UNIVERSITÉ DES MASCAREIGNES

SAVOIR, C'EST POUVOIR

Université de Limoges

# Mental Health Diagnosis

Faculty of Information and Communication Technology

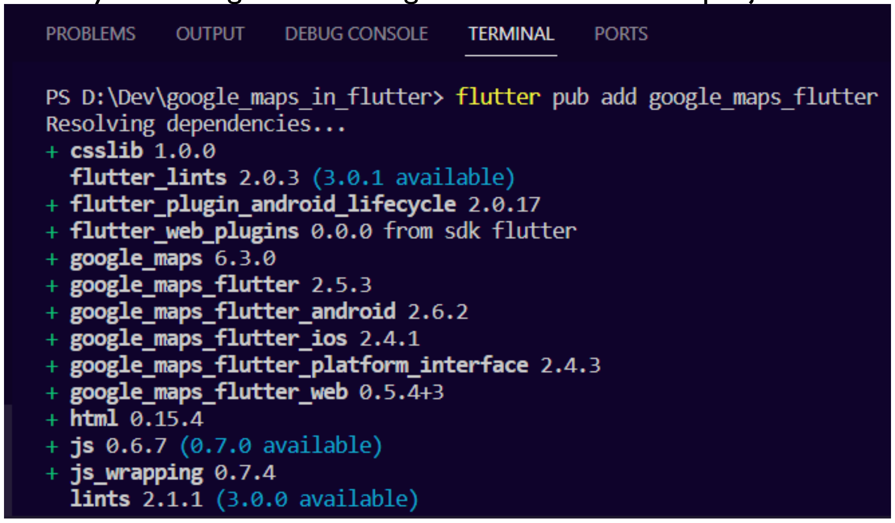
03/02/2024
PMH

Gokhool Diren
Mr Shiam Beehary

# Task-1 Adding Google Maps to a Flutter App

## Adding Google Maps Flutter plugin as a dependency

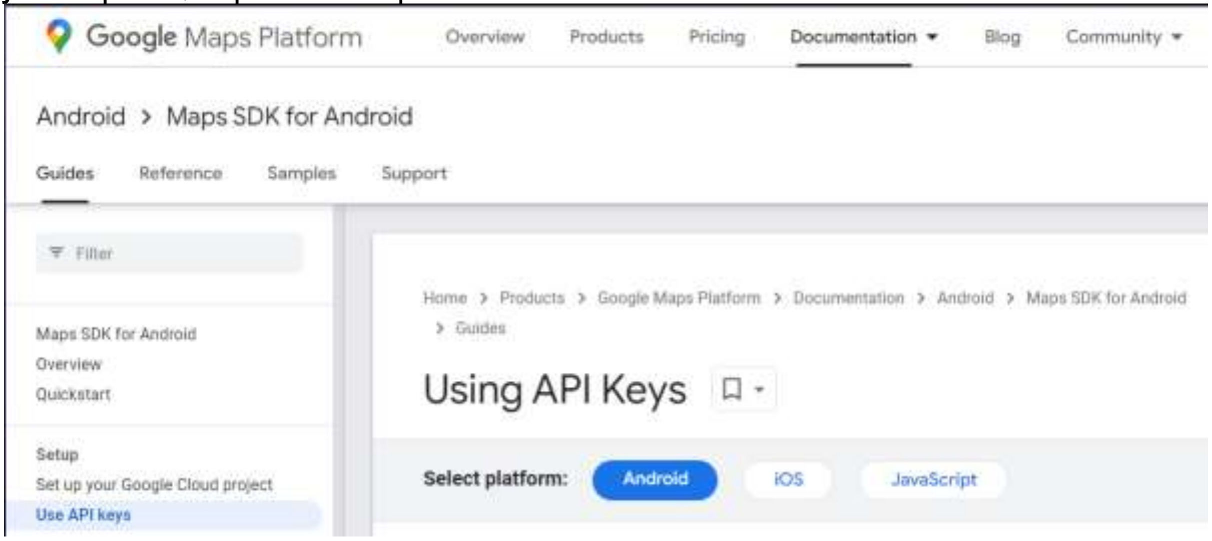
In order to include Google Maps in our app, we need to utilize the Google Maps Flutter plugin. This can be done by executing the following command within the project directory.



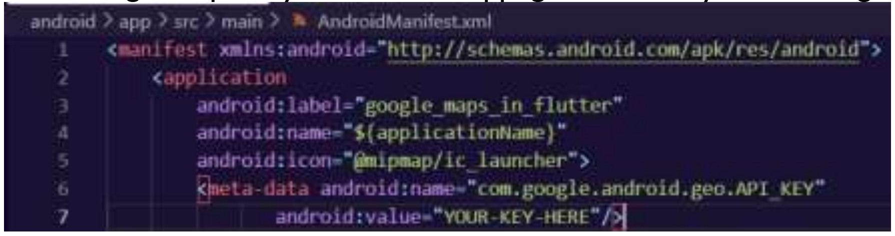
```
PS D:\Dev\google_maps_in_flutter> flutter pub add google_maps_flutter
Resolving dependencies...
+ csslib 1.0.0
  flutter_lints 2.0.3 (3.0.1 available)
+ flutter_plugin_android_lifecycle 2.0.17
+ flutter_web_plugins 0.0.0 from sdk flutter
+ google_maps 6.3.0
+ google_maps_flutter 2.5.3
+ google_maps_flutter_android 2.6.2
+ google_maps_flutter_ios 2.4.1
+ google_maps_flutter_platform_interface 2.4.3
+ google_maps_flutter_web 0.5.4+3
+ html 0.15.4
+ js 0.6.7 (0.7.0 available)
+ js_wrapping 0.7.4
  lints 2.1.1 (3.0.0 available)
```

## Including Google Maps in our app involves integrating the Google Maps functionality using the appropriate tools and plugins.

Incorporating Google Maps into a Flutter application requires setting up an API project with the Google Maps Platform. This involves configuring API keys for the Maps SDK for Android, Maps SDK for iOS, and Maps JavaScript API, as per their respective documentation.



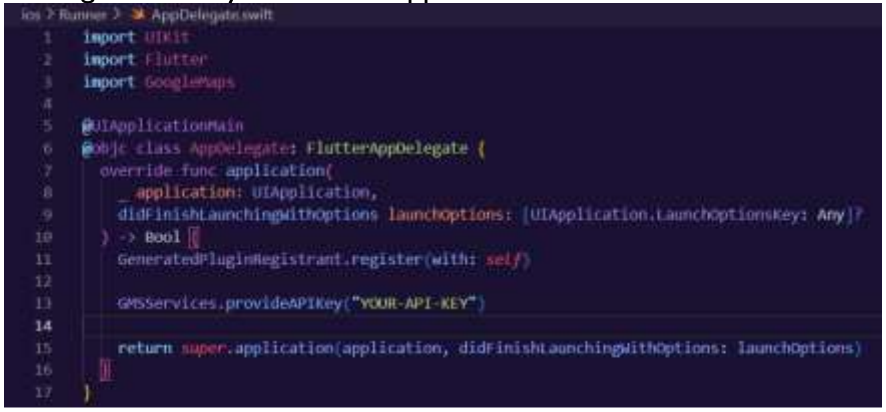
Add Google Maps to your Android app, get an API key from Google and add it to your app's settings.



android > app > src > main > AndroidManifest.xml

```xml
<manifest xmlns:android="http://schemas.android.com/apk/res/android">
    <application
        android:label="google_maps_in_flutter"
        android:name="${applicationName}"
        android:icon="@mipmap/ic_launcher">
        <meta-data android:name="com.google.android.geo.API_KEY"
               android:value="YOUR-KEY-HERE"/>
```

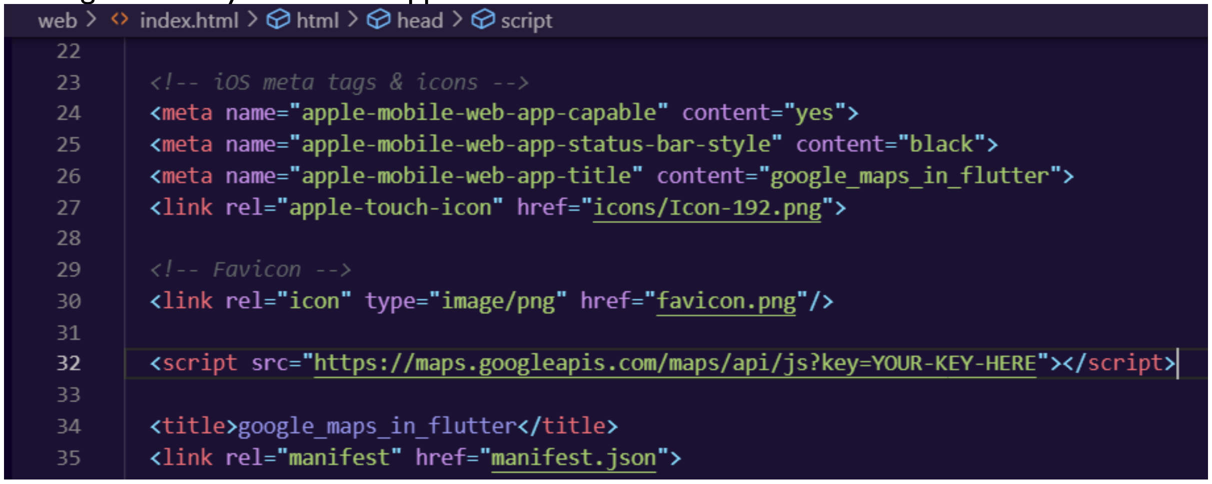
Adding an API key for an iOS app



ios > Runner > AppDelegate.swift

```swift
import UIKit
import Flutter
import GoogleMaps

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

    GMSServices.provideAPIKey("YOUR-API-KEY")

    return super.application(application, didFinishLaunchingWithOptions: launchOptions)
  }
}
```

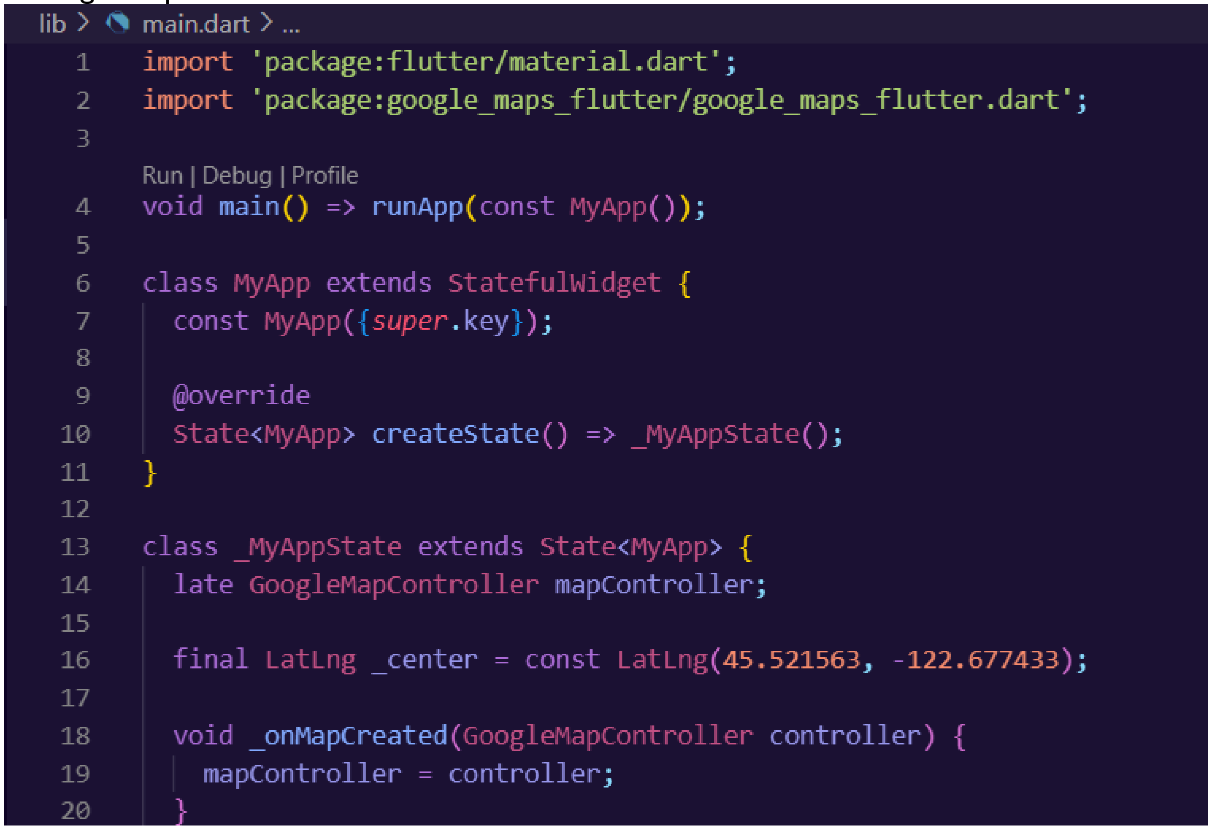
Adding an API key for a Web app



web > index.html > html > head > script

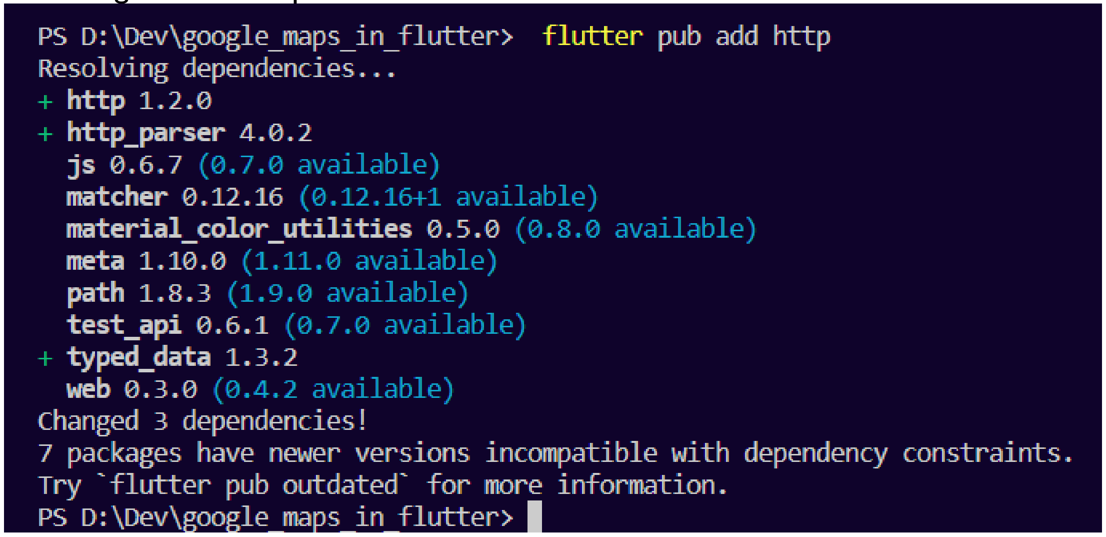
```html

  <!-- iOS meta tags & icons -->
  <meta name="apple-mobile-web-app-capable" content="yes">
  <meta name="apple-mobile-web-app-status-bar-style" content="black">
  <meta name="apple-mobile-web-app-title" content="google_maps_in_flutter">
  <link rel="apple-touch-icon" href="icons/Icon-192.png">

  <!-- Favicon -->
  <link rel="icon" type="image/png" href="favicon.png"/>

  <script src="https://maps.googleapis.com/maps/api/js?key=YOUR-KEY-HERE"></script>

  <title>google_maps_in_flutter</title>
  <link rel="manifest" href="manifest.json">
```

Putting a map on the screen



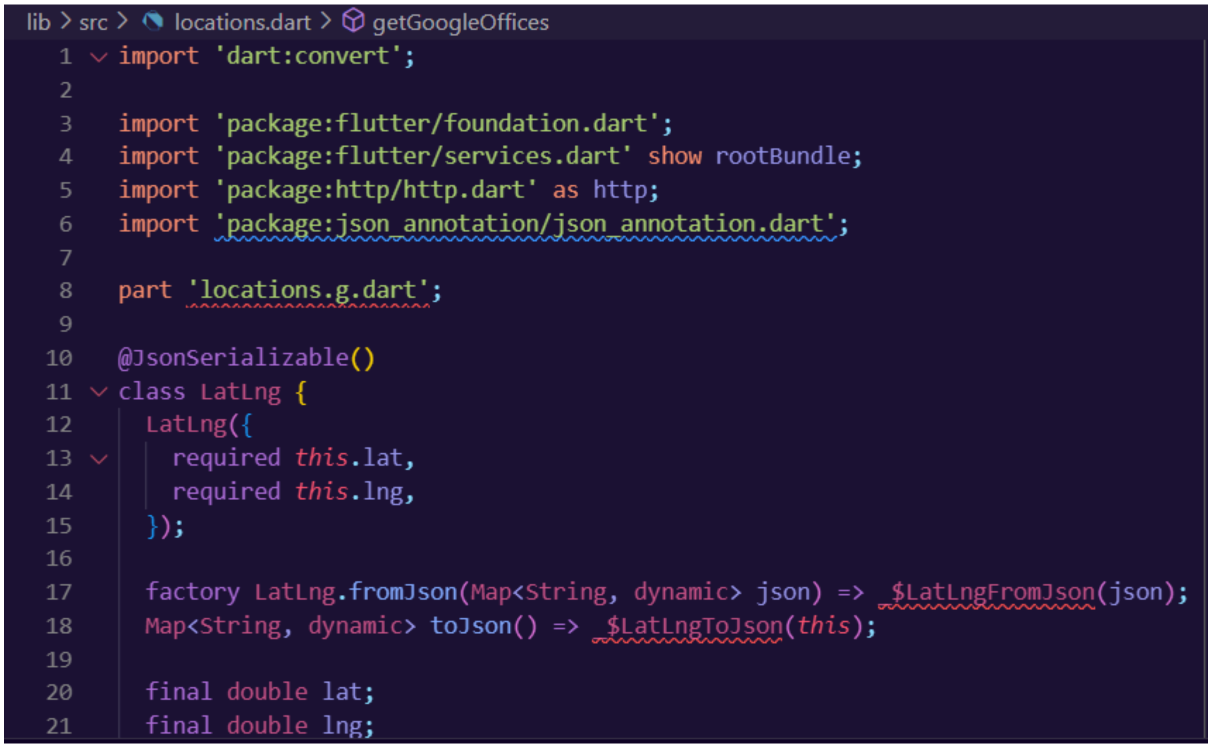
```
lib > main.dart > ...
import 'package:flutter/material.dart';
import 'package:google_maps_flutter/google_maps_flutter.dart';

Run | Debug | Profile
void main() => runApp(const MyApp());

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

  @override
  State<MyApp> createState() => _MyAppState();
}

class _MyAppState extends State<MyApp> {
  late GoogleMapController mapController;

  final LatLng _center = const LatLng(45.521563, -122.677433);

  void _onMapCreated(GoogleMapController controller) {
    mapController = controller;
  }
```

Put Google on the Map



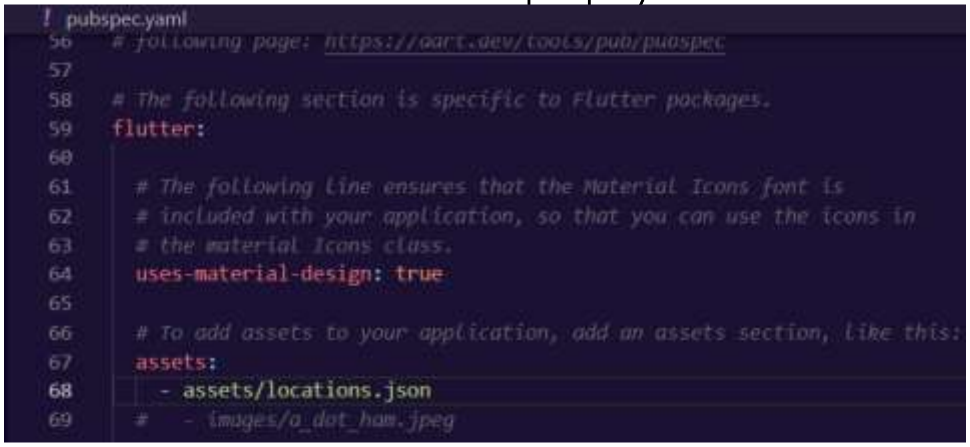
```
PS D:\Dev\google_maps_in_flutter>  flutter pub add http
Resolving dependencies...
+ http 1.2.0
+ http_parser 4.0.2
  js 0.6.7 (0.7.0 available)
  matcher 0.12.16 (0.12.16+1 available)
  material_color_utilities 0.5.0 (0.8.0 available)
  meta 1.10.0 (1.11.0 available)
  path 1.8.3 (1.9.0 available)
  test_api 0.6.1 (0.7.0 available)
+ typed_data 1.3.2
  web 0.3.0 (0.4.2 available)
Changed 3 dependencies!
7 packages have newer versions incompatible with dependency constraints.
Try `flutter pub outdated` for more information.
PS D:\Dev\google_maps_in_flutter>
```

Parsing JSON with code generation



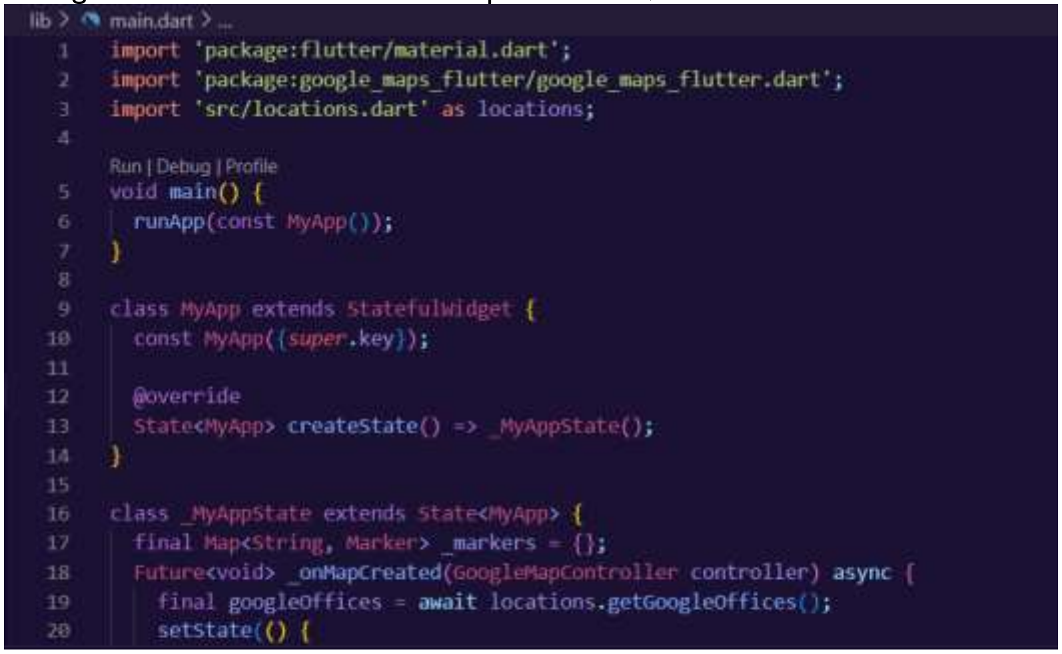
```dart
import 'dart:convert';

import 'package:flutter/foundation.dart';
import 'package:flutter/services.dart' show rootBundle;
import 'package:http/http.dart' as http;
import 'package:json_annotation/json_annotation.dart';

part 'locations.g.dart';

@JsonSerializable()
class LatLng {
  LatLng({
    required this.lat,
    required this.lng,
  });

  factory LatLng.fromJson(Map<String, dynamic> json) => _$LatLngFromJson(json);
  Map<String, dynamic> toJson() => _$LatLngToJson(this);

  final double lat;
  final double lng;
```

First, make sure to create the assets directory before pasting the data. Once the asset file is downloaded, you can add it to the "flutter" section of the pubspec.yaml file.



```yaml
# following page: https://dart.dev/tools/pub/pubspec

# 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:
    - assets/locations.json
  #   - images/a_dot_ham.jpeg
```

Change the main.dart file to fetch map data. Then, utilize the received information to mark offices on the map.



lib > main.dart > ...

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

Run | Debug | Profile
void main() {
  runApp(const MyApp());
}

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

  @override
  State<MyApp> createState() => _MyAppState();
}

class _MyAppState extends State<MyApp> {
  final Map<String, Marker> _markers = {};
  Future<void> _onMapCreated(GoogleMapController controller) async {
    final googleOffices = await locations.getGoogleOffices();
    setState(() {
```

## Problems:

This task familiarized us with integrating Google Maps into a Flutter app, which is a straightforward process. The code we developed is highly reusable for future projects. Although we couldn't test it on an emulator due to the lack of Google Maps Platform APIs, once we acquire valid API keys from an account with proper billing information, this code will function flawlessly. It will successfully display a map showcasing the locations of Google offices.