Example 7:

Develop a native application that uses GPS Location Information.

GPS Location Information:

One of the major features of android framework is location API. You can see, the location module widely used in lot of apps those provides services like food ordering, transportation, health tracking, social networking and lot more. The location module is part of Google Play Services and in the same module geofencing and activity recognition are included.

Earlier, getting location is very easy with couple of API calls. But to provide more accurate locations and optimizing the battery usage, Android introduced set APIs that should be combined to get the best results from the location API. We will be using **Fused Location API** that combines signals from **GPS**, **Wi-Fi**, and **cell networks**, as well as **accelerometer**, **gyroscope**, **magnetometer** and other sensors to provide more accurate results.

Before moving forward, we need to understand the location permissions.

Location Permissions:

If you want to get the current location of your user, then you have to add the location permission to your application. Android provides the following location permission:

- **ACCESS_COARSE_LOCATION:** By adding this in your application, allows you to use WIFI or mobile cell data(or both) to determine the device's location. The approximation by using this permission is close to the city level.

- **ACCESS_FINE_LOCATION:** This uses your GPS as well as WIFI and mobile data to get the most precise location as possible. By using this, you will get a precise location of your user.

- **ACCESS_BACKGROUND_LOCATION:** This permission was introduced in Android 10. So, for Android 10 or higher, if you want to access the user's location in the background, then along with any of the above two permissions, you need to add the ACCESS_BACKGROUND_LOCATION permission also.

Fused Location API:

In this example, we will use **Fused Location API** to get the changed location or you may say, get the current location of a user. We will be using **LocationRequest** that are used to request quality of service for location updates from the **FusedLocationProviderApi**.

Apart from getting the updated location, the **LocationRequest** includes various methods for retrieving locations like a pro. Some of the methods are:

- **setInterval(long millis):** This is used to set the desired interval after which you want to check for a location update. It is in milliseconds.
- **setFastestInterval(long millis):** This is used to set the fastest interval for a location update. This might be faster than your `setInterval(long)` in many cases because if other applications on the device are triggering location updates at an interval lesser than your interval then it will use that update.
- **setSmallestDisplacement(float smallestDisplacementMeters):** This will set the minimum displacement between location updates i.e. the smallest displacement required for a location update. It is in meters and the default value of this is 0.
- **setPriority(int priority):** This is used to set the priority of location received. It can be **PRIORITY_BALANCED_POWER_ACCURACY** (for accuracy up to **block** level) or it can be **PRIORITY_HIGH_ACCURACY**(to get the most accurate result) or it can be **PRIORITY_LOW_POWER**(to get the accuracy up to **city** level) or it can be **PRIORITY_NO_POWER**(to get the most accurate information without providing some additional power).