**Week-10 :**

**A) Fetch data from a REST API**

**B)Display the fetched data in a meaningful way in the UI.**

import 'dart:async';

import 'dart:convert';

import 'package:flutter/material.dart';

import 'package:http/http.dart' as http;

import 'package:cached\_network\_image/cached\_network\_image.dart';

class Product {

final int id;

final String title;

final double price;

final String description;

final String category;

final String image;

final double rating;

Product({

required this.id,

required this.title,

required this.price,

required this.description,

required this.category,

required this.image,

required this.rating,

});

factory Product.fromJson(Map<String, dynamic> json) {

return Product(

id: json['id'],

title: json['title'],

price: json['price'].toDouble(),

description: json['description'],

category: json['category'],

image: json['image'],

rating: json['rating']['rate'].toDouble(),

);

}

}

class ProductCard extends StatelessWidget {

final Product product;

const ProductCard({super.key,required this.product});

@override

Widget build(BuildContext context) {

return Card(

child: Padding(

padding: const EdgeInsets.all(8.0),

child: Column( // this is the column

children: [

AspectRatio(

aspectRatio: 1, // this is the ratio

child: CachedNetworkImage( // this is to fetch the image

imageUrl: product.image,

fit: BoxFit.cover,

),

),

ListTile(

title: Text(product.title),

subtitle: Text('${product.price} \$'),// this is fetch the price from the api

trailing: Row(

mainAxisSize: MainAxisSize.min,

children: [

const Icon(Icons.star, color: Colors.orange),// this will give the rating

Text('${product.rating}'),

],

),

),

],

),

),

);

}

}

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

class MyApp extends StatelessWidget {

const MyApp({super.key});

// This widget is the root of your application.

@override

Widget build(BuildContext context) {

return MaterialApp(

title: 'Flutter Demo',

theme: ThemeData(

// This is the theme of your application.

//

// TRY THIS: Try running your application with "flutter run". You'll see

// the application has a blue toolbar. Then, without quitting the app,

// try changing the seedColor in the colorScheme below to Colors.green

// and then invoke "hot reload" (save your changes or press the "hot

// reload" button in a Flutter-supported IDE, or press "r" if you used

// the command line to start the app).

//

// Notice that the counter didn't reset back to zero; the application

// state is not lost during the reload. To reset the state, use hot

// restart instead.

//

// This works for code too, not just values: Most code changes can be

// tested with just a hot reload.

colorScheme: ColorScheme.fromSeed(seedColor: Colors.deepPurple),

useMaterial3: true,

),

//home: ProductsScreen(),

home: const ProductListScreen(),

);

}

}

class ProductListScreen extends StatefulWidget {

const ProductListScreen({super.key});

@override

ProductListScreenState createState() => ProductListScreenState();

}

class ProductListScreenState extends State<ProductListScreen> {

List<Product> products = [];

@override

void initState() {

super.initState();

fetchProducts();

}

Future<void> fetchProducts() async {

final response = await http.get(Uri.parse('https://fakestoreapi.com/products'));

if (response.statusCode == 200) {

List<dynamic> jsonData = json.decode(response.body);

setState(() {

products = jsonData.map((data) => Product.fromJson(data)).toList();

});

} else {

// Handle error if needed

print('Failed to load products');

}

}

@override

Widget build(BuildContext context) {

return Scaffold(

appBar: AppBar(

title: const Text('Product List'),

),

body: ListView.builder(

// this give th length of item

itemCount: products.length,

itemBuilder: (context, index) {

// here we card the card widget

// which is in utils folder

return ProductCard(product: products[index]);

},

),

);

}

}

Run this command:

With Flutter Terminal:

$ flutter pub add cached\_network\_image

This will add a line like this to your package's pubspec.yaml (and run an implicit flutter pub get):

In pubspec.yml file add

dependencies:

Line No 37: cupertino\_icons: ^1.0.6

Line No 38: http: ^1.2.1

Line No 39: c ached\_network\_image: ^3.4.1





