1128.              onPressed: cancelClick,
1129.              child: Text("OK"),
1130.            ),
1131.          ],
1132.        );
1133.      },
1134.    );
1135.  }
1136.
1137.  void showLogoutDialog() {
1138.    showDialog(
1139.      context: context,
1140.      builder: (context) {
1141.        return AlertDialog(
1142.          title: Text("Logout"),
1143.          content: Text("Anda yakin ingin keluar?"),
1144.          actions: [
1145.            TextButton(
1146.              onPressed: cancelClick,
1147.              child: Text("Cancel"),
1148.            ),
1149.            TextButton(
1150.              onPressed: () {
1151.                Navigator.of(context).pop();
1152.                Navigator.pushReplacementNamed(context, '/login');
1153.              },
1154.              child: Text("Logout"),
1155.            ),
1156.          ],
1157.        );
1158.      },
1159.    );
1160.  }
1161.
1162.  void okClick() {
1163.    setState(() {
1164.      page = ProdukPage();
1165.    });
1166.  }
1167.
1168.  void cancelClick() {
1169.    Navigator.pop(context);
1170.  }
1171.
1172.  void showOkDialog() {
1173.    showDialog(
1174.      context: context,
1175.      builder: (context) {
1176.        return AlertDialog(
1177.          title: Text("Success"),
1178.          content: Text("Data berhasil diupdate"),
1179.          actions: [
1180.            TextButton(
1181.              onPressed: cancelClick,
1182.              child: Text("Cancel"),
1183.            ),
1184.            TextButton(
1185.              onPressed: okClick,
1186.              child: Text("OK"),
1187.            ),
1188.          ],
1189.        );
1190.      },
1191.    );
1192.  }
1193.
1194.  void showCancelDialog() {
1195.    showDialog(
1196.      context: context,
1197.      builder: (context) {
1198.        return AlertDialog(
1199.          title: Text("Cancel"),
1200.          content: Text("Data gagal diupdate"),
1201.          actions: [
1202.            TextButton(
1203.              onPressed: cancelClick,
1204.              child: Text("OK"),
1205.            ),
1206.          ],
1207.        );
1208.      },
1209.    );
1210.  }
1211.
1212.  void showLogoutDialog() {
1213.    showDialog(
1214.      context: context,
1215.      builder: (context) {
1216.        return AlertDialog(
1217.          title: Text("Logout"),
1218.          content: Text("Anda yakin ingin keluar?"),
1219.          actions: [
1220.            TextButton(
1221.              onPressed: cancelClick,
1222.              child: Text("Cancel"),
1223.            ),
1224.            TextButton(
1225.              onPressed: () {
1226.                Navigator.of(context).pop();
1227.                Navigator.pushReplacementNamed(context, '/login');
1228.              },
1229.              child: Text("Logout"),
1230.            ),
1231.          ],
1232.        );
1233.      },
1234.    );
1235.  }
1236.
1237.  void okClick() {
1238.    setState(() {
1239.      page = ProdukPage();
1240.    });
1241.  }
1242.
1243.  void cancelClick() {
1244.    Navigator.pop(context);
1245.  }
1246.
1247.  void showOkDialog() {
1248.    showDialog(
1249.      context: context,
1250.      builder: (context) {
1251.        return AlertDialog(
1252.          title: Text("Success"),
1253.          content: Text("Data berhasil diupdate"),
1254.          actions: [
1255.            TextButton(
1256.              onPressed: cancelClick,
1257.              child: Text("Cancel"),
1258.            ),
1259.            TextButton(
1260.              onPressed: okClick,
1261.              child: Text("OK"),
1262.            ),
1263.          ],
1264.        );
1265.      },
1266.    );
1267.  }
1268.
1269.  void showCancelDialog() {
1270.    showDialog(
1271.      context: context,
1272.      builder: (context) {
1273.        return AlertDialog(
1274.          title: Text("Cancel"),
1275.          content: Text("Data gagal diupdate"),
1276.          actions: [
1277.            TextButton(
1278.              onPressed: cancelClick,
1279.              child: Text("OK"),
1280.            ),
1281.          ],
1282.        );
1283.      },
1284.    );
1285.  }
1286.
1287.  void showLogoutDialog() {
1288.    showDialog(
1289.      context: context,
1290.      builder: (context) {
1291.        return AlertDialog(
1292.          title: Text("Logout"),
1293.          content: Text("Anda yakin ingin keluar?"),
1294.          actions: [
1295.            TextButton(
1296.              onPressed: cancelClick,
1297.              child: Text("Cancel"),
1298.            ),
1299.            TextButton(
1300.              onPressed: () {
1301.                Navigator.of(context).pop();
1302.                Navigator.pushReplacementNamed(context, '/login');
1303.              },
1304.              child: Text("Logout"),
1305.            ),
1306.          ],
1307.        );
1308.      },
1309.    );
1310.  }
1311.
1312.
```

```

33.      });
34.    }
35.  }
36.
37. @override
38. Widget build(BuildContext context) {
39.   return MaterialApp(
40.     title: 'Toko Kita',
41.     debugShowCheckedModeBanner: false,
42.     home: page,
43.   );
44. }
45. 
```

c) Modifikasi registrasi_page.dart

Buka file **registrasi_page.dart** pada folder **ui** kemudian modifikasi fungsi `_buttonRegistrasi` dan tambahkan fungsi dengan nama `_submit` seperti dibawah

```

116. //Membuat Tombol Registrasi
117. Widget _buttonRegistrasi() {
118.   return RaisedButton(
119.     child: Text("Registrasi"),
120.     onPressed: (){
121.       var validate = _formKey.currentState.validate();
122.       if(validate) {
123.         if(!_isLoading) _submit();
124.       }
125.     });
126.   }
127.
128. void _submit() {
129.   _formKey.currentState.save();
130.   setState(() {
131.     _isLoading = true;
132.   });
133.   RegistrasiBloc.registrasi(
134.     nama: _namaTextboxController.text,
135.     email: _emailTextboxController.text,
136.     password: _passwordTextboxController.text
137.   ).then((value) {
138.     showDialog(
139.       context: context,
140.       barrierDismissible: false,
141.       builder: (BuildContext context) => SuccessDialog(
142.         description: "Registrasi berhasil, silahkan login",
143.         onClick: () {
144.           Navigator.pop(context);
145.         },
146.       ),
147.     );
148.   }, onError: (error){
149.     showDialog(
150.       context: context,
151.       barrierDismissible: false,
152.       builder: (BuildContext context) => WarningDialog(
153.         description: "Registrasi gagal, silahkan coba lagi",
154.       )
155.     );
156.   });
157. 
```

```

155.         );
156.     });
157.     setState(() {
158.         _isLoading = false;
159.     });
160. }

```

Sehingga keselurhan kode pada **registrasi_page.dart** menjadi seperti berikut

```

1. import 'package:flutter/material.dart';
2. import 'package:tokokita/bloc/registrasi_bloc.dart';
3. import 'package:tokokita/widget/success_dialog.dart';
4. import 'package:tokokita/widget/warning_dialog.dart';
5.
6. class RegistrasiPage extends StatefulWidget {
7.     @override
8.     _RegistrasiPageState createState() => _RegistrasiPageState();
9. }
10.
11. class _RegistrasiPageState extends State<RegistrasiPage> {
12.     final GlobalKey<FormState> _formKey = GlobalKey<FormState>();
13.     bool _isLoading = false;
14.
15.     final TextEditingController _namaTextboxController = TextEditingController();
16.     final TextEditingController _emailTextboxController = TextEditingController();
17.     final TextEditingController _passwordTextboxController = TextEditingController();
18.
19.     @override
20.     Widget build(BuildContext context) {
21.         return Scaffold(
22.             appBar: AppBar(title: Text("Registrasi")),
23.             body: SingleChildScrollView(
24.                 child: Container(
25.                     child: Padding(
26.                         padding: const EdgeInsets.all(8.0),
27.                         child: Form(
28.                             key: _formKey,
29.                             child: Column(
30.                                 mainAxisAlignment: MainAxisAlignment.center,
31.                                 children: [
32.                                     _namaTextField(),
33.                                     _emailTextField(),
34.                                     _passwordTextField(),
35.                                     _passwordKonfirmasiTextField(),
36.                                     _buttonRegistrasi()
37.                                 ],
38.                             ),
39.                         ),
40.                     ),
41.                 ),
42.             ),
43.         );
44.     }
45.
46. //Membuat Textbox Nama
47. Widget _namaTextField() {
48.     return TextFormField(

```

```

49.     decoration: InputDecoration(labelText: "Nama"),
50.     keyboardType: TextInputType.text,
51.     controller: _namaTextboxController,
52.     validator: (value){
53.       if(value.length < 3){
54.         return "Nama harus diisi minimal 3 karakter";
55.       }
56.       return null;
57.     },
58.   );
59. }
60.
61. //Membuat Textbox email
62. Widget _emailTextField() {
63.   return TextFormField(
64.     decoration: InputDecoration(labelText: "Email"),
65.     keyboardType: TextInputType.emailAddress,
66.     controller: _emailTextboxController,
67.     validator: (value){
68.       //validasi harus diisi
69.       if(value.isEmpty){
70.         return 'Email harus diisi';
71.       }
72.       //validasi email
73.       Pattern pattern = r'^(([^\<@\>][\.\,\;\:\s@"]+([^\<@\>][\.\,\;\:\s@"]+)*|(\\".+\"))@(([0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3})|(([a-zA-Z\-\_0-9]+\.)+[a-zA-Z]{2,}))$';
74.       RegExp regex = new RegExp(pattern);
75.       if (!regex.hasMatch(value)) {
76.         return "Email tidak valid";
77.       }
78.       return null;
79.     },
80.   );
81. }
82.
83. //Membuat Textbox password
84. Widget _passwordTextField() {
85.   return TextFormField(
86.     decoration: InputDecoration(labelText: "Password"),
87.     keyboardType: TextInputType.text,
88.     obscureText: true,
89.     controller: _passwordTextboxController,
90.     validator: (value){
91.       //jika karakter yang dimasukkan kurang dari 6 karakter
92.       if(value.length < 6){
93.         return "Password harus diisi minimal 6 karakter";
94.       }
95.       return null;
96.     },
97.   );
98. }
99.
100. //membuat textbox Konfirmasi Password
101. Widget _passwordKonfirmasiTextField() {
102.   return TextFormField(
103.     decoration: InputDecoration(labelText: "Konfirmasi Password"),
104.     keyboardType: TextInputType.text,
105.     obscureText: true,
106.     validator: (value){

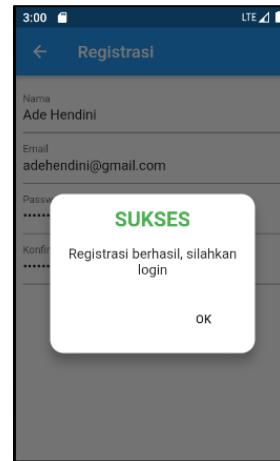
```

```

107.          //jika inputan tidak sama dengan password
108.          if(value != _passwordTextboxController.text) {
109.              return "Konfirmasi Password tidak sama";
110.          }
111.      },
112.  );
113. );
114. }
115.
116. //Membuat Tombol Registrasi
117. Widget _buttonRegistrasi() {
118.     return RaisedButton(
119.         child: Text("Registrasi"),
120.         onPressed: (){
121.             var validate = _formKey.currentState.validate();
122.             if(validate) {
123.                 if(!_isLoading) _submit();
124.             }
125.         });
126. }
127.
128. void _submit() {
129.     _formKey.currentState.save();
130.     setState(() {
131.         _isLoading = true;
132.     });
133.     RegistrasiBloc.registrasi(
134.         nama: _namaTextboxController.text,
135.         email: _emailTextboxController.text,
136.         password: _passwordTextboxController.text
137.     ).then((value) {
138.         showDialog(
139.             context: context,
140.             barrierDismissible: false,
141.             builder: (BuildContext context) => SuccessDialog(
142.                 description: "Registrasi berhasil, silahkan login",
143.                 onClick: () {
144.                     Navigator.pop(context);
145.                 },
146.             )
147.         );
148.     }, onError: (error){
149.         print(error);
150.         showDialog(
151.             context: context,
152.             barrierDismissible: false,
153.             builder: (BuildContext context) => WarningDialog(
154.                 description: "Registrasi gagal, silahkan coba lagi",
155.             )
156.         );
157.     });
158.     setState(() {
159.         _isLoading = false;
160.     });
161. }
162. }

```

Maka tampilannya akan menjadi seperti berikut



d) Modifikasi login_page.dart (fungsi login)

Buka file **login_page.dart** pada folder **ui** kemudian modifikasi fungsi `_buttonLogin` dan tambahkan fungsi dengan nama `_submit` seperti dibawah

```
79. //Membuat Tombol Login
80. Widget _buttonLogin() {
81.   return RaisedButton(
82.     child: Text("Login"),
83.     onPressed: (){
84.       var validate = _formKey.currentState.validate();
85.       if(validate) {
86.         if(!_isLoading) _submit();
87.       }
88.     });
89. }
90.
91. void _submit() {
92.   _formKey.currentState.save();
93.   setState(() {
94.     _isLoading = true;
95.   });
96.   LoginBloc.login(
97.     email: _emailTextboxController.text,
98.     password: _passwordTextboxController.text
99.   ).then((value) async{
100.     await UserInfo().setToken(value.token);
101.     await UserInfo().setUserID(value.userID);
102.     Navigator.pushReplacement(
103.       context, new MaterialPageRoute(builder: (context) => ProdukPage()));
104.   }, onError: (error){
105.     print(error);
106.     showDialog(
107.       context: context,
108.       barrierDismissible: false,
109.       builder: (BuildContext context) => WarningDialog(
110.         description: "Login gagal, silahkan coba lagi",
```

```
111.         )
112.     );
113.   });
114.   setState(() {
115.     _isLoading = false;
116.   });
117. }
```

Adapun kode secara keseluruhan menjadi seperti berikut

```
1. import 'package:flutter/material.dart';
2. import 'package:tokokita/bloc/login_bloc.dart';
3. import 'package:tokokita/helpers/user_info.dart';
4. import 'package:tokokita/ui/produk_page.dart';
5. import 'package:tokokita/ui/registrasi_page.dart';
6. import 'package:tokokita/widget/warning_dialog.dart';
7.
8. class LoginPage extends StatefulWidget {
9.   @override
10.  _LoginPageState createState() => _LoginPageState();
11. }
12.
13. class _LoginPageState extends State<LoginPage> {
14.   final GlobalKey<FormState> _formKey = GlobalKey<FormState>();
15.   bool _isLoading = false;
16.
17.   final TextEditingController _emailTextboxController = TextEditingController();
18.   final TextEditingController _passwordTextboxController = TextEditingController();
19.
20.   @override
21.   Widget build(BuildContext context) {
22.     return Scaffold(
23.       appBar: AppBar(title: Text("Login")),
24.       body: SingleChildScrollView(
25.         child: Container(
26.           child: Padding(
27.             padding: const EdgeInsets.all(8.0),
28.             child: Form(
29.               key: _formKey,
30.               child: Column(
31.                 children: [
32.                   _emailTextField(),
33.                   _passwordTextField(),
34.                   _buttonLogin(),
35.                   SizedBox(height: 30, ),
36.                   _menuRegistrasi()
37.                 ],
38.               ),
39.             ),
40.           ),
41.         ),
42.       ),
43.     );
44.   }
45.
46. //Membuat Textbox email
47. Widget _emailTextField() {
48.   return TextFormField(
49.     decoration: InputDecoration(labelText: "Email"),
```

```

50.     keyboardType: TextInputType.emailAddress,
51.     controller: _emailTextboxController,
52.     validator: (value){
53.       //validasi harus diisi
54.       if(value.isEmpty){
55.         return 'Email harus diisi';
56.       }
57.       return null;
58.     },
59.   );
60. }
61.
62. //Membuat Textbox password
63. Widget _passwordTextField() {
64.   return TextFormField(
65.     decoration: InputDecoration(labelText: "Password"),
66.     keyboardType: TextInputType.text,
67.     obscureText: true,
68.     controller: _passwordTextboxController,
69.     validator: (value){
70.       //jika karakter yang dimasukkan kurang dari 6 karakter
71.       if(value.isEmpty){
72.         return "Password harus diisi";
73.       }
74.       return null;
75.     },
76.   );
77. }
78.
79. //Membuat Tombol Login
80. Widget _buttonLogin() {
81.   return RaisedButton(
82.     child: Text("Login"),
83.     onPressed: (){
84.       var validate = _formKey.currentState.validate();
85.       if(validate) {
86.         if(!_isLoading) _submit();
87.       }
88.     });
89. }
90.
91. void _submit() {
92.   _formKey.currentState.save();
93.   setState(() {
94.     _isLoading = true;
95.   });
96.   LoginBloc.login(
97.     email: _emailTextboxController.text,
98.     password: _passwordTextboxController.text
99.   ).then((value) async{
100.     await UserInfo().setToken(value.token);
101.     await UserInfo().setUserID(value.userID);
102.     Navigator.pushReplacement(
103.       context, new MaterialPageRoute(builder: (context) => ProdukPage()));
104.   }, onError: (error){
105.     print(error);
106.     showDialog(
107.       context: context,
108.       barrierDismissible: false,
109.       builder: (BuildContext context) => WarningDialog()

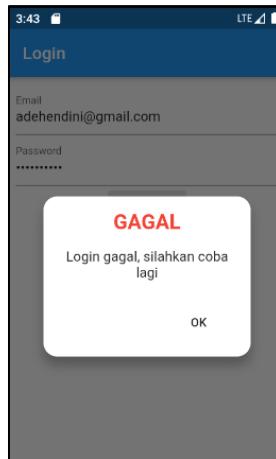
```

```

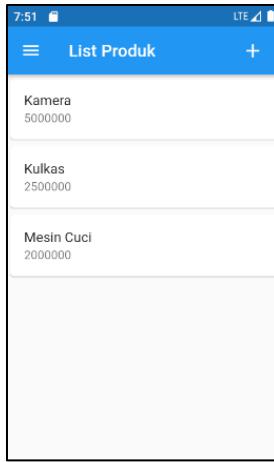
110.             description: "Login gagal, silahkan coba lagi",
111.         )
112.     );
113. );
114. setState(() {
115.     _isLoading = false;
116. });
117. }
118.
119. // Membuat menu untuk membuka halaman registrasi
120. Widget _menuRegistrasi() {
121.     return Container(
122.         child: Center(
123.             child: InkWell(
124.                 child: Text("Registrasi", style: TextStyle(color: Colors.blue),),
125.                 onTap: () {
126.                     Navigator.push(context, new MaterialPageRoute(builder: (context)=> R
127.                         egistrasiPage()));
128.                 },
129.             ),
130.         );
131.     }
132. }

```

Jika Gagal akan muncul pesan seperti berikut



Jika berhasil akan menuju ke halaman **produk_page.dart**



e) Modifikasi produk_page.dart

(1) Menambahkan fungsi logout pada drawer

Agar link logout dapat berfungsi, akan ditambahkan kode pada drawer logout, seperti berikut

```
34. drawer: Drawer(
35.         child: ListView(
36.             children: [
37.                 ListTile(
38.                     title: Text('Logout'),
39.                     trailing: Icon(Icons.logout),
40.                     onTap: () async {
41.                         await LogoutBloc.logout().then((value) {
42.                             Navigator.pushReplacement(context, new MaterialPageRoute(builder: (c
ontext) => LoginPage())));
43.                         });
44.                     },
45.                 )
46.             ],
47.         ),
48.     ),
49.     body: ListView(
```

Secara keseluruhan kode pada **produk_page.dart** menjadi seperti berikut

```
1. import 'package:flutter/cupertino.dart';
2. import 'package:flutter/material.dart';
3. import 'package:tokokita/bloc/logout_bloc.dart';
4. import 'package:tokokita/model/produk.dart';
5. import 'package:tokokita/ui/login_page.dart';
6. import 'package:tokokita/ui/produk_detail.dart';
7. import 'package:tokokita/ui/produk_form.dart';
8.
9. class ProdukPage extends StatefulWidget {
10.   @override
11.   _ProdukPageState createState() => _ProdukPageState();
12. }
13.
```

```

14. class _ProdukPageState extends State<ProdukPage> {
15.   @override
16.   Widget build(BuildContext context) {
17.     return Scaffold(
18.       appBar: AppBar(
19.         title: Text('List Produk'),
20.         actions: [
21.           Padding(
22.             padding: EdgeInsets.only(right: 20.0),
23.             child: GestureDetector(
24.               child: Icon(Icons.add, size: 26.0),
25.               onTap: () async {
26.                 Navigator.push(
27.                   context,
28.                   MaterialPageRoute(
29.                     builder: (context) => ProdukForm()));
30.               },
31.             )),
32.           ],
33.         ),
34.         drawer: Drawer(
35.           child: ListView(
36.             children: [
37.               ListTile(
38.                 title: Text('Logout'),
39.                 trailing: Icon(Icons.logout),
40.                 onTap: () async {
41.                   await LogoutBloc.logout().then((value) {
42.                     Navigator.pushReplacement(context, MaterialPageRoute(builder: (context) => LoginPage()));
43.                   });
44.                 },
45.               ),
46.               ],
47.             ),
48.           ),
49.           body: ListView(
50.             children: [
51.               ItemProduk(produk: Produk(id: 1, kodeProduk: 'A001', namaProduk: 'Kamera', hargaProduk: 5000000)),
52.               ItemProduk(produk: Produk(id: 2, kodeProduk: 'A002', namaProduk: 'Kulkas', hargaProduk: 2500000)),
53.               ItemProduk(produk: Produk(id: 3, kodeProduk: 'A003', namaProduk: 'Mesin Cuci', hargaProduk: 2000000)),
54.             ],
55.           ),
56.         );
57.       }
58.     }
59.
60. class ItemProduk extends StatelessWidget {
61.   final Produk produk;
62.
63.   ItemProduk({this.produk});
64.
65.   @override
66.   Widget build(BuildContext context) {
67.     return Container(
68.       child: GestureDetector(
69.         onTap: () {

```

```

70.         Navigator.push(
71.             context,
72.             new MaterialPageRoute(
73.                 builder: (context) => ProdukDetail(produk,)));
74.         },
75.         child: Card(
76.             child: ListTile(
77.                 title: Text(produk.namaProduk),
78.                 subtitle: Text(produk.hargaProduk.toString()),
79.             ),
80.         ),
81.     ),
82. );
83. }
84. 
```

(2) Menampilkan Data Produk dari Rest API

Pada bagian ini akan dimodifikasi file **produk_page.dart** sehingga dapat menampilkan data dari Rest API. Berikut kode keseluruhan

```

1. import 'package:flutter/cupertino.dart';
2. import 'package:flutter/material.dart';
3. import 'package:tokokita/bloc/logout_bloc.dart';
4. import 'package:tokokita/bloc/produk_bloc.dart';
5. import 'package:tokokita/model/produk.dart';
6. import 'package:tokokita/ui/login_page.dart';
7. import 'package:tokokita/ui/produk_detail.dart';
8. import 'package:tokokita/ui/produk_form.dart';
9.
10. class ProdukPage extends StatefulWidget {
11.     @override
12.     _ProdukPageState createState() => _ProdukPageState();
13. }
14.
15. class _ProdukPageState extends State<ProdukPage> {
16.     @override
17.     Widget build(BuildContext context) {
18.         return Scaffold(
19.             appBar: AppBar(
20.                 title: Text('List Produk'),
21.                 actions: [
22.                     Padding(
23.                         padding: EdgeInsets.only(right: 20.0),
24.                         child: GestureDetector(
25.                             child: Icon(Icons.add, size: 26.0),
26.                             onTap: () async {
27.                                 Navigator.push(
28.                                     context,
29.                                     new MaterialPageRoute(
30.                                         builder: (context) => ProdukForm()));
31.                             },
32.                         )),
33.                     ],
34.                 ),
35.             drawer: Drawer( 
```

```

36.         child: ListView(
37.             children: [
38.                 ListTile(
39.                     title: Text('Logout'),
40.                     trailing: Icon(Icons.logout),
41.                     onTap: () async {
42.                         await LogoutBloc.logout().then((value) {
43.                             Navigator.pushReplacement(context, new MaterialPageRoute(builder: (c
    ontext) => LoginPage())));
44.                         });
45.                     },
46.                 )
47.             ],
48.         ),
49.         body: FutureBuilder<List>(
50.             future: ProdukBloc.getProduks(),
51.             builder: (context, snapshot){
52.                 if(snapshot.hasError) print(snapshot.error);
53.                 return snapshot.hasData ? ListProduk(list: snapshot.data,) : Center(child: Ci
    rcularProgressIndicator(),);
54.             },
55.         ),
56.     );
57. }
58. }
59. }
60.
61. class ListProduk extends StatelessWidget {
62.     final List list;
63.     ListProduk({this.list});
64.
65.     @override
66.     Widget build(BuildContext context) {
67.         return ListView.builder(
68.             itemCount: list==null ? 0:list.length,
69.             itemBuilder: (context, i){
70.                 return ItemProduk(produk: list[i],);
71.             });
72.     }
73. }
74.
75.
76. class ItemProduk extends StatelessWidget {
77.     final Produk produk;
78.
79.     ItemProduk({this.produk});
80.
81.     @override
82.     Widget build(BuildContext context) {
83.         return Container(
84.             child: GestureDetector(
85.                 onTap: () {
86.                     Navigator.push(
87.                         context,
88.                         new MaterialPageRoute(
89.                             builder: (context) => ProdukDetail(produk: produk,)));
90.                 },
91.                 child: Card(
92.                     child: ListTile(
93.                         title: Text(produk.namaProduk),

```

```
94.           subtitle: Text(produk.hargaProduk.toString()),  
95.           ),  
96.           ),  
97.           ),  
98.         );  
99.     }  
100.    }
```

Adapun perubahan yang dilakukan adalah penambahan sebuah class bernama **ListProduk** dengan kode

```
61. class ListProduk extends StatelessWidget {  
62.   final List list;  
63.   ListProduk({this.list});  
64.  
65.   @override  
66.   Widget build(BuildContext context) {  
67.     return ListView.builder(  
68.       itemCount: list==null ? 0:list.length,  
69.       itemBuilder: (context, i){  
70.         return ItemProduk(produk: list[i],);  
71.       });  
72.   }  
73. }
```

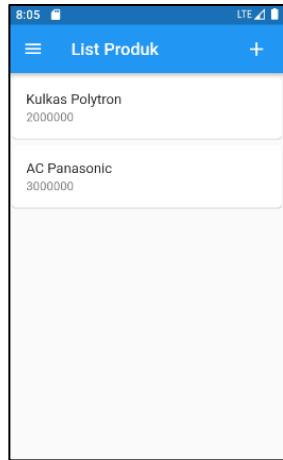
Kemudian perubahan pada bagian **body** menjadi

```
50. body: FutureBuilder<List>(  
51.   future: ProdukBloc.getProduks(),  
52.   builder: (context, snapshot){  
53.     if(snapshot.hasError) print(snapshot.error);  
54.     return snapshot.hasData ? ListProduk(list: snapshot.data,) : Center(child: Ci  
rcularProgressIndicator(),);  
55.   },  
56. ),
```

Serta memasukkan beberapa package yang diperlukan

```
1. import 'package:flutter/cupertino.dart';  
2. import 'package:flutter/material.dart';  
3. import 'package:tokokita/bloc/logout_bloc.dart';  
4. import 'package:tokokita/bloc/produk_bloc.dart';  
5. import 'package:tokokita/model/produk.dart';  
6. import 'package:tokokita/ui/login_page.dart';  
7. import 'package:tokokita/ui/produk_detail.dart';  
8. import 'package:tokokita/ui/produk_form.dart';
```

Maka tampilannya akan menjadi seperti berikut



f) Memodifikasi Form Produk (produk_form.dart)

(1) Membuat fungsi simpan

Agar tombol simpan dapat berfungsi diperlukan kode fungsi untuk menyimpan data dengan memanggil bloc **produk_bloc** yang telah dibuat sebelumnya, kita akan menambahkan sebuah fungsi dengan nama **simpan** dan memodifikasi fungsi **_buttonSubmit**

```
116. //Membuat Tombol Simpan/Ubah
117. Widget _buttonSubmit() {
118.   return RaisedButton(
119.     child: Text(tombolSubmit),
120.     onPressed: () {
121.       var validate = _formKey.currentState.validate();
122.       if(validate) {
123.         if(!_isLoading){
124.           if(widget.produk!=null){
125.             //kondisi update produk
126.
127.           }else{
128.             //kondisi tambah produk
129.             simpan();
130.           }
131.         }
132.       }
133.     );
134.   }
135.
136. simpan() {
137.   setState(() {
138.     _isLoading = true;
139.   });
140.   Produk createProduk = new Produk();
141.   createProduk.kodeProduk = _kodeProdukTextboxController.text;
142.   createProduk.namaProduk = _namaProdukTextboxController.text;
143.   createProduk.hargaProduk = int.parse(_hargaProdukTextboxController.text);
144.   ProdukBloc.addProduk(produk: createProduk).then((value) {
```

```

145.     Navigator.of(context).push(new MaterialPageRoute(builder: (BuildContext context) =>
  ProdukPage())));
146.   },onError: (error){
147.     showDialog(
148.       context: context,
149.       builder: (BuildContext context) => WarningDialog(
150.         description: "Simpan gagal, silahkan coba lagi",
151.       )
152.     );
153.   });
154.   setState(() {
155.     _isLoading = false;
156.   });
157. }

```

Dengan keseluruhan kode menjadi

```

1. import 'package:flutter/material.dart';
2. import 'package:tokokita/bloc/produk_bloc.dart';
3. import 'package:tokokita/model/produk.dart';
4. import 'package:tokokita/ui/produk_page.dart';
5. import 'package:tokokita/widget/warning_dialog.dart';
6.
7. class ProdukForm extends StatefulWidget {
8.   Produk produk;
9.   ProdukForm({this.produk});
10.  @override
11.  _ProdukFormState createState() => _ProdukFormState();
12. }
13.
14. class _ProdukFormState extends State<ProdukForm> {
15.   final GlobalKey<FormState> _formKey = GlobalKey<FormState>();
16.   bool _isLoading = false;
17.   String judul = "TAMBAH PRODUK";
18.   String tombolSubmit = "SIMPAN";
19.
20.   final _kodeProdukTextboxController = TextEditingController();
21.   final _namaProdukTextboxController = TextEditingController();
22.   final _hargaProdukTextboxController = TextEditingController();
23.
24.   @override
25.   void initState() {
26.     // TODO: implement initState
27.     super.initState();
28.     isUpdate();
29.   }
30.
31.   isUpdate(){
32.     if(widget.produk!=null){
33.       setState(() {
34.         judul = "UBAH PRODUK";
35.         tombolSubmit = "UBAH";
36.         _kodeProdukTextboxController.text = widget.produk.kodeProduk;
37.         _namaProdukTextboxController.text = widget.produk.namaProduk;
38.         _hargaProdukTextboxController.text = widget.produk.hargaProduk.toString();
39.       });
40.     }else{

```

```

41.     judul = "TAMBAH PRODUK";
42.     tombolSubmit = "SIMPAN";
43.   }
44. }
45.
46. @override
47. Widget build(BuildContext context) {
48.   return Scaffold(
49.     appBar: AppBar(title: Text(judul)),
50.     body: SingleChildScrollView(
51.       child: Container(
52.         child: Padding(
53.           padding: const EdgeInsets.all(8.0),
54.           child: Form(
55.             key: _formKey,
56.             child: Column(
57.               children: [
58.                 _kodeProdukTextField(),
59.                 _namaProdukTextField(),
60.                 _hargaProdukTextField(),
61.                 _buttonSubmit()
62.               ],
63.             ),
64.           ),
65.         ),
66.       ),
67.     ),
68.   );
69. }
70.
71. //Membuat Textbox Kode Produk
72. Widget _kodeProdukTextField() {
73.   return TextFormField(
74.     decoration: InputDecoration(labelText: "Kode Produk"),
75.     keyboardType: TextInputType.text,
76.     controller: _kodeProdukTextboxController,
77.     validator: (value) {
78.       if (value.isEmpty) {
79.         return "Kode Produk harus diisi";
80.       }
81.       return null;
82.     },
83.   );
84. }
85.
86. //Membuat Textbox Nama Produk
87. Widget _namaProdukTextField() {
88.   return TextFormField(
89.     decoration: InputDecoration(labelText: "Nama Produk"),
90.     keyboardType: TextInputType.text,
91.     controller: _namaProdukTextboxController,
92.     validator: (value) {
93.       if (value.isEmpty) {
94.         return "Nama Produk harus diisi";
95.       }
96.       return null;
97.     },
98.   );
99. }
100.

```

```

101.        //Membuat Textbox Harga Produk
102.        Widget _hargaProdukTextField() {
103.            return TextFormField(
104.                decoration: InputDecoration(labelText: "Harga"),
105.                keyboardType: TextInputType.number,
106.                controller: _hargaProdukTextboxController,
107.                validator: (value) {
108.                    if (value.isEmpty) {
109.                        return "Harga harus diisi";
110.                    }
111.                    return null;
112.                },
113.            );
114.        }
115.
116.        //Membuat Tombol Simpan/Ubah
117.        Widget _buttonSubmit() {
118.            return RaisedButton(
119.                child: Text(tombolSubmit),
120.                onPressed: () {
121.                    var validate = _formKey.currentState.validate();
122.                    if(validate) {
123.                        if(!_isLoading){
124.                            if(widget.produk!=null){
125.                                //kondisi update produk
126.
127.                            }else{
128.                                //kondisi tambah produk
129.                                simpan();
130.                            }
131.                        }
132.                    }
133.                });
134.        }
135.
136.        simpan() {
137.            setState(() {
138.                _isLoading = true;
139.            });
140.            Produk createProduk = new Produk();
141.            createProduk.kodeProduk = _kodeProdukTextboxController.text;
142.            createProduk.namaProduk = _namaProdukTextboxController.text;
143.            createProduk.hargaProduk = int.parse(_hargaProdukTextboxController.text);
144.            ProdukBloc.addProduk(produk: createProduk).then((value) {
145.                Navigator.of(context).push(new MaterialPageRoute(builder: (BuildContext context) => ProdukPage()));
146.            },onError: (error){
147.                showDialog(
148.                    context: context,
149.                    builder: (BuildContext context) => WarningDialog(
150.                        description: "Simpan gagal, silahkan coba lagi",
151.                    )
152.                );
153.            });
154.            setState(() {
155.                _isLoading = false;
156.            });
157.        }
158.    }

```

(2) Membuat fungsi ubah

Sama halnya dengan simpan, kita buat sebuah fungsi **ubah** kemudian kita sertakan pada fungsi **_buttonSubmit**

Dengan fungsi ubah sebagai berikut

```
159. ubah() {
160.     setState(() {
161.         _isLoading = true;
162.     });
163.     Produk updateProduk = new Produk();
164.     updateProduk.id = widget.produk.id;
165.     updateProduk.kodeProduk = _kodeProdukTextboxController.text;
166.     updateProduk.namaProduk = _namaProdukTextboxController.text;
167.     updateProduk.hargaProduk = int.parse(_hargaProdukTextboxController.text);
168.     ProdukBloc.updateProduk(produk: updateProduk).then((value) {
169.         Navigator.of(context).push(new MaterialPageRoute(builder: (BuildContext context) =>
    ProdukPage()));
170.     },onError: (error){
171.         showDialog(
172.             context: context,
173.             builder: (BuildContext context) => WarningDialog(
174.                 description: "Permintaan ubah data gagal, silahkan coba lagi",
175.             )
176.         );
177.     });
178.     setState(() {
179.         _isLoading = false;
180.     });
181. }
```

Kemudian kita tambahkan pada fungsi **_buttonSubmit**

```
116. //Membuat Tombol Simpan/Ubah
117.     Widget _buttonSubmit() {
118.         return RaisedButton(
119.             child: Text(tombolSubmit),
120.             onPressed: () {
121.                 var validate = _formKey.currentState.validate();
122.                 if(validate) {
123.                     if(!_isLoading){
124.                         if(widget.produk!=null){
125.                             //kondisi update produk
126.                             ubah();
127.                         }else{
128.                             //kondisi tambah produk
129.                             simpan();
130.                         }
131.                     }
132.                 }
133.             });
134. }
```

Dengan kode keseluruhan menjadi seperti berikut

```
135. import 'package:flutter/material.dart';
136. import 'package:tokokita/bloc/produk_bloc.dart';
137. import 'package:tokokita/model/produk.dart';
138. import 'package:tokokita/ui/produk_page.dart';
139. import 'package:tokokita/widget/warning_dialog.dart';
140.
141. class ProdukForm extends StatefulWidget {
142.   Produk produk;
143.   ProdukForm({this.produk});
144.   @override
145.   _ProdukFormState createState() => _ProdukFormState();
146. }
147.
148. class _ProdukFormState extends State<ProdukForm> {
149.   final GlobalKey<FormState> _formKey = GlobalKey<FormState>();
150.   bool _isLoading = false;
151.   String judul = "TAMBAH PRODUK";
152.   String tombolSubmit = "SIMPAN";
153.
154.   final TextEditingController _kodeProdukTextboxController = TextEditingController();
155.   final TextEditingController _namaProdukTextboxController = TextEditingController();
156.   final TextEditingController _hargaProdukTextboxController = TextEditingController();
157.
158.   @override
159.   void initState() {
160.     // TODO: implement initState
161.     super.initState();
162.     isUpdate();
163.   }
164.
165.   isUpdate(){
166.     if(widget.produk!=null){
167.       setState(() {
168.         judul = "UBAH PRODUK";
169.         tombolSubmit = "UBAH";
170.         _kodeProdukTextboxController.text = widget.produk.kodeProduk;
171.         _namaProdukTextboxController.text = widget.produk.namaProduk;
172.         _hargaProdukTextboxController.text = widget.produk.hargaProduk.toString();
173.       });
174.     }else{
175.       judul = "TAMBAH PRODUK";
176.       tombolSubmit = "SIMPAN";
177.     }
178.   }
179.
180.   @override
181.   Widget build(BuildContext context) {
182.     return Scaffold(
183.       appBar: AppBar(title: Text(judul)),
184.       body: SingleChildScrollView(
185.         child: Container(
186.           child: Padding(
187.             padding: const EdgeInsets.all(8.0),
188.             child: Form(
189.               key: _formKey,
```

```

190.             child: Column(
191.                 children: [
192.                     _kodeProdukTextField(),
193.                     _namaProdukTextField(),
194.                     _hargaProdukTextField(),
195.                     _buttonSubmit()
196.                 ],
197.             ),
198.         ),
199.     ),
200. ),
201. ),
202. );
203. }
204.
205. //Membuat Textbox Kode Produk
206. Widget _kodeProdukTextField() {
207.     return TextFormField(
208.         decoration: InputDecoration(labelText: "Kode Produk"),
209.         keyboardType: TextInputType.text,
210.         controller: _kodeProdukTextboxController,
211.         validator: (value) {
212.             if (value.isEmpty) {
213.                 return "Kode Produk harus diisi";
214.             }
215.             return null;
216.         },
217.     );
218. }
219.
220. //Membuat Textbox Nama Produk
221. Widget _namaProdukTextField() {
222.     return TextFormField(
223.         decoration: InputDecoration(labelText: "Nama Produk"),
224.         keyboardType: TextInputType.text,
225.         controller: _namaProdukTextboxController,
226.         validator: (value) {
227.             if (value.isEmpty) {
228.                 return "Nama Produk harus diisi";
229.             }
230.             return null;
231.         },
232.     );
233. }
234.
235. //Membuat Textbox Harga Produk
236. Widget _hargaProdukTextField() {
237.     return TextFormField(
238.         decoration: InputDecoration(labelText: "Harga"),
239.         keyboardType: TextInputType.number,
240.         controller: _hargaProdukTextboxController,
241.         validator: (value) {
242.             if (value.isEmpty) {
243.                 return "Harga harus diisi";
244.             }
245.             return null;
246.         },
247.     );
248. }
249.

```

```

250. //Membuat Tombol Simpan/Ubah
251. Widget _buttonSubmit() {
252.     return RaisedButton(
253.         child: Text(tombolSubmit),
254.         onPressed: () {
255.             var validate = _formKey.currentState.validate();
256.             if(validate) {
257.                 if(!_isLoading){
258.                     if(widget.produk!=null){
259.                         //kondisi update produk
260.                         ubah();
261.                     }else{
262.                         //kondisi tambah produk
263.                         simpan();
264.                     }
265.                 }
266.             }
267.         );
268.     }
269.
270.     simpan() {
271.         setState(() {
272.             _isLoading = true;
273.         });
274.         Produk createProduk = new Produk();
275.         createProduk.kodeProduk = _kodeProdukTextboxController.text;
276.         createProduk.namaProduk = _namaProdukTextboxController.text;
277.         createProduk.hargaProduk = int.parse(_hargaProdukTextboxController.text);
278.         ProdukBloc.addProduk(produk: createProduk).then((value) {
279.             Navigator.of(context).push(new MaterialPageRoute(builder: (BuildContext context)
=> ProdukPage())));
280.         },onError: (error){
281.             showDialog(
282.                 context: context,
283.                 builder: (BuildContext context) => WarningDialog(
284.                     description: "Simpan gagal, silahkan coba lagi",
285.                 )
286.             );
287.         });
288.         setState(() {
289.             _isLoading = false;
290.         });
291.     }
292.
293.     ubah() {
294.         setState(() {
295.             _isLoading = true;
296.         });
297.         Produk updateProduk = new Produk();
298.         updateProduk.id = widget.produk.id;
299.         updateProduk.kodeProduk = _kodeProdukTextboxController.text;
300.         updateProduk.namaProduk = _namaProdukTextboxController.text;
301.         updateProduk.hargaProduk = int.parse(_hargaProdukTextboxController.text);
302.         ProdukBloc.updateProduk(produk: updateProduk).then((value) {
303.             Navigator.of(context).push(new MaterialPageRoute(builder: (BuildContext context)
=> ProdukPage())));
304.         },onError: (error){
305.             showDialog(
306.                 context: context,
307.                 builder: (BuildContext context) => WarningDialog(

```

```

308.             description: "Permintaan ubah data gagal, silahkan coba lagi",
309.         )
310.     );
311.   });
312.   setState(() {
313.     _isLoading = false;
314.   });
315. }
316. }
```

(3) Menambahkan fungsi hapus pada Detail Produk (produk_detail.dart)

Buka file **produk_detail.dart** pada folder **ui**, kemudian kita modifikasi pada fungsi **confirmHapus** menjadi seperti berikut

```

63. void confirmHapus() {
64.   AlertDialog alertDialog = new AlertDialog(
65.     content: Text("Yakin ingin menghapus data ini?"),
66.     actions: [
67.       //tombol hapus
68.       RaisedButton(
69.         child: Text("Ya"),
70.         color: Colors.green,
71.         onPressed: (){
72.           ProdukBloc.deleteProduk(id: widget.produk.id).then((value){
73.             Navigator.of(context).push(new MaterialPageRoute(builder: (BuildContext context) => ProdukPage())));
74.           },onError: (error){
75.             showDialog(
76.               context: context,
77.               builder: (BuildContext context) => WarningDialog(
78.                 description: "Hapus data gagal, silahkan coba lagi",
79.               )
80.             );
81.           });
82.         },
83.       ),
84.       //tombol batal
85.       RaisedButton(
86.         child: Text("Batal"),
87.         color: Colors.red,
88.         onPressed: ()=>Navigator.pop(context),
89.       )
90.     ],
91.   );
92.
93.   showDialog(context: context, child: alertDialog);
94. }
```

Dengan kode keseluruhan menjadi

```
1. import 'package:flutter/material.dart';
2. import 'package:tokokita/bloc/produk_bloc.dart';
3. import 'package:tokokita/model/produk.dart';
4. import 'package:tokokita/ui/produk_form.dart';
5. import 'package:tokokita/ui/produk_page.dart';
6. import 'package:tokokita/widget/warning_dialog.dart';
7.
8. class ProdukDetail extends StatefulWidget {
9.   Produk produk;
10.  ProdukDetail({this.produk});
11.  @override
12.  _ProdukDetailState createState() => _ProdukDetailState();
13. }
14.
15. class _ProdukDetailState extends State<ProdukDetail> {
16.  @override
17.  Widget build(BuildContext context) {
18.    return Scaffold(
19.      appBar: AppBar(
20.        title: Text('Detail Produk'),
21.      ),
22.      body: Center(
23.        child: Column(
24.          children: [
25.            Text(
26.              "Kode : ${widget.produk.kodeProduk}",
27.              style: TextStyle(fontSize: 20.0),
28.            ),
29.            Text(
30.              "Nama : ${widget.produk.namaProduk}",
31.              style: TextStyle(fontSize: 18.0),
32.            ),
33.            Text(
34.              "Harga : Rp. ${widget.produk.hargaProduk.toString()}",
35.              style: TextStyle(fontSize: 18.0),
36.            ),
37.            _tombolHapusEdit()
38.          ],
39.        ),
40.      ),
41.    );
42.  }
43.
44. Widget _tombolHapusEdit() {
45.   return Row(
46.     mainAxisAlignment: MainAxisAlignment.end,
47.     children: [
48.       //Tombol Edit
49.       RaisedButton(
50.         child: Text("EDIT"), color: Colors.green, onPressed: () {
51.           Navigator.push(
52.             context,
53.             new MaterialPageRoute(
54.               builder: (context) => ProdukForm(produk: widget.produk,));
55.         },
56.         //Tombol Hapus
57.         RaisedButton(
```

```
58.             child: Text("DELETE"), color: Colors.red, onPressed: ()=>confirmHapus()),
59.         ],
60.     );
61. }
62.
63. void confirmHapus() {
64.     AlertDialog alertDialog = new AlertDialog(
65.         content: Text("Yakin ingin menghapus data ini?"),
66.         actions: [
67.             //tombol hapus
68.             RaisedButton(
69.                 child: Text("Ya"),
70.                 color: Colors.green,
71.                 onPressed: (){
72.                     ProdukBloc.deleteProduk(id: widget.produk.id).then((value){
73.                         Navigator.of(context).push(new MaterialPageRoute(builder: (BuildContext context) => ProdukPage()));
74.                     },onError: (error){
75.                         showDialog(
76.                             context: context,
77.                             builder: (BuildContext context) => WarningDialog(
78.                                 description: "Hapus data gagal, silahkan coba lagi",
79.                             )
80.                         );
81.                     });
82.                 },
83.             ),
84.             //tombol batal
85.             RaisedButton(
86.                 child: Text("Batal"),
87.                 color: Colors.red,
88.                 onPressed: ()=>Navigator.pop(context),
89.             )
90.         ],
91.     );
92.
93.     showDialog(context: context, child: alertDialog);
94. }
95. }
```

PERTEMUAN 13-15

PRESENTASI TUGAS BESAR

PERTEMUAN 16

UJIAN AKHIR SEMESTER

Daftar Pustaka