

CSE 162 Mobile Computing

Lab 4 Location and Map Programming

Hua Huang

Tasks

- ❖ Creating a google map
- ❖ Using Location services

Maps SDK for Android

- ❖ Set up the development environment
- ❖ Set up an Android device
- ❖ Create a Google Maps project in Android Studio
- ❖ Set up in Cloud Console
- ❖ Add the API key to your app
- ❖ Deploy and run the app

Set up the development environment

- ❖ Android Studio is required. If you haven't already done so, download and install it.
- ❖ Add the Google Play services SDK to Android Studio. The Maps SDK for Android is distributed as part of the Google Play services SDK, which you can add through the SDK Manager.

Set up an Android device

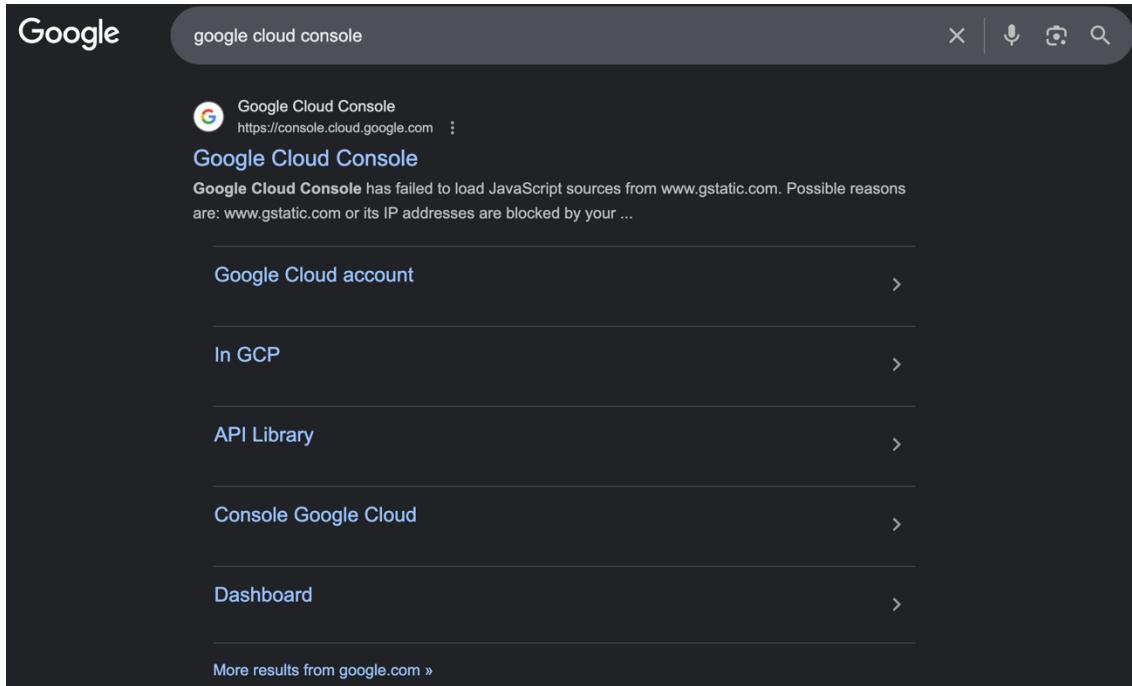
- >To run an app that uses the Maps SDK for Android, you must deploy it to an Android device or Android emulator that is based on Android 4.0 or higher and includes the Google APIs.

Create a Google Maps project in Android Studio

- ✦ Open Android Studio, and click Create New Project.
- ✦ In the New Project window, under the Phone category, select the Empty Activity
- ✦ Complete the Google Maps Activity form:
 - ☛ Set Language to Java
 - ☛ Set Minimum SDK to an SDK version compatible with your test device.
You must select a version greater than the minimum version required by the Maps SDK for Android version 18.0.x, which is currently Android API Level 19 (Android 4.4, KitKat) or higher.

Set up in Cloud Console

❖ On your browser look up Google Cloud Console



Set up in Cloud Console

- ❖ You need a Gmail account
- ❖ For first time users get \$300 free credits, click Try for free

The screenshot shows the Google Cloud homepage. At the top, there is a banner with a gift icon and the text "Start your Free Trial with \$300 in credit. Don't worry—you won't be charged if you run out of credits. [Learn more](#)". To the right of the banner are "Dismiss" and "Start free" buttons. Below the banner is the Google Cloud navigation bar, which includes the "Google Cloud" logo, a "Select a project" dropdown, a search bar with the placeholder "Search (/) for resources, docs, products, and more", and a "Search" button. To the right of the search bar are several icons: a star, a document, a bell, a question mark, and a user profile. The main content area features a "Welcome, test" message and a large call-to-action button labeled "Try Google Cloud with \$300 in free credits". Below this button is a list of three benefits with checkmarks: "Access to Google Cloud products and services", "90 days to spend your credits", and "No billing during trial". At the bottom of this section is a blue "Try for free" button. To the right of the main content, there is a sidebar with the heading "Other options" and links to "Try Gemini 2.0 Flash" and "Try Gemini". Below these links is a description of Google Cloud setup: "Configure Google Cloud for scalable, production-ready enterprise workloads." At the very bottom of the page, there is a section titled "Popular getting started resources" with a "Filter by" dropdown and several category buttons: "Web, Mobile, Game, Storage", "Containers, VMs, Hybrid/Multi, Move Workload", "Data, AI/ML", "Maps, APIs", and "General".

Popular getting started resources

Filter by [Web, Mobile, Game, Storage](#) [Containers, VMs, Hybrid/Multi, Move Workload](#) [Data, AI/ML](#) [Maps, APIs](#) [General](#)

Set up in Cloud Console

- You will need to setup a billing account and enable the Google Maps API.

 Try Google Cloud for free

Step 1 of 2 Account Information

 test
lunareclipse987654321@gmail.com [Switch account](#)

Country

By using this application, you agree to the [Google Cloud Platform](#), [Supplemental Free Trial](#), and [any applicable services and APIs](#).
Terms of Service.

[Agree & continue](#)

 Try Google Cloud for free

Step 2 of 2 Payment Information Verification

Don't worry, this trial is still free. Collecting your payment information helps us verify your identity to reduce fraud. You won't be charged unless you manually activate a full pay-as-you-go account or choose to prepay.

Contact information
 [Change](#)

Payment method
 [+](#)

Your info is saved in a [payments profile](#) and shared across Google services

[Start free](#)

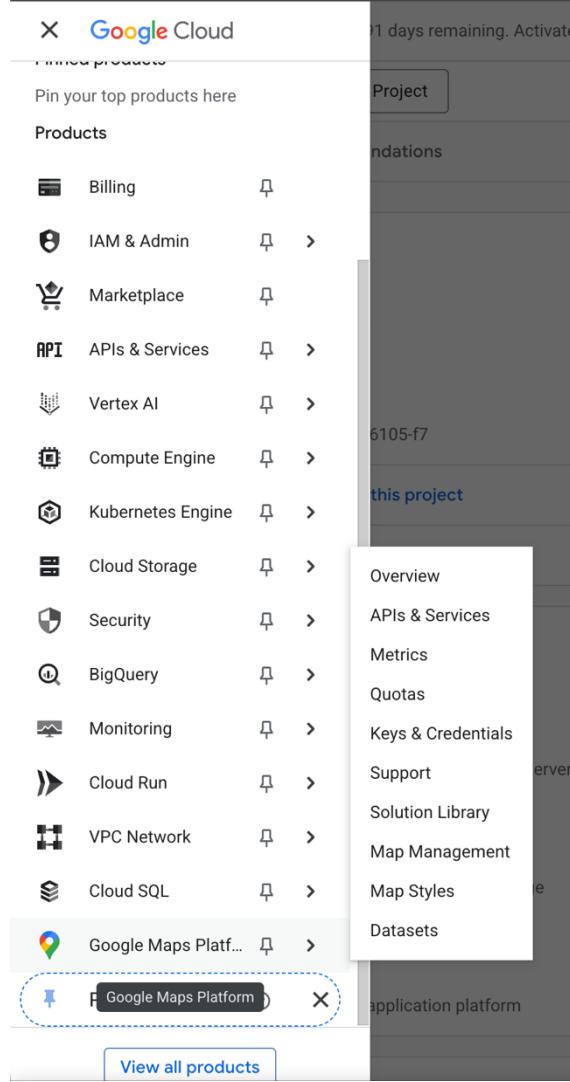
Access to Google Cloud products
Get everything you need to build and run your apps, websites and services, including Firebase and the Google Maps API.

\$300 in free credit
Try Google Cloud with \$300 in credit to spend over the next 90 days.

No automatic charges
You only start paying if you decide to activate a full, pay-as-you-go account or choose to prepay. Any remaining free credit is yours to keep.

Set up in Cloud Console

- ❖ Once you've created a billing account, and new project
- ❖ From the navigation menu choose Google Maps Platform option -> API&Services



Set up in Cloud Console

- After you click API & Services locate Maps SDK for Android and Enable it

 Maps SDK for Android Disable

Maps for your native Android app.

[Maps](#) [!\[\]\(4b44fd96c5b2c5c0a53a590089e166b6_img.jpg\)](#) [!\[\]\(2145567205625be51f8c0942c237fe22_img.jpg\) Keys](#) [!\[\]\(4e16fecd574d2675f9e366026e1f980f_img.jpg\) Metrics](#) [!\[\]\(e6be26a27d4ced49cc03f8bffef4522b_img.jpg\) Guides ↗](#)

Set up in Cloud Console

- Once enabled you will see your API key which will be used later.

API Keys

●	Name	Creation date	Restrictions ↑	Actions
<input checked="" type="checkbox"/>	Maps Platform API Key	Oct 23, 2025	31 APIs ...	Show key 

Creating a map api key string

Inside layout -> values -> strings.xml

```
</> strings.xml ×  
  
ⓘ Edit translations for all locales in the translations editor.  
  
1 <resources>  
2     <string name="app_name">Maps</string>  
3     ⚡ <string name="google_maps_key">YOUR_KEY_HERE</string>  
4  
5 </resources>
```

You need to create a new key and paste it over the “YOUR_KEY_HERE” placeholder

Add the API key to your app

- ➡ In Android Studio, open your project-level build.gradle file and add the following code to the dependencies element under buildscript.

```
dependencies {  
    implementation libs.appcompat  
    implementation libs.material  
    testImplementation libs.junit  
    androidTestImplementation libs.ext.junit  
    androidTestImplementation libs.espresso.core  
  
    implementation 'com.google.android.gms:play-services-maps:18.1.0'  
    implementation 'com.google.android.gms:play-services-location:21.0.1'  
}
```

- Next, open your module-level build.gradle file and add the following code to the plugins element.

```
id 'com.google.android.libraries.mapsplatform.secrets-grad  
le-plugin'
```

★ Save the file and sync your project with Gradle.

Reference the API Key in Manifest

- In your AndroidManifest.xml file, go to com.google.android.geo.API_KEY and update the android:value attribute as follows:

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

Location Acquisition

Registering for Location Updates

- The LocationManager handles registrations for GPS and network location updates
- In order for an object to receive updates from GPS, it must implement the LocationListener interface
- Once the LocationManager is obtained, an object registers for updates by calling requestLocationUpdates (there are multiple versions you can use)
- The arguments passed into the requestLocationUpdates method determine the granularity of location changes that will generate an event
 - send updates that are at least X meters apart
 - send updates at least this far apart in time
 - send updates that have this minimum accuracy

Getting Location Info and updates

```
2 usages
88     private void startLocationUpdates() {
89         LocationRequest request = new LocationRequest.Builder(
90             Priority.PRIORITY_HIGH_ACCURACY, intervalMillis: 10000)
91             .setWaitForAccurateLocation(true)
92             .build();
93
94         locationCallback = new LocationCallback() {
95             no usages
96             @Override
97             public void onLocationResult(@NonNull LocationResult result) {
98                 Location location = result.getLastLocation();
99                 if (location != null) {
100                     double lat = location.getLatitude();
101                     double lon = location.getLongitude();
102                     Log.i(TAG, msg: "Location update: " + lat + ", " + lon);
103
104                     locationInfo.setText(String.format("lat: %.5f, Lng: %.5f", lat, lon));
105
106                     LatLng pos = new LatLng(lat, lon);
107                     mMap.clear();
108                     mMap.addMarker(new MarkerOptions().position(pos).title("You are here"));
109                     mMap.moveCamera(CameraUpdateFactory.newLatLngZoom(pos, DEFAULT_ZOOM));
110                 }
111             }
112         };
113
114         if (ActivityCompat.checkSelfPermission(context: this, Manifest.permission.ACCESS_FINE_LOCATION)
115             != PackageManager.PERMISSION_GRANTED &&
116             ActivityCompat.checkSelfPermission(context: this, Manifest.permission.ACCESS_COARSE_LOCATION)
117             != PackageManager.PERMISSION_GRANTED) {
118             // TODO: Consider calling
119             // ActivityCompat#requestPermissions
120             // here to request the missing permissions, and then overriding
121             // public void onRequestPermissionsResult(int requestCode, String[] permissions,
122             //                                         int[] grantResults)
123             // to handle the case where the user grants the permission. See the documentation
124             // for ActivityCompat#requestPermissions for more details.
125             return;
126         }
127         fusedLocationClient.requestLocationUpdates(request, locationCallback, Looper.getMainLooper());
128     }
```

Location Providers

- ❑ The phone's location can be determined from multiple providers
 - ❑ GPS
 - ❑ Network
- ❑ GPS location updates consume significantly more power than network location updates but are more accurate
 - ❑ GPS: 25 seconds * 140mA = 1mAh
 - ❑ Network: 2 seconds * 180mA = 0.1mAh
- ❑ The provider argument determines which method will be used to get a location for you
- ❑ You can also register for the PASSIVE_PROVIDER which only updates you if another app is actively using GPS / Network location

<code>String</code>	<code>GPS_PROVIDER</code>
Name of the GPS location provider.	
<code>String</code>	<code>NETWORK_PROVIDER</code>
Name of the network location provider.	
<code>String</code>	<code>PASSIVE_PROVIDER</code>
A special location provider for receiving locations without actually initiating a location fix.	

Being a Good Citizen...

- ❖ It is very important that you unregister your App when you no longer need updates
- ❖ For example, you should always unregister your listener when your Activity is paused!
- ❖ If you unregister when you pause, you must also reregister when you resume
 - ☞ This is true for all sensors!

```
protected void onPause() {
    super.onPause();
    try{
        locationManager.removeUpdates(this);
        catch(SecurityException e){ Log.e("Err", "No Location update permission remover");}
    }
    protected void onResume() {
        super.onResume();
        if(ContextCompat.checkSelfPermission(this,
Manifest.permission.ACCESS_COARSE_LOCATION)== PackageManager.PERMISSION_GRANTED)
            locationManager.requestLocationUpdates(LocationManager.GPS_PROVIDER,
10, Criteria.ACCURACY_COARSE, this);
    }
}
```

MapsActivity.java Overview

MapsActivity.java

```
1 package rahul.cse_162_2l.maps;
2
3
4 import android.Manifest;
5 import android.content.pm.PackageManager;
6 import android.location.Location;
7 import android.os.Bundle;
8 import android.os.Looper;
9 import android.util.Log;
10 import android.widget.ImageButton;
11 import android.widget.TextView;
12 import android.widget.Toast;
13
14 import androidx.annotation.NonNull;
15 import androidx.appcompat.app.AppCompatActivity;
16 import androidx.core.app.ActivityCompat;
17 import androidx.core.content.ContextCompat;
18
19 import com.google.android.gms.location.FusedLocationProviderClient;
20 import com.google.android.gms.location.LocationCallback;
21 import com.google.android.gms.location.LocationRequest;
22 import com.google.android.gms.location.LocationResult;
23 import com.google.android.gms.location.LocationServices;
24 import com.google.android.gms.location.Priority;
25 import com.google.android.gms.maps.CameraUpdateFactory;
26 import com.google.android.gms.maps.GoogleMap;
27 import com.google.android.gms.maps.MapsInitializer;
28 import com.google.android.gms.maps.OnMapReadyCallback;
29 import com.google.android.gms.maps.SupportMapFragment;
30 import com.google.android.gms.maps.model.LatLng;
31 import com.google.android.gms.maps.model.MarkerOptions;
```

Import section

MapsActivity.java Overview

```
32
33 ></> public class MapsActivity extends AppCompatActivity implements OnMapReadyCallback {
34
35     1 usage
36     private static final String TAG = "MAPS_AUTO";
37     2 usages
38
39     ? private static final int LOCATION_PERMISSION_REQUEST_CODE = 101;
40     1 usage
41     private static final float DEFAULT_ZOOM = 15f;
42
43     8 usages
44     private GoogleMap mMap;
45     4 usages
46     private FusedLocationProviderClient fusedLocationClient;
47     4 usages
48     private LocationCallback locationCallback;
49     2 usages
50     private TextView locationInfo;
51     1 usage
52     private ImageButton mylocationBtn;
53
54     @Override
55     protected void onCreate(Bundle savedInstanceState) {
56         super.onCreate(savedInstanceState);
57         setContentView(R.layout.activity_maps);
58
59         locationInfo = findViewById(R.id.location_info);
60         myLocationBtn = findViewById(R.id.btn_my_location);
61         fusedLocationClient = LocationServices.getFusedLocationProviderClient(this);
62
63         SupportMapFragment mapFragment =
64             (SupportMapFragment) getSupportFragmentManager().findFragmentById(R.id.map);
65         if (mapFragment != null) mapFragment.getMapAsync(callback: this);
66     }
67 }
```

OnCreate() function

MapsActivity.java

Variables, onCreate, onMapReady

```
61      no usages
62  @Override
63      public void onMapReady(@NonNull GoogleMap googleMap) {
64          mMap = googleMap;
65          MapsInitializer.initialize( context: this, MapsInitializer.Renderer.LATEST, callback: null );
66          mMap.getUiSettings().setZoomControlsEnabled(true);
67          mMap.getUiSettings().setMyLocationButtonEnabled(false);
68          checkLocationPermission();
69      }
70
71      1 usage
72      private void checkLocationPermission() {
73          if (ContextCompat.checkSelfPermission( context: this, Manifest.permission.ACCESS_FINE_LOCATION )
74              == PackageManager.PERMISSION_GRANTED) {
75              enableLocationFeatures();
76          } else {
77              ActivityCompat.requestPermissions( activity: this,
78                  new String[]{Manifest.permission.ACCESS_FINE_LOCATION},
79                  LOCATION_PERMISSION_REQUEST_CODE );
80          }
81      }
82
83      2 usages
84      private void enableLocationFeatures() {
85          if (ActivityCompat.checkSelfPermission( context: this, Manifest.permission.ACCESS_FINE_LOCATION )
86              != PackageManager.PERMISSION_GRANTED) return;
87          mMap.setMyLocationEnabled(true);
88          startLocationUpdates();
89      }
90
```

```
88     2 usages
89     private void startLocationUpdates() {
90         LocationRequest request = new LocationRequest.Builder(
91             Priority.PRIORITY_HIGH_ACCURACY, intervalMillis: 10000)
92             .setWaitForAccurateLocation(true)
93             .build();
94
95         locationCallback = new LocationCallback() {
96             no usages
97             @Override
98             public void onLocationResult(@NonNull LocationResult result) {
99                 Location location = result.getLastLocation();
100                if (location != null) {
101                    double lat = location.getLatitude();
102                    double lon = location.getLongitude();
103                    Log.i(TAG, msg: "Location update: " + lat + ", " + lon);
104
105                    locationInfo.setText(String.format("Lat: %.5f, Lng: %.5f", lat, lon));
106
107                    LatLng pos = new LatLng(lat, lon);
108                    mMap.clear();
109                    mMap.addMarker(new MarkerOptions().position(pos).title("You are here"));
110                    mMap.moveCamera(CameraUpdateFactory.newLatLngZoom(pos, DEFAULT_ZOOM));
111                }
112            }
113
114            if (ActivityCompat.checkSelfPermission(context: this, Manifest.permission.ACCESS_FINE_LOCATION)
115                != PackageManager.PERMISSION_GRANTED &&
116                ActivityCompat.checkSelfPermission(context: this, Manifest.permission.ACCESS_COARSE_LOCATION)
117                != PackageManager.PERMISSION_GRANTED) {
118                // TODO: Consider calling
119                //    ActivityCompat#requestPermissions
120                // here to request the missing permissions, and then overriding
121                //    public void onRequestPermissionsResult(int requestCode, String[] permissions,
122                //                                              int[] grantResults)
123                // to handle the case where the user grants the permission. See the documentation
124                // for ActivityCompat#requestPermissions for more details.
125                return;
126            }
127            fusedLocationClient.requestLocationUpdates(request, locationCallback, Looper.getMainLooper());
128        }
129    }
```

MapsActivity.java

- `startLocationUpdates()`
listens for location change

```
@Override  
protected void onPause() {  
    super.onPause();  
    if (fusedLocationClient != null && locationCallback != null) {  
        fusedLocationClient.removeLocationUpdates(locationCallback);  
    }  
}  
  
@Override  
protected void onResume() {  
    super.onResume();  
    if (mMap != null &&  
        ContextCompat.checkSelfPermission(context: this, Manifest.permission.ACCESS_FINE_LOCATION)  
            == PackageManager.PERMISSION_GRANTED) {  
        startLocationUpdates();  
    }  
}
```

14 usages

```
@Override  
public void onRequestPermissionsResult(int requestCode,  
                                      @NonNull String[] permissions,  
                                      @NonNull int[] grantResults) {  
    super.onRequestPermissionsResult(requestCode, permissions, grantResults);  
    if (requestCode == LOCATION_PERMISSION_REQUEST_CODE) {  
        if (grantResults.length > 0 && grantResults[0] == PackageManager.PERMISSION_GRANTED) {  
            enableLocationFeatures();  
        } else {  
            Toast.makeText(context: this, text: "Location permission denied", Toast.LENGTH_SHORT).show();  
        }  
    }  
}
```

MapsActivity.java

- OnResume() and OnPause()

activity_maps.xml

```
1  <?xml version="1.0" encoding="utf-8"?>
2  <androidx.constraintlayout.widget.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"
3      xmlns:app="http://schemas.android.com/apk/res-auto"
4      xmlns:tools="http://schemas.android.com/tools"
5      android:layout_width="match_parent"
6      android:layout_height="match_parent"
7      tools:context=".MapsActivity">
8
9      <fragment
10         android:id="@+id/map"
11         android:name="com.google.android.gms.maps.SupportMapFragment"
12         android:layout_width="0dp"
13         android:layout_height="0dp"
14         app:layout_constraintBottom_toBottomOf="parent"
15         app:layout_constraintEnd_toEndOf="parent"
16         app:layout_constraintStart_toStartOf="parent"
17         app:layout_constraintTop_toTopOf="parent" />
18
19      <ImageButton
20          android:id="@+id/btn_my_location"
21          android:layout_width="56dp"
22          android:layout_height="56dp"
23          android:layout_marginEnd="26dp"
24          android:layout_marginBottom="85dp"
25          android:backgroundTint="@color/purple_500"
26          android:src="@android:drawable/ic_menu_mylocation"
27          app:layout_constraintBottom_toBottomOf="parent"
28          app:layout_constraintEnd_toEndOf="parent"
29          app:tint="@android:color/white" />
30
```

activity_maps.xml

```
31     <TextView  
32         android:id="@+id/location_info"  
33         android:layout_width="wrap_content"  
34         android:layout_height="wrap_content"  
35         android:layout_marginStart="16dp"  
36         android:layout_marginTop="16dp"  
37         android:background="#80000000"  
38         android:padding="8dp"  
39         android:text="Locating..."  
40         android:textColor="@android:color/white"  
41         android:textSize="14sp"  
42         app:layout_constraintStart_toStartOf="parent"  
43         app:layout_constraintTop_toTopOf="parent" />  
44     </androidx.constraintlayout.widget.ConstraintLayout>  
45
```

MainActivity.java Overview

```
© MainActivity.java ×

1 package rahul.cse_162_21.maps;
2
3
4 import android.content.Intent;
5 import android.os.Bundle;
6
7 import androidx.appcompat.app.AppCompatActivity;
8
9 public class MainActivity extends AppCompatActivity {
10     @Override
11     protected void onCreate(Bundle savedInstanceState) {
12         super.onCreate(savedInstanceState);
13         setContentView(R.layout.activity_main);
14
15         // Launch the map automatically
16         Intent intent = new Intent(packageContext: MainActivity.this, MapsActivity.class)
17         startActivity(intent);
18
19         // Optional: close MainActivity so back button doesn't return here
20         finish();
21     }
22 }
23
```

activity_main.xml Overview

```
</> activity_main.xml ×

1   <?xml version="1.0" encoding="utf-8"?>
2   <androidx.constraintlayout.widget.ConstraintLayout
3       xmlns:android="http://schemas.android.com/apk/res/android"
4       xmlns:app="http://schemas.android.com/apk/res-auto"
5       xmlns:tools="http://schemas.android.com/tools"
6       android:layout_width="match_parent"
7       android:layout_height="match_parent"
8       tools:context=".MainActivity">
9
10
11      <TextView
12          android:id="@+id/titleText"
13          android:layout_width="wrap_content"
14          android:layout_height="wrap_content"
15          android:text="Launching Map..."
16          android:textSize="22sp"
17          android:textStyle="bold"
18          app:layout_constraintTop_toTopOf="parent"
19          app:layout_constraintStart_toStartOf="parent"
20          app:layout_constraintEnd_toEndOf="parent"
21          app:layout_constraintBottom_toBottomOf="parent"/>
22
23      </androidx.constraintlayout.widget.ConstraintLayout>
```

M AndroidManifest.xml ×

```
1  <?xml version="1.0" encoding="utf-8"?>
2  <manifest xmlns:android="http://schemas.android.com/apk/res/android"
3      xmlns:tools="http://schemas.android.com/tools">
4          <uses-permission android:name="android.permission.ACCESS_FINE_LOCATION" />
5          <uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION" />
6
7          <application
8              android:allowBackup="true"
9              android:dataExtractionRules="@xml/data_extraction_rules"
10             android:fullBackupContent="@xml/backup_rules"
11             android:icon="@mipmap/ic_launcher"
12             android:label="Maps"
13             android:roundIcon="@mipmap/ic_launcher_round"
14             android:supportsRtl="true"
15             android:theme="@style/Theme.Maps">
16              <!-- Google Maps API Key --&gt;
17              &lt;meta-data
18                  android:name="com.google.android.geo.API_KEY"
19                  android:value="@string/google_maps_key" /&gt;
20
21              &lt;activity android:name=".MapsActivity" /&gt;
22              &lt;activity android:name=".MainActivity"
23                  android:exported="true"&gt;
24                  &lt;intent-filter&gt;
25                      &lt;action android:name="android.intent.action.MAIN" /&gt;
26                      &lt;category android:name="android.intent.category.LAUNCHER" /&gt;
27                  &lt;/intent-filter&gt;
28              &lt;/activity&gt;
29          &lt;/application&gt;
30
31      &lt;/manifest&gt;</pre>
```

AndroidManifest.xml

Build grade (project)

build.gradle (Maps) ×

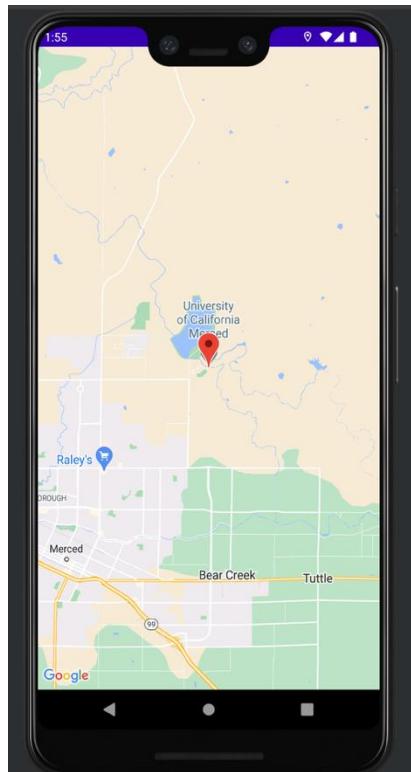
```
1 // Top-level build file where you can add configuration options common to all sub-projects/modules.  
2 plugins {  
3     alias(libs.plugins.android.application) apply false  
4 }
```

Build grade (app/module)

```
build.gradle (:app) ×

1  plugins {
2      alias(libs.plugins.android.application)
3  }
4
5  android {
6      namespace 'rahul.cse_162_2l.maps'
7      compileSdk 36
8
9      defaultConfig { DefaultConfig it ->
10         applicationId "rahul.cse_162_2l.maps"
11         minSdk 24
12         targetSdk 36
13         versionCode 1
14         versionName "1.0"
15     }
16
17     testInstrumentationRunner "androidx.test.runner.AndroidJUnitRunner"
18 }
19
20 buildTypes { NamedDomainObjectContainer<BuildType> it ->
21     release {
22         minifyEnabled false
23         proguardFiles getDefaultProguardFile('proguard-android-optimize.txt'), 'proguard-rules.pro
24     }
25 }
26 compileOptions { CompileOptions it ->
27     sourceCompatibility JavaVersion.VERSION_11
28     targetCompatibility JavaVersion.VERSION_11
29 }
30
31 dependencies {
32
33     implementation libs.appcompat
34     implementation libs.material
35     testImplementation libs.junit
36     androidTestImplementation libs.ext.junit
37     androidTestImplementation libs.espresso.core
38     implementation 'com.google.android.gms:play-services-maps:18.1.0'
39     implementation 'com.google.android.gms:play-services-location:21.0.1'
40 }
```

Final App



Assignments

- ★Display the map
- ★Set the map view based on the GPS tracking results