

**LAPORAN PRAKTIKUM
PEMROGRAMAN PERANGKAT BERGERAK**

**MODUL XII
MAPS & PLACES**



Disusun Oleh :
Aditya Prabu Mukti/ 2211104037

SE-06-02

Asisten Praktikum :
Muhammad Faza Zulian Gesit Al Barru
Aisyah Hasna Aulia

Dosen Pengampu :
Yudha Islami Sulistya

PROGRAM STUDI S1 REKAYASA PERANGKAT LUNAK
FAKULTAS INFORMATIKA
2024

GUIDED

1. LOGIC

Sebelum kita pergi ke tampilan flutter project, Langkah yang pertama kita lakukan adalah mengatur file services untuk google maps.

A. PUBSPEC.YAML
Sourcecode:

```

name: pertemuan12
description: "A new Flutter project."
# The following line prevents the package from being accidentally
published to
# pub.dev using `flutter pub publish`. This is preferred for private
packages.
publish_to: 'none' # Remove this line if you wish to publish to
pub.dev

# The following defines the version and build number for your
application.
# A version number is three numbers separated by dots, like 1.2.43
# followed by an optional build number separated by a +.
# Both the version and the builder number may be overridden in
flutter
# build by specifying --build-name and --build-number, respectively.
# In Android, build-name is used as versionName while build-number
used as versionCode.
# Read more about Android versioning at
https://developer.android.com/studio/publish/versioning
# In iOS, build-name is used as CFBundleShortVersionString while
build-number is used as CFBundleVersion.
# Read more about iOS versioning at
#
https://developer.apple.com/library/archive/documentation/General/Ref
erence/InfoPlistKeyReference/Articles/CoreFoundationKeys.html
# In Windows, build-name is used as the major, minor, and patch parts
# of the product and file versions while build-number is used as the
build suffix.
version: 1.0.0+1

environment:
  sdk: ^3.5.4

# Dependencies specify other packages that your package needs in
order to work.
# To automatically upgrade your package dependencies to the latest
versions
# consider running `flutter pub upgrade --major-versions`.
Alternatively,
# dependencies can be manually updated by changing the version
numbers below to
# the latest version available on pub.dev. To see which dependencies
have newer
# versions available, run `flutter pub outdated`.
dependencies:
  flutter:
    sdk: flutter

# The following adds the Cupertino Icons font to your application.
# Use with the CupertinoIcons class for iOS style icons.
cupertino_icons: ^1.0.8
google_maps_flutter: ^2.10.0

dev_dependencies:
  flutter_test:
    sdk: flutter

# The "flutter_lints" package below contains a set of recommended
lints to
# encourage good coding practices. The lint set provided by the
package is
# activated in the `analysis_options.yaml` file located at the root
of your
# package. See that file for information about deactivating
specific lint
# rules and activating additional ones.
flutter_lints: ^4.0.0

# For information on the generic Dart part of this file, see the
# following page: https://dart.dev/tools/pub/pubspe

# The following section is specific to Flutter packages.
flutter:

# The following line ensures that the Material Icons font is
# included with your application, so that you can use the icons in
# the material Icons class.
uses-material-design: true

# To add assets to your application, add an assets section, like
this:
# assets:
#   - images/a_dot_burr.jpeg
#   - images/a_dot_ham.jpeg

# An image asset can refer to one or more resolution-specific
"variants", see
# https://flutter.dev/to/resolution-aware-images

# For details regarding adding assets from package dependencies,
see
# https://flutter.dev/to/asset-from-package

# To add custom fonts to your application, add a fonts section
here,
# in this "flutter" section. Each entry in this list should have a
# "family" key with the font family name, and a "fonts" key with a
# list giving the asset and other descriptors for the font. For
# example:
# fonts:
#   - family: Schyler
#     fonts:
#       - asset: fonts/Schyler-Regular.ttf
#       - asset: fonts/Schyler-Italic.ttf
#         style: italic
#   - family: Trajan Pro
#     fonts:
#       - asset: fonts/TrajanPro.ttf
#       - asset: fonts/TrajanPro_Bold.ttf
#         weight: 700
#
# For details regarding fonts from package dependencies,
# see https://flutter.dev/to/font-from-package

```

Penjelasan:

File `pubspec.yaml` adalah file konfigurasi utama dalam proyek Flutter yang menyimpan informasi penting mengenai aplikasi, seperti nama, deskripsi, dan versi. Selain itu, file ini juga mengatur lingkungan pengembangan (SDK Dart), serta mendefinisikan dependensi yang diperlukan, seperti `google_maps_flutter` untuk menambahkan fitur peta, `place_picker_google` untuk pemilihan lokasi, dan `flutter_lints` untuk memastikan standar penulisan kode yang baik. Pada bagian flutter, file ini memungkinkan penggunaan desain material dan menambah aset atau font khusus jika dibutuhkan. Dengan demikian, file `pubspec.yaml` mempermudah Flutter dalam mengelola dependensi dan konfigurasi aplikasi dengan lebih efisien.

B. BUILD.GRADLE (ANDROID/APP)

Sourcecode:

```
plugins {
    id "com.android.application"
    id "kotlin-android"
    // The Flutter Gradle Plugin must be applied after the Android
    and Kotlin Gradle plugins.
    id "dev.flutter.flutter-gradle-plugin"
}

android {
    namespace = "com.example.pertemuan12"
    compileSdk = flutter.compileSdkVersion
    ndkVersion = flutter.ndkVersion

    compileOptions {
        sourceCompatibility = JavaVersion.VERSION_1_8
        targetCompatibility = JavaVersion.VERSION_1_8
    }

    kotlinOptions {
        jvmTarget = JavaVersion.VERSION_1_8
    }

    defaultConfig {
        // TODO: Specify your own unique Application ID
        // (https://developer.android.com/studio/build/application-
        id.html)
        applicationId = "com.example.pertemuan12"
        // You can update the following values to match your
        application needs.
        // For more information, see:
        // https://flutter.dev/to/reviewgradle-config.
        minSdkVersion 21
        targetSdk = flutter.targetSdkVersion
        versionCode = flutter.versionCode
        versionName = flutter.versionName
    }

    buildTypes {
        release {
            // TODO: Add your own signing config for the release
            build.
            // Signing with the debug keys for now, so `flutter run --
            release` works.
            signingConfig = signingConfigs.debug
        }
    }
}

flutter {
    source = "../.."
}
```

Penjelasan:

File build.gradle ini berfungsi sebagai konfigurasi untuk membangun aplikasi Android dalam proyek Flutter. Plugin com.android.application digunakan untuk membangun aplikasi Android, kotlin-android mendukung penggunaan bahasa pemrograman Kotlin, dan dev.flutter.flutter-gradle-plugin mengintegrasikan Flutter dengan Android. Pada bagian android, file ini mengatur berbagai pengaturan seperti namespace aplikasi, versi SDK (termasuk compileSdk, minSdkVersion, dan targetSdk), serta kompatibilitas dengan Java 1.8 melalui pengaturan compileOptions dan kotlinOptions. Di bagian defaultConfig, file ini mendefinisikan informasi aplikasi seperti applicationId, versionCode, dan versionName. Sementara itu, di bagian buildTypes, tipe build untuk release diatur dengan menggunakan kunci debug sementara. Direktori sumber Flutter juga ditentukan melalui properti flutter. Secara keseluruhan, file ini memastikan integrasi yang optimal antara Android dan Flutter untuk menjalankan aplikasi.

C. ANDROIDMANIFEST.XML

Sourcecode:

```
<manifest xmlns:android="http://schemas.android.com/apk/res/android">

    <uses-permission android:name="android.permission.INTERNET" />
    <uses-permission
        android:name="android.permission.ACCESS_COARSE_LOCATION" />
    <uses-permission
        android:name="android.permission.ACCESS_FINE_LOCATION" />

    <application
        android:label="pertemuan12"
        android:name="${applicationName}"
        android:icon="@mipmap/ic_launcher">

        <meta-data
            android:name="com.google.android.geo.API_KEY"
            android:value="AIzaSyCNRctT0VUJgTb_C_ukssCpr6Uok9NGn_g"
        />

        <activity
            android:name=".MainActivity"
            android:exported="true"
            android:launchMode="singleTop"
            android:taskAffinity=""
            android:theme="@style/LaunchTheme"

            android:configChanges="orientation|keyboardHidden|keyboard|screenSize|smallestScreenSize|locale|layoutDirection|fontScale|screenLayout|density|uiMode"
            android:hardwareAccelerated="true"
            android:windowSoftInputMode="adjustResize">

            <!-- Specifies an Android theme to apply to this Activity
            as soon as the Android process has started. This theme is
            visible to the user while the Flutter UI initializes. After that, this
            theme continues to determine the Window background behind the
            Flutter UI. -->
            <meta-data
                android:name="io.flutter.embedding.android.NormalTheme"
                android:resource="@style/NormalTheme" />

            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category
                    android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>

        <!-- Don't delete the meta-data below.
            This is used by the Flutter tool to generate
            GeneratedPluginRegistrant.java -->
        <meta-data
            android:name="flutterEmbedding"
            android:value="2" />

    </application>

    <!-- Required to query activities that can process text, see:
        https://developer.android.com/training/package-visibility
        and
        https://developer.android.com/reference/android/content/Intent#ACTION_PROCESS_TEXT.

        In particular, this is used by the Flutter engine in
        io.flutter.plugin.text.ProcessTextPlugin. -->
    <queries>
        <intent>
            <action android:name="android.intent.action.PROCESS_TEXT"
                />

            <data android:mimeType="text/plain" />
        </intent>
    </queries>
</manifest>
```

Penjelasan:

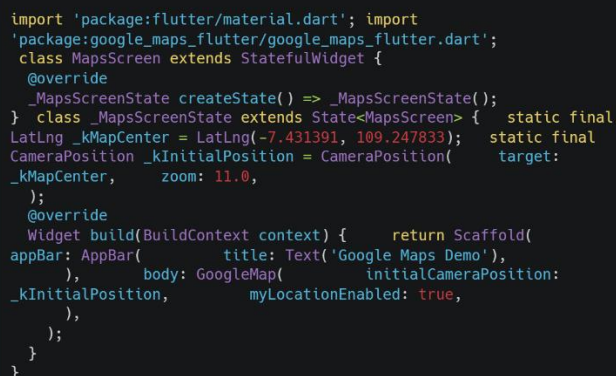
File `AndroidManifest.xml` ini merupakan konfigurasi penting yang mendefinisikan izin, metadata, dan pengaturan utama aplikasi Android. Izin yang diberikan termasuk akses internet (`INTERNET`), lokasi kasar (`ACCESS_COARSE_LOCATION`), dan lokasi presisi (`ACCESS_FINE_LOCATION`) yang diperlukan untuk fitur peta. Bagian `<application>` menyertakan nama aplikasi, ikon, serta API key Google Maps melalui elemen `<meta-data>`. Aktivitas utama (`MainActivity`) diatur dengan atribut seperti mode peluncuran, tema awal (`LaunchTheme`), dan intent filter untuk menjadikan aplikasi sebagai titik masuk utama. Metadata `flutterEmbedding` memastikan integrasi dengan Flutter versi 2. Bagian `<queries>` mendukung aplikasi untuk memproses teks, memastikan kompatibilitas dengan fitur pencarian berbasis teks. File ini memungkinkan aplikasi untuk memanfaatkan fitur peta dan lokasi dengan integrasi Android yang optimal.

2. TAMPILAN APLIKASI

Setelah melakukan konfigurasi untuk Firebase Notification, selanjutnya melakukan coding untuk tampilan aplikasinya

A. MY_MAPS.DART

Sourcecode:



```
import 'package:flutter/material.dart'; import
'package:google_maps_flutter/google_maps_flutter.dart';
class MapsScreen extends StatefulWidget {
  @override
  _MapsScreenState createState() => _MapsScreenState();
} class _MapsScreenState extends State<MapsScreen> { static final
LatLng _kMapCenter = LatLng(-7.431391, 109.247833); static final
CameraPosition _kInitialPosition = CameraPosition( target:
_kMapCenter, zoom: 11.0,
);
@override
Widget build(BuildContext context) { return Scaffold(
appBar: AppBar( title: Text('Google Maps Demo'),
), body: GoogleMap( initialCameraPosition:
_kInitialPosition, myLocationEnabled: true,
),
);
}
}
```

Penjelasan:

Kode ini merupakan implementasi peta interaktif menggunakan Flutter dengan plugin `google_maps_flutter`. Kelas `MapsScreen` adalah sebuah `StatefulWidget` yang menampilkan peta, dimulai dengan lokasi awal di Banyumas yang ditentukan oleh koordinat `_kMapCenter`, dan menggunakan properti `CameraPosition` dengan tingkat zoom 11. Pada widget `GoogleMap`, properti `initialCameraPosition` mengatur posisi awal kamera saat peta dimuat, sementara `myLocationEnabled: true` memungkinkan peta menampilkan lokasi pengguna jika diizinkan. Aplikasi ini juga dilengkapi dengan `AppBar` sederhana yang menampilkan judul "Google Maps Demo". Kode ini menyediakan tampilan dasar peta Google yang dapat dikembangkan lebih lanjut.

B. MAIN.DART

Sourcecode:

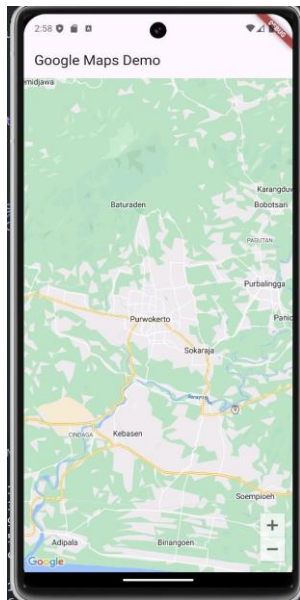
```
import 'package:flutter/material.dart'; import
'package:pertemuan12/my_maps.dart';
void main() { runApp(const MyApp());
} class MyApp extends StatelessWidget { const MyApp({super.key});

  @override
  Widget build(BuildContext context) { return MaterialApp(
    title: 'Flutter Demo', theme: ThemeData( colorScheme:
    ColorScheme.fromSeed(seedColor: Colors.deepPurple),
    useMaterial3: true,
    ), home: MapsScreen(),
  );
}
```

Penjelasan:

Kode main.dart ini merupakan titik masuk utama aplikasi Flutter. Fungsi main() memanggil runApp() untuk memulai aplikasi dengan widget MyApp sebagai widget utama. Kelas MyApp adalah sebuah StatelessWidget yang mengonfigurasi aplikasi. Di dalam metode build(), widget MaterialApp digunakan untuk menyusun aplikasi dengan judul "Flutter Demo" dan tema berbasis Material 3, menggunakan warna utama ungu gelap (deepPurple). Properti home dari MaterialApp diatur untuk menampilkan halaman pertama, yaitu MapsScreen, yang diimpor dari file my_maps.dart. Kode ini menyediakan struktur dasar aplikasi Flutter dengan navigasi awal menuju peta.

Output:



Penjelasan:

Aplikasi Flutter ini adalah contoh sederhana yang menggunakan plugin `google_maps_flutter` untuk menampilkan peta interaktif. Pada layar utama, aplikasi memuat peta Google Maps dengan lokasi awal yang ditetapkan di koordinat Purwokerto (latitude -7.431391, longitude 109.247833). Widget `GoogleMap` digunakan untuk menampilkan peta ini, yang mendukung fitur zoom dan navigasi dasar. Judul "Google Maps Demo" muncul di `AppBar` sebagai antarmuka pengguna. Fitur `myLocationEnabled` diaktifkan untuk menunjukkan lokasi pengguna jika izin lokasi diberikan. Aplikasi ini menyediakan tampilan dasar peta yang responsif dan siap untuk dikembangkan dengan fitur tambahan seperti penanda lokasi atau pencarian tempat.

UNGUIDED

Dari tugas guided yang telah dikerjakan, lanjutkan hingga ke bagian place picker untuk memberikan informasi mengenai lokasi yang ditunjuk di peta.

Berikut merupakan sourcecode dari program

A. MY_MAPS.DART

Sourcecode:

```
import 'dart:io';
import 'package:flutter/material.dart';
import 'package:google_maps_flutter/google_maps_flutter.dart';
import 'package:place_picker_google/place_picker_google.dart';

class MapsScreen extends StatefulWidget {
  @override
  _MapsScreenState createState() => _MapsScreenState();
}

class _MapsScreenState extends State<MapsScreen> {
  static final LatLng _initialLocation = LatLng(-7.431391,
109.247833);

  void _showPlacePicker() async {
    if (Platform.isAndroid) {
      final result = await Navigator.of(context).push(
        MaterialPageRoute(
          builder: (context) => PlacePicker(
            apiKey: "AIzaSyCNRctT0VUJgTb_CukssCpr6Uok9NGn_g",
            onPlacePicked: (LocationResult result) {
              debugPrint("Place picked: ${result.formattedAddress}");
              Navigator.of(context).pop();
            },
            initialLocation: _initialLocation,
          ),
        ),
      );
      if (result != null) {
        showDialog(
          context: context,
          builder: (context) => AlertDialog(
            title: Text("Lokasi Dipilih"),
            content: Text(result.formattedAddress ?? "Tidak ada
alamat"),
            actions: [
              TextButton(
                onPressed: () => Navigator.of(context).pop(),
                child: Text("OK"),
              ),
            ],
          ),
        );
      } else {
        debugPrint("PlacePicker hanya mendukung Android saat ini.");
      }
    }
  }

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: Text('Google Maps Demo'),
      ),
      body: Stack(
        children: [
          GoogleMap(
            initialCameraPosition: CameraPosition(
              target: _initialLocation,
              zoom: 11.0,
            ),
            myLocationEnabled: true,
          ),
          Positioned(
            bottom: 16,
            left: 16,
            right: 16,
            child: ElevatedButton(
              onPressed: _showPlacePicker,
              child: Text('Pilih Lokasi'),
            ),
          ),
        ],
      ),
    );
  }
}
```

B. MAIN.DART

Sourcecode:

```

import 'package:flutter/material.dart';
import 'package:pertemuan12/my_maps.dart';

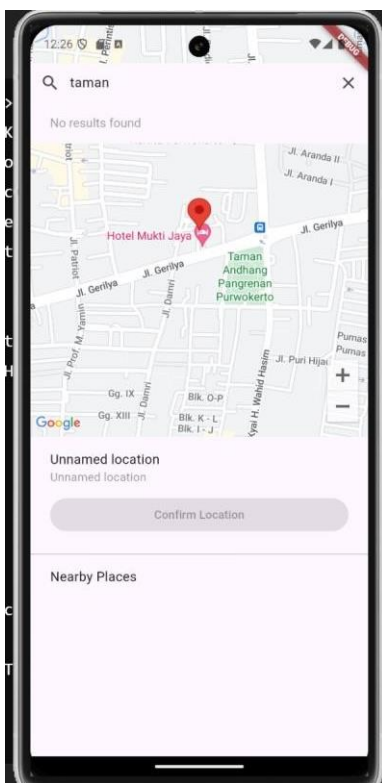
void main() {
  runApp(const MyApp());
}

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

  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      title: 'Flutter Demo',
      theme: ThemeData(
        colorScheme: ColorScheme.fromSeed(seedColor:
Colors.deepPurple),
        useMaterial3: true,
      ),
      home: MapsScreen(),
    );
  }
}

```

Output:



Penjelasan:

Aplikasi Flutter ini memanfaatkan plugin Google Maps dan Place Picker untuk menampilkan peta interaktif dengan fitur pencarian dan pemilihan lokasi. Pengguna dapat mencari lokasi tertentu, seperti "taman," yang kemudian ditandai dengan pin merah pada peta. Lokasi yang dipilih akan ditampilkan secara rinci di bawah peta, lengkap dengan opsi untuk mengonfirmasi lokasi tersebut. Aplikasi ini juga mencakup fitur daftar "Nearby Places," meskipun hasil pencarian tidak ditampilkan di layar. Dengan desain yang sederhana, aplikasi ini bertujuan mempermudah pengguna dalam menjelajahi dan memilih lokasi secara visual melalui antarmuka berbasis peta.

