```
package com.example.covid_19alertapp.activities;


import androidx.fragment.app.FragmentActivity;


import android.app.AlertDialog;

import android.content.DialogInterface;

import android.os.Bundle;

import android.util.Log;

import android.widget.Toast;


import com.example.covid_19alertapp.R;

import com.example.covid_19alertapp.models.MapMarkerLocation;

import com.example.covid_19alertapp.roomdatabase.VisitedLocations;

import com.example.covid_19alertapp.roomdatabase.VisitedLocationsDao;

import com.example.covid_19alertapp.roomdatabase.VisitedLocationsDatabase;

import com.example.covid_19alertapp.sharedPreferences.UserInfoSharedPreferences;

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.LatLng;

import com.google.android.gms.maps.model.Marker;

import com.google.android.gms.maps.model.MarkerOptions;


import java.util.ArrayList;
import java.util.List;

public class MyLocationsMapsActivity extends FragmentActivity implements
        OnMapReadyCallback, GoogleMap.OnMarkerClickListener {

    private GoogleMap mMap;

    // model
    private List<MapMarkerLocation> locations = new ArrayList<>();
    private int listPosition;

    // local database
    private VisitedLocationsDatabase roomDatabase;
    private VisitedLocationsDao visitedLocationsDao;
    private List<VisitedLocations> visitedLocationsList = new ArrayList<>();

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);

        roomDatabase = VisitedLocationsDatabase.getDatabase(this);
        visitedLocationsDao = roomDatabase.visitedLocationsDao();

        setContentView(R.layout.activity_my_locations_maps);
        // Obtain the SupportMapFragment and get notified when the map is ready to be used.
        SupportMapFragment mapFragment = (SupportMapFragment) getSupportFragmentManager()
                .findFragmentById(R.id.map);
        mapFragment.getMapAsync(this);
    }



    @Override
    public void onMapReady(GoogleMap googleMap) {
        mMap = googleMap;

        // move camera to home
        moveCameraToHome();

        // fetch locations from local db and plot on map
        fetchNShowLocationMarkers();

        mMap.setOnMarkerClickListener(this);
    }

    private void moveCameraToHome() {

        // home = latitude,longitude
        String[] home = UserInfoSharedPreferences.getHomeLatLng(this).split(",");

        LatLng homeLatLng = new LatLng( Double.valueOf(home[0]), Double.valueOf(home[1]));
        mMap.moveCamera(CameraUpdateFactory.newLatLngZoom(homeLatLng, 16.5f));

    }


    private void fetchNShowLocationMarkers() {

        roomDatabase.databaseWriteExecutor.execute(new Runnable() {
            @Override
            public void run() {


                visitedLocationsList = visitedLocationsDao.fetchAll();

                listPosition = 0;

                for (VisitedLocations visitedLocation: visitedLocationsList) {

                    // pk = latLon_dateTime
                    String[] splitLLDT = visitedLocation.splitPrimaryKey();

                    final MapMarkerLocation location = new MapMarkerLocation(splitLLDT[0], splitLLDT[1]);
                    locations.add(location);

                    // plot marker on map
```

```
          runOnUiThread(new Runnable() {
            @Override
            public void run() {

              String markerTitle = locations.get(listPosition).getMeaningfulDateTime();
              LatLng markerLatLng = new LatLng(locations.get(listPosition).getLatitude(),
locations.get(listPosition).getLongitude());

              // show the marker
              Marker myLocationMarker =
                  mMap.addMarker( new MarkerOptions().position(markerLatLng).title(markerTitle));

              // tag = primary key of local db
              myLocationMarker.setTag(
                  locations.get(listPosition).getRawLatLon() +
                  "_"+
                  locations.get(listPosition).getRawDateTime()
              );

              listPosition++;

            }
          });

        }
        }
    });


}

@Override
public boolean onMarkerClick(final Marker marker) {

    // present delete location option to user

    // marker tag has local db PK
    final String tag = (String) marker.getTag();

    AlertDialog Dialog = new AlertDialog.Builder(this)
            .setMessage(marker.getTitle())
            .setPositiveButton("Delete", new DialogInterface.OnClickListener() {
                @Override
                public void onClick(DialogInterface dialog, int which) {

                    // delete location from local db
                    roomDatabase.databaseWriteExecutor.execute(new Runnable() {
                        @Override
                        public void run() {
                            visitedLocationsDao.deletebyPrimaryKey(tag);


                        }
                    });


                    marker.remove();
                    Toast.makeText(MyLocationsMapsActivity.this, "location removed", Toast.LENGTH_LONG)
                            .show();


                    dialog.dismiss();


                }
            })
            .setNegativeButton("dismiss", new DialogInterface.OnClickListener() {
                @Override
                public void onClick(DialogInterface dialog, int which) {
                    dialog.dismiss();
                }
            })
            .show();



    return false;
}
```

```
}
```