TUGAS PENDAHULUAN
PEMROGRAMAN PERANGKAT BERGERAK

MODUL XII

MAPS & PLACES

Disusun Oleh :

Nama : Ganes Gemi Putra

Kelas : SE-07-02

NIM : (2311104075)

Dosen : YUDHA ISLAMI SULISTYA S.Kom M.Cs

PROGRAM STUDI S1 SOFTWARE ENGINEERING

FAKULTAS INFORMATIKA

TELKOM UNIVERSITY

PURWOKERTO

2025

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

2. **Menampilkan Google Maps**

a. Tuliskan kode untuk menampilkan Google Map di Flutter menggunakan widget GoogleMap.
b. Bagaimana cara menentukan posisi awal kamera (camera position) pada Google Maps di Flutter?
c. Sebutkan properti utama dari widget GoogleMap dan fungsinya.

3. **Menambahkan Marker**

a. Tuliskan kode untuk menambahkan marker di posisi tertentu (latitude: -6.2088, longitude: 106.8456) pada Google Maps.
b. Bagaimana cara menampilkan info window saat marker diklik?

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

JAWABAN

1. Menambahkan Google Maps Package

1.a Nama package & langkah menambahkan

Nama package-nya:

google_maps_flutter

Langkah-langkah menambahkan Google Maps ke proyek Flutter:

Aktifkan API di Google Cloud Console

- Buka Google Cloud Console.
- Buat / pilih project.
- Aktifkan:
  - **Maps SDK for Android**
  - **Maps SDK for iOS**
  - (opsional, nanti untuk Place Picker: Places API)
- Buat **API Key**.

2. Tambahkan dependency di pubspec.yaml:

```
dependencies:
  flutter:
    sdk: flutter
  google_maps_flutter: ^2.6.0  # versi bisa
menyesuaikan
```

3. Jalankan:

```
flutter pub get
```

4. **Android – tambahkan API key di AndroidManifest.xml:**

File: android/app/src/main/AndroidManifest.xml

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

    <meta-data

android:name="com.google.android.geo.API_KEY"
        android:value="YOUR_API_KEY_HERE" />
</application>
```

5.iOS – tambahkan API key di AppDelegate / Info.plist:

Di ios/Runner/AppDelegate.swift (contoh Swift):

```
import UIKit
import Flutter
import GoogleMaps   // import

@UIApplicationMain
@objc class AppDelegate: FlutterAppDelegate {
  override func application(
    _ application: UIApplication,
    didFinishLaunchingWithOptions launchOptions:
[UIApplication.LaunchOptionsKey: Any]?
  ) -> Bool {
```

```
    GMSServices.provideAPIKey("YOUR_API_KEY_HERE")   // API
Key di sini
    GeneratedPluginRegistrant.register(with: self)
    return super.application(application,
didFinishLaunchingWithOptions: launchOptions)
  }
}
```

1.b Alasan perlu API Key & di mana diatur

- **Mengapa perlu API Key?**
  - Untuk **autentikasi** ke layanan Google Maps Platform.
  - Untuk **membatasi & memonitor penggunaan** (quota, billing).
  - Untuk **keamanan**: Google tahu aplikasi mana yang memakai service-nya.
- **Di mana API Key diatur di Flutter?**
  - **Android:** di AndroidManifest.xml pada tag <application> sebagai:

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

iOS: di AppDelegate dengan GMSServices.provideAPIKey("YOUR_API_KEY_HERE")
(atau di Info.plist sesuai dokumentasi plugin).

2. Menampilkan Google Maps

2.a Kode menampilkan Google Map dengan widget GoogleMap

Contoh minimal:

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

class SimpleMapPage extends StatefulWidget {
  const SimpleMapPage({super.key});

  @override
  State<SimpleMapPage> createState() => _SimpleMapPageState();
}

class _SimpleMapPageState extends State<SimpleMapPage> {
  GoogleMapController? _mapController;

  static const CameraPosition _initialCameraPosition = CameraPosition(
    target: LatLng(-6.2088, 106.8456), // Jakarta
    zoom: 12,
  );

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: const Text('Google Map Demo'),
      ),
      body: GoogleMap(
        initialCameraPosition: _initialCameraPosition,
        onMapCreated: (controller) {
          _mapController = controller;
        },
      ),
    );
  }
```

```
}
```

2.b Menentukan posisi awal kamera

Posisi awal kamera diatur dengan **CameraPosition** lalu diberikan ke properti initialCameraPosition pada widget GoogleMap.

```
static const CameraPosition _initialCameraPosition = CameraPosition(
  target: LatLng(-6.2088, 106.8456), // latitude, longitude
  zoom: 12.0,
  bearing: 0,     // arah kompas (derajat)
  tilt: 0,        // kemiringan kamera (derajat)
);

// lalu dipakai:
GoogleMap(
  initialCameraPosition: _initialCameraPosition,
  ...
);
```

2.c Properti utama GoogleMap dan fungsinya

Beberapa properti penting:

- initialCameraPosition
  Posisi awal kamera (wajib).
- onMapCreated
  Callback ketika peta selesai dibuat, memberikan GoogleMapController.
- markers
  Kumpulan marker (Set<Marker>) yang akan ditampilkan di peta.
- mapType
  Jenis tampilan peta, misal MapType.normal, satellite, terrain, dll.
- myLocationEnabled
  Menampilkan titik lokasi pengguna (kalau izin lokasi diaktifkan).
- myLocationButtonEnabled
  Menampilkan tombol untuk fokus ke lokasi pengguna.
- zoomGesturesEnabled, scrollGesturesEnabled, rotateGesturesEnabled
  Mengatur apakah user boleh zoom, scroll (drag), dan rotate peta.
- onTap, onLongPress
  Callback saat user mengetuk / tekan lama suatu titik di peta (dapat latLng-nya).

3. Menambahkan Marker

3.a Kode marker di posisi (-6.2088, 106.8456)

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

class MarkerMapPage extends StatefulWidget {
  const MarkerMapPage({super.key});

  @override
  State<MarkerMapPage> createState() => _MarkerMapPageState();
}

class _MarkerMapPageState extends State<MarkerMapPage> {
  final Set<Marker> _markers = {};

  static const LatLng _jakartaLatLng = LatLng(-6.2088, 106.8456);

  static const CameraPosition _initialCameraPosition = CameraPosition(
```

```
  target: _jakartaLatLng,
  zoom: 12,
 );

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

 void _addJakartaMarker() {
  const marker = Marker(
   markerId: MarkerId('jakarta_marker'),
   position: _jakartaLatLng,
   infoWindow: InfoWindow(
    title: 'Jakarta',
    snippet: 'Latitude: -6.2088, Longitude: 106.8456',
   ),
  );

  _markers.add(marker);
 }

 @override
 Widget build(BuildContext context) {
  return Scaffold(
   appBar: AppBar(
    title: const Text('Marker Example'),
   ),
   body: GoogleMap(
    initialCameraPosition: _initialCameraPosition,
    markers: _markers,
   ),
  );
 }
}
```

3.b Menampilkan info window saat marker diklik
Cara default (paling umum):

- Kita set properti infoWindow pada Marker (seperti di atas).
- Ketika pengguna **klik marker**, info window otomatis muncul.

```
const Marker(
  markerId: MarkerId('jakarta_marker'),
  position: LatLng(-6.2088, 106.8456),
  infoWindow: InfoWindow(
    title: 'Jakarta',
    snippet: 'Ibu kota Indonesia',
  ),
);
```

Kalau ingin memunculkan info window **secara manual** via kode (misal setelah map load):

```
late GoogleMapController _controller;

...
```

```
onMapCreated: (controller) {
  _controller = controller;
  _controller.showMarkerInfoWindow(const MarkerId('jakarta_marker'));
},
```

4. Menggunakan Place Picker

4.a Apa itu Place Picker, cara kerja, dan nama package

- **Place Picker** adalah fitur/komponen yang memungkinkan user:
  - memilih lokasi di peta (drag & drop pin), dan/atau
  - mencari tempat dengan nama/alamat (autocomplete),
    lalu mengembalikan **informasi lokasi** seperti alamat dan koordinat (latitude, longitude).
- **Cara kerjanya di Flutter (gambaran umum):**
  - Kita pakai package yang membungkus Google Maps + Places API.
  - Ketika Place Picker dibuka:
    - akan menampilkan peta dan/atau search box.
    - user memilih lokasi.
    - widget mengembalikan objek hasil pilihan (misal LocationResult) yang berisi latLng, alamat, dst.
- **Contoh nama package Place Picker di Flutter:**
  - place_picker (banyak dipakai di tutorial kampus)
  - (Ada juga package lain seperti google_maps_place_picker_mb, dll.)

Untuk jawaban tugas, cukup sebut:
Package Place Picker: place_picker

4.b Kode menampilkan Place Picker & mengembalikan latitude/longitude

Contoh dengan place_picker:

1. **Tambahkan dependency di pubspec.yaml:**

```
dependencies:
  flutter:
    sdk: flutter
  google_maps_flutter: ^2.6.0
  place_picker: ^0.10.0   # versi contoh, sesuaikan
```

2. Gunakan PlacePicker di kode:

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

class PlacePickerPage extends StatefulWidget {
  const PlacePickerPage({super.key});

  @override
  State<PlacePickerPage> createState() => _PlacePickerPageState();
}

class _PlacePickerPageState extends State<PlacePickerPage> {
  String? _selectedLat;
  String? _selectedLng;

  // Ganti dengan API Key milikmu (yang sudah mengaktifkan Places API)
  final String _googleApiKey = "YOUR_GOOGLE_MAPS_API_KEY";

  void _openPlacePicker() async {
    // Buka halaman place picker
    LocationResult? result = await Navigator.of(context).push(
      MaterialPageRoute(
        builder: (context) => PlacePicker(_googleApiKey),
```

```
    ),
  );

  if (result != null && result.latLng != null) {
    setState(() {
      _selectedLat = result.latLng!.latitude.toString();
      _selectedLng = result.latLng!.longitude.toString();
    });

    // Di sini kamu sudah punya latitude & longitude
    // Misal mau print:
    debugPrint("Latitude: $_selectedLat, Longitude: $_selectedLng");
  }
}

@override
Widget build(BuildContext context) {
  return Scaffold(
    appBar: AppBar(
      title: const Text('Place Picker Example'),
    ),
    body: Center(
      child: Column(
        mainAxisSize: MainAxisSize.min,
        children: [
          ElevatedButton(
            onPressed: _openPlacePicker,
            child: const Text('Pilih Lokasi'),
          ),
          const SizedBox(height: 16),
          if (_selectedLat != null && _selectedLng != null)
            Text(
              'Lokasi dipilih:\n'
              'Latitude: $_selectedLat\n'
              'Longitude: $_selectedLng',
              textAlign: TextAlign.center,
            ),
        ],
      ),
    ),
  );
}
}
```

Di contoh di atas:

- Saat tombol **“Pilih Lokasi”** ditekan, Place Picker terbuka.
- Setelah user memilih lokasi dan kembali, kita ambil:

```
result.latLng!.latitude
result.latLng!.longitude
```

- Nilai tersebut kemudian bisa disimpan, ditampilkan, atau dikirim ke server.