

Modul

Fake SHOP APP DEVELOPMENT

Setelah kemarin kita belajar flutter widget mulai dari column, row, dll. Tibalah saatnya kita akan belajar membuat sebuah aplikasi : **FAKE SHOP**. Pada pembuatan aplikasi ini, kita akan fokus di beberapa hal yang menjadi acuan dalam pembuatan aplikasi di flutter. berikut taks yang akan kita kerjakan :

1. add package from pub.dev
2. get data from api (<https://fakestoreapi.com/products>)
3. transform data to model
4. state managemen (GetX)
5. desain UI

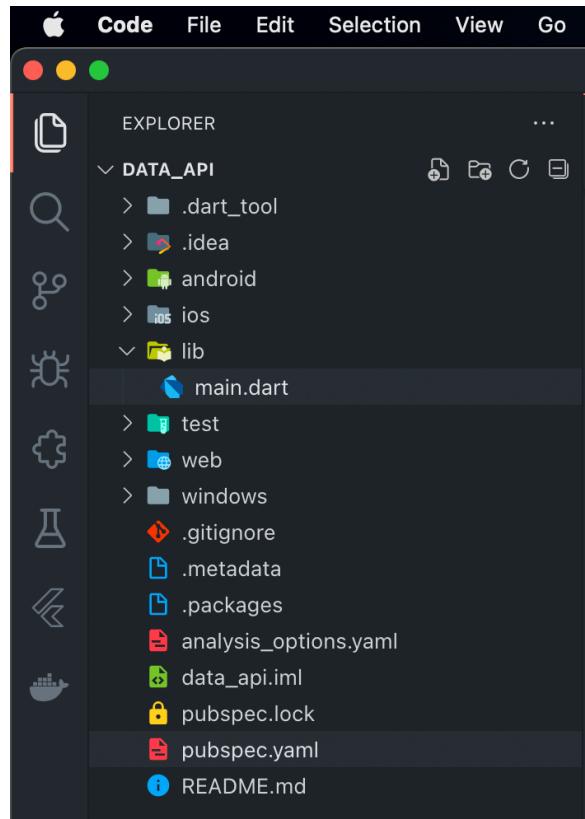
Kita akan membuat aplikasi belanja biasa yang menampilkan daftar produk. Semua produk diambil melalui internet dan ditampilkan di aplikasi Flutter. kita akan mengintegrasikan API Fakestore. Ini adalah REST API online gratis yang dapat kamu gunakan kapanpun kamu membutuhkan data Pseudo-real untuk aplikasi FAKE SHOP tanpa menjalankan kode sisi server apapun (BACKEND).

1. Menggunakan Pub untuk Manajemen Paket pada Flutter



Apa itu Pub?

Pub adalah package manager untuk Dart. Fungsinya untuk menginstal package ke dalam proyek flutter. Package adalah library yang berisi fungsi-fungsi, class, dan program Dart yang bisa digunakan ulang. Pub menggunakan file [pubspec.yaml](#) untuk menentukan package yang akan digunakan pada proyek.

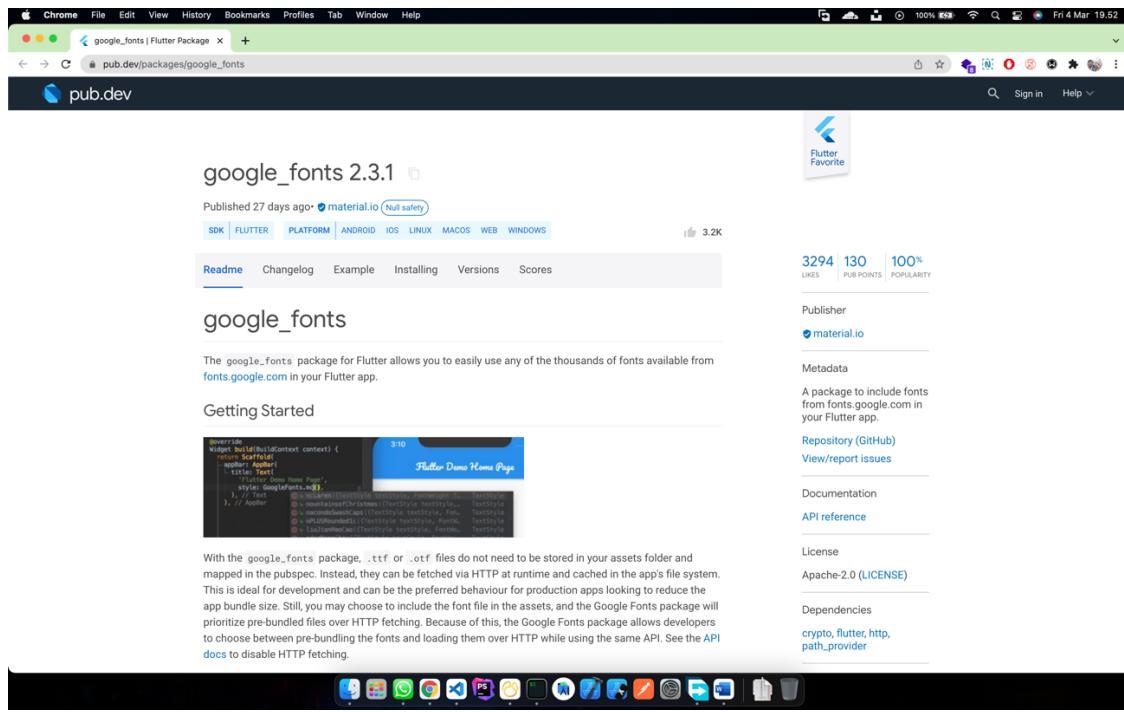


Jika kita baru belajar flutter maka sudah hal yang wajib untuk mengerti menambahkan dependency. Di flutter banyak sekali library-library yang menarik sayang rasanya jika kita tidak menggunakannya untuk mepermudah pembuatan aplikasi kita. ok to the point langsung saja kita membuat project baru, lalu bukalah pubspec.yaml di project yang telah buat

```
● ● ●  
1 name: data_api  
2 description: A new Flutter project.  
3 publish_to: 'none' # Remove this line if you wish to publish to pub.dev  
4 version: 1.0.0+1  
5  
6 environment:  
7   sdk: "≥2.16.1 <3.0.0"  
8  
9 dependencies:  
10   flutter:  
11     sdk: flutter  
12     cupertino_icons: ^1.0.2  
13  
14 dev_dependencies:  
15   flutter_test:  
16     sdk: flutter  
17     flutter_lints: ^1.0.0  
18  
19 flutter:  
20   uses-material-design: true
```

Comment sudah saya hapus.

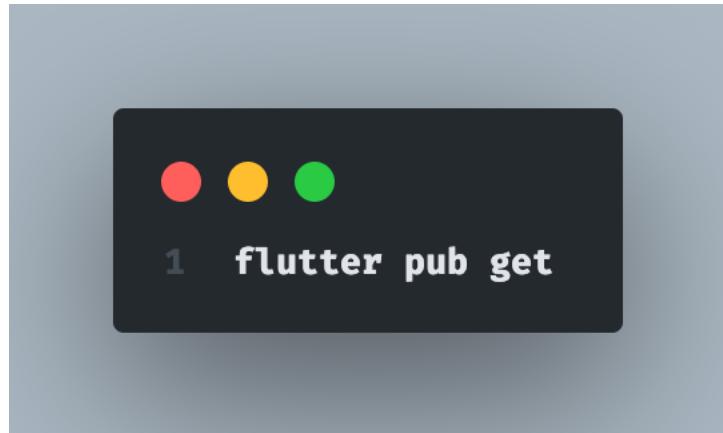
selanjutnya kita perlu library yang ingin kita tambahkan, di modul ini saya akan mencontohkan library Google fonts : https://pub.dev/packages/google_fonts.



di halaman pub.dev dokumentasi nya sudah cukup lengkap mulai dari install nya dan cara memakainnya. ok tambahkan depedency nya:

```
1 name: data_api
2 description: A new Flutter project.
3 publish_to: 'none' # Remove this line if you wish to publish to pub.dev
4 version: 1.0.0+1
5
6 environment:
7   sdk: "≥2.16.1 <3.0.0"
8
9 dependencies:
10   flutter:
11     sdk: flutter
12   cupertino_icons: ^1.0.2
13   google_fonts: ^2.3.1 #ini adalah package yang sudah ditambahkan
14
15 dev_dependencies:
16   flutter_test:
17     sdk: flutter
18   flutter_lints: ^1.0.0
19
20 flutter:
21   uses-material-design: true
```

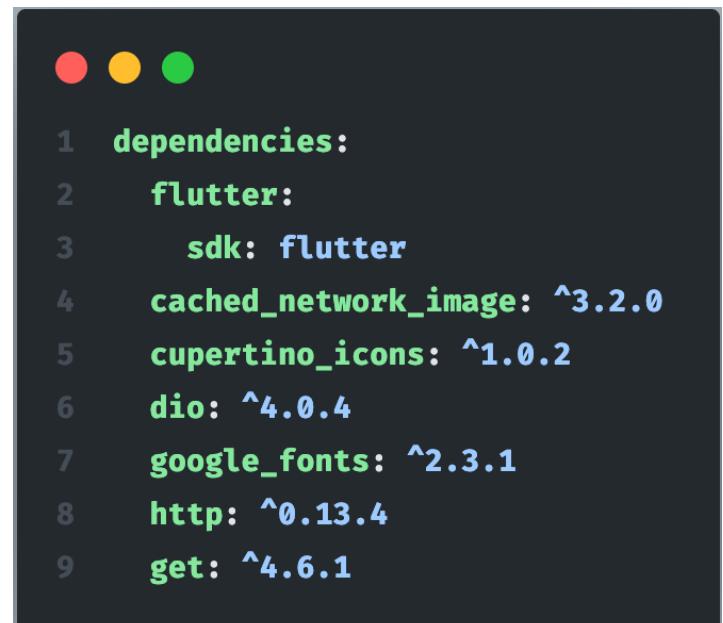
setelah menambahkan dependency IDE biasanya melakukan pub get secara otomatis tapi jika tidak, kita bisa melakukannya secara manual. ketikkan perintah berikut di terminal project atau arahkan terminalnya ke project :



lalu terminal akan menambahkan library ke project kita dan sekarang kita bisa menggunakan dependency baru saja kita tambahkan. silahkan kunjungi : <https://pub.dev> untuk lebih jelasnya. setelah berhasil menambahkan package google font, kita akan menambahkan package:

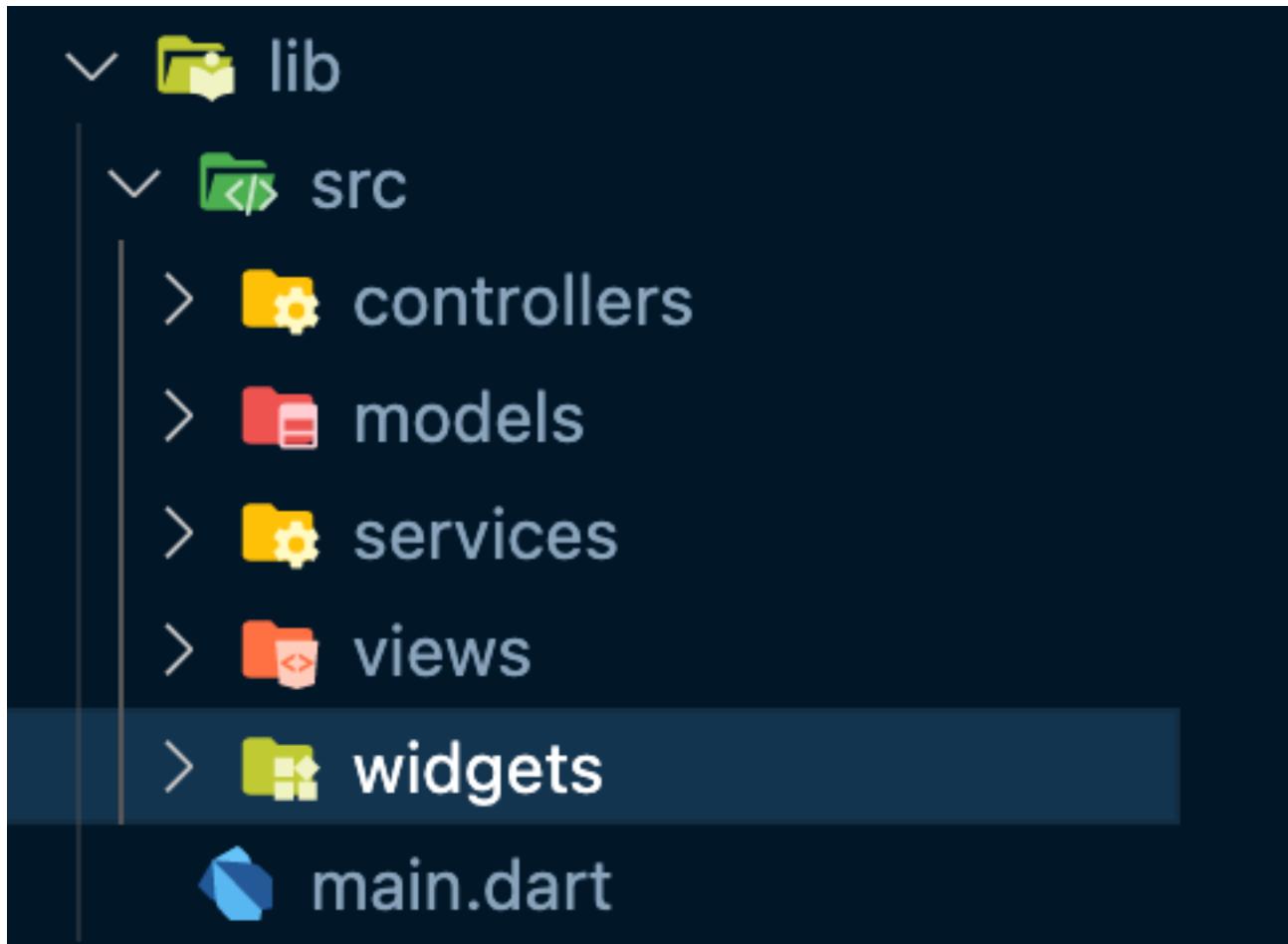
1. Dio : untuk memanggil url
2. Http : untuk memanggil url
3. get : untuk state management
4. cache network image : untuk menyimpan image kedalam app

caranya masih sama seperti diatas. hanya yg berbeda adalah nama & versi package yang kita tambahkan.



2. Get Data From API

Untuk membuat komunikasi dengan server, kita akan menggunakan berbagai API yang memerlukan beberapa jenis metode HTTP untuk dieksekusi. Jadi kita akan membuat kelas `HttpService`, yang akan membantu kita berkomunikasi dengan server kita. Di dalam folder `services`. buat file bernama `http_service.dart`. tapi sebelum itu, mari kita buat struktur project kita seperti berikut:



berikut adalah potongan kode yang sudah saya buat.

```
1 import 'package:http/http.dart' as http;
2
3 class HttpService {
4     static Future<List<ProductsModel>> fetchProducts() async {
5         var response =
6             await http.get(Uri.parse("https://fakestoreapi.com/products"));
7         if (response.statusCode == 200) {
8             var data = response.body;
9             return productsModelFromJson(data);
10        } else {
11            throw Exception();
12        }
13    }
14 }
15 }
```

Kode ini cukup jelas. kita membuat kelas `HttpService` dan kita membuat metode statis yang disebut `fetchProducts()`. Kita membuat permintaan HTTP GET ke URL yang merupakan API Fakeshop. Jika kita berhasil mencapai endpoint, maka kita akan mendapatkan kode respon (`statusCode`) 200 dan akan mengembalikan objek atau kita akan melempar `Exception` apabila gagal. Oke, kita jadi tahu tentang kelas dan fungsi dan apa fungsinya **tapi apa itu `List<ProductionModel>` dan `productsModelFromJson()` ?** kita akan membuat model untuk menjawab pertanyaan ini.

3. Transform data to model.

Sama seperti bahasa pemrograman pada umumnya pasti ada yang namanya pengolahan data yang berbentuk json termasuklah itu di Flutter. Sebelum kita mempelajari bagaimana melakukan Rest API Call di Flutter alangkah lebih baiknya kita memahami penggunaan dasar json dimana, pada artikel ini kita fokus tentang bagaimana kita melakukan parsing json secara manual. contoh Json :



```
1 {  
2   "name": "Agung Wahyudi",  
3   "age": 30  
4 }
```

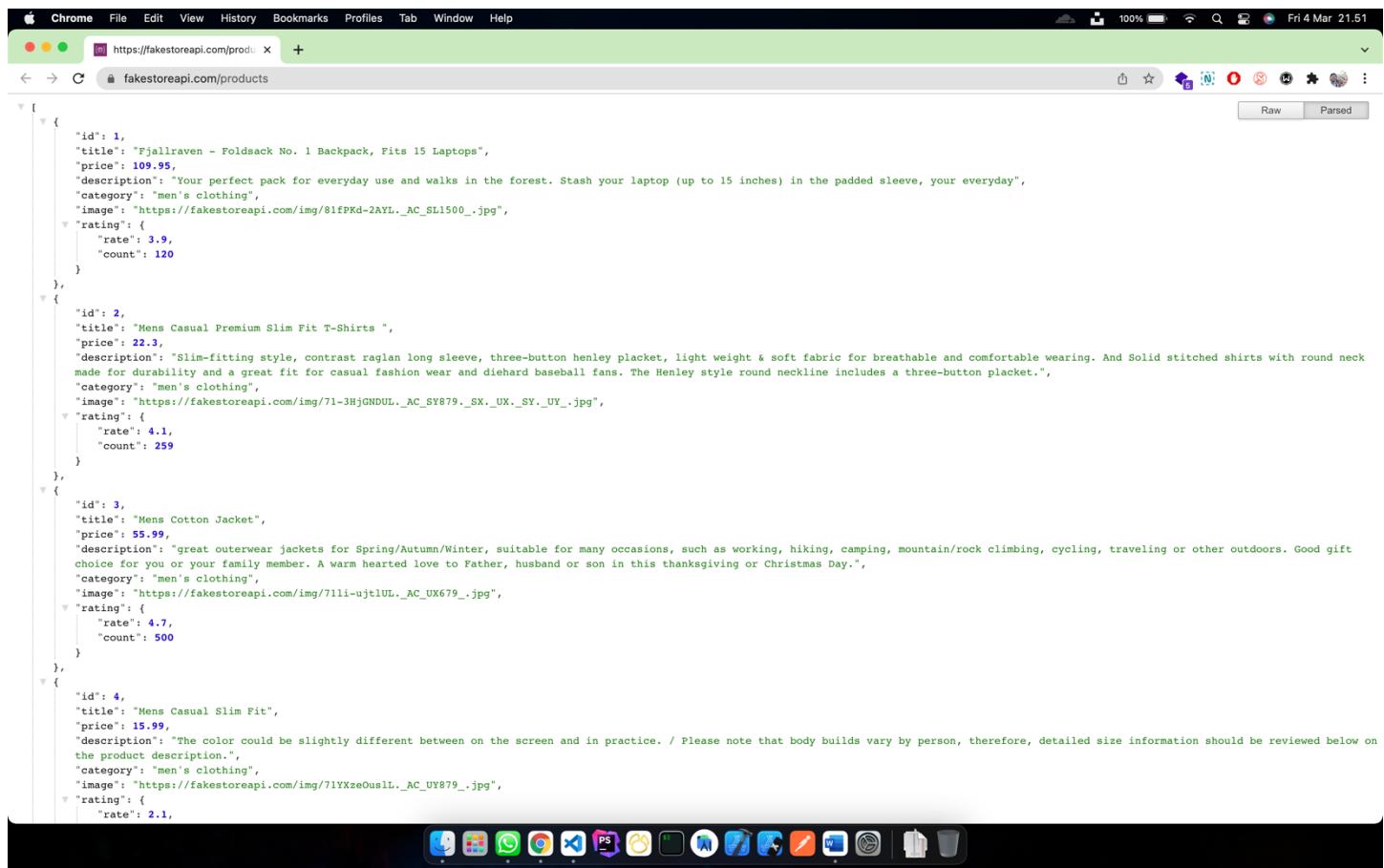
Lalu, kita buat class model-nya seperti berikut:



```
class Sample {  
  String name;  
  int age;  
  Sample({this.name, this.age});  
  @override  
  String toString() {  
    return 'Sample{name: $name, age: $age}';  
  }  
  factory Sample.fromJson(Map<String, dynamic> json) {  
    return Sample(  
      name: json["name"],  
      age: json["age"]  
    );  
  }  
}
```

Pada class model tersebut, bisa kita lihat terdapat 2 variable yaitu variable **name** dan **age**. Lalu, kita juga ada buat constructor-nya dan method **toString()**-nya. Selain itu, ada satu lagi method yang mungkin asing kita lihat untuk pertama kalinya yaitu method **fromJson()**. Sebenarnya method ini bernama Named Constructor di Dart. Named Constructor tersebut kita buat untuk mengubah data json kedalam bentuk class model. Lalu, kenapa kita pakai type **dynamic** untuk value di Map-nya? Karena, kita tidak pernah tahu didalam json tersebut kira-kira type apa saja yang bakalan tersedia. Contohnya seperti json yang diatas dimana, disitu ada 2 type yang tersedia yaitu type string dan number.

Nah bagaimana dengan response dari url yang sudah kita tentukan di file **http_service.dart** ?

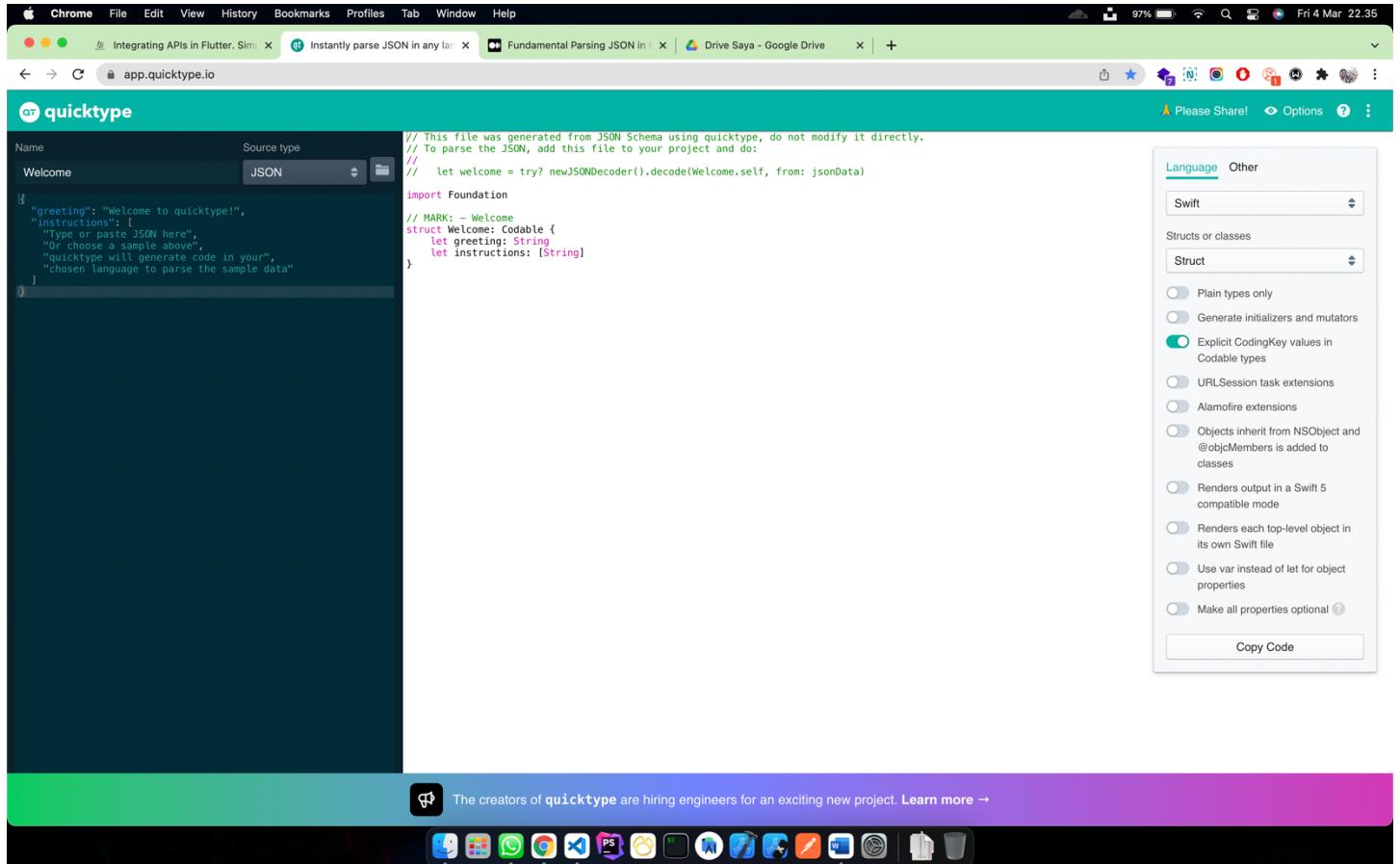


The screenshot shows a Chrome browser window with the URL <https://fakestoreapi.com/products> in the address bar. The page content displays a JSON array of product items. Each item has an id, title, price, description, category, image URL, and a rating object with rate and count properties. The JSON is formatted with collapsible sections for each item.

```
[{"id": 1, "title": "Fjallraven - Foldsack No. 1 Backpack, Fits 15 Laptops", "price": 109.95, "description": "Your perfect pack for everyday use and walks in the forest. Stash your laptop (up to 15 inches) in the padded sleeve, your everyday", "category": "men's clothing", "image": "https://fakestoreapi.com/img/81fPKd-2AYL._AC_SL1500_.jpg", "rating": { "rate": 3.9, "count": 120 }, "id": 2, "title": "Mens Casual Premium Slim Fit T-Shirts ", "price": 22.3, "description": "Slim-fitting style, contrast raglan long sleeve, three-button henley placket, light weight & soft fabric for breathable and comfortable wearing. And Solid stitched shirts with round neck made for durability and a great fit for casual fashion wear and diehard baseball fans. The Henley style round neckline includes a three-button placket.", "category": "men's clothing", "image": "https://fakestoreapi.com/img/71-HjGNDUL._AC_SX879_.SX._UX._SY._UY_.jpg", "rating": { "rate": 4.1, "count": 259 }, "id": 3, "title": "Mens Cotton Jacket", "price": 55.99, "description": "great outerwear jackets for Spring/Autumn/Winter, suitable for many occasions, such as working, hiking, camping, mountain/rock climbing, cycling, traveling or other outdoors. Good gift choice for you or your family member. A warm hearted love to Father, husband or son in this thanksgiving or Christmas Day.", "category": "men's clothing", "image": "https://fakestoreapi.com/img/71liI-uJtLUL._AC_UX679_.jpg", "rating": { "rate": 4.7, "count": 500 }, "id": 4, "title": "Mens Casual Slim Fit", "price": 15.99, "description": "The color could be slightly different between on the screen and in practice. / Please note that body builds vary by person, therefore, detailed size information should be reviewed below on the product description.", "category": "men's clothing", "image": "https://fakestoreapi.com/img/71YXzeOus1L._AC_UY879_.jpg", "rating": { "rate": 2.1, "count": 100 }]
```

apakah kita akan membuatnya secara manual ? pasti agak capek & memeras keringat. dimana kita harus tahu dari posisi yang benar dalam memberikan nama variable.

Saat berurusan dengan API, kita mungkin mendapatkan sejumlah besar data dan yang mungkin memiliki banyak bidang sehingga mengkodekan setiap bidang JSON ke dalam Objek Dart (ini juga disebut parsing JSON) akan berguna. Untuk mengatasi masalah ini akan menggunakan situs web terkenal bernama **quicktype.io** yang mengubah respons JSON kami menjadi Objek Dart.



Buat file bernama **products_model.dart** di dalam folder **models**, kemudian akses API Fakestore kita dengan postman atau cukup salin-tempel tautan ke browser. kamu akan mendapatkan respons JSON cukup salin seluruh respons dan tempel ke quicktype.io. untuk lebih jelasnya, silahkan kalian tonton video [anime-wibu-kelas-berat](#) tutorial yang sudah saya siapkan.

URL VIDEO : <https://bit.ly/jsontomodel>

kemudian sisipkan atau masukan kode :

```
import 'package:data_api/models/products_model.dart';
```

di file **http_service.dart** yang sudah kita buat pada posisi paling atas.

* ganti **:data_api** dengan nama project kalian. contoh : **package:fake_shop**

4. State Management.

Setiap kali data berubah, kita harus memperbarui UI aplikasi kita sesuai dengan kebutuhan. Misalnya, ketika kita membuat permintaan data dari URL, kita harus menunjukkan **PROGRESSBAR** pengguna hingga permintaan data selesai, setelah selesai, kita harus menunjukkan UI yang sesuai. Jika permintaan gagal, kita harus menunjukkan pesan yang sesuai kepada pengguna. Di aplikasi ini, kita akan menampilkan dua status, **PROGRESSBAR** saat melakukan permintaan data ke URL, setelah selesai akan menampilkan data yang kita ambil dan memperbarui UI. Oke, sekarang bagaimana kita akan melakukannya? Mari kita buat file bernama **product_controller.dart** di dalam folder **controllers**.

```
● ● ●

import 'package:data_api/services/http_service.dart';
import 'package:get/get.dart';

class ProductController extends GetxController {
    var isLoading = true.obs;
    var productList = [].obs;

    @override
    void onInit() {
        fetchProducts();
        super.onInit();
    }

    void fetchProducts() async {
        try {
            isLoading(true);
            var products = await HttpService.fetchProducts();
            productList.value = products;
        } finally {
            isLoading(false);
        }
    }
}
```

Seperti yang saya sebutkan, kita menggunakan **GetX** untuk State Management, kita meng-**extends** kelas GetxController yang memberi metode **onInit()** . Ini adalah metode yang pertama kali dipanggil setiap kali kita membuat kelas **instance objek**. kita telah mendeklarasikan dua variabel, satu adalah **boolean (isLoading)** dan satu lagi adalah **list (productList)**. Perhatikan bahwa kita telah menambahkan **.obs** ke variabel yang berarti dapat diamati. Itu berarti kedua variabel ini dapat diperbarui dan kita harus mengamatinya. Ini semua hal GetX tidak perlu memahami apa yang terjadi. GetX melakukan semua ini untuk kita. kita telah menulis metode **fetchProducts()** di mana kita membuat permintaan data ke URL. Sebelum membuat permintaan ke URL, kita memperbarui variabel **boolean** kita **isLoading** ke **true** sehingga UI kita dapat memperlihatkan **PROGRESSBAR** dan di blok akhirnya kita akan memperbarui variabel yang sama ke **false** sehingga UI dapat mengetahui data telah diambil dari URL dan memperbarui UI yang sesuai.

5. its time to creating f*ck*ng UI !

sebelum kita memulainya, mari kita rubah code yang ada di main.dart menjadi berikut.



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

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

class MyApp extends StatelessWidget {
  const MyApp({Key? key}) : super(key: key);

  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      home: Scaffold(
        appBar: AppBar(
          title: const Text('Fake Shop'),
          backgroundColor: Colors.indigo,
        ),
        body: Container(),
      ),
    );
  }
}
```

agar kita bisa fokus dalam pembuatan aplikasi. dan berikut adalah hasil dari debug-nya.

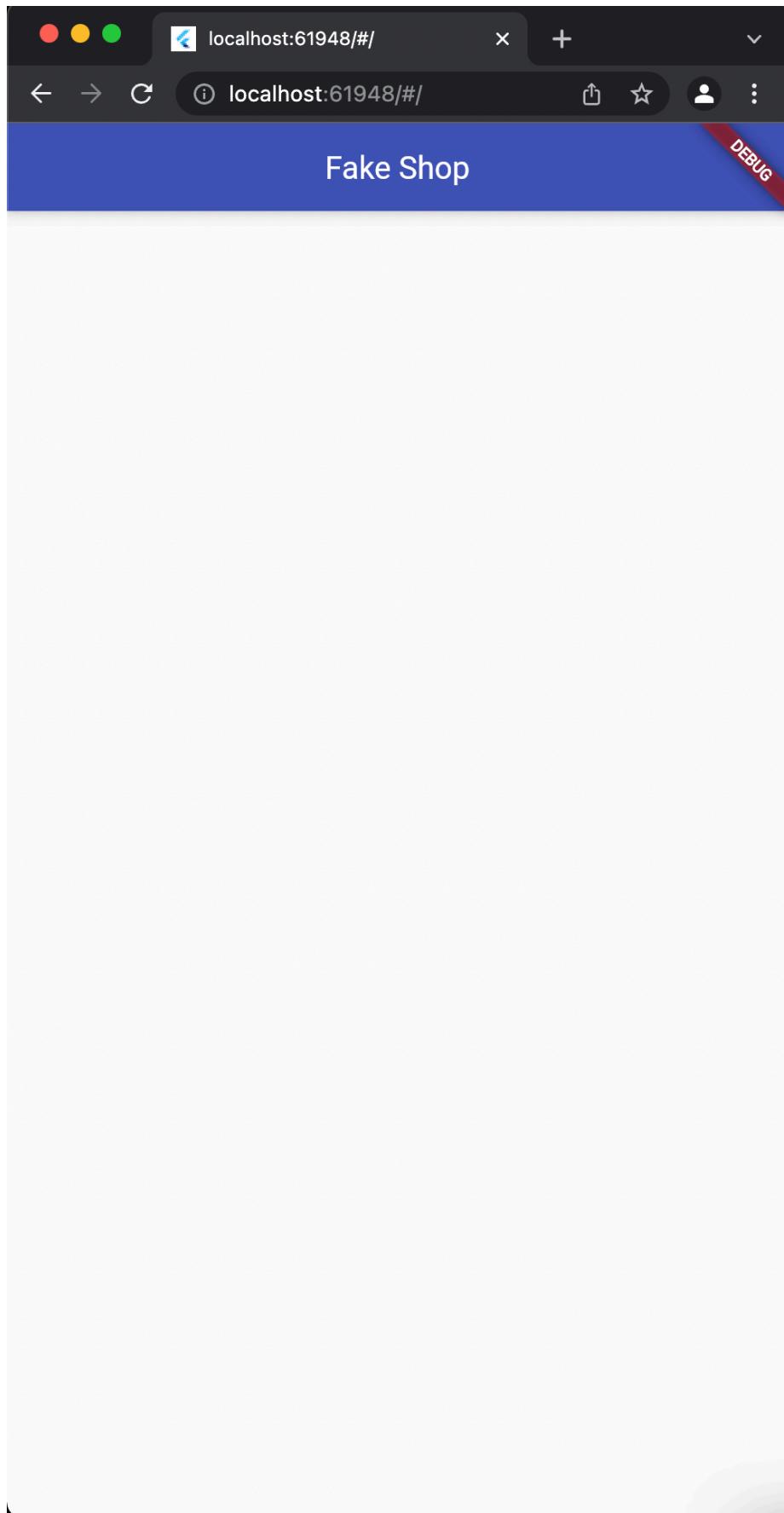
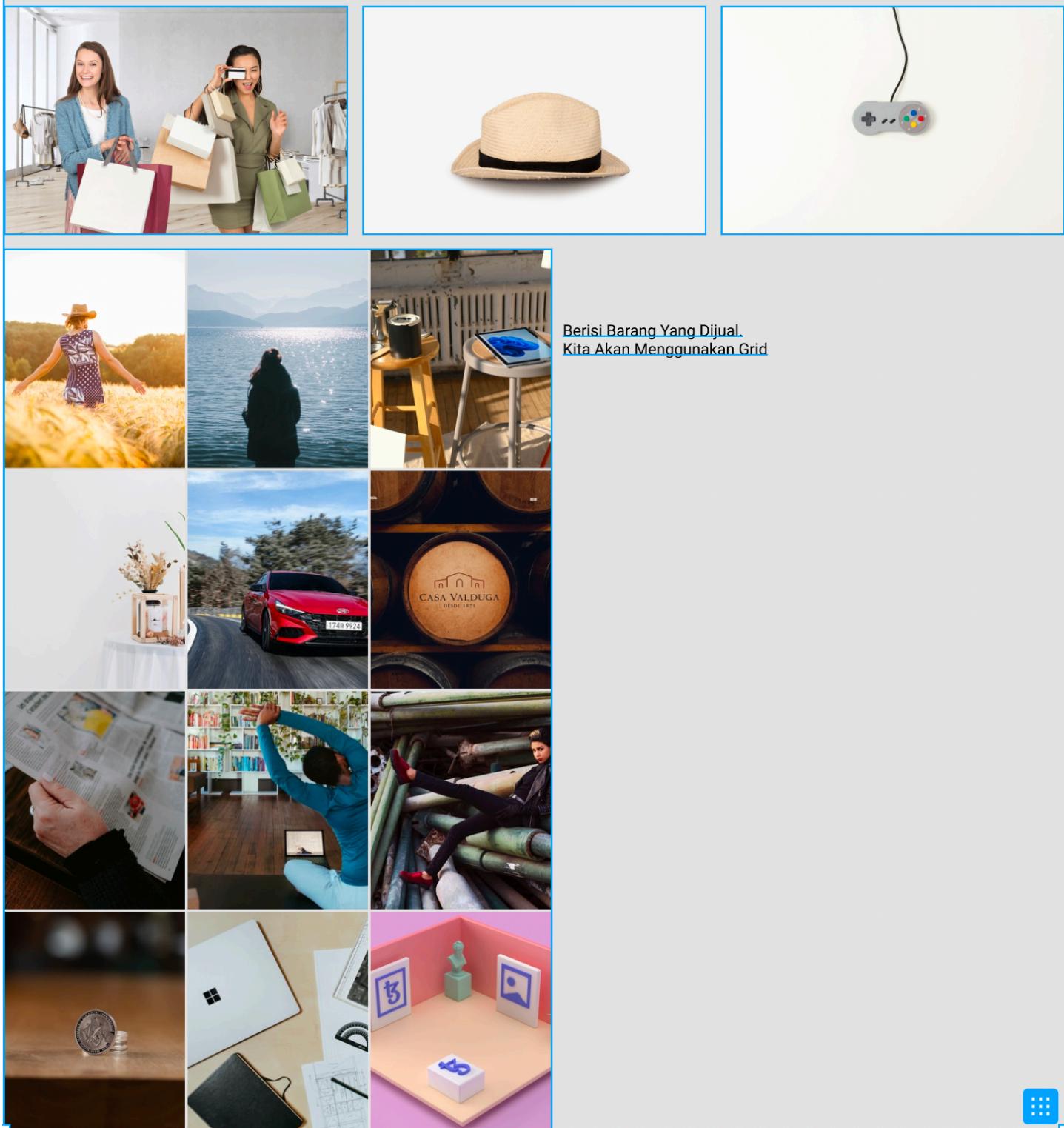


image slider yang berisi foto-foto produk yang dijual

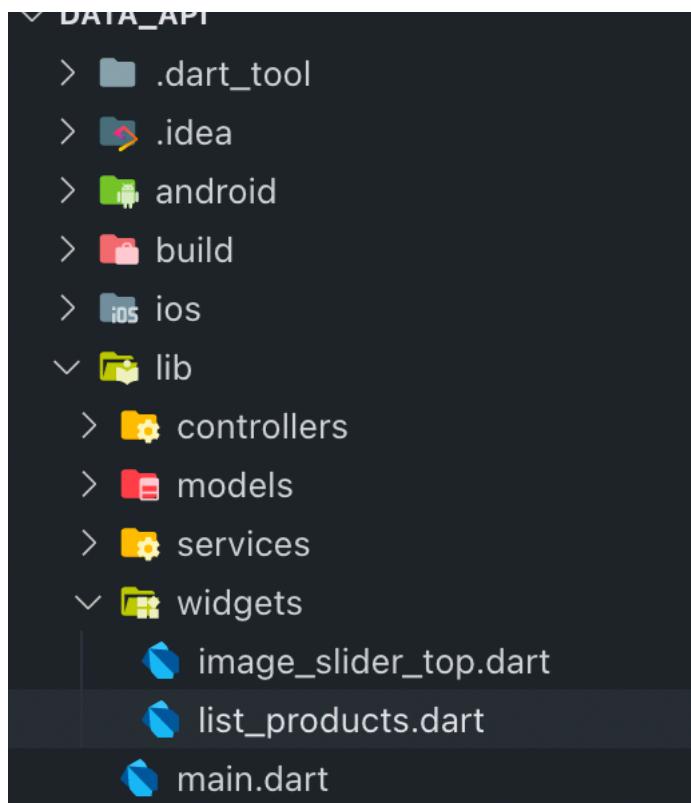


kita akan membuat layout seperti diatas. kalau kita cermati, kita hanya membutuhkan 2 widget.

yaitu:

1. widget untuk menampung image slider
2. widget Gridview untuk menampung data product dari URL yang telah kita tentukan.

oleh sebab itu, kita akan membuat 2 file dart untuk menampung masing-masing widget diatas.



berikut ada baris kode untuk menampung image slider

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

class ImageListPage extends StatefulWidget {
  const ImageListPage({Key? key}) : super(key: key);

  @override
  State<ImageListPage> createState() => _ImageListPageState();
}

class _ImageListPageState extends State<ImageListPage> {
  @override
  Widget build(BuildContext context) {
    // Kita Menggunakan Widget "CachedNetworkImage"
    // yang berguna untuk mempercepat loading data image
    // dari internet
    return Container(
      color: Colors.white,
      height: 250,
      child: ListView(
        scrollDirection: Axis.horizontal,
        children: [
          Padding(
            padding: const EdgeInsets.all(8.0),
            child: CachedNetworkImage(
              imageUrl:
                'https://fakestoreapi.com/img/81fPKd-2AYL._AC_SL1500_.jpg',
            ),
          ),
          Padding(
            padding: const EdgeInsets.all(8.0),
            child: CachedNetworkImage(
              imageUrl:
                'https://fakestoreapi.com/img/71-3HjGNDUL._AC_SY879._SX._UX._SY._UY_.jpg',
            ),
          ),
          Padding(
            padding: const EdgeInsets.all(8.0),
            child: CachedNetworkImage(
              imageUrl:
                'https://fakestoreapi.com/img/71pWzhDJNwL._AC_UL640_QL65_ML3_.jpg',
            ),
          ),
          Padding(
            padding: const EdgeInsets.all(8.0),
            child: CachedNetworkImage(
              imageUrl:
                'https://fakestoreapi.com/img/61mtL65D4cL._AC_SX679_.jpg',
            ),
          ),
        ],
      );
    }
}
```

setelah itu, kita panggil widget **ImageListPage** di file **main.dart**

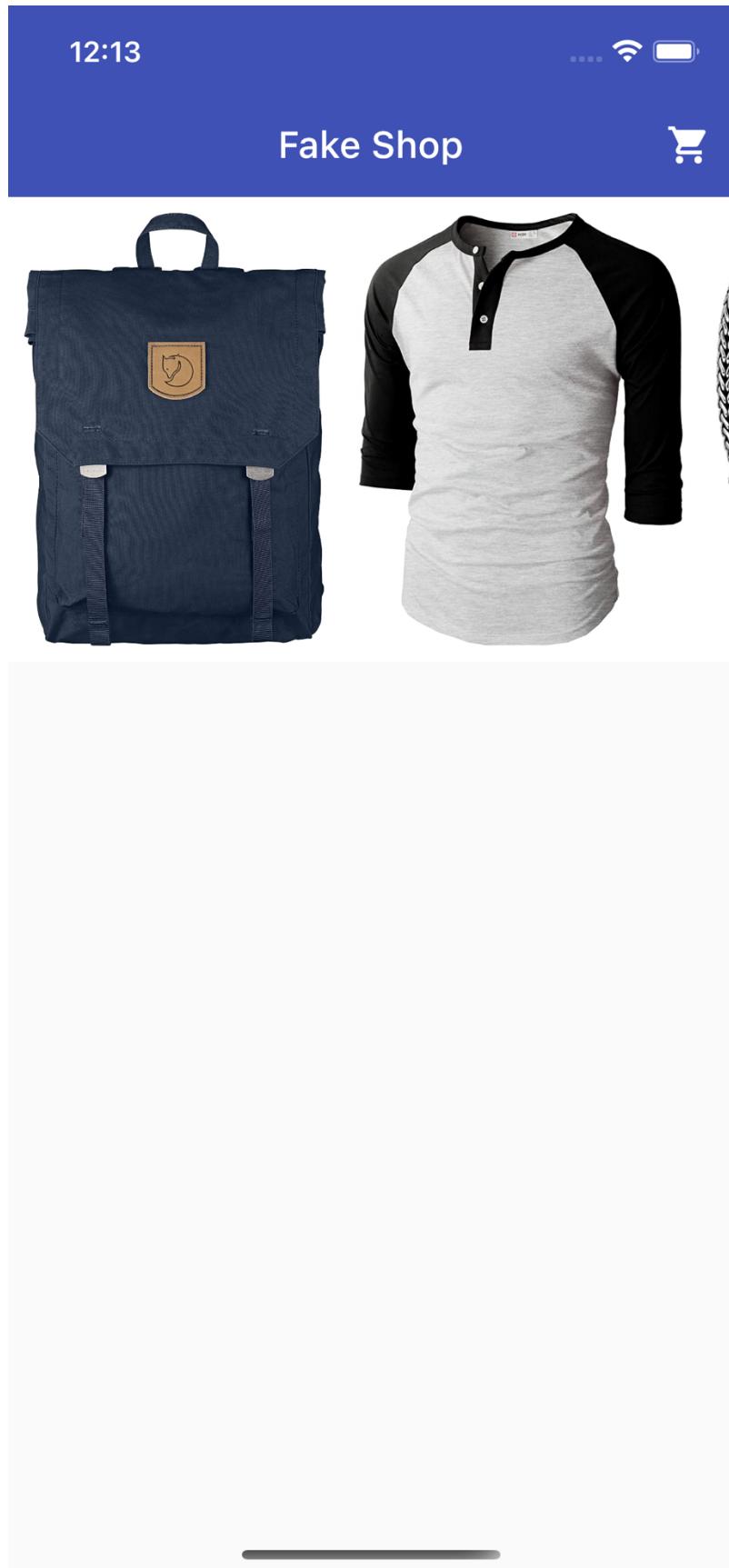
```
import 'package:data_api/widgets/image_slider_top.dart';
import 'package:flutter/material.dart';

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

class MyApp extends StatelessWidget {
  const MyApp({Key? key}) : super(key: key);

  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      debugShowCheckedModeBanner: false,
      home: Scaffold(
        appBar: AppBar(
          elevation: 0,
          title: const Text('Fake Shop'),
          backgroundColor: Colors.indigo,
          actions: [
            IconButton(
              onPressed: () {},
              icon: const Icon(Icons.shopping_cart),
            ),
          ],
        ),
        body: Column(
          children: const [
            ImageListPage(), //ini adalah clas yg sudah kita buat tadi
          ],
        ),
      );
  }
}
```

Hasil Debug:



selanjutnya, kita akan membuat sebuah widget untuk menampung data dari internet. kita buka file **list_products.dart** & isikan code sebagai berikut : (menggunakan stateless)



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

class ListProducts extends StatelessWidget {
  const ListProducts({Key? key}) : super(key: key);

  @override
  Widget build(BuildContext context) {
    return Container();
  }
}
```

kemudian tambahkan code berikut di dalam class **ListProduct**



```
final ProductController productController = Get.put(ProductController());
```

kode diatas berfungsi sebagai variable untuk menampung data dari ProductController yang telah kita buat.

mari kita buat code untuk menampilkan data-nya sekarang.



```
// ignore_for_file: prefer_const_constructors

import 'package:cached_network_image/cached_network_image.dart';
import 'package:data_api/controllers/product_controller.dart';
import 'package:flutter/material.dart';
import 'package:get/get.dart';

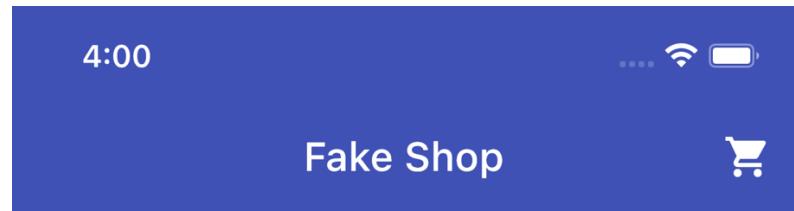
class ListProducts extends StatelessWidget {
    final ProductController productController = Get.put(ProductController());
    ListProducts({Key? key}) : super(key: key);

    @override
    Widget build(BuildContext context) {
        return Expanded(
            child: Obx(
                () {
                    // Ox disini sebagai statemanagmen dari GetX
                    if (productController.isLoading.value) {
                        return Center(
                            child: const CircularProgressIndicator(),
                        );
                    }
                    // Setelah berhasil, maka akan menampilkan data yang disimpan
                    // di widget ListView.builder
                    else {
                        return ListView.builder(
                            itemCount: productController.productList.length,
                            itemBuilder: (context, index) {
                                return Padding(
                                    padding: const EdgeInsets.all(8.0),
                                    child: Container(
                                        height: 100,
                                        color: Colors.amber,
                                    ),
                                );
                            },
                        );
                    }
                }
            )
        );
    }
}
```

kemudian kita panggil class ListProduct di main.dart

```
...  
Padding(  
  padding: const EdgeInsets.all(16),  
  child: Row(  
    children: const [  
      Expanded(  
        child: Text(  
          "Produk Saat Ini",  
          style: TextStyle(  
            fontFamily: "avenir",  
            fontSize: 25,  
            fontWeight: FontWeight.w900),  
        ),  
        ),  
      ],  
    ),  
  ),  
ListProducts(), // ini adalah class yang kita panggil  
...
```

Hasil Debug :



Produk Saat Ini



bisa kita lihat, data dari model berhasil dimuat, kita akan memberi informasi product di masing-masing container yang berwarna amber. mari kita buat !!



```
...
return Padding(
    padding: const EdgeInsets.all(8.0),
    child: Container(
        decoration: BoxDecoration(
            borderRadius: BorderRadius.circular(10),
            color: Colors.indigo,
        ),
        height: 200,
        child: Row(
            crossAxisAlignment: CrossAxisAlignment.start,
            children: [
                SizedBox(
                    width: 100,
                    child: Padding(
                        padding: const EdgeInsets.all(8.0),
                        child: CachedNetworkImage(
                            imageUrl:
                                productController.productList[index].image,
                            ),
                    ),
                ),
                Expanded(
                    child: Container(
                        color: Colors.amber,
                        child: Column(
                            children: [
                                Text(
                                    productController.productList[index].title,
                                    textAlign: TextAlign.center,
                                    style: TextStyle(
                                        fontSize: 15,
                                    ),
                                ),
                                ],
                            ),
                ),
            ],
        );
    },
);
```

4:16



Fake Shop



New Product



Produk Saat Ini



Fjallraven - Foldsack No. 1 Backpack,
Fits 15 Laptops



Mens Casual Premium Slim Fit T-Shirts

Mens Cotton Jacket

selanjutnya kita akan menambahkan kategori dari masing-masing produk.



...

```
Text(  
    productController.productList[index].title,  
    style: TextStyle(  
        fontSize: 15,  
    ),  
)  
,  
SizedBox(  
    height: 10,  
)  
,  
Text(  
    'Kategori Produk : ' +  
    productController  
        .productList[index].category,  
)  
...
```

setelah itu, kita akan menambahkan informasi harga.

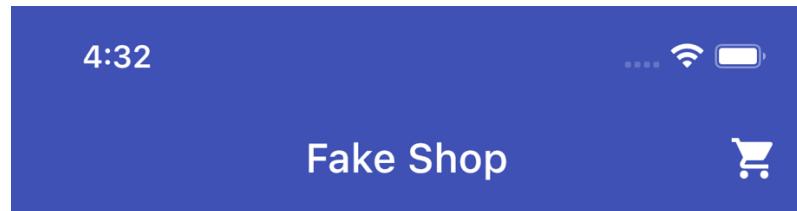
```
...  
SizedBox(  
    height: 10,  
)  
Text(  
    'Rp.' +  
        productController.productList[index].price  
            .toString(),  
    style: TextStyle(fontSize: 18),  
)  
...  
...
```

dan terakhir, kita akan menambahkan rating dari produk.

```
...
SizedBox(
height: 10,
),
if (productController
    .productList[index].rating != null)
Container(
decoration: BoxDecoration(
color: Colors.green,
borderRadius:
    BorderRadius.circular(12),
),
padding: const EdgeInsets.symmetric(
    horizontal: 4,
    vertical: 2,
),
child: Row(
mainAxisSize: MainAxisSize.min,
children: [
Text(
    productController
        .productList[index]
        .rating!
        .rate
        .toString(),
style: const TextStyle(
    color: Colors.white),
),
const Icon(
    Icons.star,
    size: 16,
    color: Colors.white,
),
],
),
...

```

Hasil Debug :



New Product



Produk Saat Ini



Fjallraven - Foldsack No. 1
Backpack, Fits 15 Laptops

Kategori Produk : men's clothing

Rp.109.95

3.9★



Mens Casual Premium Slim Fit T-
Shirts

Kategori Produk : men's clothing

Rp.22.3

4.1★

Untuk selanjutnya, silahkan kalian berkreasi dengan tampilan dari produk yang sudah berhasil dibuat seperti diatas.