گزارش پروژه پایانی درس شبکه‌های موبایل

علیرضا اسلامی‌خواه-۹۹۵۲۱۰۶۴
امیرعلی فرازمند-۹۹۵۲۲۳۲۹

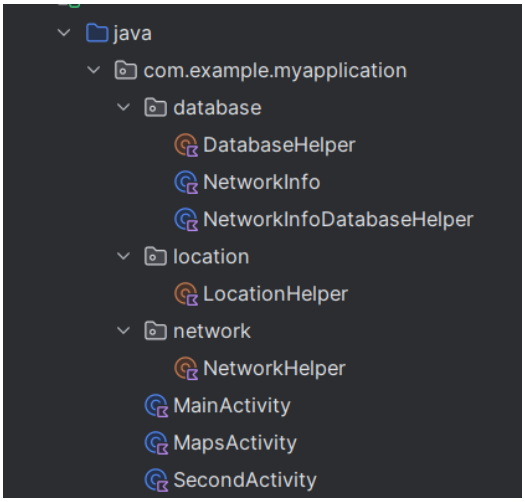
۱۵ تیر ۱۴۰۳

معرفی پروژه

پروژه ساماریم در واقع یک پروژه در بستر فریمورک زبان کاتلین و با استفاده از ابزارهایی مانند اندرویداستودیو و شبیه‌سازهای دیگر پیاده‌سازی شده است.

ساختار پروژه

پروژه ما از یک ساختار چندکلاسه تبعیت میکند. بدین صورت که در شکل هم مشخص است فایل های مربوط به هندل کردن دیتابیس درون پوشه database، فایل های هندل کردن مکان و دسترسی های آن در پوشه location، فایل های دسترسی های شبکه و تمام مشتقات آن درون network و در آخر هم در ریشه پکیج دو فایل MapsActivity و SecondActivity قرار دارند که وظیفه مدیریت کردن این فایل ها را بر عهده دارند.



البته که کنار اینها فایل های XML درون پوشه layout وجود دارند که وظیفه User Interface پروژه را عهده دار هستند که مختصرا به آنها میپردازیم.

MainActivity

در ابتدا از اصلی ترین بخش پروژه یعنی فایل MainActivity شروع به توضیح دادن میکنیم. این فایل وظیفه مدیریت صفحه اول و اصلی پروژه و همچنین مدیریت کردن ماژول های دیگر را عهده دار است. در پروژه ما از یک handler استفاده کرده ایم که وظیفه این را دارد که کدها را به طور متناوب و دلخواه روی صفحه به نمایش در بیاورد. همچنین این ماژول توانایی هماهنگی permission ها کاربر از طرف دستگاه خود را دارا است و در صورت fail شدن به کاربر اطلاع میدهد.

```xml
<uses-permission android:name="android.permission.ACCESS_FINE_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" />
```

LocationHelper

فایل دوم LocationHelper میباشد که وظیفه اعلان زمان رخ داد و مکان کاربر از لحاظ طولی و عرضی جغرافیایی را عهده دارد.

کتابخانه FusedLocationProviderClient از Google Play Services برای دسترسی به مکان کاربر استفاده می کند. این کتابخانه به دلیل دقت بالا، مصرف کم باتری و توانایی استفاده از منابع مختلف مانند GPS، Wi-Fi و شبکه‌های موبایل، انتخاب شده است.

هدف اصلی این کلاس این است که مکان کاربر را به‌طور دقیق از طریق GPS به‌دست آورد. با این حال، زمانی که GPS در دسترس نباشد، روش‌های جایگزینی برای گرفتن مکان کاربر استفاده می‌شود. در ادامه به توضیح بخش‌های مختلف کد و علت استفاده از هر بخش می‌پردازیم.

در ابتدا، تلاش می‌شود تا مکان کاربر از طریق GPS و با استفاده از کلاس FusedLocationProviderClient به دست آید.

```
Alireza Eslamikhah
@RequiresApi(Build.VERSION_CODES.P)
fun getLastLocation(activity: MainActivity, locationText: TextView, eventTimeText: TextView) {
    fusedLocationClient = getFusedLocationProviderClient(activity)

    if (ActivityCompat.checkSelfPermission(activity, Manifest.permission.ACCESS_FINE_LOCATION) !=
        ActivityCompat.checkSelfPermission(activity, Manifest.permission.ACCESS_COARSE_LOCATION)
        return
    }

    fusedLocationClient.lastLocation
        .addOnSuccessListener { location: Location? ->
            if (location != null) {
                updateLocationUI(location, locationText, eventTimeText)
            } else {
                // Handle the case where the location is null
                requestLocationUpdates(activity, locationText, eventTimeText)
            }
        }
        .addOnFailureListener { exception ->
            // Handle failure in getting location
            requestLocationUpdates(activity, locationText, eventTimeText)
        }
}
```

در این بخش، ابتدا تلاش می‌شود تا مکان آخر کاربر با استفاده از fusedLocationClient.lastLocation به دست آید. اگر مکان موفقیت‌آمیز بود، به روزرسانی UI با استفاده از متد updateLocationUI انجام می‌شود. اگر مکان ناموفق بود یا به‌دست نیامد، روش جایگزین فراخوانی می‌شود.

```
Alireza Eslamikhah
private fun requestLocationUpdates(activity: MainActivity, locationText: TextView, eventTimeText: TextView) {
    val locationRequest = LocationRequest.create().apply { this: LocationRequest
        interval = 10000
        fastestInterval = 5000
        priority = LocationRequest.PRIORITY_HIGH_ACCURACY
    }

    val locationCallback = object : LocationCallback() {
        override fun onLocationResult(locationResult: LocationResult) {
            locationResult.locations.forEach { location ->
                updateLocationUI(location, locationText, eventTimeText)
            }
            fusedLocationClient.removeLocationUpdates(this)
        }
    }
```

در این بخش، یک درخواست مکان جدید ایجاد می‌شود که هر ۱۰ ثانیه مکان را به‌روز می‌کند و در هر ۵ ثانیه مکان دقیق‌تر را به‌دست می‌آورد.
این درخواست مکان از longtitudeLocationRequest.PRIORITY$_{H}IGH_{A}CCURACY$. و latitude در صفحه اول چاپ میشوند.

NetworkHelper

در این فایل وظیفه محاسبه و نمایش یکسری پارامترها را عهده دار هستیم. این فایل شامل یک کلاس از NetworkHelper است که برای نمایش اطلاعات شبکه سلولی و سیگنال کاربر در یک برنامه اندروید استفاده می‌شود. این اطلاعات شامل فناوری سلولی، شناسه‌های مکانی سلول و کمیت و کیفیت سیگنال است.
فایل DatabaseHelper نقش مهمی در مدیریت اطلاعات شبکه و سیگنال در برنامه ایفا می‌کند. با استفاده از این شیء، اطلاعات مهمی مانند قدرت سیگنال، شناسه‌های مکانی سلول و فناوری شبکه سلولی به‌دست آمده و در پایگاه داده ذخیره می‌شود. در ادامه به توضیح بخش‌های مختلف کد و نحوه استفاده از اندروید TelephonyManager می‌پردازیم.
کتابخانه android.telephony در اندروید برای دسترسی به اطلاعات شبکه و سیگنال سلولی استفاده می‌شود. این کتابخانه امکان دسترسی به جزئیات شبکه از جمله نوع شبکه، قدرت سیگنال و شناسه‌های مکانی را فراهم می‌کند. کلاس‌هایی مانند TelephonyManager, CellInfo, CellIdentityLte, CellInfoLte و غیره برای استخراج و مدیریت این اطلاعات به کار می‌روند.
در ابتدا، با استفاده از TelephonyManager نوع شبکه سلولی که دستگاه به آن متصل است، شناسایی می‌شود. در این بخش، با استفاده از telephonyManager.networkType نوع شبکه شناسایی شده و به کاربر نمایش داده می‌شود.

```
<uses-permission android:name="android.permission.ACCESS_FINE_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" />
```

در ادامه، اطلاعات شبکه سلولی شامل شناسه‌های مکانی سلول و کیفیت سیگنال استخراج می‌شود و در نهایت، اطلاعات به‌دست آمده در پایگاه داده ذخیره می‌شود.

DatabaseHelper

فایل DatabaseHelper شامل یک کلاس برای مدیریت ذخیره‌سازی اطلاعات شبکه و سیگنال در پایگاه داده برنامه است. این کلاس اطلاعات مختلفی از جمله قدرت سیگنال، شناسه‌های مکانی سلول و فناوری شبکه سلولی را در پایگاه داده ذخیره می کند. در ادامه، بخش‌های مختلف این فایل توضیح داده می‌شود.

متد insertNetworkInfoToDatabase اطلاعات شبکه و سیگنال را دریافت و در پایگاه داده ذخیره می کند. پارامترهای این متد شامل قدرت سیگنال، شناسه‌های مکانی سلول و جزئیات فناوری شبکه است.

متد calculateSituation براساس قدرت سیگنال، وضعیت سیگنال را در یکی از دسته‌های ”Excellent” - ”Good” - ”Fair” - ”Poor” - ”Very Poor” قرار می‌دهد.

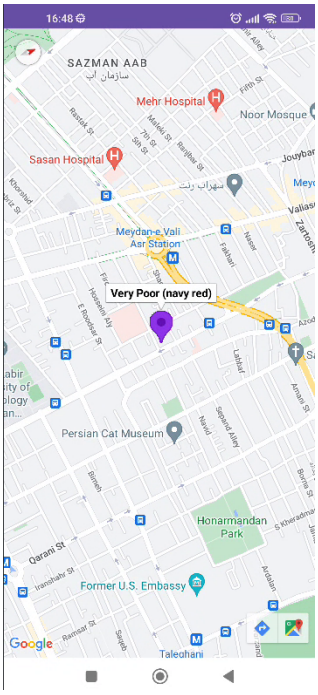
```
Alireza Eslamikhah
private fun calculateSituation(signalStrength: Int?): String {
    return when (signalStrength) {
        null -> "Unknown"
        10000 -> "Unknown"
        in -85 ≤ .. ≤ Int.MAX_VALUE -> "Excellent"
        in -95 ≤ .. ≤ -86 -> "Good"
        in -105 ≤ .. ≤ -96 -> "Fair"
        in -115 ≤ .. ≤ -106 -> "Poor"
        else -> "Very Poor"
    }
}
```

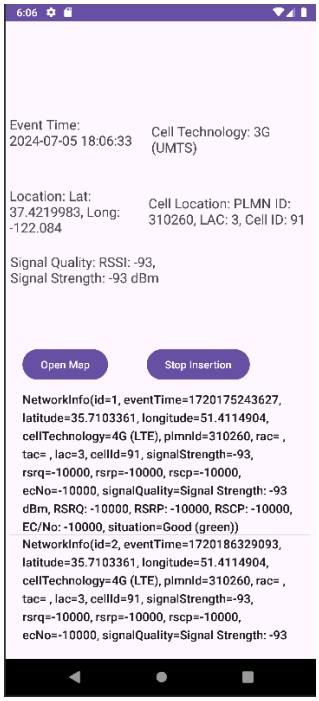
همینطور ساختار داده ما در دیتابیس که با sqlite تبیین شده بدین شکل مشخص است :

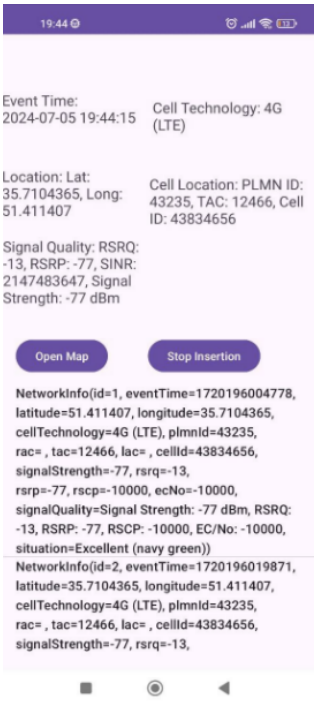
```kotlin
package com.example.myapplication.database

import androidx.room.PrimaryKey
 Alireza Eslamikhah
data class NetworkInfo(
    @PrimaryKey(autoGenerate = true) val id: Int = 0,
    val eventTime: Long,
    val latitude: Double,
    val longitude: Double,
    val cellTechnology: String,
    val plmnId: String?,
    val rac: String?,
    val tac: String?,
    val lac: String?,
    val cellId: String?,
    val signalStrength: Int?,
    val rsrq: Int?,
    val rsrp: Int?,
    val rscp: Int?,
    val ecNo: Int?,
    val signalQuality: String,
    val situation: String
)
```

خروجی

در بخش آخر هم تصاویری از خروجی این اپلیکشین میبینیم که در آنها جزئیاتی همچون زمان رخداد و تکنولوژی سلول و لوکیشن و CELL-ID , LAC , TAC , RAC , PLMN-ID و آخر هم کیفیت و قدرت سیگنال تعریف شده اند.


16:48
SAZMAN AAB
Mehr Hospital
Noor Mosque
Sasan Hospital
Meydan-e Vali Asr Station
Very Poor (navy red)
Persian Cat Museum
Honarmandan Park
Former U.S. Embassy
Taleghani
Google


6:06
Event Time: 2024-07-05 18:06:33
Cell Technology: 3G (UMTS)
Location: Lat: 37.4219983, Long: -122.084
Cell Location: PLMN ID: 310260, LAC: 3, Cell ID: 91
Signal Quality: RSSI: -93, Signal Strength: -93 dBm
Open Map
Stop Insertion
NetworkInfo(id=1, eventTime=1720175243627, latitude=35.7103361, longitude=51.4114904, cellTechnology=4G (LTE), plmnId=310260, rac= , tac= , lac=3, cellId=91, signalStrength=-93, rsrq=-10000, rsrp=-10000, rscp=-10000, ecNo=-10000, signalQuality=Signal Strength: -93 dBm, RSRQ: -10000, RSRP: -10000, RSCP: -10000, EC/No: -10000, situation=Good (green))
NetworkInfo(id=2, eventTime=1720186329093, latitude=35.7103361, longitude=51.4114904, cellTechnology=4G (LTE), plmnId=310260, rac= , tac= , lac=3, cellId=91, signalStrength=-93, rsrq=-10000, rsrp=-10000, rscp=-10000, ecNo=-10000, signalQuality=Signal Strength: -93


19:44
Event Time: 2024-07-05 19:44:15
Cell Technology: 4G (LTE)
Location: Lat: 35.7104365, Long: 51.411407
Cell Location: PLMN ID: 43235, TAC: 12466, Cell ID: 43834656
Signal Quality: RSRQ: -13, RSRP: -77, SINR: 2147483647, Signal Strength: -77 dBm
Open Map
Stop Insertion
NetworkInfo(id=1, eventTime=1720196004778, latitude=51.411407, longitude=35.7104365, cellTechnology=4G (LTE), plmnId=43235, rac= , tac=12466, lac= , cellId=43834656, signalStrength=-77, rsrq=-13, rsrp=-77, rscp=-10000, ecNo=-10000, signalQuality=Signal Strength: -77 dBm, RSRQ: -13, RSRP: -77, RSCP: -10000, EC/No: -10000, situation=Excellent (navy green))
NetworkInfo(id=2, eventTime=1720196019871, latitude=35.7104365, longitude=51.411407, cellTechnology=4G (LTE), plmnId=43235, rac= , tac=12466, lac= , cellId=43834656, signalStrength=-77, rsrq=-13,