
دانشکده مهندسی کامپیوتر

# پروژه پایانی درس شبکه های تلفن همراه

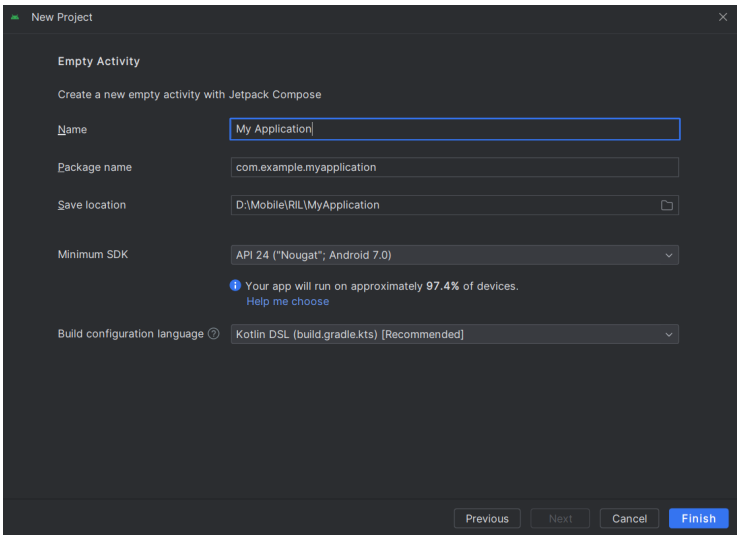
# پروژه ساماریوم

نام و نام خانوادگی اعضای گروه: پوریا رحیمی - علی سلطانی

نیم‌سال دوم
سال تحصیلی ۰۳-۱۴۰۲

# بخش اول

در بخش اول ما در ابتدا شروع به نصب برنامه اندروید استودیو و اجرای آن می‌کنیم. پس از آنکه SDK را دانلود کردیم شروع به کدنویسی می‌کنیم. برای همین در ابتدا یک پروژه به صورت زیر ایجاد می‌کنیم.




شکل ۱: محیط ایجاد یک پروژه جدید

# بخش دوم

در این بخش شروع به پیاده‌سازی فایل‌های مختلف می‌کنیم:

## ۱ معرفی کد MapActivity

کد MapActivity در پروژه اندروید به منظور نمایش نقشه و مارکرها با استفاده از Google Maps API استفاده می‌شود. این کد به طور خاص داده‌های ذخیره شده در پایگاه داده محلی را بازیابی می‌کند و آن‌ها را بر روی نقشه به همراه جزئیات اضافی مانند کیفیت سیگنال و تکنولوژی شبکه نمایش می‌دهد.

## ۲ ویژگی‌های نرم‌افزاری

این نرم‌افزار دارای ویژگی‌های زیر است:

- **بازیابی داده‌ها از پایگاه داده:** استفاده از DatabaseHelper برای دریافت اطلاعات ذخیره شده.
- **نمایش مارکرها:** اضافه کردن مارکرها بر روی نقشه برای نمایش موقعیت‌ها.
- **نمایش کیفیت سیگنال:** تعیین کیفیت سیگنال بر اساس تکنولوژی شبکه و سیگنال دریافت شده.
- **رسم مسیر:** رسم مسیر طی شده بر روی نقشه با استفاده از Polyline.
- **پنجره اطلاعات سفارشی:** نمایش اطلاعات بیشتر درباره هر مارکر با استفاده از پنجره اطلاعات سفارشی.

## ۳ الگوریتم‌های مورد استفاده

در این بخش به توضیح الگوریتم‌های اصلی مورد استفاده در کد می‌پردازیم:

### ۱.۳ دریافت کیفیت سیگنال

کیفیت سیگنال برای سه نوع تکنولوژی GSM، CDMA و WCDMA با استفاده از الگوریتم‌های زیر تعیین می‌شود:

- **الگوریتم GSM:**

```
fun getGsmSignalQuality(signalStrength: Int): Int {
    return when {
        signalStrength >= -75 -> 4 // Very good
        signalStrength >= -80 -> 3 // Good
```

```
            signalStrength >= -90 -> 2 // Fair
            signalStrength > -100 -> 1 // Poor
            else -> -1 // Unknown or no signal
        }
    }
```

در این الگوریتم، کیفیت سیگنال بر اساس قدرت سیگنال به چهار دسته مختلف تقسیم می‌شود: بسیار خوب، خوب، متوسط، ضعیف و ناشناخته.

- **الگوریتم CDMA:**

```
fun getCdmaSignalQuality(signalStrength: Int): Int {
    return when {
        signalStrength >= -70 -> 4 // Very good
        signalStrength >= -80 -> 3 // Good
        signalStrength >= -90 -> 2 // Fair
        signalStrength > -100 -> 1 // Poor
        else -> -1 // Unknown or no signal
    }
}
```

الگوریتم مشابهی برای تکنولوژی CDMA استفاده می‌شود، اما با محدوده‌های متفاوت.

- **الگوریتم WCDMA:**

```
fun getWcdmaSignalQuality(signalStrength: Int): Int {
    return when {
        signalStrength >= -70 -> 4 // Very good
```

```
            signalStrength >= -80 -> 3 // Good
            signalStrength >= -90 -> 2 // Fair
            signalStrength > -100 -> 1 // Poor
            else -> -1 // Unknown or no signal
        }
    }
```

الگوریتم مشابهی برای تکنولوژی WCDMA نیز استفاده می‌شود.

### ۲.۳ تبدیل کیفیت سیگنال به رنگ

برای نمایش مارکرها بر اساس کیفیت سیگنال، رنگ مارکرها تعیین می‌شود:

```
fun getColorForSignalQuality(signalQuality: Int): Float {
    return when (signalQuality) {
        4 -> BitmapDescriptorFactory.HUE_BLUE   // Very good
        3 -> BitmapDescriptorFactory.HUE_GREEN  // Good
        2 -> BitmapDescriptorFactory.HUE_YELLOW // Fair
        1 -> BitmapDescriptorFactory.HUE_ORANGE // Poor
        else -> BitmapDescriptorFactory.HUE_RED // Very poor or unknown
    }
}
```

در این الگوریتم، هر کیفیت سیگنال به یک رنگ خاص نسبت داده می‌شود که مارکرها را با توجه به کیفیت سیگنال رنگ‌آمیزی می‌کند.

### ۳.۳ فرمت کردن زمان

برای نمایش زمان به صورت خوانا، از الگوریتم زیر استفاده می‌شود:

```
fun formatTimestamp(timestamp: String): String {
    val sdf = SimpleDateFormat("yyyy-MM-dd HH:mm:ss", Locale.getDefault())
    val date = Date(timestamp.toLong())
    return sdf.format(date)
}
```

این الگوریتم زمان را از فرمت timestamp به فرمت خوانای yyyy-MM-dd HH:mm:ss تبدیل می‌کند.

## ۴ روند کلی اجرای کد

۱. **راه‌اندازی نقشه:** با استفاده از SupportMapFragment نقشه گوگل بارگذاری می‌شود.
۲. **بازیابی داده‌ها:** داده‌های مورد نیاز از پایگاه داده محلی بازیابی می‌شوند.
۳. **اضافه کردن مارکرها:** مارکرها بر اساس موقعیت‌های جغرافیایی و کیفیت سیگنال بر روی نقشه اضافه می‌شوند.
۴. **رسم مسیر:** مسیر طی شده بین نقاط مختلف بر روی نقشه رسم می‌شود.
۵. **نمایش اطلاعات مارکرها:** با کلیک بر روی هر مارکر، پنجره اطلاعات سفارشی نمایش داده می‌شود که جزئیات بیشتری را در اختیار کاربر قرار می‌دهد.

## ۵ نتیجه‌گیری

کد MapActivity با استفاده از Google Maps API و داده‌های محلی، نقشه‌ای تعاملی با مارکرها و مسیرها ایجاد می‌کند. این نرم‌افزار با تحلیل کیفیت سیگنال‌های مختلف

و نمایش اطلاعات مرتبط با آن‌ها، ابزاری مفید برای کاربران فراهم می‌کند تا اطلاعات جغرافیایی و تکنولوژی‌های شبکه را به‌صورت بصری مشاهده کنند.

# ۶ توضیحات کلی

## ۱.۶ CellularService.kt

این فایل کلاس `CellularService` را تعریف می‌کند که مسئول جمع‌آوری داده‌های موقعیت مکانی و اطلاعات سلولی است. این کلاس از `TelephonyManager` و `LocationManager` برای دریافت اطلاعات سلولی و موقعیت مکانی استفاده می‌کند.

## ۲.۶ DatabaseHelper.kt

این فایل کلاس `DatabaseHelper` را تعریف می‌کند که مسئول مدیریت پایگاه داده SQLite است. این کلاس شامل توابعی برای ایجاد، بروزرسانی و درج داده در پایگاه داده است.

## ۳.۶ MainActivity.kt

این فایل کلاس `MainActivity` را تعریف می‌کند که فعالیت اصلی برنامه را مدیریت می‌کند. در این کلاس درخواست‌های دسترسی به مجوزها انجام می‌شود و در صورت دریافت مجوزها، سرویس جمع‌آوری داده‌های سلولی شروع می‌شود.

# ۷ ویژگی‌های نرم‌افزاری

## ۱.۷ مجوزهای مورد نیاز

برای دریافت داده‌های موقعیت مکانی و اطلاعات سلولی، برنامه نیاز به مجوزهای `ACCESS_FINE_LOCATION`، `ACCESS_COARSE_LOCATION` و `READ_PHONE_STATE` دارد. در صورت عدم وجود

این مجوزها، برنامه درخواست مجوزها را انجام می‌دهد.

### ۲.۷ جمع‌آوری داده‌های موقعیت مکانی

CellularService از LocationManager برای درخواست بروزرسانی موقعیت مکانی از دو منبع GPS_PROVIDER و NETWORK_PROVIDER استفاده می‌کند. داده‌های موقعیت مکانی شامل عرض و طول جغرافیایی و همچنین زمان جمع‌آوری داده‌ها است.

### ۳.۷ جمع‌آوری اطلاعات سلولی

CellularService از TelephonyManager برای دریافت اطلاعات سلولی شامل نوع شبکه، شناسه سلول، قدرت سیگنال و کیفیت سیگنال استفاده می‌کند. اطلاعات سلولی از انواع مختلف شبکه‌ها شامل LTE، GSM، CDMA، WCDMA و NR جمع‌آوری می‌شود.

### ۴.۷ ذخیره‌سازی داده‌ها

داده‌های جمع‌آوری شده توسط کلاس DatabaseHelper در پایگاه داده SQLite ذخیره می‌شوند. جدول cellular_info شامل ستون‌های مختلفی برای ذخیره‌سازی اطلاعات موقعیت مکانی، نوع شبکه، شناسه سلول، قدرت سیگنال و کیفیت سیگنال است.

## ۸ الگوریتم‌های استفاده شده

### ۱.۸ محاسبه فاصله پیموده شده

در CellularService، تابع calculateDistanceWalked فاصله پیموده شده از آخرین موقعیت مکانی ذخیره شده تا موقعیت مکانی جدید را محاسبه می‌کند. این تابع از تابع distanceTo کلاس Location برای محاسبه فاصله استفاده می‌کند.

### ۲.۸ کیفیت سیگنال

برای ارزیابی کیفیت سیگنال در شبکه‌های مختلف، توابع getGsmSignalQuality، getCdmaSignalQuality و getWcdmaSignalQuality استفاده می‌شوند. این توابع بر اساس قدرت سیگنال دریافتی، کیفیت سیگنال را به صورت مقادیر ۱ تا ۴ (ضعیف تا عالی) ارزیابی می‌کنند.

## ۹ AndroidManifest

AndroidManifest شامل مجوزها و تنظیمات برنامه است. مجوزها شامل ACCESS_FINE_LOCATION، ACCESS_COARSE_LOCATION، READ_PHONE_STATE، INTERNET و WRITE_EXTERNAL_STORAGE است که برای دسترسی به مکان‌یابی دقیق، مکان‌یابی تقریبی، خواندن وضعیت تلفن، دسترسی به اینترنت و نوشتن در حافظه خارجی استفاده می‌شود. همچنین، تنظیمات برنامه شامل اطلاعات مربوط به آیکون برنامه، پشتیبانی از RTL، و تنظیمات مربوط به نسخه‌های مختلف اندروید است.

[language=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”>

”android.permission.READ$_{P}HONE_{S}TATE$”

”android.permission.ACCESS$_{F}INE_{L}OCATION$”

”android.permission.ACCESS$_{C}OARSE_{L}OCATION$”

”android.permission.INTERNET”

”android.permission.WRITE$_{E}XTERNAL_{S}TORAGE$”

<uses-feature android:glEsVersion=”0x00020000” android:required=”true”
/>

<application

android:allowBackup=”true”

android:dataExtractionRules=”@xml/data$_e xtraction_r ules$”

android:fullBackupContent=”@xml/backup$_r ules$”

android:icon=”@mipmap/ic$_l auncher$”

android:label=”@string/app$_n ame$”

android:roundIcon=”@mipmap/ic$_l auncher_r ound$”

android:supportsRtl=”true”

android:theme=”@style/Theme.Samarium”

tools:targetApi=”31”>

<meta-data android:name=”com.google.android.geo.API$_K EY$”

android:value=”AIzaSyAUb7PkYrFw5cs8gDY4nkSMZr03T0H4B2Y”/>

<activity android:name=”.MainActivity”

android:exported=”true”

android:label=”@string/app$_n ame$” >

<intent-filter>

<action android:name=”android.intent.action.MAIN” /> <category android:name=”android.intent.category.LAUNCHER” />

</intent-filter> </activity>

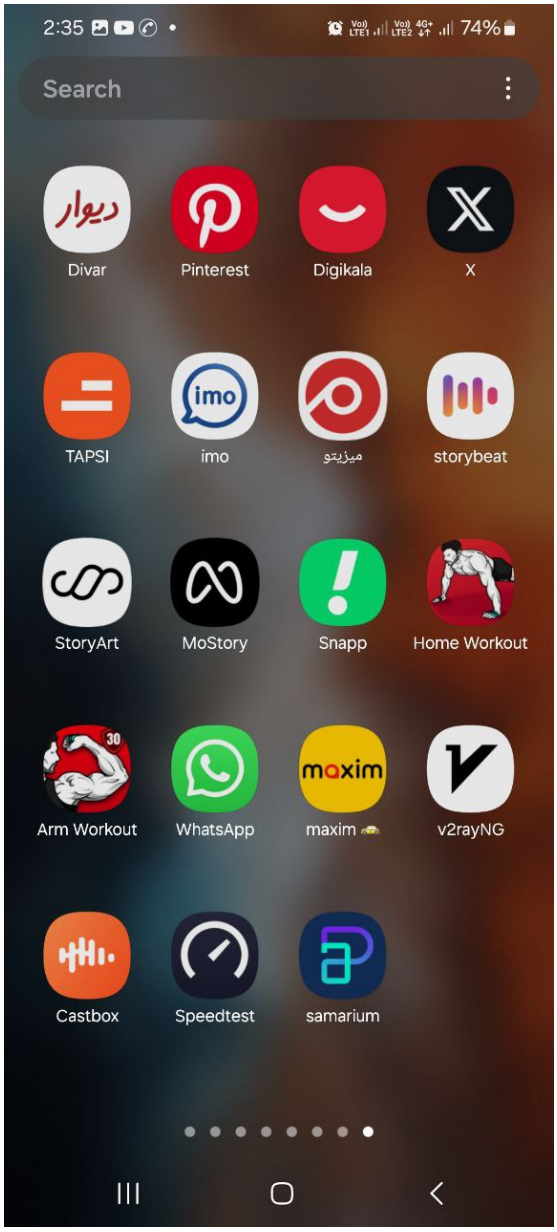
<activity android:name=”.MapActivity” android:exported=”true” android:label=”@string/app$_name$” >< /*activity* >

</application> </manifest>

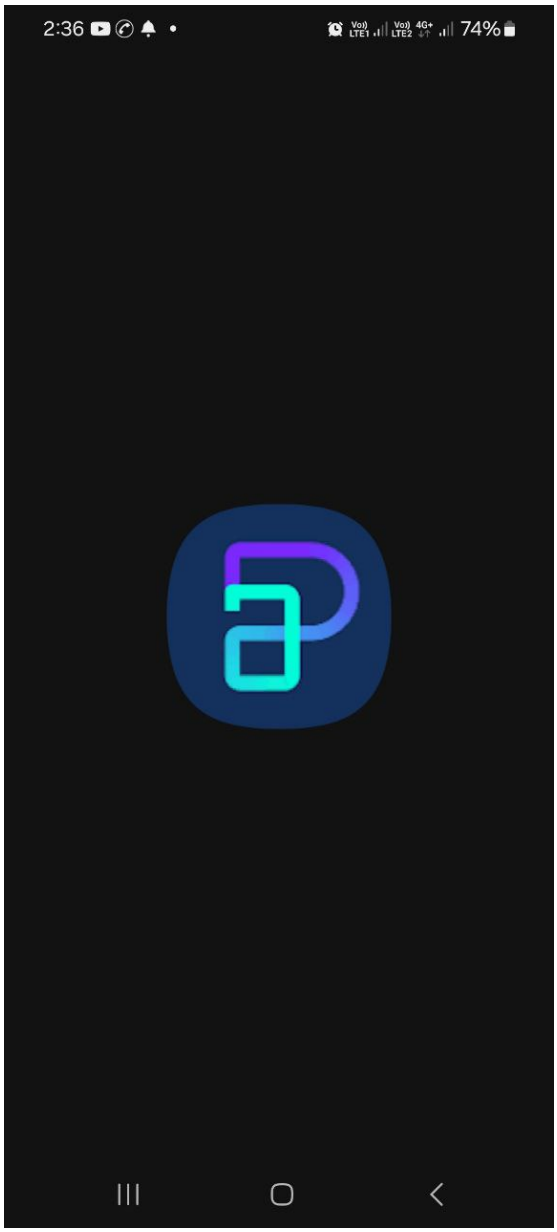
## ۱۰ نتیجه‌گیری

نرم‌افزار ساماریوم با استفاده از مجوزهای لازم، اطلاعات موقعیت مکانی و سلولی را جمع‌آوری کرده و در پایگاه داده ذخیره می‌کند. این اطلاعات می‌توانند برای تحلیل‌های مختلفی مورد استفاده قرار گیرند. الگوریتم‌های استفاده شده در این نرم‌افزار، دقت بالایی در جمع‌آوری و ذخیره‌سازی داده‌ها ارائه می‌دهند.

## ۱۱ خروجی نهایی پروژه

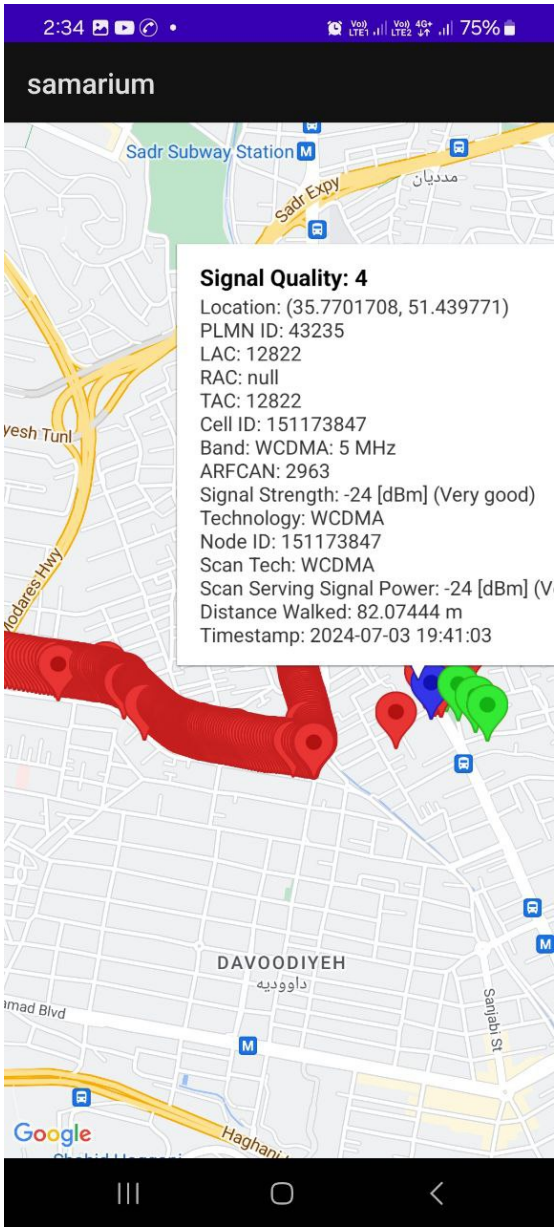



شکل ۲: برنامه ساماریوم موجود بر دیوایس سخت افزاری




شکل ۳: اجرای برنامه

## ۱۲ اجرای برنامه و دیدن نتیجه




شکل ۴: اجرای برنامه