```
package com.example.latihanmaplbs;

import android.Manifest;
import android.content.pm.PackageManager;
import android.location.Location;
import android.os.Build;

import android.os.Bundle;
import android.widget.Toast;

import androidx.annotation.NonNull;
import androidx.annotation.Nullable;
import androidx.core.app.ActivityCompat;
import androidx.core.content.ContextCompat;
import androidx.fragment.app.FragmentActivity;

import com.google.android.gms.common.ConnectionResult;
import com.google.android.gms.common.api.GoogleApiClient;
import com.google.android.gms.location.LocationListener;
import com.google.android.gms.location.LocationRequest;
import com.google.android.gms.location.LocationServices;
import com.google.android.gms.maps.CameraUpdateFactory;
import com.google.android.gms.maps.GoogleMap;
import com.google.android.gms.maps.OnMapReadyCallback;
import com.google.android.gms.maps.SupportMapFragment;
import com.google.android.gms.maps.model.CameraPosition;
import com.google.android.gms.maps.model.LatLng;
import com.google.android.gms.maps.model.Marker;

public class LokasiUpdate extends FragmentActivity implements OnMapReadyCallback,
GoogleApiClient.ConnectionCallbacks, GoogleApiClient.OnConnectionFailedListener,
LocationListener {
    public static final int MY_PERMISSIONS_REQUEST_LOCATION = 99;
    GoogleApiClient mGoogleApiClient;
    private GoogleMap mMap;
    private LocationRequest mLocationRequest;
    private Marker mCurrLocationMarker;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_lokasi_update);
        // Memuat SupportMapFragment dan memberi notifikasi saat telah siap.
        SupportMapFragment mapFragment = (SupportMapFragment)
getSupportFragmentManager().findFragmentById(R.id.locMap);
        mapFragment.getMapAsync(this);
    }

    @Override
    public void onMapReady(GoogleMap googleMap) {
        mMap = googleMap;
        //Memulai Google Play Services
        if (android.os.Build.VERSION.SDK_INT >= Build.VERSION_CODES.M) {
            if (ContextCompat.checkSelfPermission(this,
android.Manifest.permission.ACCESS_FINE_LOCATION) ==
```

```
PackageManager.PERMISSION_GRANTED) {
                buildGoogleApiClient();
                mMap.setMyLocationEnabled(true);
            }
        } else {
            buildGoogleApiClient();
            mMap.setMyLocationEnabled(true);
        }
    }

    private void buildGoogleApiClient() {
        mGoogleApiClient = new GoogleApiClient.Builder(this)
                .addConnectionCallbacks(this)
                .addOnConnectionFailedListener(this)
                .addApi(LocationServices.API).build();
        mGoogleApiClient.connect();
    }

    @Override
    public void onConnected(@Nullable Bundle bundle) {
        LocationRequest mLocationRequest = new LocationRequest();
        mLocationRequest.setInterval(1000);
        mLocationRequest.setFastestInterval(1000);

mLocationRequest.setPriority(LocationRequest.PRIORITY_BALANCED_POWER_ACCURACY);
        if (ContextCompat.checkSelfPermission(this,
android.Manifest.permission.ACCESS_FINE_LOCATION) ==
PackageManager.PERMISSION_GRANTED) {

LocationServices.FusedLocationApi.requestLocationUpdates(mGoogleApiClient,
mLocationRequest, this);
        }
    }

    @Override
    public void onConnectionSuspended(int i) {
    }

    @Override
    public void onConnectionFailed(@NonNull ConnectionResult connectionResult) {
    }

    @Override
    public void onLocationChanged(Location location) {
        if (mCurrLocationMarker != null) {
            mCurrLocationMarker.remove();
        }
        LatLng latLng = new LatLng(location.getLatitude(), location.getLongitude());
        CameraPosition cameraPosition = new CameraPosition.Builder().target(new
LatLng(latLng.latitude, latLng.longitude)).zoom(16).build();
        mMap.animateCamera(CameraUpdateFactory.newCameraPosition(cameraPosition));
        if (mGoogleApiClient != null) {
            LocationServices.FusedLocationApi.removeLocationUpdates(mGoogleApiClient,
this);
        }
```

```
    }

    public boolean checkLocationPermission() {
        if (ContextCompat.checkSelfPermission(this,
android.Manifest.permission.ACCESS_FINE_LOCATION) !=
PackageManager.PERMISSION_GRANTED) {
            if (ActivityCompat.shouldShowRequestPermissionRationale(this,
android.Manifest.permission.ACCESS_FINE_LOCATION)) {
                ActivityCompat.requestPermissions(this, new
String[]{android.Manifest.permission.ACCESS_FINE_LOCATION},
MY_PERMISSIONS_REQUEST_LOCATION);
            } else {
                ActivityCompat.requestPermissions(this, new
String[]{android.Manifest.permission.ACCESS_FINE_LOCATION},
MY_PERMISSIONS_REQUEST_LOCATION);
            }
            return false;
        } else {
            return true;
        }
    }

    @Override
    public void onRequestPermissionsResult(int requestCode, String[] permissions,
@NonNull int[] grantResults) {
        super.onRequestPermissionsResult(requestCode, permissions, grantResults);
        if (requestCode == MY_PERMISSIONS_REQUEST_LOCATION) {
            if (grantResults.length > 0 && grantResults[0] ==
PackageManager.PERMISSION_GRANTED) {
                // Izin diberikan.
                if (ContextCompat.checkSelfPermission(this,
Manifest.permission.ACCESS_FINE_LOCATION) == PackageManager.PERMISSION_GRANTED) {
                    if (mGoogleApiClient == null) {
                        buildGoogleApiClient();
                    }
                    mMap.setMyLocationEnabled(true);
                }
            } else {
                // Izin ditolak.
                Toast.makeText(this, "permission denied", Toast.LENGTH_LONG).show();
            }
        }
    }
}
```