

**TUGAS PENDAHULUAN  
PEMROGRAMAN PERANGKAT BERGERAK**

**MODUL XII  
MAPS & PLACES**



**Disusun Oleh :**  
**Yogi Hafidh Maulana / 2211104061**  
**SE06-02**

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

**Dosen Pengampu :**  
**Yudha Islami Sulistya, S.Kom., M.Cs.**

**PROGRAM STUDI S1 SOFTWARE ENGINEERING**  
**FAKULTAS INFORMATIKA**  
**TELKOM UNIVERSITY PURWOKERTO**

**2024**

## TUGAS PENDAHULUAN


### SOAL

#### 1. Menambahkan Google Maps Package

- a. Apa nama package yang digunakan untuk mengintegrasikan Google Maps di Flutter dan sebutkan langkah-langkah yang diperlukan untuk menambahkan package Google Maps ke dalam proyek Flutter.

Package untuk mengintegrasikan Google Maps di Flutter `google_maps_flutter`. Berikut langkah langkah untuk menambahkan package tersebut:

- Buka file `pubspec.yaml` di proyek Anda
- Tambahkan dependency `google_maps_flutter` seperti berikut



```
dependencies:  
  flutter:  
    sdk: flutter  
  google_maps_flutter: ^2.2.0
```

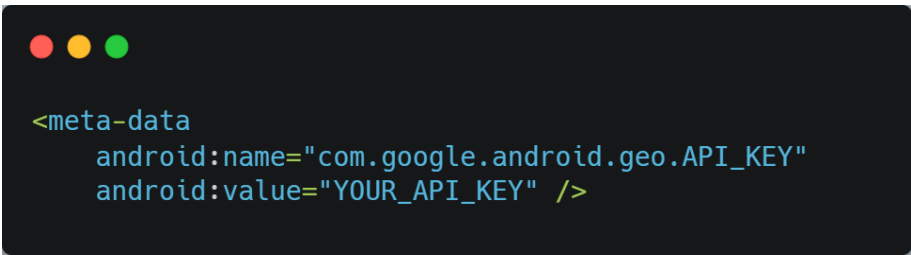
- Jalankan perintah `flutter pub get` untuk mengunduh dependency
- Konfigurasi android dengan menambahkan izin lokasi di `AndroidManifest.xml` dan menambahkan meta-data API Key di dalam tag `<application>` pada `AndroidManifest.xml`.
- Dapatkan API Key dari Google Cloud Console dengan mengaktifkan Maps SDK for Android dan Maps SDK for iOS.
- Tambahkan widget `GoogleMap` ke dalam file Dart untuk menampilkan peta.
- Jalankan aplikasi Flutter dengan `flutter run`.

- b. Mengapa kita perlu menambahkan API Key, dan di mana API Key tersebut diatur dalam aplikasi Flutter?

API Key diperlukan untuk otorisasi, memastikan hanya aplikasi yang sah dapat mengakses layanan Google Maps, serta untuk mengenali proyek Anda di Google Cloud Console sehingga Google dapat memantau dan membatasi penggunaan sesuai kuota atau aturan yang ditetapkan. Selain itu, API Key memungkinkan pengelolaan akses layanan, seperti menentukan layanan yang diaktifkan (misalnya, Maps SDK for Android/iOS, Places API, atau Directions API), dan dapat diatur di file `AndroidManifest.xml` pada Android serta `AppDelegate.swift` pada iOS untuk mengintegrasikan layanan Google Maps ke dalam aplikasi Flutter.

Dimana API Key Diatur dalam Aplikasi Flutter?

- Untuk Android, API Key ditambahkan di file `AndroidManifest.xml`, di dalam tag `<application>`:

A screenshot of a code editor showing a snippet of XML code for an Android application. The code is highlighted in a dark theme with syntax coloring. It shows a <meta-data tag with android:name="com.google.android.geo.API\_KEY" and android:value="YOUR\_API\_KEY".

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

- Untuk iOS, API Key ditambahkan di file `AppDelegate.swift`, biasanya dalam metode `didFinishLaunchingWithOptions`:

A screenshot of a code editor showing a snippet of Swift code for an iOS application. The code is highlighted in a dark theme with syntax coloring. It shows the GMSServices.provideAPIKey method call with the argument "YOUR\_API\_KEY".

```
GMSServices.provideAPIKey("YOUR_API_KEY")
```

## 2. Menampilkan Google Maps

- a. Tuliskan kode untuk menampilkan Google Map di Flutter menggunakan widget `GoogleMap`.

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

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

class MyApp extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      home: GoogleMapScreen(),
    );
  }
}

class GoogleMapScreen extends StatefulWidget {
  @override
  _GoogleMapScreenState createState() => _GoogleMapScreenState();
}

class _GoogleMapScreenState extends State<GoogleMapScreen> {
  // Initial position dari map
  static const CameraPosition _initialPosition = CameraPosition(
    // Coordinates untuk Bandung, Indonesia
    target: LatLng(-6.917464, 107.619123),
    zoom: 14.0,
  );

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: Text('Google Map Example'),
        backgroundColor: Colors.blue,
      ),
      body: GoogleMap(
        initialCameraPosition: _initialPosition,
        mapType: MapType.normal,
        onMapCreated: (GoogleMapController controller) {
          print('Google Map is ready');
        },
      ),
    );
  }
}
```

- b. Bagaimana cara menentukan posisi awal kamera (camera position) pada Google Maps di Flutter?

Untuk menentukan posisi awal kamera (camera position) pada Google Maps di Flutter, Anda dapat menggunakan properti `initialCameraPosition` pada widget `GoogleMap`. Posisi awal kamera ditentukan menggunakan kelas `CameraPosition`. Kelas ini digunakan untuk menentukan lokasi awal kamera, level zoom, dan orientasi peta. Contoh Implementasi Posisi Awal Kamera Berikut adalah contoh menentukan posisi awal kamera

```
static const CameraPosition _initialPosition = CameraPosition(  
  target: LatLng(-6.917464, 107.619123),  
  zoom: 14.0,  
);
```

Gunakan `initialCameraPosition` di `GoogleMap` Setelah posisi kamera didefinisikan, tambahkan properti `initialCameraPosition` pada widget `GoogleMap`:

```
GoogleMap(  
  initialCameraPosition: _initialPosition,  
);
```

- c. Sebutkan properti utama dari widget `GoogleMap` dan fungsinya.
1. `initialCameraPosition`: Menentukan posisi awal kamera pada peta, termasuk koordinat lokasi, tingkat zoom, arah rotasi, dan sudut kemiringan.
  2. `mapType`: Mengatur jenis tampilan peta, seperti normal, satelit, hybrid, atau terrain.
  3. `markers`: Menambahkan penanda pada peta untuk menunjukkan lokasi tertentu.
  4. `onMapCreated`: Callback yang dipanggil saat peta selesai dimuat, memungkinkan kontrol lebih lanjut melalui instance `GoogleMapController`.
  5. `myLocationEnabled`: Menampilkan lokasi pengguna saat ini pada peta jika izin lokasi telah diberikan.
  6. `zoomControlsEnabled`: Menentukan apakah tombol kontrol zoom ditampilkan pada peta.
  7. `polylines`: Menggambar garis pada peta untuk menunjukkan rute atau jalur tertentu.
  8. `circles`: Menambahkan lingkaran pada peta untuk menunjukkan area tertentu dengan radius tertentu.
  9. `onTap`: Callback yang dipanggil saat pengguna mengetuk lokasi pada peta.
  10. `myLocationButtonEnabled`: Menentukan apakah tombol "lokasi saya" ditampilkan pada peta.

### 3. Menambahkan Marker

- a. Tuliskan kode untuk menambahkan marker di posisi tertentu (latitude: -6.2088, longitude: 106.8456) pada Google Maps.

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

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

class MyApp extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      home: GoogleMapScreen(),
    );
  }
}

class GoogleMapScreen extends StatefulWidget {
  @override
  _GoogleMapScreenState createState() => _GoogleMapScreenState();
}

class _GoogleMapScreenState extends State<GoogleMapScreen> {
  static const LatLng _markerPosition = LatLng(-6.2088, 106.8456);
  static const CameraPosition _initialPosition = CameraPosition(
    target: _markerPosition,
    zoom: 14.0,
  );

  final Set<Marker> _markers = {};

  @override
  void initState() {
    super.initState();
    _addMarker();
  }

  void _addMarker() {
    _markers.add(
      Marker(
        markerId: MarkerId('marker_1'),
        position: _markerPosition,
        infoWindow: InfoWindow(title: 'Jakarta', snippet: 'Capital of Indonesia'),
      ),
    );
  }

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: Text('Google Maps with Marker'),
      ),
      body: GoogleMap(
        initialCameraPosition: _initialPosition,
        markers: _markers,
      ),
    );
  }
}
```

- b. Bagaimana cara menampilkan info window saat marker diklik?
1. Tambahkan InfoWindow pada Marker  
Setiap marker dapat memiliki properti infoWindow, yang digunakan untuk menampilkan informasi saat marker diklik. Properti ini diatur dengan objek InfoWindow yang mencakup detail seperti judul dan deskripsi.
  2. Tentukan Teks InfoWindow  
Pada InfoWindow, Anda dapat menentukan teks yang ditampilkan melalui parameter title (untuk judul utama) dan snippet (untuk teks tambahan di bawah judul). Judul biasanya berupa informasi utama, sedangkan snippet digunakan untuk deskripsi singkat.
  3. Pastikan Marker Dimasukkan ke dalam Set  
Semua marker yang ingin ditampilkan pada peta harus dimasukkan ke dalam sebuah properti bertipe Set<Marker>. Marker dengan infoWindow yang telah diatur akan otomatis menampilkan Info Window saat marker tersebut diklik oleh pengguna.

#### **4. Menggunakan Place Picker**

- a. Apa itu Place Picker, dan bagaimana cara kerjanya di Flutter dan sebutkan nama package yang digunakan untuk implementasi Place Picker di Flutter.

Place Picker adalah fitur yang memungkinkan pengguna untuk memilih lokasi tertentu pada peta. Fitur ini biasanya digunakan untuk mendapatkan alamat, koordinat geografis (latitude dan longitude), atau informasi lain tentang tempat yang dipilih. Place Picker sering digunakan dalam aplikasi seperti pemesanan, navigasi, atau layanan berbasis lokasi.

Place Picker adalah fitur yang memungkinkan pengguna memilih lokasi pada peta secara interaktif untuk mendapatkan alamat, koordinat, atau detail lokasi lainnya. Di Flutter, Place Picker bekerja dengan menampilkan peta Google Maps, memungkinkan pengguna memilih lokasi, dan mengembalikan data lokasi melalui callback. Package yang sering digunakan untuk implementasinya adalah place\_picker atau google\_maps\_place\_picker, dengan integrasi API Google Places untuk detail lokasi.

Nama Package yang Digunakan

1. place\_picker: Salah satu package populer untuk menambahkan fitur Place Picker menggunakan Google Maps.
2. google\_maps\_place\_picker: Alternatif lain yang memungkinkan integrasi Google Maps dan Place Picker.

- b. Tuliskan kode untuk menampilkan Place Picker, lalu kembalikan lokasi yang dipilih oleh pengguna dalam bentuk latitude dan longitude.

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

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

class MyApp extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      home: PlacePickerExample(),
    );
  }
}

class PlacePickerExample extends StatelessWidget {
  final String googleApiKey = "AIzaSyD-EXAMPLE1234567890qwertyuiopasdfghjk";

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: Text("Place Picker Example"),
      ),
      body: Center(
        child: ElevatedButton(
          onPressed: () async {
            // Tampilkan Place Picker
            LocationResult result = await Navigator.push(
              context,
              MaterialPageRoute(
                builder: (context) => PlacePicker(googleApiKey),
              ),
            );
            // Cek jika lokasi dipilih
            if (result != null) {
              print("Latitude: ${result.latLng?.latitude}");
              print("Longitude: ${result.latLng?.longitude}");
            } else {
              print("No location selected");
            }
          },
          child: Text("Pick a Place"),
        ),
      ),
    );
  }
}
```