Modul PBO/Mobile

1. **Overview**

   Mobile programming atau pemrograman aplikasi bergerak telah menjadi salah satu bidang yang berkembang pesat dalam dunia teknologi. Dengan semakin meluasnya penggunaan smartphone di seluruh dunia, permintaan akan aplikasi mobile yang inovatif dan fungsional juga semakin meningkat.

   Mobile programming adalah proses pengembangan aplikasi atau perangkat lunak yang dirancang khusus untuk digunakan pada perangkat mobile, seperti smartphone atau tablet. Aplikasi mobile umumnya dibuat untuk sistem operasi mobile tertentu, seperti Android atau iOS.

   Ada beberapa bahasa pemrograman yang umum digunakan untuk pengembangan aplikasi mobile. Beberapa bahasa pemrograman yang populer untuk mobile programming termasuk:

   - **Java**: Bahasa pemrograman Java adalah salah satu yang paling umum digunakan untuk pengembangan aplikasi Android. Java menggunakan platform Android SDK (Software Development Kit) untuk membangun aplikasi Android.

   - **Kotlin**: Kotlin adalah bahasa pemrograman modern yang mendapatkan dukungan resmi dari Google sebagai bahasa pemrograman untuk pengembangan aplikasi Android. Kotlin memiliki sintaks yang lebih ringkas dan lebih aman dibandingkan dengan Java.

   - **Swift**: Swift adalah bahasa pemrograman yang dikembangkan oleh Apple untuk pengembangan aplikasi iOS, macOS, watchOS, dan tvOS. Swift menggantikan Objective-C sebagai bahasa utama untuk pengembangan aplikasi iOS.

   - Objective-C: Meskipun Swift telah menjadi bahasa pemrograman utama untuk pengembangan aplikasi iOS, Objective-C masih digunakan secara luas, terutama untuk memelihara dan memperbarui aplikasi yang sudah ada.

   - **C#:** C# adalah bahasa pemrograman yang digunakan dalam platform Xamarin, yang memungkinkan pengembangan aplikasi lintas platform untuk Android, iOS, dan Windows menggunakan bahasa yang sama.

- **JavaScript**: JavaScript digunakan dalam pengembangan aplikasi mobile lintas platform menggunakan kerangka kerja seperti React Native dan NativeScript. Dengan menggunakan JavaScript, pengembang dapat membuat aplikasi mobile untuk Android dan iOS secara bersamaan.

- **Dart**: Dart adalah bahasa pemrograman yang digunakan dalam kerangka kerja Flutter untuk pengembangan aplikasi mobile lintas platform. Flutter memungkinkan pengembang untuk membuat aplikasi yang berjalan di Android dan iOS dengan antarmuka pengguna native menggunakan satu basis kode.

2. **Tools**

   Dalam pembelajaran ini bahasa yang akan digunakan adalah **Dart** dengan menggunakan framework **Flutter**

Komponen:

- **Widget**: Widget adalah komponen dasar dalam Flutter yang digunakan untuk membangun antarmuka pengguna (UI). Semua elemen visual dalam Flutter, seperti tombol, teks, gambar, layout, animasi, dll., merupakan turunan dari widget. Widget dapat berupa StatelessWidget (tidak berubah) atau StatefulWidget (dapat berubah berdasarkan waktu atau keadaan).

- **Material Components**: Material Design adalah pedoman desain dari Google untuk membangun antarmuka pengguna yang konsisten di berbagai platform. Flutter menyediakan sejumlah besar widget yang diimplementasikan sesuai dengan Material Design, seperti AppBar, Button, Card, dan lain-lain. Ini memungkinkan pengembang untuk dengan mudah membuat aplikasi yang memiliki tampilan dan perilaku yang konsisten dengan aplikasi Google lainnya.

- **Cupertino Components:** Cupertino adalah gaya desain yang digunakan oleh Apple dalam produk-produknya. Flutter juga menyediakan widget yang mengikuti gaya desain Cupertino, seperti CupertinoNavigationBar, CupertinoButton, dan lain-lain. Ini memungkinkan pengembang untuk membuat aplikasi yang memiliki tampilan yang konsisten dengan aplikasi iOS.

- **Layouts**: Flutter menyediakan berbagai macam widget untuk mengatur tata letak (layout) dari elemen-elemen dalam aplikasi. Beberapa contoh layout widget yang umum digunakan termasuk Row, Column, Stack, Container, dan GridView. Layout widget memungkinkan pengembang untuk mengorganisasi dan menata elemen-elemen dalam antarmuka pengguna dengan fleksibilitas yang tinggi.

- **Gestures**: Flutter memiliki widget untuk menangani berbagai macam input pengguna seperti tap, swipe, drag, dll. Widget GestureDetector digunakan untuk menangkap gesture dan menghubungkannya dengan perilaku aplikasi, seperti melakukan navigasi halaman atau mengubah data.

- **Animations:** Animasi adalah bagian penting dari pengalaman pengguna modern. Flutter menyediakan berbagai macam widget untuk membuat animasi, seperti AnimatedContainer, AnimatedOpacity, dan lain-lain. Pengembang juga dapat membuat animasi kompleks menggunakan widget AnimationController dan Tween.

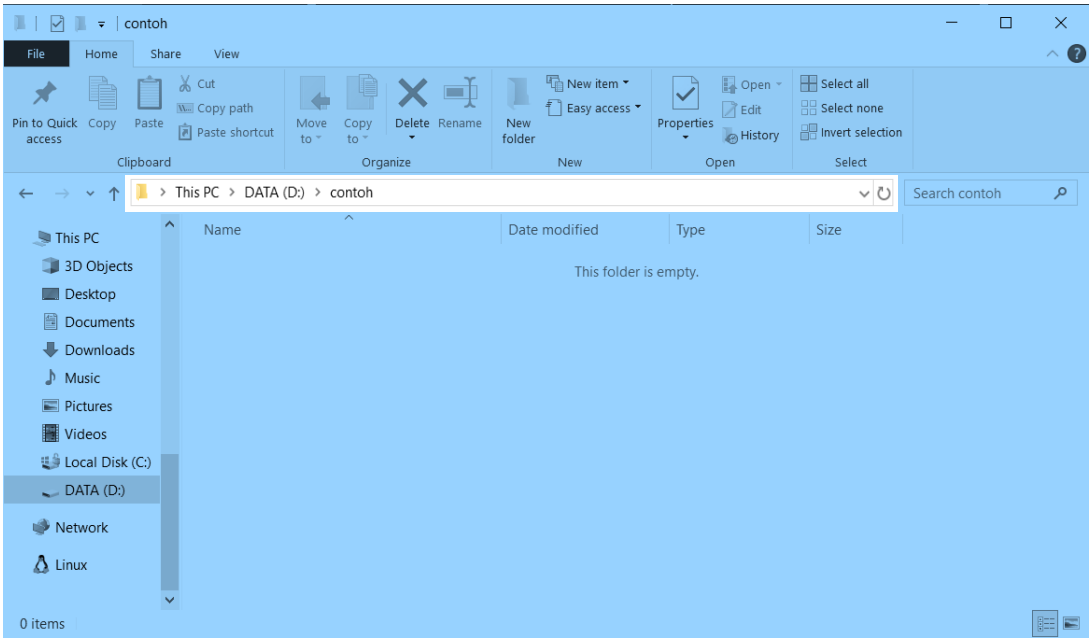
- **Routing & Navigation:** Untuk mengatur navigasi antar halaman dalam aplikasi Flutter, terdapat widget Navigator yang memungkinkan navigasi stack-based. Widget ini dapat digunakan untuk menangani rute (routes) dan mengelola tumpukan navigasi.

- **State Management:** Flutter memiliki pendekatan yang fleksibel terhadap manajemen state. Pengembang dapat menggunakan setState() untuk memperbarui state dalam StatefulWidget atau menggunakan pendekatan manajemen state yang lebih canggih seperti Provider, Bloc (Business Logic Component), Redux, atau GetX untuk mengelola state aplikasi secara lebih terstruktur.

- **Plugins:** Flutter memiliki dukungan yang luas untuk plugin yang memungkinkan pengembang untuk mengakses fitur-fitur platform khusus seperti kamera, lokasi, sensor, dan lain-lain. Plugin-plugin ini memungkinkan aplikasi Flutter untuk berinteraksi dengan perangkat keras dan perangkat lunak di platform yang dituju.

3. Pembuatan Project

1. Untuk memulai pastikan sudah menginstall Flutter
2. Pada windows explorer, buka folder yang akan digunakan untuk membuat project lalu klik pada address bar atau tekan tombol f4 pada keyboard



3. Ketikkan “cmd” lalu tekan enter, maka sebuah command prompt akan terbuka, ketikkan “flutter create nama_project”, tunggu hingga selesai

4. Project sudah siap untuk digunakan

Pembahasan

Secara default flutter akan membuat file Bernama main.dart di dalam folder lib, ada banyak coding untuk keperluan tes di dalam folder ini, namun yang kita butuhkan hanya sampai line 37 saja, silahkan hapus sisanya

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

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 purple 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: const MyHomePage(title: 'Flutter Demo Home Page'),
    );
  }
}

```

Untuk membuat halaman baru, tambahkan file baru di folder lib dengan ekstensi .dart contoh “home.dart”

Ketikkan “stl” pada file home.dart lalu tekan tab untuk membuat stateless widget atau “stful” untuk membuat stateful widget

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

class Home extends StatelessWidget {
  const Home({super.key});

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

Beri nama pada class nya, lalu ubah bagian home di file main.dart sesuai dengan nama class yang dibuat

```dart
        colorScheme: ColorScheme.fromSeed(seedColor: Colors.deepPurple),
        useMaterial3: true,
      ),
      home: const Home(),
    );
  }
}
```

Untuk mengetes silahkan run aplikasi menggunakan tombol f5

1. **Layouting**

   Salah satu widget yang digunakan untuk pengaturan layout di flutter adalah Column dan Row.

   Column akan Menyusun widget secara vertical sedangkan row secara horizontal