PRAKTIKUM PEMROGRAMAN PERANGKAT BERGERAK
TUGAS GUIDED & UNGUIDED

MODUL XII
MAPS AND PLACES

Disusun Oleh :

Rizky Hanifa Afania / 2211104017

SE-06-01

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

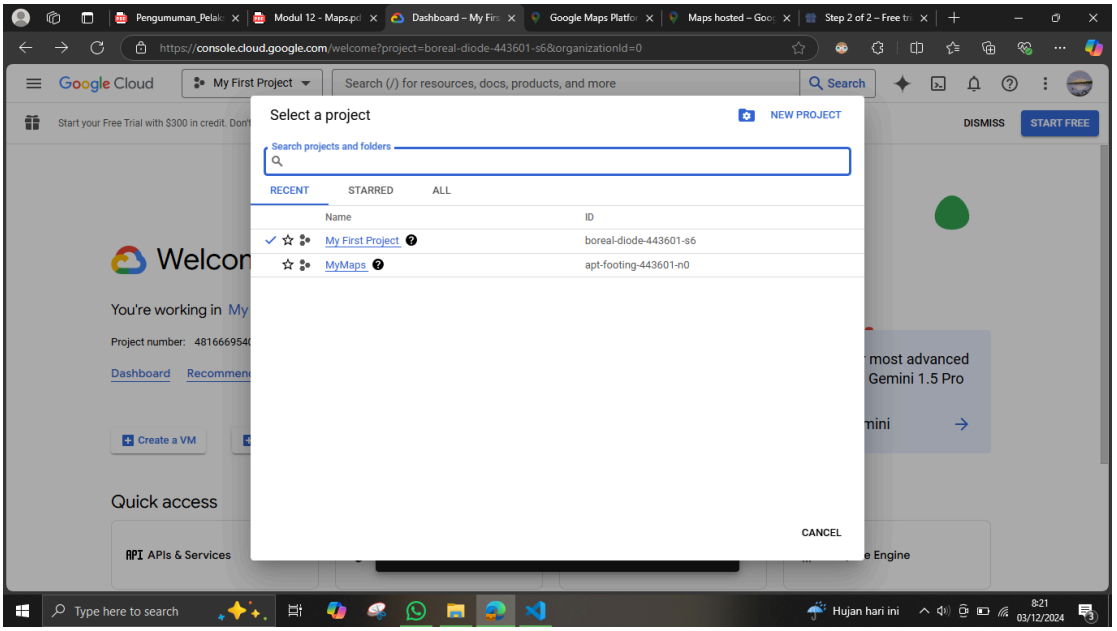
GUIDED

A. GOOGLE MAPS API

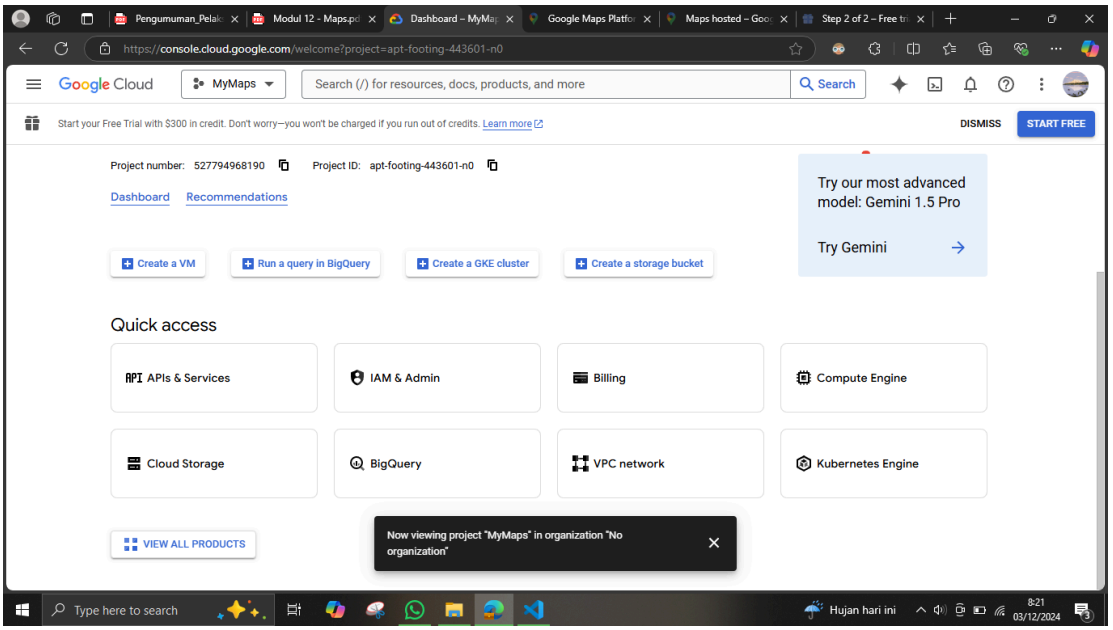
Google Maps API adalah layanan dari Google yang memudahkan pengembang untuk membuat aplikasi yang memanfaatkan fitur peta atau maps. Dengan Google Maps API, kita dapat menambahkan marker, menggunakan fitur rute, mencari lokasi, dan berbagai fitur lainnya. Untuk mengimplementasikan Google Maps API pada Flutter, kita dapat menggunakan paket Google Maps.

Berikut adalah langkah-langkah yang dapat diikuti untuk menambahkan Google Maps API:

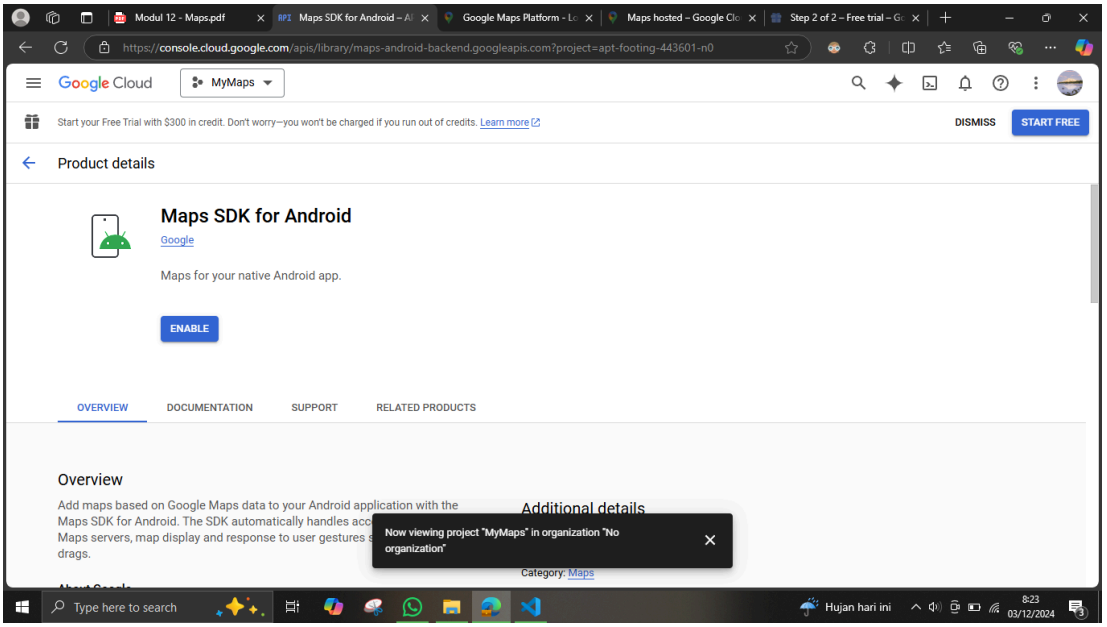
1. Dapatkan API key melalui link berikut https://cloud.google.com/maps-platform/
2. Selanjutnya, enable Google Map SDK di tiap platform yang akan menggunakan Google Maps.
    a. Pergi ke https://console.cloud.google.com/ (Google Delevopers Console)
    b. Pilih project yang ingin menggunakan Google Maps



    c. Pilih pada navigation menu, lalu pilih “Google Maps”
    d. Pilih “APIs” di bawah menu Google Maps



e. Untuk mengaktifkan Google Maps di Android, pilih “Maps SDK for Android” pada section “Additional APIs”, lalu pilih “ENABLE”



f. Untuk mengaktifkan Google Maps di iOS, pilih “Maps SDK for iOS” pada section “Additional APIs”, lalu pilih “ENABLE”

g. Pastikan bahwa APIs telah aktif pada section “Enabled APIs”

h. Untuk lebih detail bisa cek di https://developers.google.com/maps/gmp-get-started

3. Android
    a. Set minSdkVersion di **android/app/build.gradle**:

```
defaultConfig {
    // TODO: Specify your own unique Applic
    applicationId = "com.example.modul_12"
    // You can update the following values
    // For more information, see: https://f
    minSdkVersion 20
    targetSdk = flutter.targetSdkVersion
    versionCode = flutter.versionCode
    versionName = flutter.versionName

}
```

    b. Tambahkan API key pada manifest aplikasi **android/app/src/main/AndroidManifest.xml**

```
<!-- Tambahkan API Key disini -->
<meta-data android:name="com.google.android.geo.API_KEY"
    android:value="AIzaSyAUYRND6xGwbPx-hJfbdZiFtblqh2G5zpA"/>
```

    Langkah diatas untuk menambahkan Google Maps API ke aplikasi.

B. Menambah Packages Google Maps

Setelah mengikuti langkah diatas, sekarang adalah langkah-langkah menambahkan Google Maps ke layar aplikasi Flutter:

1. Pergi ke https://www.pub.dev lalu cari packages Google Maps. Nama packagesnya adalah google_maps_flutter.
2. Cari versi yang paling terbaru lalu tambahkan pada file pubspec.yaml

```
dependencies:
  flutter:
    sdk: flutter
  cupertino_icons: ^1.0.8
  google_maps_flutter: ^2.10.0
```

3. Selanjutnya, import packages ke dalam file Dart

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

4. Lalu, tambahkan widget GoogleMap ke file Dart

```
25          body: GoogleMap(
26            initialCameraPosition: _kInitialPosition,
```

GoogleMap diberi _kInitialPosition, yang dimana untuk menyimpan lokasi default saat aplikasi dijalankan atau dimuat

5. Buat fungsi _kMapCenter dan _kInitialPosition dengan kode sebagai berikut:

```
12    static final LatLng _kMapCenter =
13        const LatLng(19.018255973653343, 72.84793849278007);
14    static final CameraPosition _kInitialPosition =
15        CameraPosition(target: _kMapCenter, zoom: 11.0, tilt: 0, bearing: 0);
```

6. Berikut adalah tampilan kode yang lengkap
7. Berikut tampilannya

C. Menambahkan Akses di Manifest

Secara default, map akan menampilkan lokasi yang sudah kita definisikan pada initialCameraPosition yang ada pada parameter widget. Jika pengguna ingin menampilkan lokasi mereka, ubah pengaturan myLocationEnable menjad true.

Berikut barisan kode untuk menampilkan lokasi kita saat ini:

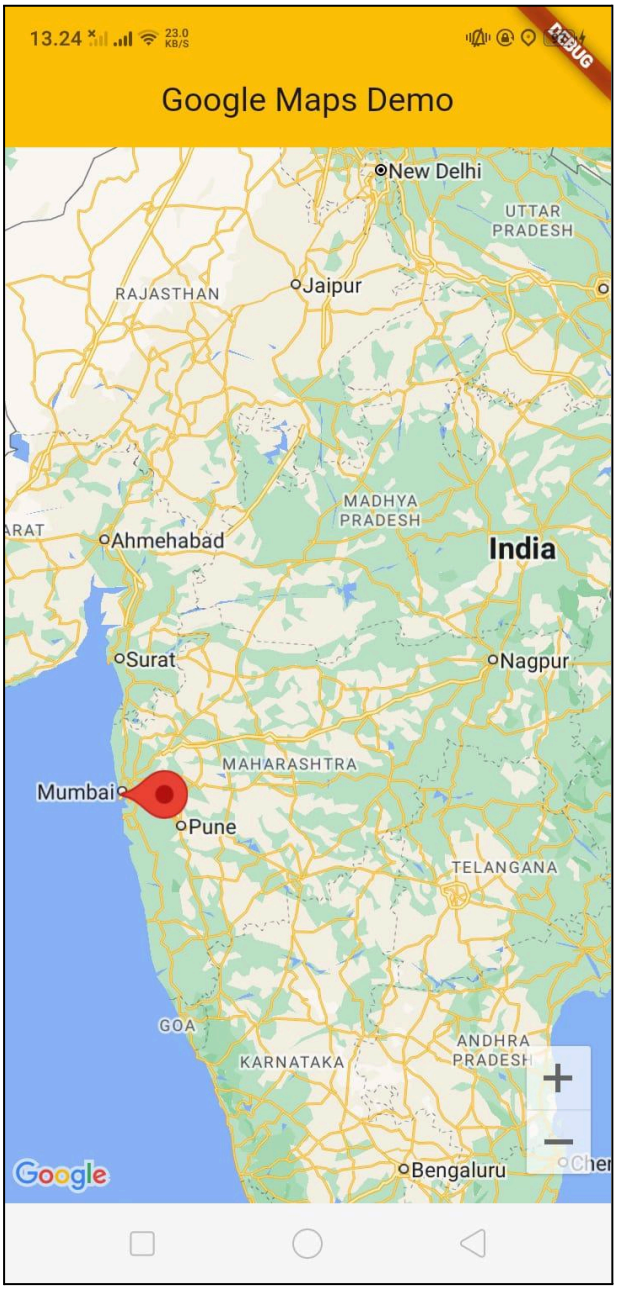
```
body: GoogleMap(
  initialCameraPosition: _kInitialPosition,
  //onMapCreated: onMapCreated,
  myLocationEnabled: true,
```

D. Menambahkan Marker pada Google Maps

Marker merupakan cara untuk menunjukkan lokasi tertentu. Untuk membuat marker pada map, berikut adalah barisan kodenya:

```
void _createMarker() {
  _markers.add(
    Marker(
      markerId: const MarkerId("marker_1"),
      position: _kMapCenter,
      infoWindow: const InfoWindow(title: 'Marker 1'),
      rotation: 0,
    ),
  );

  _markers.add(
    Marker(
      markerId: const MarkerId("marker_2"),
      position: const LatLng(-6.9733165, 107.6281415),
      infoWindow: const InfoWindow(title: 'Marker 2'),
    ),
  );
}
```

Berikut tampilan untuk Marker 1



Berikut tampilan untuk Marker 2

UNGUIDED

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

Note: Jangan lupa sertakan source code, screenshoot output, dan deskripsi program.

Kreatifitas menjadi nilai tambah

Source Code

main.dart

```dart
import 'package:flutter/material.dart';
import 'package:google_maps_flutter/google_maps_flutter.dart';
import 'package:modul_12/homepage.dart';


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

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

  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      title: 'Unguided Modul 12',
      theme: ThemeData(
        colorScheme: ColorScheme.fromSeed(seedColor: Colors.deepPurple),
        useMaterial3: true,
      ),
      home: const MyHomePage(),
    );
  }
}
```

homepage.dart

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

```dart
import 'dart:convert';

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

  @override
  State<MyHomePage> createState() => _MyHomePageState();
}

class _MyHomePageState extends State<MyHomePage> {
  static final LatLng _kMapCenter = const LatLng(-6.9733165, 107.6281415);
  static final CameraPosition _kInitialPosition =
      CameraPosition(target: _kMapCenter, zoom: 14.0, tilt: 0, bearing: 0);

  String? selectedPlaceName; // Untuk menyimpan nama lokasi
  List<dynamic>? nearbyPlaces; // Untuk menyimpan daftar tempat terdekat

  final String apiKey = "AIzaSyAUYRND6xGwbPx-hJfbdZiFtblqh2G5zpA";
  final Set<Marker> _markers = {}; // Set untuk menyimpan marker

  @override
  void initState() {
    super.initState();
    _createMarker(); // Tambahkan marker awal saat inisialisasi
  }

  // Fungsi untuk menambahkan marker awal
  void _createMarker() {
    _markers.add(
      Marker(
        markerId: const MarkerId("marker_1"),
        position: _kMapCenter,
        infoWindow: const InfoWindow(title: 'Marker 1'),
        rotation: 0,
      ),
    );

    _markers.add(
      Marker(
        markerId: const MarkerId("marker_2"),
        position: const LatLng(-6.974, 107.629),
        infoWindow: const InfoWindow(title: 'Marker 2'),
      ),
```

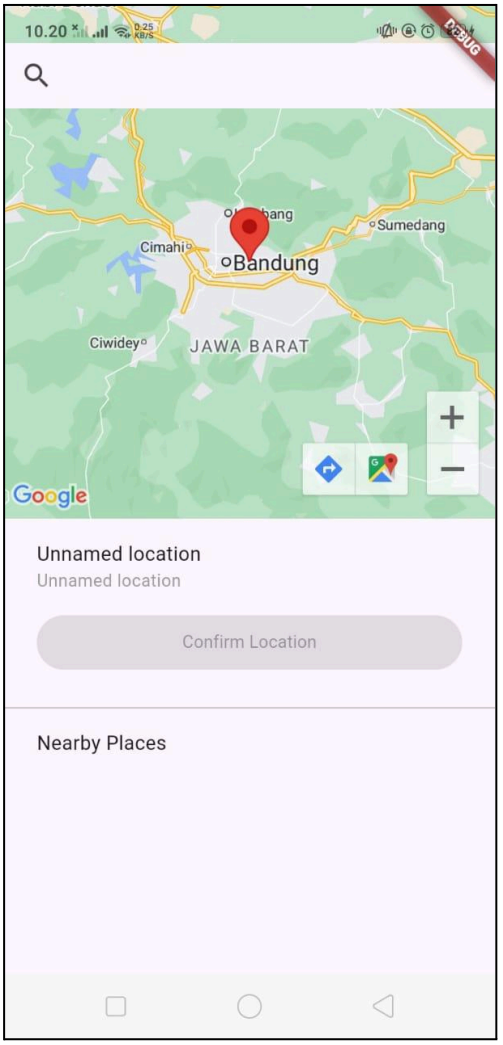
```
    );
  }

  // Fungsi untuk mendapatkan nama lokasi berdasarkan koordinat
  Future<String> getPlaceName(double lat, double lng) async {
    final url =

"https://maps.googleapis.com/maps/api/geocode/json?latlng=$lat,$lng&key=$apiK
ey";

    final response = await http.get(Uri.parse(url));

    if (response.statusCode == 200) {
      final data = json.decode(response.body);
      if (data["results"] != null && data["results"].isNotEmpty) {
        return data["results"][0]["formatted_address"] ?? "Unnamed Location";
      } else {
        return "Unnamed Location";
      }
    } else {
      throw Exception("Failed to fetch place name");
    }
  }

  // Fungsi untuk mendapatkan nearby places berdasarkan koordinat
  Future<Map<String, dynamic>> fetchNearbyPlaces(double lat, double lng)
async {
    final url =

"https://maps.googleapis.com/maps/api/place/nearbysearch/json?location=$lat,$
lng&radius=500&key=$apiKey";

    final response = await http.get(Uri.parse(url));

    if (response.statusCode == 200) {
      return json.decode(response.body);
    } else {
      throw Exception("Failed to fetch nearby places");
    }
  }

  // Fungsi untuk menampilkan Place Picker
  void showPlacePicker() async {
```

```
    LocationResult? result = await Navigator.of(context).push(
      MaterialPageRoute(
        builder: (context) => PlacePicker(
          apiKey: apiKey,
          initialLocation: _kMapCenter,
        ),
      ),
    );

    if (result != null) {
      final lat = result.latLng?.latitude ?? 0.0;
      final lng = result.latLng?.longitude ?? 0.0;

      try {
        final placeName = await getPlaceName(lat, lng);
        final places = await fetchNearbyPlaces(lat, lng);

        setState(() {
          selectedPlaceName = placeName;
          nearbyPlaces = places['results'];
        });
      } catch (e) {
        print("Error: $e");
      }
    }
  }

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: const Text('Google Maps Demo'),
        centerTitle: true,
        backgroundColor: Colors.amber,
      ),
      body: Column(
        children: [
          Expanded(
            child: GoogleMap(
              initialCameraPosition: _kInitialPosition,
              myLocationEnabled: true,
              markers: _markers, // Tambahkan marker ke peta
              onTap: (LatLng position) async {
```

```
                try {
                  final placeName =
                      await getPlaceName(position.latitude,
position.longitude);
                  final places = await fetchNearbyPlaces(
                      position.latitude, position.longitude);

                  setState(() {
                    selectedPlaceName = placeName;
                    nearbyPlaces = places['results'];
                  });
                } catch (e) {
                  print("Error: $e");
                }
              },
            ),
          ),
          Container(
            padding: const EdgeInsets.all(16.0),
            color: Colors.white,
            child: Column(
              crossAxisAlignment: CrossAxisAlignment.start,
              children: [
                Text(
                  selectedPlaceName ?? "No location selected",
                  style: const TextStyle(
                    fontSize: 16,
                    fontWeight: FontWeight.bold,
                  ),
                ),
                const SizedBox(height: 8.0),
                if (nearbyPlaces != null)
                  Column(
                    crossAxisAlignment: CrossAxisAlignment.start,
                    children: [
                      const Text(
                        "Nearby Places:",
                        style: TextStyle(
                            fontSize: 14, fontWeight: FontWeight.bold),
                      ),
                      ...nearbyPlaces!.map((place) {
                        return Text(
                          place['name'] ?? "Unknown",
```

```
                        style: const TextStyle(fontSize: 14),
                      );
                    }).toList(),
                  ],
                )
              else
                const Text("No nearby places found"),
            ],
          ),
        ),
      ],
    ),
    floatingActionButton: FloatingActionButton(
      onPressed: showPlacePicker,
      child: const Icon(Icons.place),
      backgroundColor: Colors.amber,
    ),
  );
 }
}
```

Screenshoot output



Deskripsi Program:

Program ini mengimplementasikan fitur peta menggunakan Google Maps API dan Google Places API agar pengguna bisa memilih lokasi. Fitur utamanya adalah Place Picker, yang menggunakan widge untuk membantu pengguna mencari dan memilih lokasi melalui antarmuka yang mudah digunakan. Selain itu, program ini menampilkan marker pada peta menggunakan widget GoogleMap untuk memberikan petunjuk visual pada lokasi tertentu.

Program memanggil beberapa fungsi penting seperti fungsi showPlacePicker yang digunakan untuk menampilkan Place Picker dan mengembalikan hasil berupa koordinat lokasi yang dipilih oleh pengguna. Fungsi getPlaceName berfungsi memanggil Google Geocoding API untuk mendapatkan nama lokasi berdasarkan koordinat yang dipilih. Lalu, fungsi fetchNearbyPlaces digunakan untuk mengambil daftar tempat terdekat dari Google Places API, yang kemudian ditampilkan pada layar. Marker awal ditambahkan ke peta menggunakan fungsi _createMarker, yang dipanggil saat inisialisasi program.