Nama : Hadi Iswanto
NIM : 181011401680
Kelas : 06TPLE017

Jawaban UAS Mobile Programing

1. Mobile programming adalah proses pembuatan aplikasi untuk perangkat mobile baik aplikasi yang bersifat offline maupun online.

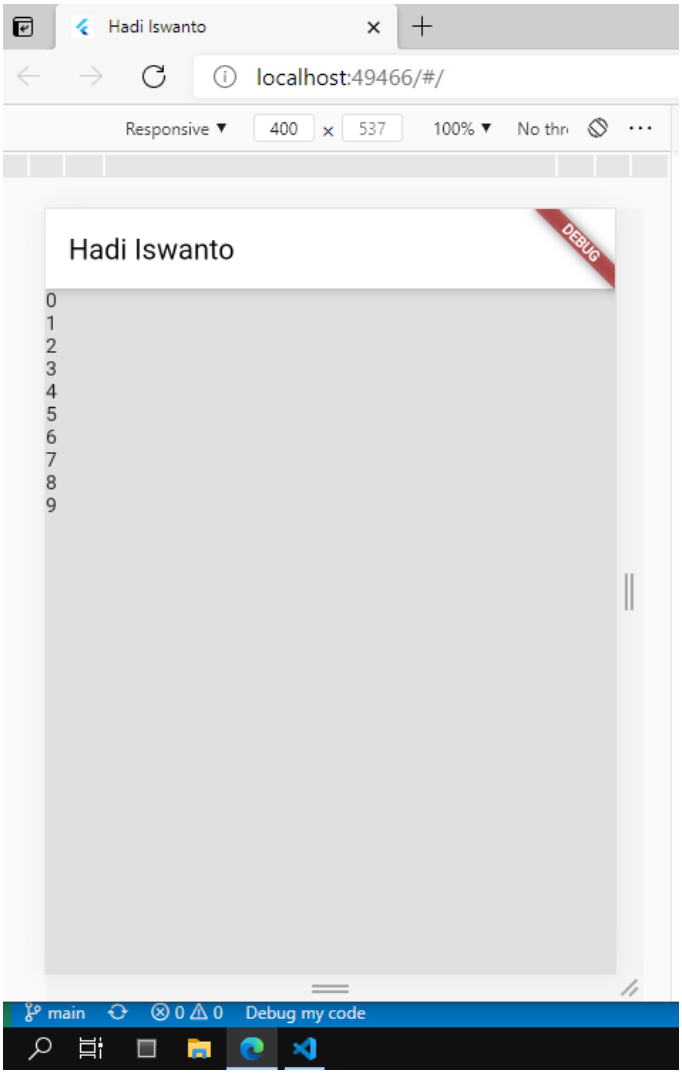
2. User Interface adalah tampilan visual sebuah produk yang menjembatani sistem dengan pengguna *(user).* Tampilan UI dapat berupa bentuk, warna, dan tulisan yang didesain semenarik mungkin. Secara sederhana, UI adalah bagaimana tampilan sebuah produk dilihat oleh pengguna.

3. API atau Application Programming Interface adalah sebuah interface yang dapat menghubungkan aplikasi satu dengan aplikasi lainnya. Jadi, API berperan sebagai perantara antar berbagai aplikasi berbeda, baik dalam satu platform yang sama atau lintas platform.

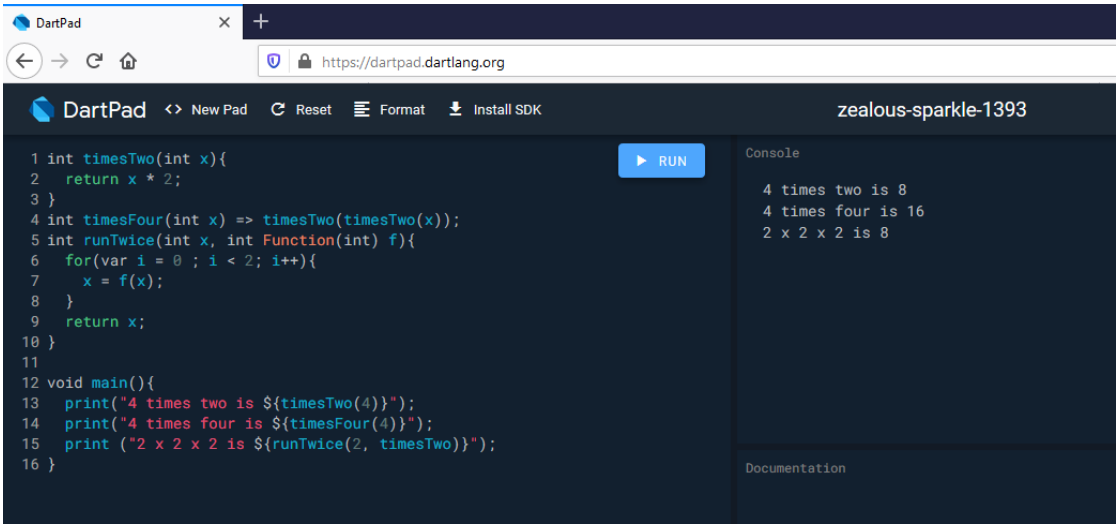
4. ***Aplikasi native*** adalah aplikasi yang dibangun dengan bahasa pemrograman yang spesifik dan hanya dapat digunakan di platform tertentu. Aplikasi *native* bisa juga disebut aplikasi asli. Salah satu kelebihan dari aplikasi *native* adalah UI/UX-nya yang sangat baik. Aplikasi ini hanya dapat digunakan di satu platform dan biaya pengembangannya pun relatif tinggi. ***Aplikasi hybrid*** adalah aplikasi yang dibuat menggunakan bahasa pemrograman web dengan bantuan *software development kit* (SDK) *native* dari berbagai platform sistem operasi. Aplikasi *hybrid* menggabungkan elemen aplikasi *native* dan web, seperti ditulis Gist. Selain itu, jenis aplikasi ini juga menggabungkan berbagai fitur sistem operasi. Jadi jika ingin aplikasimu dapat digunakan di Android, iOs dan sistem operasi lainnya, kamu bisa memilih aplikasi *hybrid.* Salah satu kelebihan aplikasi *hybrid* adalah biasanya lebih mudah dan lebih cepat untuk dikembangkan. Aplikasi ini juga membutuhkan lebih sedikit *maintenance* atau perawatan. Namun, umumnya performa aplikasi *hybrid* belum bisa mengungguli aplikasi *native.*

5. Fungsi Github adalah suatu platform dimana Anda bisa bekerja bersama-sama dengan rekan dari berbagai berlahan dunia, merencanakan proyek, dan bahkan *tracking* (melacak) pekerjaan Anda. GitHub juga merupakan salah satu storehouse online terbesar di dunia untuk pekerjaan kolaborasi.

6. Manampilkan list dinamis sebanyak 10 data



7. tampilan



8. Tahap Pertama kita akan membuat file halamanJson.dart dalam project kita. Kemudian isi dengan kode di bawah ini.

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

void main() {
  runApp(new MaterialApp(
    title: "My Apps",
    home: new HalamanJson(),
  ));
}

class HalamanJson extends StatefulWidget {
  @override
  _HalamanJsonState createState() => _HalamanJsonState();
}

class _HalamanJsonState extends State {

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: Text("Data JSON"),
      ),
      drawer: DrawerApp(),
      body: Center(
          child: Text("Data JSON")
      ),
    );
  }
}
```

Selanjutnya kita membutuhkan beberapa package diantaranya sebagai berikut. Tambahkan http dependencies dalam file pubspecs.yaml

```
dependencies:
  flutter:
    sdk: flutter

  http: ^0.12.0+1
```

Lalu import dalam file .dart

```
  import 'dart:convert';
  import 'package:http/http.dart' as http;
  import 'dart:async';
```

Selanjutnya kita akan menggunakan **Future** untuk menjalankan http.get.

```
List datadariJSON;

Future ambildata() async {
  http.Response hasil = await http.get(
    Uri.encodeFull("https://jsonplaceholder.typicode.com/users"),
    headers: {"Accept": "application/json"});

  this.setState(() {
   datadariJSON = json.decode(hasil.body);
  });
}
```

Pada contoh **Future** di atas, sebelumnya kita telah membuat List terlebih dahulu yang bernama **datadariJSON** yang dimaksudkan akan menampung data yang akan diparsing dari url. Jadi sebelum Future **ambildata** dijalankan maka List **datadariJSON** masih bernilai **null**. setState di sini berfungsi untuk merubah state dari **datadariJSON** yang tadinya null menjadi berisi data dari hasil parsing. Lalu untuk menjalankan Future ambil data kita menggunakan **initState.**

```
@override
void initState() {
  this.ambildata();
}
```

Singkatnya Full Code-nya akan menjadi seperti di bawah ini. Di sini saya menampilkan List **datadariJSON** menggunakan **ListView.builder** dimana setiap ListTile nya dapat diubah secara custom sesuai keinginan masing-masing. Untuk melihat contoh Listview custom **di sini.**

```
import 'dart:convert';
import 'package:flutter/material.dart';
import 'package:http/http.dart' as http;
import 'dart:async';

void main() {
 runApp(new MaterialApp(
  title: "My Apps",
  home: new HalamanJson(),
 ));
}

class HalamanJson extends StatefulWidget {
 @override
 _HalamanJsonState createState() => _HalamanJsonState();
}

class _HalamanJsonState extends State {
```

```
List datadariJSON;


Future ambildata() async {
  http.Response hasil = await http.get(
      Uri.encodeFull("https://jsonplaceholder.typicode.com/users"),
      headers: {"Accept": "application/json"});

  this.setState(() {
    datadariJSON = json.decode(hasil.body);
  });
}

@override
void initState() {
  this.ambildata();
}

@override
Widget build(BuildContext context) {
  return Scaffold(
    appBar: AppBar(
      title: Text("Data JSON"),
    ),
    body: Container(
      child: ListView.builder(
        itemCount: datadariJSON == null ? 0 : datadariJSON.length,
        itemBuilder: (context, i){
          return ListTile(
            title: Text(datadariJSON[i]['name']),
          );
        }
      ),
    ),
  );
 }
}
```

Saat menampilkan data dari List kita seperti menampilkan array pada umumnya dan disesuaikan dengan struktur dari JSON yang tersedia. Contohnya di sini menampilkan **name** dengan cara **datadariJSON[i]['name'].** Contoh lain jika ingin menampilkan nama jalan maka kita gunakan cara **datadariJSON[i]['address']['street'].**