CODING & SOLUTIONING

7.1 FEATURE 1 (ADDING GEOFENCE)

- Geofence is like a round wall covering the given location. So parents can use them to mark the locationwhere their children are going.

```
package com.example.geofence;
import android.app.PendingIntent;
import android.content.Context;
import android.content.ContextWrapper;
import android.content.Intent;import android.widget.Toast;
import com.google.android.gms.common.api.ApiException;
import com.google.android.gms.location.Geofence;
import com.google.android.gms.location.GeofenceStatusCodes;
import com.google.android.gms.location.GeofencingRequest;
import com.google.android.gms.maps.model.LatLng;
public class GeofenceHelper extends ContextWrapper {

        private static final String TAG =
        "GeofenceHelper";PendingIntent pendingIntent;

        public GeofenceHelper(Context base) {
                super(base);
        }
        public GeofencingRequest getGeofencingRequest(Geofence
geofence) {return new GeofencingRequest.Builder()
                                .addGeofence(geofence)

                .setInitialTrigger(GeofencingRequest.INITIAL_TRIGGER_ENTER)
                                .build();
        }
```

```
        public Geofence getGeofence(String ID, LatLng latLng, float
radius,int transitionTypes) {
                        return new Geofence.Builder()
                                .setCircularRegion(latLng.latitude,
                                latLng.longitude,
radius)


            .setRequestId(ID)
            .setTransitionTypes(transitionTypes)
            .setLoiteringDelay(5000)
            .setExpirationDuration(Geofence.NEVER_EXPIRE)
                .build();
}

public PendingIntent getPendingIntent() {
        if (pendingIntent != null)  {
                return pendingIntent;
        }
        Intent intent = new Intent(this,
GeofenceBroadcastReceiver.class);
        pendingIntent = PendingIntent.getBroadcast(this, 2607, intent,
PendingIntent.FLAG_IMMUTABLE);
                        return pendingIntent;
                }

public String getErrorString(Exception e) {
if (e instanceof ApiException) {
        ApiException apiException = (ApiException) e;
        switch (apiException.getStatusCode()) {
                case GeofenceStatusCodes
.                       GEOFENCE_NOT_AVAILABLE:
                return "GEOFENCE_NOT_AVAILABLE";
case GeofenceStatusCodes
```

```
GEOFENCE_NOT_AVAILABLE:
                return "GEOFENCE_NOT_AVAILABLE";
case GeofenceStatusCodes
                        .GEOFENCE_TOO_MANY_GEOFENCES:
                return "GEOFENCE_TOO_MANY_GEOFENCES";
        case GeofenceStatusCodes
                .GEOFENCE_TOO_MANY_PENDING_INTENTS:
        return "GEOFENCE_TOO_MANY_PENDING_INTENTS";}}
```

7.2 FEATURE 2 (ALERT NOTIFICATION)

- Once geofence is added, when the child enters the geofence a notification will be sent
- When the child leaves the geofence a notification will be sent.

```
        package com.example.geofence;
import android.content.BroadcastReceiver;import
android.content.Context;
import android.content.Intent; import android.location.Location;import
android.os.CountDownTimer;import android.util.Log;
import android.widget.Toast;
import  com.google.android.gms.location.Geofence;
import com.google.android.gms.location.GeofencingEvent
import java.util.List;
import android.os.Handler;
public class GeofenceBroadcastReceiver extends
BroadcastReceiver {
        private static final String TAG =
"GeofenceBroadcastReceiv";
        @Override
        public void onReceive(Context context, Intent intent) {
        // TODO: This method is called when the
```

```
        BroadcastReceiver is receiving
        // an Intent broadcast
        //.
        /*Toast.makeText(context, "GEOFENCE_ENTERED",
Toast.LENGTH_SHORT).show();

final Toast mToastToShow;
int toastDurationInMilliSeconds = 1200000;
mToastToShow = Toast.makeText(context, "GEOFENCE_EXITED",
Toast.LENGTH_LONG);
// Set the countdown to display the toast
        CountDownTimer toastCountDown;
        toastCountDown = new
CountDownTimer(toastDurationInMilliSeconds, 100000) {
                public void onTick(long millisUntilFinished) {
                        mToastToShow.show();
                }

                public void onFinish() {
                        mToastToShow.cancel();
                }
        };
// Show the toast and starts the countdown
        mToastToShow.show();
        toastCountDown.start();*/
```

```
  NotificationHelper notificationHelper = new
NotificationHelper(context);
notificationHelper.sendHighPriorityNotification("GEOFENCE_TRAN
SITION_EN TER",
 "", MapsActivity.class);

GeofencingEvent geofencingEvent =
GeofencingEvent.fromIntent(intent);If
(geofencingEvent.hasError())

        Log.d(TAG, "onReceive: Error receiving
                geofence event...");return;
}

        List<Geofence>
geofenceList =
geofencingEvent.getTriggeringGeofenc
es();
        for (Geofence geofence: geofenceList) {
                Log.d(TAG, "onReceive: " + geofence.getRequestId());
        }
//      Location location =
         geofencingEvent.getTriggeringLocation(); int
         transitionType =
         geofencingEvent.getGeofenceTransition();

        switch (transitionType) {
                case Geofence.GEOFENCE_TRANSITION_ENTER:

        notificationHelper.sendHighPriorityNotification(
"Entered theLocation", "", MapsActivity.class);
                break;

        case Geofence.GEOFENCE_TRANSITION_EXIT:
notificationHelper.sendHighPriorityNotification("Exited the Location ",
"",MapsActivity.class);
                break;}  }  }
```