

**TUGAS PENDAHULUAN
PEMROGRAMAN PERANGKAT BERGERAK**

**MODUL XII
MAPS & PLACES**



Disusun Oleh :

Fikri Khairul Fajri/2211104052

SE-06-02

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.
- 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

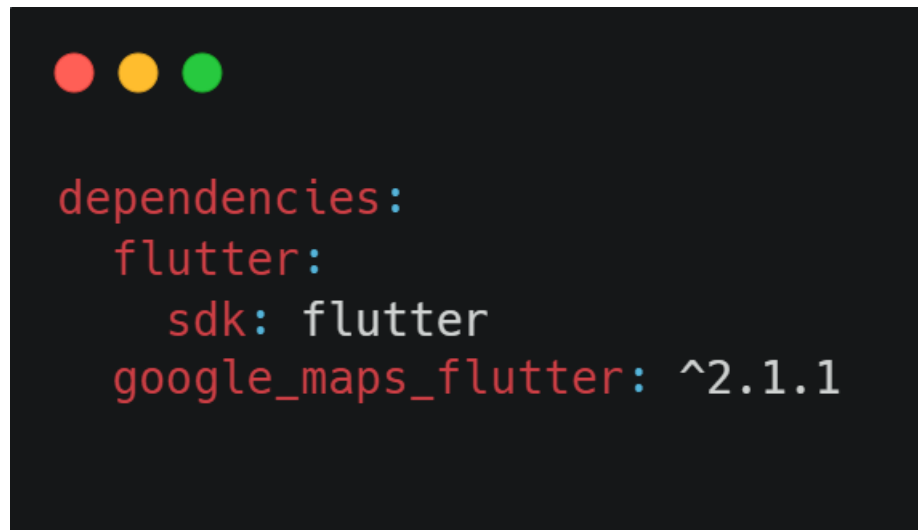
- a. Apa itu Place Picker, dan bagaimana cara kerjanya di Flutter dan sebutkan nama package yang digunakan untuk implementasi Place Picker di Flutter.
- b. Tuliskan kode untuk menampilkan Place Picker, lalu kembalikan lokasi yang dipilih oleh pengguna dalam bentuk latitude dan longitude.

5. Jawaban Soal :

1. Menambahkan Google Maps Package

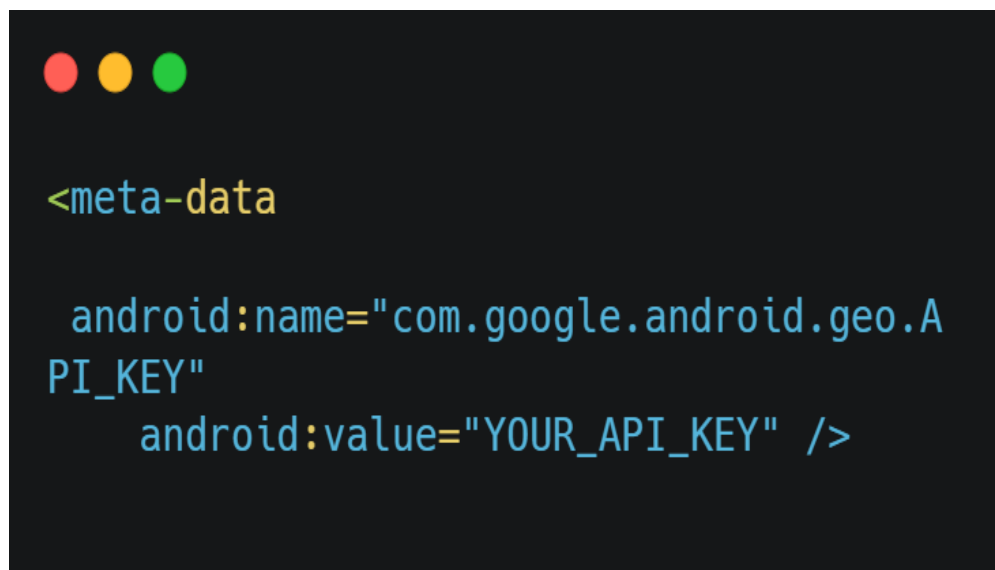
a. Package yang digunakan untuk mengintegrasikan Google Maps di Flutter adalah `google_maps_flutter`.

- Langkah-langkah menambahkan package Google Maps :
- Tambahkan dependensi di `pubspec.yaml`

A screenshot of a code editor showing the dependencies section of a pubspec.yaml file. The code is as follows:

```
dependencies:  
  flutter:  
    sdk: flutter  
  google_maps_flutter: ^2.1.1
```

- Install dependensi: Setelah menambahkan dependensi.
- Aktifkan API Google Maps.
- Konfigurasi API Key untuk Android:
- Masukkan API Key di file `android/app/src/main/AndroidManifest.xml` pada bagian `<application>`.

A screenshot of a code editor showing the meta-data section of an AndroidManifest.xml file. The code is as follows:

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

- Bisa mulai menggunakan Google Maps di widget Flutter.

b. Mengapa Kita Perlu Menambahkan API Key dan Di Mana API Key Tersebut Diatur dalam Aplikasi Flutter :

- Keamanan dan Pembatasan Penggunaan: API Key digunakan untuk mengidentifikasi aplikasi Anda saat berkomunikasi dengan layanan Google Maps. Ini memastikan bahwa hanya aplikasi yang memiliki kunci yang valid yang dapat mengakses API, dan membantu membatasi penggunaan API agar tidak disalahgunakan.
- Pengelolaan Penggunaan dan Biaya: Google menggunakan API Key untuk memantau dan menghitung penggunaan API oleh aplikasi Anda. Ini penting agar Anda dapat melacak penggunaan dan mengelola biaya layanan API.
- Mengaktifkan Akses ke Fitur Google Maps: Beberapa fitur di Google Maps, seperti pemetaan, geocoding, dan pembuatan rute, memerlukan API Key untuk diaktifkan.

2. Kode untuk Menampilkan Google Map di Flutter:

- a. Source Code :

```

import 'package:flutter/material.dart';
import
'package:google_maps_flutter/google_map
s_flutter.dart';

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

class MyApp extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      home: Scaffold(
        appBar: AppBar(
          title: Text('Google Maps
Flutter'),
        ),
        body: GoogleMapWidget(),
      ),
    );
  }
}

class GoogleMapWidget extends
StatefulWidget {
  @override
  _GoogleMapWidgetState createState()
=> _GoogleMapWidgetState();
}

class _GoogleMapWidgetState extends
State<GoogleMapWidget> {
  late GoogleMapController
mapController;

  // Menentukan posisi awal kamera
(CameraPosition)
  static const CameraPosition
_initialPosition = CameraPosition(
    target: LatLng(-6.1751, 106.8650),
    // Koordinat Jakarta
    zoom: 10, // Zoom level awal
  );

  // Fungsi untuk menyimpan kontroler
peta
  void
  _onMapCreated(GoogleMapController
controller) {
    mapController = controller;
  }

  @override
  Widget build(BuildContext context) {
    return GoogleMap(
      onMapCreated: _onMapCreated, //
Memanggil fungsi saat peta dibuat
      initialCameraPosition:
_initialPosition, // Menentukan posisi
awal kamera
      markers: Set<Marker>.of([
        Marker(
          markerId: MarkerId('1'),
          position: LatLng(-6.1751,
106.8650), // Menambahkan marker di
Jakarta
          infoWindow: InfoWindow(
            title: 'Jakarta',
            snippet: 'Kota di
Indonesia',
          ),
        ),
      ]),
    );
  }
}

```

b. Untuk menentukan posisi awal kamera pada Google Maps di Flutter, kita menggunakan properti `initialCameraPosition` pada widget `GoogleMap` :

- `CameraPosition` adalah objek yang digunakan untuk mengatur posisi awal kamera, termasuk target dan tingkat zoom.

1. `target`: Menentukan koordinat latitude dan longitude yang menjadi pusat peta.

2. `zoom`: Menentukan level zoom awal peta.

c. Properti Utama dari Widget `GoogleMap` dan Fungsinya

- `onMapCreated`:

Fungsi callback yang dipanggil ketika peta selesai dibuat dan siap digunakan. Di sini, Anda bisa mendapatkan referensi ke `GoogleMapController` yang memungkinkan Anda untuk mengontrol peta.

- `initialCameraPosition`:

Menentukan posisi awal kamera peta saat pertama kali ditampilkan. Ini mencakup target (koordinat latitude dan longitude) dan zoom level.

- `markers`:

Menentukan koleksi marker yang akan ditampilkan di peta. Marker digunakan untuk menandai lokasi tertentu di peta.

3. Menambahkan Marker :

- Source Code :

```

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: Scaffold(
        appBar: AppBar(
          title: Text('Google Maps Marker with InfoWindow'),
        ),
        body: GoogleMapWidget(),
      ),
    );
  }
}

class GoogleMapWidget extends StatefulWidget {
  @override
  _GoogleMapWidgetState createState() => _GoogleMapWidgetState();
}

class _GoogleMapWidgetState extends State<GoogleMapWidget> {
  late GoogleMapController
  mapController;

  // Menentukan posisi awal kamera
  static const CameraPosition
  _initialPosition = CameraPosition(
    target: LatLng(-6.1751, 106.8650),
    // Koordinat Jakarta
    zoom: 10,
  );

  // Menyimpan marker
  final Marker _marker = Marker(
    markerId: MarkerId('marker_1'),
    position: LatLng(-6.2088,
    106.8456), // Posisi yang ditentukan
    infoWindow: InfoWindow(
      title: 'Lokasi Baru',
      snippet: 'Latitude: -6.2088,
      Longitude: 106.8456',
    ),
  );

  // Fungsi untuk menyimpan kontroler
  peta
  void
  _onMapCreated(GoogleMapController
  controller) {
    mapController = controller;
  }

  @override
  Widget build(BuildContext context) {
    return GoogleMap(
      onMapCreated: _onMapCreated, //
      // Memanggil fungsi saat peta dibuat
      initialCameraPosition:
      _initialPosition, // Menentukan posisi
      // awal kamera
      markers:
      Set<Marker>.of([_marker]), //
      // Menambahkan marker ke peta
      onMarkerTapped: (Marker marker) {
        // Menampilkan info window
        // ketika marker diklik
        print('Marker
        ${marker.markerId} tapped');
      },
    );
  }
}

```

- Menampilkan Info Window Saat Marker Diklik :

1. onMarkerTapped adalah properti yang dipanggil ketika marker diklik. Di dalam callback ini, kita bisa menampilkan info window atau melakukan tindakan lain. Cara ini, ketika pengguna mengetuk marker di peta, info window dengan informasi yang sudah ditentukan (seperti judul dan snippet) akan muncul di atas marker tersebut

4. Menggunakan Place Picker

- a. Place Picker adalah sebuah widget atau fitur yang memungkinkan pengguna untuk memilih lokasi geografis (tempat) di peta. Dengan fitur ini, pengguna dapat memilih lokasi menggunakan peta yang menampilkan berbagai tempat yang dapat dipilih, seperti restoran, hotel, atau lokasi geografis lainnya. Di Flutter, Place Picker biasanya digunakan untuk mempermudah pengguna dalam memilih lokasi tanpa harus memasukkan alamat secara manual. Fitur ini mengintegrasikan peta dengan layanan pencarian tempat untuk memberikan pengalaman yang lebih interaktif dan user-friendly. Untuk mengimplementasikan Place Picker di Flutter, kita biasanya menggunakan package `flutter_google_places` yang mengintegrasikan Google Places API. Package ini memungkinkan pengguna untuk memilih tempat dan mendapatkan data terkait tempat tersebut, seperti nama tempat, alamat, latitude, dan longitude.
- b. kode untuk menampilkan Place Picker di Flutter dan mengembalikan latitude dan longitude lokasi yang dipilih oleh pengguna :


```

import 'package:flutter/material.dart';
import
'package:flutter_google_places/flutter_
google_places.dart';
import
'package:google_maps_flutter/google_map
s_flutter.dart';
import
'package:google_maps_webservice/places.
dart';
import
'package:google_maps_flutter/google_map
s_flutter.dart';

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

class MyApp extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      home: Scaffold(
        appBar: AppBar(
          title: Text('Place Picker
Example'),
        ),
        body: PlacePickerExample(),
      ),
    );
  }
}

class PlacePickerExample extends
StatefulWidget {
  @override
  _PlacePickerExampleState
createState() =>
_PlacePickerExampleState();
}

class _PlacePickerExampleState extends
State<PlacePickerExample> {
  late GoogleMapController
mapController;
  LatLng _pickedLocation =
LatLng(-6.1751, 106.8650); // Default
location (Jakarta)

  // API Key Anda dari Google Cloud
Console
  final String apiKey =
"YOUR_GOOGLE_API_KEY";

  // Fungsi untuk menampilkan Place
Picker
  Future<void> _pickPlace() async {
    // Memanggil PlacePicker untuk
memilih lokasi
    Prediction? prediction = await
PlacesAutocomplete.show(
      context: context,
      apiKey: apiKey,
      mode: Mode.overlay,
      language: 'id', // Pilih bahasa
sesuai kebutuhan
      // Optionally, you can pass a
location bias to filter results
      types: ['geocode'],
    );

    if (prediction != null) {
      // Mendapatkan detail tempat yang
dipilih
      final GoogleMapsPlaces _places =
GoogleMapsPlaces(apiKey: apiKey);
      PlacesDetailsResponse detail =
await
_places.getDetailsByPlaceId(prediction.
placeId!);
      final lat =
detail.result.geometry!.location.lat;
      final lng =
detail.result.geometry!.location.lng;

      setState(() {
        _pickedLocation = LatLng(lat,
lng); // Mengupdate lokasi yang dipilih
      });
    }
  }

  @override
  Widget build(BuildContext context) {
    return Column(
      children: <Widget>[
        ElevatedButton(
          onPressed: _pickPlace,
          child: Text('Pilih Lokasi'),
        ),
        Expanded(
          child: GoogleMap(
            initialCameraPosition:
CameraPosition(
              target: _pickedLocation,
              zoom: 14,
            ),
            markers: {
              Marker(
                markerId:
MarkerId('picked-location'),
                position:
_pickedLocation,
              ),
            },
            onMapCreated:
(GoogleMapController controller) {
              mapController =
controller;
            },
          ),
          Text(
            'Latitude:
${_pickedLocation.latitude}, Longitude:
${_pickedLocation.longitude}',
            style: TextStyle(fontSize:
16),
          ),
        ),
      ],
    );
  }
}

```