دانشکده مهندسی کامپیوتر

گزارش پروژه

درس شبکه های تلفن همراه

اعضای گروه: فرگل فریدونی و نیایش خانی

نیم‌سال دوم

سال تحصیلی۱۴۰۳-۱۴۰۲

۱.۰ توضیحات پروژه

پس از نصب اندروید استودیو و sdk های لازم به سراغ ساخت پروژه می رویم و یک تمپلت را انتخاب می کنیم.
ابتدا به بررسی فایل MainActivity.kt می پردازیم.

MainActivity.kt :

```kotlin
class MainActivity : AppCompatActivity(), OnMapReadyCallback {  niayesh-khani

    private lateinit var dbHelper: DatabaseHelper
    private lateinit var googleMap: GoogleMap
    private lateinit var fusedLocationClient: FusedLocationProviderClient
    private lateinit var cellInfoTextView: TextView
    private lateinit var locationTextView: TextView
    private val LOCATION_PERMISSION_REQUEST_CODE = 1
    val cellInfoText = StringBuilder()

    private val requestPermissionLauncher = registerForActivityResult(
        ActivityResultContracts.RequestMultiplePermissions()
    ) { permissions ->
        if (permissions[Manifest.permission.READ_PHONE_STATE] == true && permissions[Manifest.permission.ACCESS_FINE_LOCATION] == tr
            getCellInfo()
            getCurrentLocation()
        } else {
            Log.d( tag: "MainActivity",  msg: "Permissions denied")
        }
    }
```

شکل ۱: MainActivity.kt

ابتدا کلاس اصلی برنامه را تعریف می‌کنیم که از AppCompatActivity ارث‌بری می‌کند و OnMapReadyCallback را پیاده‌سازی می‌کند.سپس متغیرها و ثابت‌های مختلف برای استفاده در کلاس تعریف می‌شوند که شامل دیتابیس استفاده شده، موارد لازم برای مپ مانند گوگل مپ و ... است.

سپس از requestPermissionLauncher برای درخواست مجوزهای لازم (خواندن وضعیت تلفن و دسترسی به موقعیت مکانی) استفاده می‌شود و در صورت دریافت مجوزها، اطلاعات موقعیت و شبکه را جمع‌آوری می‌کند.

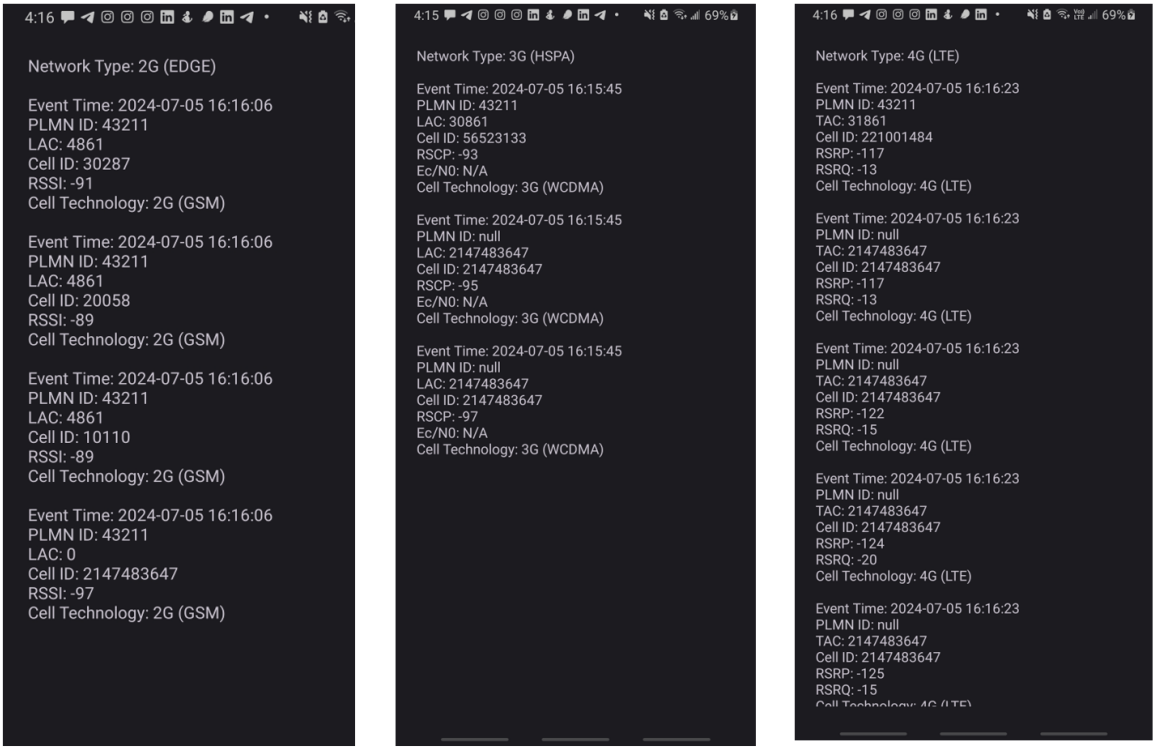
بعد در متد onCreate واسط کاربری تنظیم می‌شود، متغیرها مقداردهی اولیه می‌شوند و نقشه گوگل بارگذاری می‌شود. همچنین چک می‌کند که آیا مجوزهای لازم موجود هستند یا نه.

زمانی که نقشه آماده است، onMapReady فراخوانی می‌شود و تنظیمات اولیه نقشه انجام می‌شود. سپس موقعیت مکانی کنونی دستگاه را دریافت کرده و روی نقشه نمایش می‌دهیم. ضمن اینکه اطلاعات موقعیت را در دیتابیس ذخیره می کنیم.

پس از اینکه موقعیت مکانی کاربر را دریافت کردیم، آن را در دیتابیس ذخیر می کنیم.

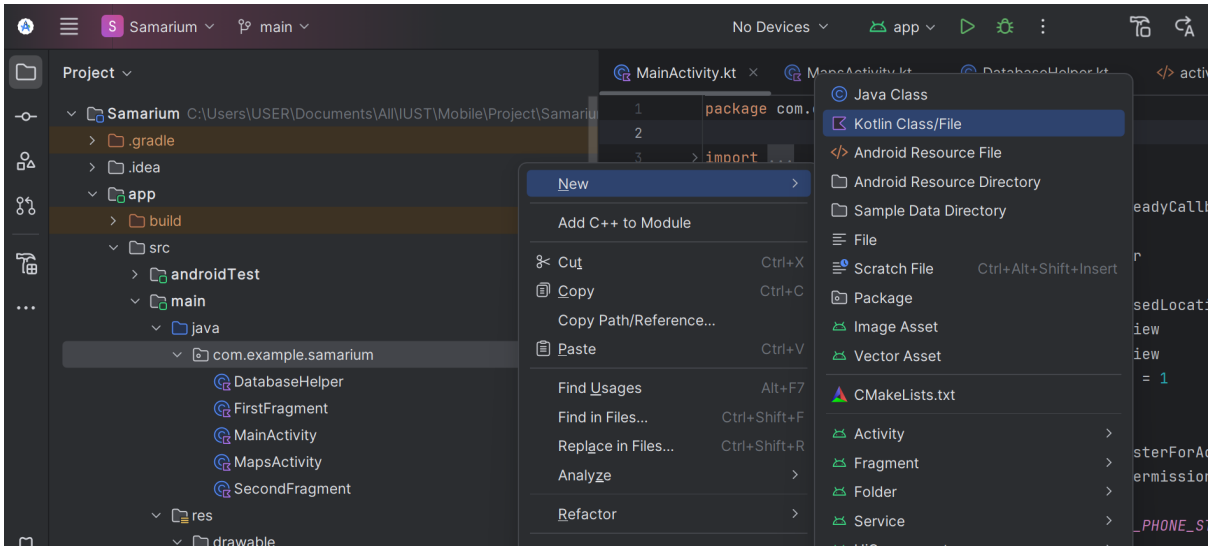
سپس در متد getCellInfo شروع به دریافت اطلاعات می کنیم. این اطلاعات برای هر کدام از نسل ها به صورت جداگانه دریافت می شوند. در اینجا با استفاده از getSystemService متغیر telephonyManager را تعریف می کنیم. با استفاده از متد getNetwork-Type(telephonyManager.networkType) تایپ اینترنت کاربر را دریافت می کنیم. حال با استفاده از متد های آماده برای دریافت پارامترهای مورد نظراز متد های cellIdentity برای دریافت rsrp ca، tac و ...استفاده می کنیم. سپس پارامترهای دریافت شده را در ستون های خود در دیتابیس ذخیره کرده و به cellInfoText می دهیم تا بر روی صفحه نمایش دهیم.

این کار را برای هر سه نسل انجام می دهیم. و در آخر این اطلاعات را در TextView نمایش می دهیم.



شکل ۲: نمونه دیتاهای دریافت شده با تکنولوژی های متفاوت

برای ایجاد دیتابیس باید طبق عکس زیر یک فایل ایجاد کنیم.



شکل ۳: درست کردن فایل DatabaseHelper

در فایل DatabaaseHelper از دیتابیس SQLiteDatabase استفاده می کنیم و با ساختن جدول و ستون های مورد نظر برای هر پارامتر پایگاه داده خود را ایجاد می کنیم.

در کنار این فایل ها، FirstFragment و SecondFragment نیز وجود دارد که به دلیل داشتن تنها یک صفحه، مورد استفاده و تغییر قرار نگرفته اند. در مقایسه SecondFragment با FirstFragment میتوان گفت، هر دو Fragment عملکرد مشابهی دارند و از روش‌های یکسانی برای ایجاد و مدیریت نمایی (UI) استفاده می‌کنند. تفاوت اصلی آنها در destination و شناسه‌های منابع استفاده شده در Listener های button ها است. این تفاوت‌ها مشخص می‌کنند که کدام Fragment به کدام مقصد خواهد رفت. (هر دو از FragmentBinding (یعنی FragmentFirstBinding و FragmentSecondBinding برای اتصال (UI) استفاده می‌کنند.هر دو به این صورت تعریف شده‌اند که binding فقط بین onCreateView و onDestroyView معتبر خواهد بود.)

MapActivity.kt :

```kotlin
class MapsActivity : AppCompatActivity(), OnMapReadyCallback {  niayesh-khani

    private lateinit var mMap: GoogleMap
    private lateinit var binding: ActivityMapsBinding
    private lateinit var fusedLocationClient: FusedLocationProviderClient
    private val LOCATION_PERMISSION_REQUEST_CODE = 1

    override fun onCreate(savedInstanceState: Bundle?) {  niayesh-khani
        super.onCreate(savedInstanceState)

        binding = ActivityMapsBinding.inflate(layoutInflater)
        setContentView(binding.root)

        // Obtain the SupportMapFragment and get notified when the map is ready to be used.
        val mapFragment = supportFragmentManager
            .findFragmentById(R.id.map) as SupportMapFragment
        mapFragment.getMapAsync( callback: this)

        // Initialize the FusedLocationProviderClient
        fusedLocationClient = LocationServices.getFusedLocationProviderClient( activity: this)
    }
```

شکل ۴: MapActivity.kt

با انتخاب تمپلت گوگل مپ و نصب dependency های لازم، این فایل را ایجاد می کنیم.

پس از تعریف کردن متغیر های لازم، در متد ،onCreate واسط کاربری تنظیم می‌شود، متغیرها مقداردهی اولیه می‌شوند و نقشه گوگل بارگذاری می‌شود. mapFragment.getMapAsync(this) تنظیم می‌کند که وقتی نقشه آماده شد، onMapReady فراخوانی شود. همچنین، FusedLocationProviderClient برای دریافت موقعیت مکانی کاربر مقداردهی اولیه می‌شود.

در متد onMapReady زمانی که نقشه آماده است، فراخوانی می‌شود و تنظیمات اولیه نقشه انجام می‌شود. همچنین چک می‌کند که آیا مجوز دسترسی به موقعیت مکانی موجود است یا نه. اگر مجوز موجود باشد، موقعیت مکانی کاربر روی نقشه نمایش داده می‌شود. در غیر این صورت، درخواست مجوز دسترسی به موقعیت مکانی ارسال می‌شود.

سپس در متد getCurrentLocation موقعیت مکانی کنونی دستگاه را دریافت کرده و روی نقشه نمایش می‌دهد. ابتدا چک می‌کند که آیا مجوز دسترسی به موقعیت مکانی وجود دارد یا نه. سپس، اگر مجوز وجود داشته باشد، آخرین موقعیت مکانی کاربر را از FusedLocation-ProviderClient دریافت می‌کند و یک مارکر روی نقشه قرار می‌دهد و نقشه را به موقعیت کاربر منتقل می‌کند.

در آخر در متد onRequestPermissionsResult نتایج درخواست مجوز را بررسی کرده و در صورت دریافت مجوز، موقعیت مکانی کنونی کاربر را دریافت و روی نقشه نمایش می‌دهد. اگر مجوز داده نشود، برنامه باید به صورت مناسب این مسئله را مدیریت کند.

در فایل AndroidManifest.xml دسترسی های لازم و فعالیت ها و ... را قرار می دهیم.

```xml
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    package="com.example.samarium">

    <uses-permission android:name="android.permission.ACCESS_FINE_LOCATION" />
    <uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION" />
    <uses-permission android:name="android.permission.READ_PHONE_STATE"/>
    <uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION" />
    <uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />
    <uses-permission android:name="android.permission.INTERNET" />
    <uses-permission android:name="android.permission.ACCESS_WIFI_STATE" />
    <application
        android:allowBackup="true"
        android:dataExtractionRules="@xml/data_extraction_rules"
        android:fullBackupContent="@xml/backup_rules"
        android:icon="@mipmap/ic_launcher"
        android:label="Samarium"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportsRtl="true"
        android:theme="@style/Theme.Samarium"
        tools:targetApi="31">
```

شکل ۵: AndroidManifest.xml

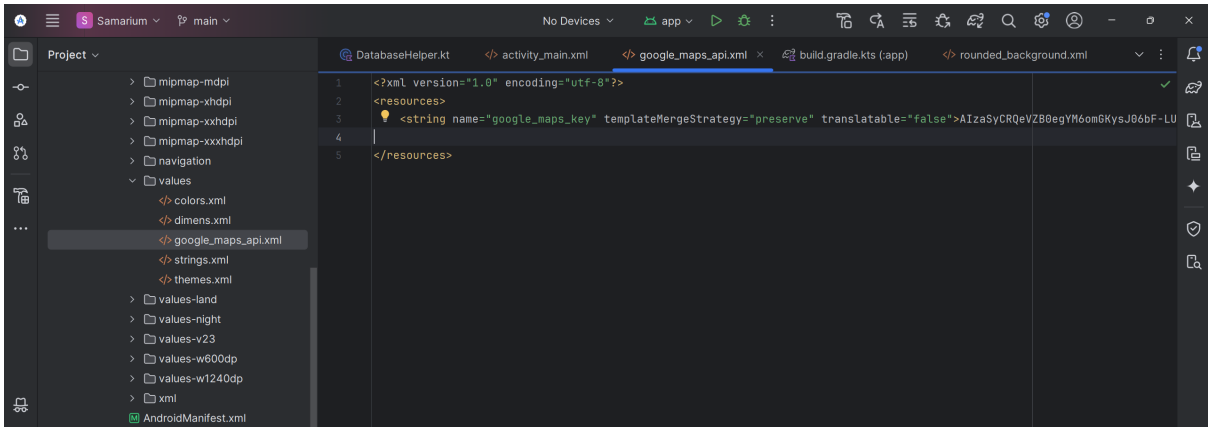
ابتدا دسترسی‌های مورد نیاز برنامه را تعریف می کنیم:

ACCESS-FINE-LOCATION و ACCESS-COARSE-LOCATION برای دسترسی به موقعیت جغرافیایی دقیق و تقریبی.
READ-PHONE-STATE برای دسترسی به اطلاعات وضعیت تلفن.
ACCESS-NETWORK-STATE و ACCESS-WIFI-STATE برای دسترسی به وضعیت شبکه و وای‌فای.
INTERNET برای دسترسی به اینترنت.

سپس تنظیمات کلی برنامه را قرار می دهیم. همچنین کلید API گوگل مپ را تعریف می کنیم که برای استفاده از نقشه‌های گوگل در برنامه ضروری است.

دریافت Key API نقشه‌های گوگل: درکنسول Google Cloud Platform یک پروژه جدید ایجاد کرده‌ایم. در بخش "APIs Services" نیز API Maps SDK for Android و برخی موارد ضروری دیگر را فعال می کنیم.یک Key API ایجاد و کپی خواهیم کرد.

افزودن Key API به فایل google-maps-api: افزودن Key API به فایل google-maps-api: در مسیر res/values یک فایل به نام google-maps-api.xml ایجاد کردیم و API Key خود را درون آن قرار خواهیم داد:

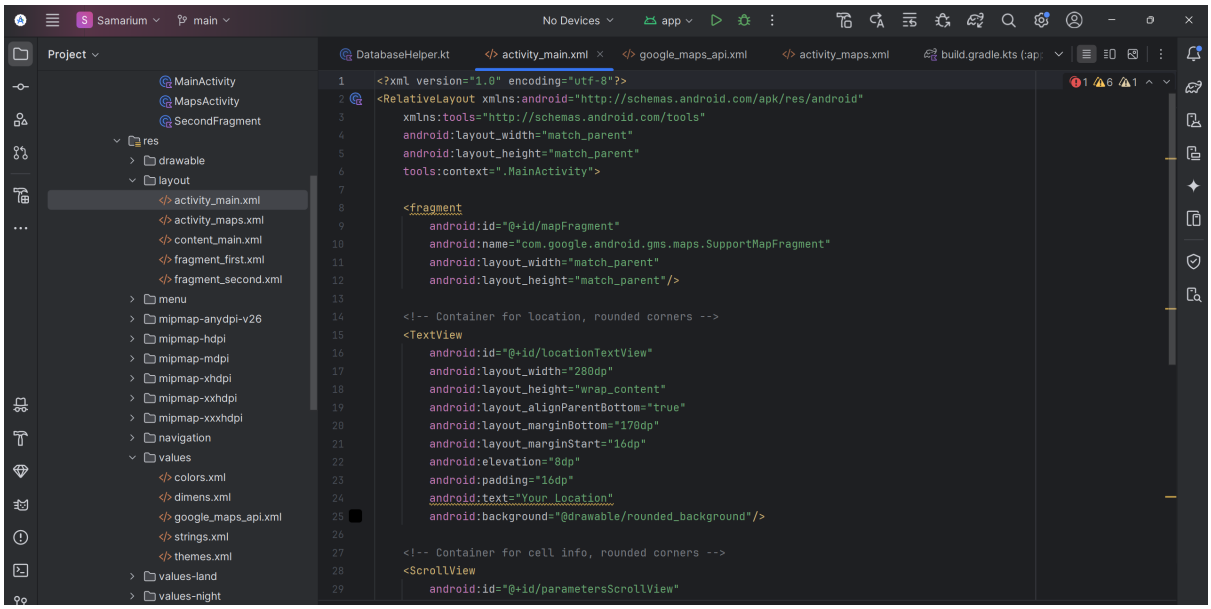


شکل ۶: google-maps-api.xml

در آخر نیز فعالیت ها را مشخص می کنیم.

لازم به ذکر است که تمامی dependency های لازم، plugin ها و ... در فایل biuld.gradles.kts قرار می گیرند.

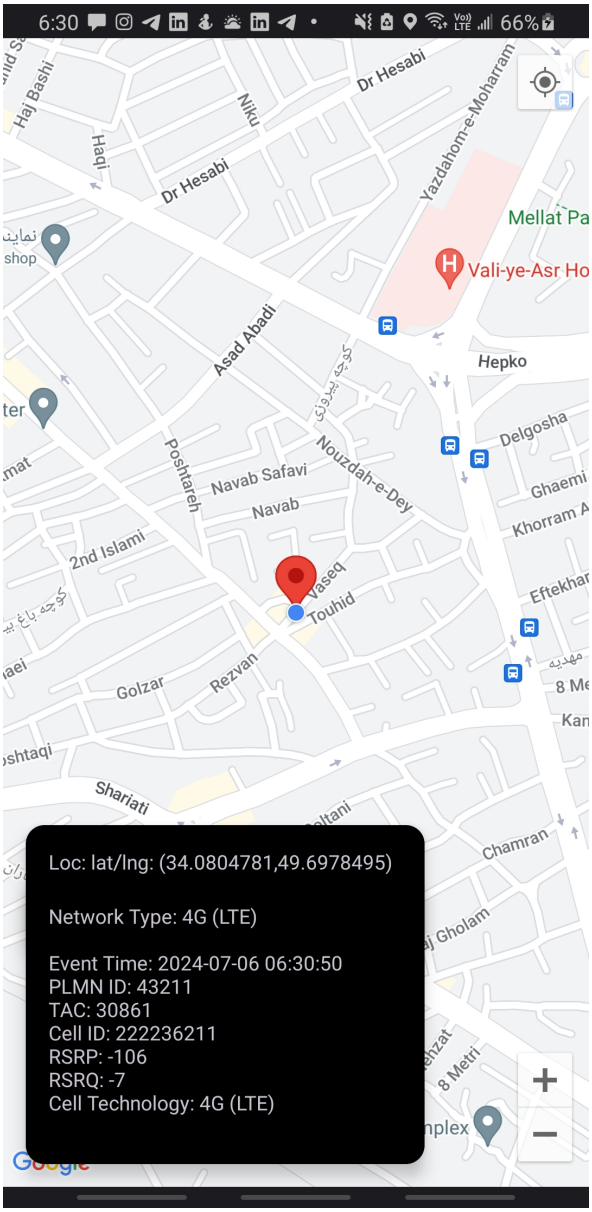
کارهای مربوط به دیزاین و نمایش دادن موارد مربوطه در فایل activity-main.xml صورت گرفته است.



شکل ۷: activity-main.xml

همانطور که مشاهده می شود در این صفحه از TextView، ScrollView برای باکس پارامترها، فرگمنت مپ و ... تعریف شده است.

در نهایت خروجی به این صورت خواهد بود :



شکل ۸: خروجی مورد نظر

همانطور که مشاهده می شود، مکان کاربر با مارکر نمایش داده شده است.

همچنین در باکس، موقعیت مکانی کاربر، زمان ثبت رخداد، فناوری سلولی که گوشی بر روی آن اردو زده است از جمله LTE و ...، شناسه های مکانی سلول و کمیت و کیفیت سیگنال قابل مشاهده است. با اسکرول کردن بر روی باکس می توان اطلاعات دیگر سلول ها را نیز مشاهده کرد.