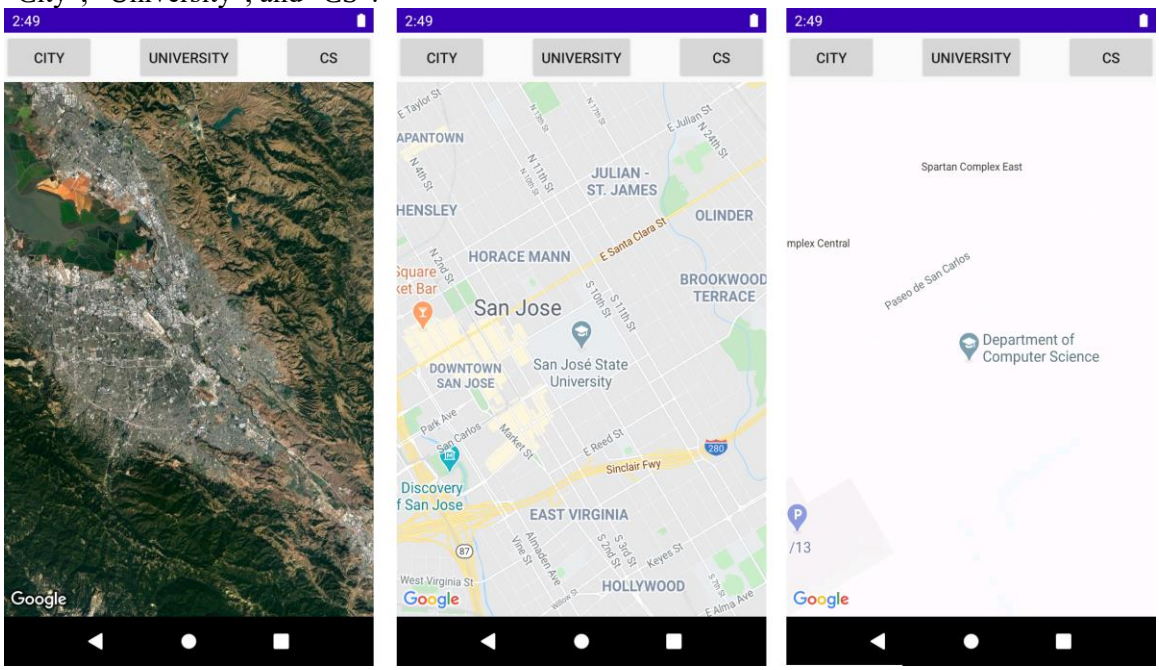
Assignment #4
Mobile Device Development
Google Maps and SQLite
Due Date: 11/16/2021@11:59pm

In this assignment you will extend My locations and Google map exercise. The assumption is that you have already completed the example and the following screens appear by clicking on the "City", "University", and "CS".



In this assignment you will store and retrieve Google Maps locations in SQLite database from Google Maps Android API v2.

- On taping a location in the Google Map, a marker will be drawn at the taped location and the corresponding latitude longitude coordinates with Google Map zoom level will be saved in SQLite database.
- On restarting the application, the saved locations are retrieved from the SQLite database and redrawn in the Google Maps.
- On a long tap, all locations will be removed from the map and database.

The database is accessed via *Content Providers*. The insertion and deletion operations are executed using *AsyncTask* objects and the data retrieval is done using *CursorLoader*.

Note: You may use the same API key you got for lab example number 2 assuming that you won't change project's package name. If you don't have the key, please follow the below link to get the API key for Google Maps Android API v2.
https://developers.google.com/maps/documentation/android/start

1) Create a class named *LocationsDB* that extends *SQLiteOpenHelper* and define the following variables:
    a. Database name
    b. Table name

c. Field 1: primary key
d. Field 2: latitude
e. Field 3: longitude
f. Field 4: zoom level of the map

2) In *LocationsDB* class develop the following methods:
   a. A callback method that is invoked when the method getReadableDatabase()/getWritableDatabase() is called provided the database doesn't exist.
   b. A method that inserts a new location to the table.
   c. A method that deletes all locations from the table.
   d. A method that returns all the locations from the table.
3) Write a custom Content Provider class named *LocationsContentProvider* and define the following variables:
   a. Content Provider's name
   b. A URI for Content Provider to do operations on location table.

   You are not limited to the above variables alone. Define any variable as needed.
4) In *LocationsContentProvider* class develop the following methods:
   a. A callback method that is invoked when insert operation is requested on this content provider.
   b. A callback method that is invoked when delete operation is requested on this content provider.
   c. A callback method that is invoked by default content URI.
5) Change the *MainActivity* class definition so it also implements *LoaderManager.LoaderCallbacks<Cursor>*

   a. Complete *onMapReady()* method as follow:

```
@Override
public void onMapReady(GoogleMap googleMap) {
    map = googleMap;
    // Invoke LoaderCallbacks to retrieve and draw already saved locations in map

    map.setOnMapClickListener(new GoogleMap.OnMapClickListener() {
        @Override
        public void onMapClick(LatLng point) {
            // Add a maker to the map
            // Creating an instance of LocationInsertTask
            // Storing the latitude, longitude and zoom level to SQList database
            // Display "Maker is added to the Map" message
        }
    });

    map.setOnMapLongClickListener(new GoogleMap.OnMapLongClickListener() {
        @Override
        public void onMapLongClick(LatLng point) {
            // Removing all markers from the Google Map
            // Creating an instance of LocationDeleteTask
            // Deleting all the rows from SQLite database table
            // Display "All makers are removed" message
        }
    });
}
```

b. Complete LocationInsertTask() method as follow:

```
private class LocationInsertTask extends AsyncTask<ContentValues, Void, Void>{
    @Override
    protected Void doInBackground(ContentValues... contentValues) {
        // Setting up values to insert the clicked location into SQLite database */
        return null;
    }
}
```

c. Complete LocationDeleteTask() method as follow:

```
private class LocationDeleteTask extends AsyncTask<Void, Void, Void>{
    @Override
    protected Void doInBackground(Void... params) {
        //Deleting all the locations stored in SQLite database */
        return null;
    }
}
```

d. Complete onCreateLoader() method as follow:

```
public Loader<Cursor> onCreateLoader(int arg0, Bundle arg1) {
    Loader<Cursor> c=null;
    // Uri to the content provider LocationsContentProvider
    // Fetches all the rows from locations table

    return c;
}
```

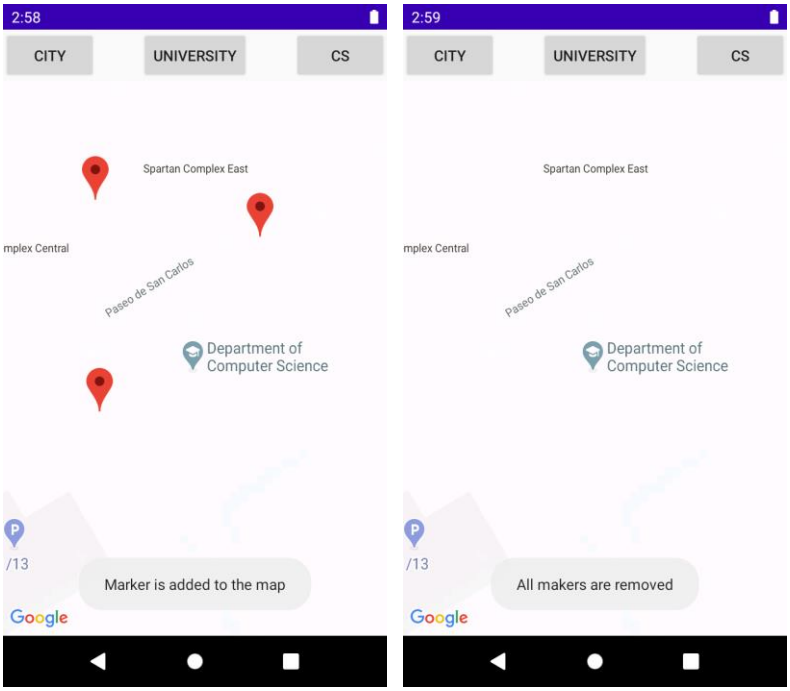
e. Complete onLoadFinished() method as follow:

```
public void onLoadFinished(Loader<Cursor> arg0, Cursor arg1) {
    int locationCount = 0;
    double lat=0;
    double lng=0;
    float zoom=0;
    // Number of locations available in the SQLite database table
        if  (arg1 != null) {
            locationCount = arg1.getCount();
    // Move the current record pointer to the first row of the table
            arg1.moveToFirst();
        }
        else {
            locationCount=0;
        }
     for(int i=0;i<locationCount;i++){
        // Get the latitude
        // Get the longitude
        // Get the zoom level
        // Creating an instance of LatLng to plot the location in Google Maps
        // Drawing the marker in the Google Maps
        // Traverse the pointer to the next row
     }
    if(locationCount>0){
        // Moving CameraPosition to last clicked position
        // Setting the zoom level in the map on last position  is clicked
    }
}
```

6) Add provider to manifest file (authorities field is your packagePath.tableName that you defined in *LocationsDB* (step 1b)

```
<provider
        android:name="LocationsContentProvider"
        android:authorities="edu.sjsu.android.homework4.locations"
        android:exported="false" />
```

Screenshots of the app running:



Submission

1. Push your project directory along with the source to remote bitbucket repository by the due date.
2. Share your project repository the Grader (mohibkhanayubkhan.pathan@sjsu.edu) and Instructor (ramin.moazeni@sjsu.edu).
3. Submit a Readme.pdf to Canvas including your name, repository access link, instructions to run your program (if any), snapshot of your running program
4. Your project directory will be graded according to the state your project directory was in at due time when fetched.