UCS6611 - MOBILE APPLICATION DEVELOPMENT LABORATORY

Ex. No. 6 Develop a native application that uses GPS location information

Date: 30/9/21 **Name:** Srinath S

Class: CSE-C **Roll:** 185001205

Question:

Develop a native application that uses GPS location information

Code:

Activitymain.xml:

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity">

    xmlns:tools="https://schemas.android.com/tools"
        android:id="@+id/activity_main"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:paddingBottom="@dimen/activity_vertical_margin"
        android:paddingLeft="@dimen/activity_horizontal_margin"
        android:paddingRight="@dimen/activity_horizontal_margin"
        android:paddingTop="@dimen/activity_vertical_margin"
        tools:context="com.journaldev.gpslocationtracking.MainActivity">

        <Button
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:id="@+id/btn"
            android:layout_centerInParent="true"
            android:text="GET LOCATION" />

</androidx.constraintlayout.widget.ConstraintLayout>
```

Mainactivity.java:

```
package com.example.geolocation;

import android.annotation.TargetApi;
import android.app.AlertDialog;
import android.content.DialogInterface;
```

```
import android.content.pm.PackageManager;
import android.os.Build;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.Toast;

import androidx.appcompat.app.AppCompatActivity;

import java.util.ArrayList;

import static android.Manifest.permission.ACCESS_COARSE_LOCATION;
import static android.Manifest.permission.ACCESS_FINE_LOCATION;

public class MainActivity extends AppCompatActivity {

    private ArrayList permissionsToRequest;
    private ArrayList permissionsRejected = new ArrayList();
    private ArrayList permissions = new ArrayList();

    private final static int ALL_PERMISSIONS_RESULT = 101;
    LocationTrack locationTrack;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        permissions.add(ACCESS_FINE_LOCATION);
        permissions.add(ACCESS_COARSE_LOCATION);

        permissionsToRequest = findUnAskedPermissions(permissions);
        if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.M) {

            if (permissionsToRequest.size() > 0)
                requestPermissions((String[]) permissionsToRequest.toArray(new
String[permissionsToRequest.size()]), ALL_PERMISSIONS_RESULT);
        }

        Button btn = (Button) findViewById(R.id.btn);

        btn.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {

                locationTrack = new LocationTrack(MainActivity.this);

                if (locationTrack.canGetLocation()) {
```

```
                    double longitude = locationTrack.getLongitude();
                    double latitude = locationTrack.getLatitude();

                    Toast.makeText(getApplicationContext(), "Longitude:" +
Double.toString(longitude) + "\nLatitude:" + Double.toString(latitude),
Toast.LENGTH_SHORT).show();
                } else {

                    locationTrack.showSettingsAlert();
                }

            }
        });

    }

    private ArrayList findUnAskedPermissions(ArrayList wanted) {
        ArrayList result = new ArrayList();

        for (Object perm : wanted) {
            if (!hasPermission((String) perm)) {
                result.add(perm);
            }
        }

        return result;
    }

    private boolean hasPermission(String permission) {
        if (canMakeSmores()) {
            if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.M) {
                return (checkSelfPermission(permission) ==
PackageManager.PERMISSION_GRANTED);
            }
        }
        return true;
    }

    private boolean canMakeSmores() {
        return (Build.VERSION.SDK_INT > Build.VERSION_CODES.LOLLIPOP_MR1);
    }

    @TargetApi(Build.VERSION_CODES.M)
    @Override
    public void onRequestPermissionsResult(int requestCode, String[]
permissions, int[] grantResults) {
```

```
        switch (requestCode) {

            case ALL_PERMISSIONS_RESULT:
                for (Object perms : permissionsToRequest) {
                    if (!hasPermission((String) perms)) {
                        permissionsRejected.add(perms);
                    }
                }

                if (permissionsRejected.size() > 0) {

                    if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.M) {
                        if (shouldShowRequestPermissionRationale((String)
permissionsRejected.get(0))) {
                            showMessageOKCancel("These permissions are
mandatory for the application. Please allow access.",
                                    new DialogInterface.OnClickListener() {
                                        @Override
                                        public void onClick(DialogInterface
dialog, int which) {
                                            if (Build.VERSION.SDK_INT >=
Build.VERSION_CODES.M) {
                                                requestPermissions((String[])
permissionsRejected.toArray(new String[permissionsRejected.size()]),
ALL_PERMISSIONS_RESULT);
                                            }
                                        }
                                    });
                            return;
                        }
                    }

                }

                break;
        }

    }

    private void showMessageOKCancel(String message,
DialogInterface.OnClickListener okListener) {
        new AlertDialog.Builder(MainActivity.this)
                .setMessage(message)
                .setPositiveButton("OK", okListener)
                .setNegativeButton("Cancel", null)
                .create()
                .show();
    }
```

```
    @Override
    protected void onDestroy() {
        super.onDestroy();
        locationTrack.stopListener();
    }
}
```

Location.java:

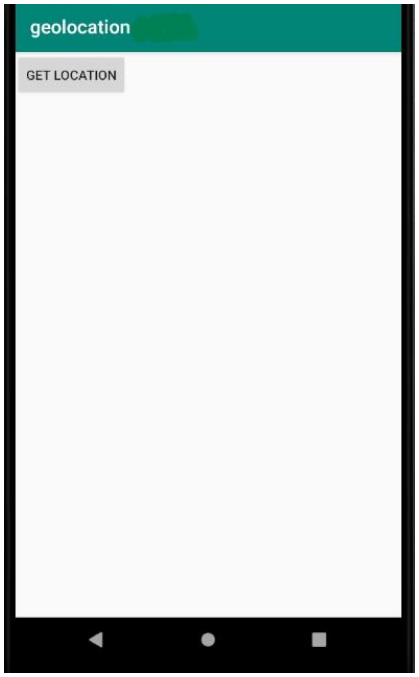
```
package com.example.geolocation;

import android.Manifest;
import android.app.AlertDialog;
import android.app.Service;
import android.content.Context;
import android.content.DialogInterface;
import android.content.Intent;
import android.content.pm.PackageManager;
import android.location.Location;
import android.location.LocationListener;
import android.location.LocationManager;
import android.os.Bundle;
import android.os.IBinder;
import android.provider.Settings;
import android.widget.Toast;

import androidx.core.app.ActivityCompat;

public class LocationTrack extends Service implements LocationListener {

    private final Context mContext;

    boolean checkGPS = false;

    boolean checkNetwork = false;

    boolean canGetLocation = false;

    Location loc;
    double latitude;
    double longitude;

    private static final long MIN_DISTANCE_CHANGE_FOR_UPDATES = 10;

    private static final long MIN_TIME_BW_UPDATES = 1000 * 60 * 1;
    protected LocationManager locationManager;
```

```
    public LocationTrack(Context mContext) {
        this.mContext = mContext;
        getLocation();
    }

    private Location getLocation() {

        try {
            locationManager = (LocationManager) mContext
                    .getSystemService(LOCATION_SERVICE);

            checkGPS = locationManager
                    .isProviderEnabled(LocationManager.GPS_PROVIDER);

            checkNetwork = locationManager
                    .isProviderEnabled(LocationManager.NETWORK_PROVIDER);

            if (!checkGPS && !checkNetwork) {
                Toast.makeText(mContext, "No Service Provider is available",
Toast.LENGTH_SHORT).show();
            } else {
                this.canGetLocation = true;
                if (checkGPS) {

                    if (ActivityCompat.checkSelfPermission(mContext,
Manifest.permission.ACCESS_FINE_LOCATION) != PackageManager.PERMISSION_GRANTED
&& ActivityCompat.checkSelfPermission(mContext,
Manifest.permission.ACCESS_COARSE_LOCATION) !=
PackageManager.PERMISSION_GRANTED) {

                    }
                    locationManager.requestLocationUpdates(
                            LocationManager.GPS_PROVIDER,
                            MIN_TIME_BW_UPDATES,
                            MIN_DISTANCE_CHANGE_FOR_UPDATES, this);
                    if (locationManager != null) {
                        loc = locationManager
                                .getLastKnownLocation(LocationManager.GPS_PROV
IDER);
                        if (loc != null) {
                            latitude = loc.getLatitude();
                            longitude = loc.getLongitude();
                        }
                    }

                }
            }
```

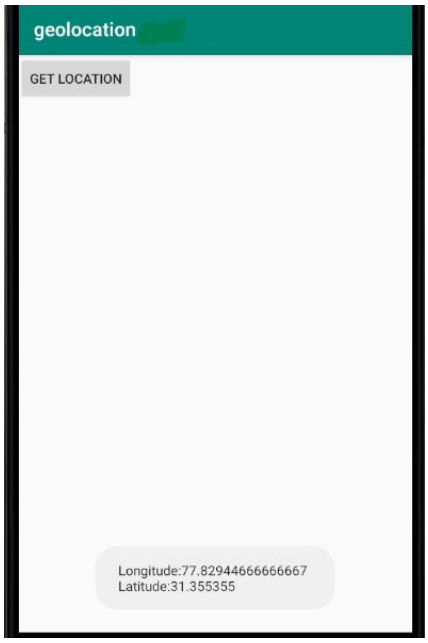
```
        } catch (Exception e) {
            e.printStackTrace();
        }

        return loc;
    }

    public double getLongitude() {
        if (loc != null) {
            longitude = loc.getLongitude();
        }
        return longitude;
    }

    public double getLatitude() {
        if (loc != null) {
            latitude = loc.getLatitude();
        }
        return latitude;
    }

    public boolean canGetLocation() {
        return this.canGetLocation;
    }

    public void showSettingsAlert() {
        AlertDialog.Builder alertDialog = new AlertDialog.Builder(mContext);

        alertDialog.setTitle("GPS is not Enabled!");

        alertDialog.setMessage("Do you want to turn on GPS?");

        alertDialog.setPositiveButton("Yes", new
DialogInterface.OnClickListener() {
            public void onClick(DialogInterface dialog, int which) {
                Intent intent = new
Intent(Settings.ACTION_LOCATION_SOURCE_SETTINGS);
                mContext.startActivity(intent);
            }
        });

        alertDialog.setNegativeButton("No", new
DialogInterface.OnClickListener() {
            public void onClick(DialogInterface dialog, int which) {
                dialog.cancel();
            }
        });
```

```
        alertDialog.show();
    }

    public void stopListener() {
        if (locationManager != null) {

            if (ActivityCompat.checkSelfPermission(mContext,
Manifest.permission.ACCESS_FINE_LOCATION) != PackageManager.PERMISSION_GRANTED
&& ActivityCompat.checkSelfPermission(mContext,
Manifest.permission.ACCESS_COARSE_LOCATION) !=
PackageManager.PERMISSION_GRANTED) {
                return;
            }
            locationManager.removeUpdates(LocationTrack.this);
        }
    }

    @Override
    public IBinder onBind(Intent intent) {
        return null;
    }

    @Override
    public void onLocationChanged(Location location) {

    }

    @Override
    public void onStatusChanged(String s, int i, Bundle bundle) {

    }

    @Override
    public void onProviderEnabled(String s) {

    }

    @Override
    public void onProviderDisabled(String s) {

    }
}
```

Output:


geolocation
GET LOCATION


Search
Kyrgyzstan
Tajikistan
Pakistan
New Delhi
नई दिल्ली
Nepal
India
Mumbai
मुंबई
Hyderabad
Chennai
Bengaluru


geolocation
GET LOCATION
Longitude:77.82944666666667
Latitude:31.355355