МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ОДЕСЬКИЙ ДЕРЖАВНИЙ ЕКОЛОГІЧНИЙ УНІВЕРСИТЕТ

Інститут післядипломної освіти

Кваліфікаційна робота бакалавра

на тему: Розробка мобільного додатку для сповіщення про знаходження небезпечних об'єктів

Виконав студент групи КН-5
спеціальності 122 Комп'ютерні науки
Алакберлі Ніяз Яшар Огли

Керівник к.т.н., доцент
Фразе-Фразенко Олексій Олексійович

Консультант

Рецензент т.в.о. директора КП
"Обласний інформаційно-
аналітичний центр"
Попов Володимир Леонідович

Одеса 2023

ЗМІСТ

СКОРОЧЕННЯ ТА УМОВНІ ПОЗНАКИ 5
ВСТУП 6
1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ ТА ІСНУЮЧИХ ПРОГРАМНИХ СИСТЕМ 8
1.1 Аналіз предметної області 8
1.2 Аналіз існуючих програмних систем 12
2 ВИБІР ТА ОБҐРУНТУВАННЯ ЗАСОБІВ РОЗРОБКИ 15
2.1 Функціональні та нефункціональні вимоги 15
2.2 Обґрунтування вибору операційної системи 15
2.3 Вибір середовища розробки 17
3 ГЕОІНФОРМАЦІЙНІ ТЕХНОЛОГІЇ ТА ГЕОЛОКАЦІЯ 20
3.1 Основні поняття геопозиціонування 20
3.2 Типи даних геолокації 21
3.3 Місцезнаходження та картографування в Android 23
3.4 Огляд картографічних служб 24
3.5 Основні принципи роботи зі службами визначення місцезнаходження 29
4 ПРОЕКТУВАННЯ МОБІЛЬНОГО ЗАСТОСУНКУ 32
4.1 Діаграма прецедентів 32
5 РЕАЛІЗАЦІЯ МОБІЛЬНОГО ЗАСТОСУНКУ 34
5.1. Використання бібліотеки osmdroid 34
5.2 Використання Firebase 38
5.3 Опис та тестування застосунку 43
ВИСНОВКИ 49
ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ 50

СКОРОЧЕННЯ ТА УМОВНІ ПОЗНАКИ

API – Application Programming Interface, програмний інтерфейс програми

GPS – Global Positioning System, система глобального позиціонування

IDE – Integrated Development Environment, інтегроване середовище розробки

IED – Improvised Explosive Device, саморобний вибуховий пристрій

IP – Internet Protocol, інтернет протокол

LBS – Location-based services, послуги на основі місцезнаходження

OSM – OpenStreetMap, «відкрита вулична мапа»

UI – User Interface, інтерфейс користувача

UML – Unified Modeling Language, уніфікована мова моделювання

URL – Uniform Resource Locator, єдиний вказівник на ресурс

VPN – Virtual Private Network, віртуальна приватна мережа

ДСНС – Державна служба України з надзвичайних ситуацій

НВБ – Боєприпас, що не вибухнув

ОС – Операційна система

РСЗВ – Реактивна система залпового вогню

ВСТУП

Розвиток технологій відкрив нові можливості для підвищення громадської безпеки, зокрема за допомогою мобільних застосунків. Одним із таких застосунків, який може зробити вагомий внесок у цій області, є застосунок для сповіщення про знаходження небезпечних об’єктів. Метою кваліфікаційної роботи є розробка такого застосунку.

Триваючий воєнний конфлікт та загроза терористичних атак породили нагальну потребу в ефективній системі оповіщення громадськості про виявлення небезпечних об’єктів. Мобільний застосунок може служити надійним інструментом для швидкого інформування користувачів про будь-які небезпечні матеріали, вибухівку чи зброю, які були виявлені поблизу. Ця інформація може допомогти запобігти нещасним випадкам, травмам і врятувати життя.

Застосунок також може бути цінним інструментом для правоохоронних органів і органів безпеки, оскільки він може допомогти їм швидко поширювати інформацію про небезпечні об’єкти та координувати свою реакцію на будь-які потенційні загрози.

Розробка мобільного застосунку має переваги, зокрема надання користувачам інформації в режимі реального часу про наявність небезпечних об’єктів, можливість швидко повідомляти про будь-які такі об’єкти та можливість безпосередньо сповіщати органи ДСНС. Користувачі також можуть отримувати сповіщення про стан ситуації та будь-які оновлення щодо потенційних загроз. Крім того, програма може допомогти у створенні безпечнішого середовища, сприяючи обізнаності громадськості про потенційні небезпеки, що може призвести до культури безпеки.

Для досягнення мети кваліфікаційної роботи були сформульовані такі завдання:

- аналіз предметної області;

- порівняльний аналіз існуючих програм-аналогів;
- обґрунтування вибору програмних засобів розробки та технологій;
- проектування системи;
- реалізація застосунку;
- тестування.

Структура кваліфікаційної роботи бакалавра складається з вступу, 5 розділів, висновків, переліку посилань на 12 найменувань.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ ТА ІСНУЮЧИХ ПРОГРАМНИХ СИСТЕМ

1.1 Аналіз предметної області

Під час конфліктів і військових дій руйнівний вплив на життя людей та інфраструктуру є очевидним. Однак ще одним наслідком війни, який часто забувають, є її згубний вплив на довкілля, зокрема забруднення землі небезпечними предметами. Від боєприпасів, що не вибухнули, до хімічної зброї та покинутого військового обладнання, наявність цих небезпек створює довгострокові ризики для екосистем, здоров’я людей і майбутніх поколінь.

Однією з найбільш серйозних загроз для забруднення земель у постраждалих від війни районах є боєприпаси, що не розірвалися. НВБ – це вибухові речовини, такі як наземні міни, касетні бомби та артилерійські снаряди, які не спрацювали під час удару. Ці смертоносні пережитки війни становлять серйозну небезпеку для цивільного населення, перешкоджаючи постконфліктному відновленню та забруднюючи землю небезпечними матеріалами. Нестабільність НВБ може призвести до випадкових детонацій, що спричинить подальші руйнування та жертви.

Використання хімічної зброї під час конфліктів має руйнівні наслідки не лише для людських життів, а й для навколишнього середовища. Хімічні агенти, такі як нервово-паралітичні речовини, іприт і токсичні промислові хімікати, забруднюють ґрунт і ґрунтові води, становлячи довгострокову загрозу як для здоров’я людей, так і для навколишнього середовища. Стійкість цих агентів може призвести до серйозного забруднення, що вплине на сільськогосподарські угіддя, джерела води та дику природу з потенційними наслідками, які триватимуть десятиліттями.

Під час війни значні обсяги військової техніки, включаючи транспортні засоби, танки, літаки та боєприпаси, залишаються на полі бою. З часом ці матеріали піддаються корозії, вивільняючи небезпечні речовини в ґрунт, воду

та повітря. Паливно-мастильні матеріали витікають, важкі метали просочуються в землю, а пластикові та гумові компоненти руйнуються, спричиняючи забруднення, яке може зберігатися роками, впливаючи на місцеві екосистеми та людські спільноти.

У зонах конфлікту часто спостерігається збільшення виробництва відходів через порушення інфраструктури та основних послуг. Накопичення відходів може призвести до створення імпровізованих звалищ, де шкідливі речовини утилізуються неналежним чином, що підвищує ризик забруднення земель. Крім того, саморобні вибухові пристрої (IED), виготовлені з легкодоступних матеріалів, включаючи хімічні та вибухові компоненти, становлять значну загрозу для забруднення землі та погіршення стану навколишнього середовища.

Забруднення землі небезпечними об’єктами має серйозні наслідки для здоров’я людини та екосистем. Токсичні речовини, що виділяються з цих об’єктів, можуть потрапляти в харчовий ланцюг, впливаючи на сільське господарство та тваринництво. Крім того, забруднення джерел води може призвести до втрати водного життя та поставити під загрозу наявність чистої питної води для громад у постраждалих районах. Тривалий вплив цих забруднюючих речовин може призвести до різних проблем зі здоров’ям, включаючи респіраторні захворювання, рак і неврологічні розлади.

Визнаючи нагальну потребу у боротьбі із забрудненням землі, спричиненим небезпечними об’єктами, зусилля з постконфліктного очищення та відновлення відіграють вирішальну роль у відновленні постраждалих територій. Ці ініціативи передбачають очищення від невибухаючих боєприпасів, дезактивацію хімічних ділянок, належну утилізацію покинутого військового обладнання та створення безпечних систем поводження з відходами. Міжнародні організації, уряди та неурядові організації повинні співпрацювати, щоб розподіляти ресурси, підвищувати обізнаність і впроваджувати комплексні стратегії очищення.

Військове вторгнення Росії в Україну, яке почалося в 2014 році, має глибокий вплив на країну як політично, так і економічно. Одним із найруйнівніших наслідків конфлікту також стало забруднення території України вибухонебезпечними предметами. З 2014 року, за оцінками, постраждало 150-200 тисяч квадратних кілометрів, що становить 25-30% всієї країни. За даними ООН це робить Україну однією з найбільш замінованих країн у світі.

Завдання очистити цю величезну територію від вибухонебезпечних пристроїв лежить на українських саперах і рятувальниках, які стикаються з величезними труднощами, намагаючись зробити територію безпечною для мирного населення. Щодня вони мають знешкоджувати від 2 до 6 тисяч різних видів снарядів і мін.

Протипіхотні міни.

Невеликий пластиковий об'єкт, що використовується для знищення живої сили противника за допомогою вибухової сили міни. Принцип дії - тиск. Активуються після того, як на них наступають. Пошкодження залежать від типу міни, але найчастіше трапляються пошкодження кінцівок.

Особливо небезпечними є міни типу "метелик" або "пелюстка". Ці міни мають розмір 12 см, важать приблизно 80 грамів і розкидаються з касетних боєприпасів. Вони вибухають при контакті або мають детонатор з годинниковим механізмом, тобто механізм самоліквідації. Через кілька годин міни можуть здетонувати самостійно.

Протипіхотні осколкові міни.

На відміну від фугасних мін, осколкові міни завдають шкоди за рахунок вибухового компонента в корпусі міни, а не за рахунок вибухової сили. Вони можуть бути зариті під землею або замасковані в деревах. Такі міни вибухають при попаданні на натягнутий дріт або спрацьовують за допомогою дистанційного керування. Їхня дія може бути спрямованою (осколки вилітають з одного боку міни) або круговою (вони розлітаються на 360°).

Однією з найнебезпечніших осколкових мін є ОЗМ-72. Вона має розмір літрової пляшки. Важить вона 5 кг. Вона містить всередині 2400 осколків і може пролетіти 50 метрів. Коли людина потрапляє під пусковий дріт, заряд, що "вибиває", відправляє міну в політ на висоту до одного метра, після чого вибухає. Шансів вижити практично немає. Протипіхотні міни заборонені Оттавською конвенцією з 1999 року.

Протитранспортні (протитанкові) міни.

Протитранспортні міни також закопують під землею або встановлюють на поверхні; вони вибухають при вазі 120 кг або більше. Протипіхотні міни часто встановлюються під протитранспортними мінами. Якщо вони вибухнуть, шанси на виживання майже нульові.

Касетні боєприпаси.

РСЗВ "Град", "Смерч", "Ураган" та "Точка У" можуть бути оснащені касетними боєприпасами. Такі боєприпаси сконструйовані за принципом російської "матрьошки". Випущені боєприпаси містять 30-70 дрібних боєприпасів і розлітаються на площі в кілька гектарів. Ці елементи мають розмір 12 см і механізм самознищення.

Цей тип боєприпасів заборонений Міжнародною конвенцією про касетні боєприпаси з 2010 року, однак російські військові використовували їх неодноразово.

Саморобні вибухові пристрої.

Цей тип боєприпасів важко ідентифікувати, вони можуть бути заховані в повсякденних предметах, встановлені в транспортних засобах або закріплені всередині будівель. Саморобні вибухові пристрої відрізняються за розміром, кольором і вагою. Принцип дії - довільний.

До повномасштабного вторгнення російських військ у лютому 2022 року в Україні вже були мінно небезпечні райони, зокрема на територіях Донецької та Луганської областей. За оцінками Асоціації саперів України, близько 7 тисяч квадратних кілометрів території цих регіонів потребували підвищеної уваги саперів.

Після російського вторгнення 2022 масштаби проблеми надзвичайно зросли. Станом на серпень міністр екології Руслан Стрілець повідомляв, що майже третина території України – приблизно 200 тисяч квадратних кілометрів – потребує розмінування. У перспективі ця площа приблизно еквівалентна всій території Великої Британії.

Незважаючи на зусилля українських саперів і рятувальників, процес розмінування йде повільно і кропітко, прогресу часто заважає конфлікт, що триває. Аби повністю очистити територію України від вибухових пристроїв, потрібні величезні ресурси, як людські, так і фінансові. За даними Асоціації саперів України, розмінування 200 тисяч квадратних кілометрів обійдеться щонайменше в 500 мільярдів доларів.

Вплив забруднення території України вибуховими пристроями виходить далеко за межі безпосередньої небезпеки для цивільного населення. Наявність мін та інших вибухових пристроїв ускладнює повернення людей додому та відновлення повсякденного життя, а також становить значну загрозу для аграрного сектору країни. Фермери не можуть використовувати великі площі землі через ризик вибухів, що має серйозний вплив на продовольчу безпеку та економіку в цілому [1].

1.2 Аналіз існуючих програмних систем

На даний час створюються різноманітні застосунки і сервіси, що надають можливість отримувати сповіщення про знаходження небезпечних об'єктів. Розглянемо деякі з них.

MineFree від Free Ukraine

MineFree - це безкоштовний мобільний застосунок, створений благодійною організацією Free Ukraine за підтримки ДСНС для попередження про виявлення мін, вибухонебезпечних та інших підозрілих предметів на

території України (рис. 1.1). Застосунок доступний на платформах iOS та Android.

Зареєстровані користувачі можуть повідомити про місцезнаходження вибухонебезпечних та підозрілих предметів за допомогою електронної форми, яка містить фото, геолокацію та опис. Ця інформація надасть можливість ДСНС оперативно реагувати на повідомлення для подальшої ідентифікації та знешкодження таких предметів.

Усі користувачі отримують доступ до мапи з територіями, які потенційно можуть бути забруднені вибухонебезпечними предметами. На цій мапі відображаються місця, на яких вже виявлено або ймовірно знаходяться боєприпаси згідно наявної інформації у ДСНС. Застосунок також надає доступ до довідника ДСНС з фотографіями та описом вибухонебезпечних предметів. Ця інформація доповнюється, щоб інформувати громадян України про правила поведінки та можливі ризики пов'язані з вибухонебезпечними предметами [2]. Застосунок також містить статистику щодо кількості знайдених та знищених мін на території України.

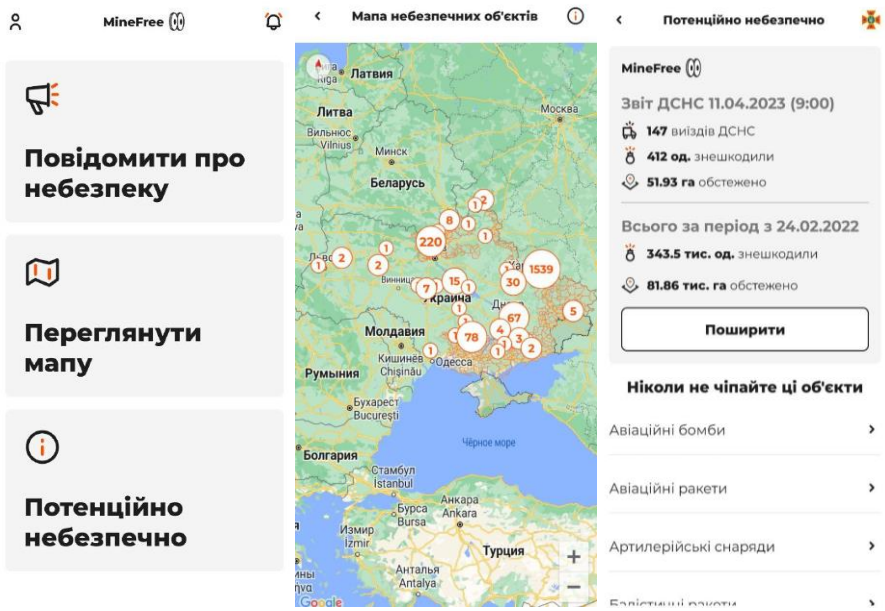

Рисунок 1.1 – Інтерфейс застосунку MineFree

Розмінування України від ДСНС

Розмінування України - це офіційне джерело повідомлення органам ДСНС про виявлені вибухонебезпечні та підозрілі предмети [3] (рис. 1.2).

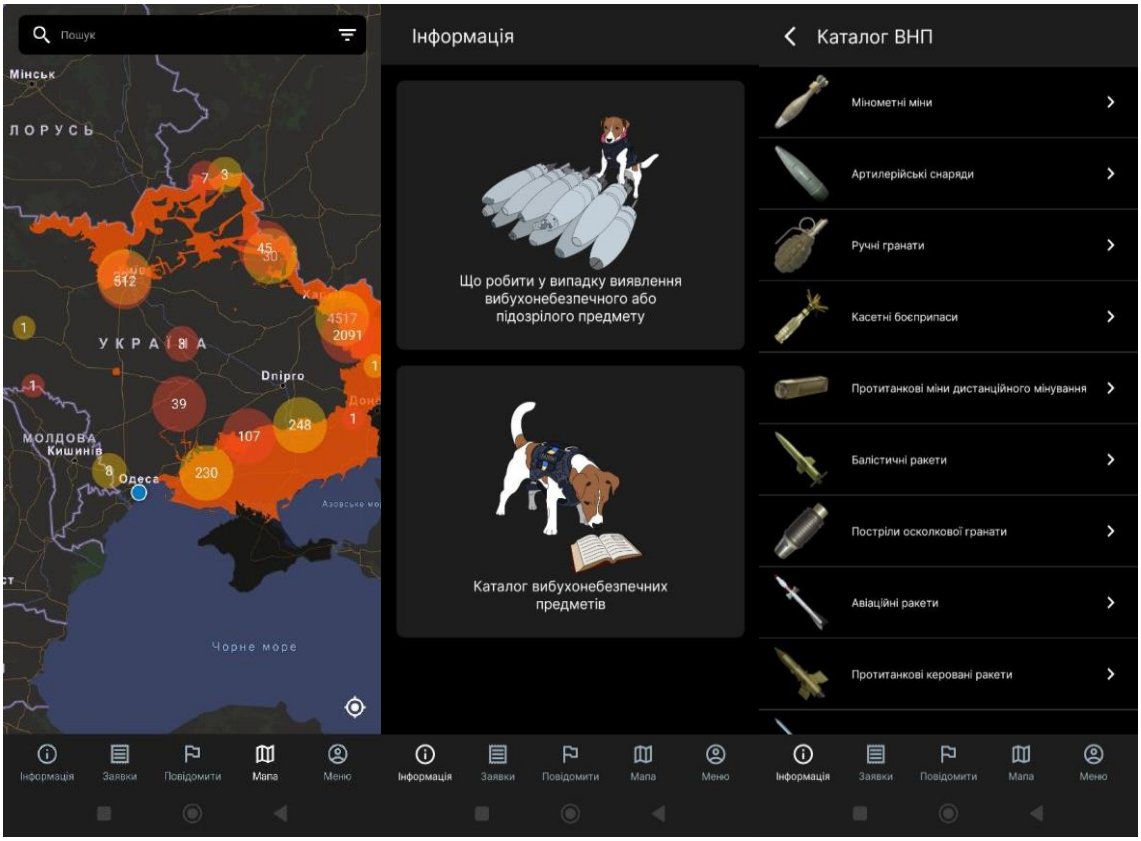

Рисунок 1.2 – Інтерфейс застосунку Розмінування України

2 ВИБІР ТА ОБҐРУНТУВАННЯ ЗАСОБІВ РОЗРОБКИ

2.1 Функціональні та нефункціональні вимоги

Перед початком розробки застосунку були визначені функціональні та нефункціональні вимоги до нього.

До функціональних вимог належать:

- відображення небезпечних об’єктів на карті;
- сповіщення про небезпечні об’єкти на пристрої користувача;
- відображення інформації про тип небезпечного об’єкту;
- можливість встановити радіус, в межах якого надходять повідомлення про небезпечні об’єкти;
- відображення детальної інформації про кожний об'єкт;
- можливість додати інформацію про знайдений небезпечний об’єкт.

До нефункціональних вимог належать:

- простий і зрозумілий інтерфейс;
- швидкість та ефективність у використанні;
- смартфон з ОС Android 8+;
- підключення до Інтернету.

2.2 Обґрунтування вибору операційної системи

При виборі операційної системи для розробки застосунку необхідно враховувати ряд факторів. Основні операційні системи: Android, iOS та Windows.

Android має найбільшу частку ринку серед усіх мобільних платформ: за даними Statista, вона перевищує 72%, порівняно з 27% для iOS і 0,3% для Windows Phone. Така популярність операційної системи Android забезпечує додаткам велику кількість користувачів, сприяючи їх поширенню та використанню.

Android - це платформа з відкритим вихідним кодом, що дозволяє розробникам створювати додатки за допомогою різноманітних інструментів. Її відкритість також дозволяє створювати велику кількість застосунків з різною функціональністю та інтерфейсами. З іншого боку, iOS і Windows Phone є закритими платформами.

Функціональність мобільного пристрою Android можна розширювати, підлаштовуючи його під потреби користувача, наприклад, змінюючи інтерфейс, додаючи нові функції та додатки. На iOS та Windows Phone такі можливості обмежені.

Google Play, магазин додатків для Android, має менше обмежень на публікацію додатків, ніж App Store на iOS або Microsoft Store на Windows Phone. Наприклад, Google Play дозволяє публікувати додатки з безкоштовних джерел, тоді як App Store - ні. Крім того, щоб опублікувати додаток в App Store або Microsoft Store, необхідно сплачувати щорічний членський внесок, тоді як Google Play не стягує такої плати, тільки одноразову при реєстрації.

Android має широкі можливості інтеграції з іншими сервісами та додатками, що дозволяє створювати більш складні застосунки з розширеною функціональністю. Наприклад, їх можна інтегрувати з такими сервісами Google, як Google Maps, Gmail і Google Drive, щоб полегшити обробку даних і забезпечити більш зручний користувацький досвід.

Android не має обов'язкових вимог до організаційної структури компанії, яка розробляє застосунок. Це дає можливість розробляти застосунки як великим компаніям, так і невеликим стартапам. iOS і Windows Phone вимагають, щоб розробник був зареєстрованим і мав відповідну структуру компанії, щоб опублікувати свій застосунок.

Android працює на широкому спектрі пристроїв, що дозволяє охопити широку базу користувачів; інші платформи, такі як iOS, працюють лише на обмеженій кількості пристроїв, що зменшує кількість потенційних користувачів вашого додатку.

Android надає користувачам більше можливостей для налаштування своїх пристроїв і застосунків, що забезпечує більш гнучкий користувацький досвід. Наприклад, застосунки можна налаштувати на автоматичний запуск при запуску пристрою, що корисно для тих, які потребують постійного моніторингу.

Оскільки Android є відкритою системою, розробники можуть створювати застосунки, які взаємодіють з різними компонентами пристрою, такими як звуковий динамік, камера або модуль Bluetooth. Вони також можуть змінювати системні налаштування пристрою, що дає їм більше контролю над поведінкою пристрою.

Оскільки Android працює на пристроях широкого спектру виробників, ціна на ці пристрої є значно нижчою, ніж на аналогічні пристрої від Apple та Microsoft. Це робить Android-пристрої доступними для більшої кількості людей і збільшує популярність Android.

Таким чином, розглядаючи переваги розробки додатків для Android, стає зрозуміло, що ця платформа має багато переваг перед iOS і Windows Phone. Зокрема, це багаті можливості кастомізації, свобода розробки та публікації додатків, ширший спектр пристроїв і більш доступні ціни. Крім того, свобода маніпулювання додатками в Android полегшує розробку та підтримку нових функцій.

2.3 Вибір середовища розробки

Поширеними середовищами розробки під ОС Android є Android Studio, Xamarin, Eclipse.

Android Studio - офіційне середовище розробки для Android від Google. Воно базується на IntelliJ IDEA, загальному середовищі розробки, але з додатковими інструментами, призначеними для розробки додатків для Android. Основними перевагами Android Studio є:

- Інтегроване середовище розробки (IDE), яке включає всі інструменти, необхідні для розробки Android-застосунків.
- Просте у налаштуванні та використанні.
- Містить велику кількість шаблонів і зразків для розробки додатків.
- Використовує Java, одну з найпопулярніших мов програмування у світі.
- Має велику та активну спільноту розробників, з регулярними оновленнями та підтримкою середовища.

Недоліки: вимагає досить потужного комп'ютера для роботи та може повільно працювати на старих комп'ютерах.

Eclipse - популярне середовище розробки, яке можна використовувати для створення застосунків для Android. Це вільне програмне забезпечення з великою та активною спільнотою розробників. Основними перевагами середовища Eclipse є:

- Багато плагінів та інструментів для розробки додатків.
- Простий у налаштуванні та використанні.
- Використовує мову програмування Java, одну з найпопулярніших мов програмування у світі.
- Має велику та активну спільноту розробників і забезпечує регулярні оновлення та підтримку середовища.

Недоліки:

- Не так глибоко інтегрована з Android SDK, як Android Studio.

- Менш потужний і функціональний, ніж Android Studio.

- Потребує додаткового налаштування для підтримки мови програмування Kotlin.

Xamarin - це середовище розробки для розробки додатків для Android, iOS і Windows Phone з використанням мови програмування C#. Основними перевагами Xamarin є:

- Розробка застосунків для різних мобільних платформ з єдиною мовою програмування.

- Можна використовувати всі функції та можливості платформи Android.

- Можна використовувати бібліотеки та інструменти NET Framework, що дозволяє розробляти додатки швидко та ефективно.

- Підтримка мови програмування Kotlin.

Недоліки:

- Розробка додатків вимагає використання спеціального редактора коду, який не так глибоко інтегрований з Android SDK, як Android Studio.

- Він менш популярний і менш трудомісткий у розробці, ніж Android Studio та Eclipse.

- При розробці додатків можуть виникати проблеми сумісності зі сторонніми бібліотеками.

Для розробки Android-додатків краще використовувати Android Studio, оскільки вона глибоко інтегрована з Android SDK, дозволяє швидко створювати і тестувати застосунки, має доступ до низки корисних інструментів та плагінів. Також Android Studio має найбільше ресурсів, включаючи навчальні посібники, документацію та спільноту розробників [4].

3 ГЕОІНФОРМАЦІЙНІ ТЕХНОЛОГІЇ ТА ГЕОЛОКАЦІЯ

3.1 Основні поняття геопозиціонування

Геолокація — це можливість відстежувати місцезнаходження пристрою за допомогою GPS, веж стільникового зв'язку, точок доступу WiFi або їх комбінації. Оскільки пристрої використовуються окремими особами, геолокація використовує системи позиціонування для відстеження місцезнаходження особи за координатами широти та довготи або, більш практично, за фізичною адресою. Геолокацію можуть використовувати як мобільні, так і настільні пристрої.

Геопозиціонування використовує різні візуальні та електронні методи, включаючи позиційні лінії та позиційні кола, астронавігацію, радіонавігацію та використання супутникових навігаційних систем (рис. 3.1).

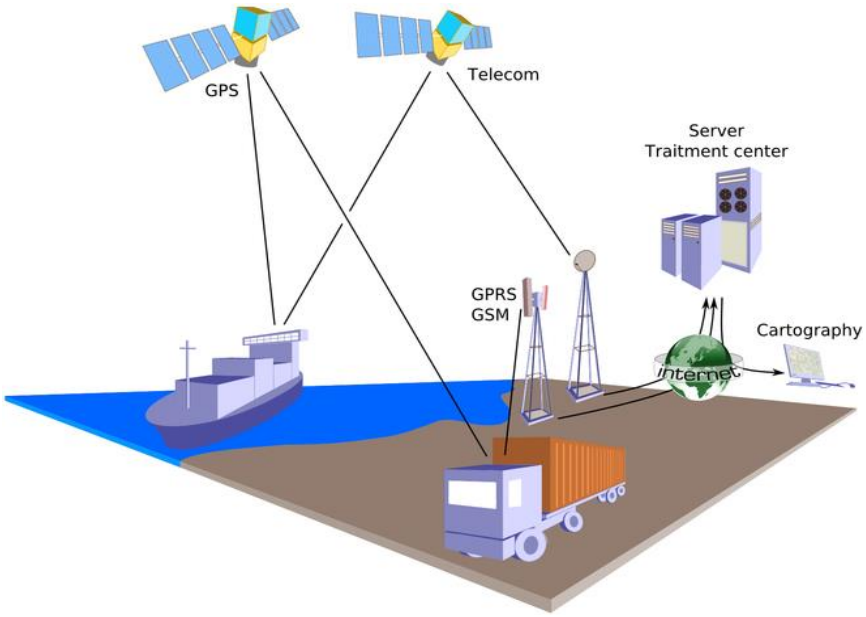

Рисунок 3.1 – Принцип роботи геопозиціонування

Розрахунок вимагає вимірювань або спостережень відстаней або кутів до опорних точок, положення яких відомі. У 2D-зйомках спостереження за трьома опорними точками достатньо для обчислення положення в двовимірній

площині. На практиці спостереження піддаються помилкам, спричиненим різними фізичними та атмосферними факторами, які впливають на вимірювання відстаней і кутів.

Зона сумнівів, що оточує визначення позиції, називається еліпсом помилки. Щоб мінімізувати помилку, електронні навігаційні системи зазвичай використовують більше трьох опорних точок для обчислення фіксованого положення, щоб збільшити надлишковість даних. Оскільки додається більше надлишкових опорних точок, фіксація позиції стає точнішою, а площа результуючого еліпса помилки зменшується.

Процес об’єднання кількох спостережень для обчислення визначення місця розташування еквівалентний розв’язанню системи лінійних рівнянь. Навігаційні системи використовують такі алгоритми регресії, як найменші квадрати, щоб обчислити фіксовану позицію в 3D-просторі. Найчастіше це робиться шляхом поєднання вимірювань відстані до 4 або більше супутників GPS, які обертаються навколо Землі за відомими шляхами.

3.2 Типи даних геолокації

Збір даних на основі пристрою

Мобільні пристрої (наприклад, смартфони, планшети, ноутбуки, смарт-годинники та фітнес-трекери) досить інтуїтивно зрозумілі з точки зору їхньої корисності з геолокацією. Збір даних на основі пристрою базується на GPS і стільникових мережах, тому він точніший у місцях з великою кількістю людей, оскільки існує точніша тріангуляція. Проте, чим нижча щільність населення, тим нижча точність. У таких випадках зазвичай виникають затримки або паузи в даних.

Поки ввімкнено служби визначення місцезнаходження, а у користувача є чіп GPS і сигнал стільникової мережі, можна отримати доступ до цих послуг для визначення свого загального місцезнаходження за допомогою тріангуляції

GPS-вежі. Очевидно, що Інтернет-сервіси, які мають доступ до подібних даних, викликають проблеми з конфіденційністю. Тому для збору даних на основі пристрою:

- Користувачі повинні дозволити визначення місцезнаходження на кожному пристрої (і для кожної програми).

- Веб-сайти мають запитувати місцезнаходження відвідувача.

- Починаючи з Chrome 50 API геолокації HTML працюватиме лише через захищені з'єднання веб-сайтів (https:// в URL-адресі).

Збір даних на сервері

Інший метод геолокації використовує збір даних на сервері, прив'язаних до IP-адреси вашого пристрою через з'єднання WiFi або Ethernet. IP-адреси зберігаються в базах даних, де фізичні місцезнаходження пов'язані з цими IP-адресами, зіставленими за роками аналізу даних. Ці дані продаються сторонніми сервісними службами, що означає, що точність настільки висока, наскільки дані сервісного центру. Щоразу, коли цінність даних ґрунтується на точності, а джерело даних ґрунтується на доступності, цілісність даних стає підозрілою.

Бази даних про місцезнаходження на основі IP мають власні критерії щодо того, як дані були отримані, що дозволяє їм надавати спеціальні рішення геолокації. Наприклад, дані одного популярного постачальника рішень надходять від постачальників послуг, які використовують запити, що вводяться користувачем, що є прямим підходом до отримання інформації, наприклад, просто просять відвідувачів ввести свої адреси у форму. Коли інформацію аналізують за такими самими чи подібними відгуками про місцезнаходження (підтверджувальні дані), а також перевіряють за допомогою алгоритмів визначення місцезнаходження, вона вважається точною або настільки точною, наскільки це дозволяють доступні дані.

Таким чином, якщо введено неправильну інформацію або недостатньо інформації, бази даних вгадують. Отже, точність геолокації IP-адреси

базується на кількості даних, що стосуються конкретного місця, а також на своєчасності отримання цих даних через бази даних сторонніх служб.

У Сполучених Штатах геолокація IP-адреси точна приблизно на 90 відсотків на рівні країни. На рівні міста точність падає від 50 до 70 відсотків. Зважаючи на це, геолокацію IP найкраще використовувати для ширших категорій визначення місцезнаходження, наприклад країни відвідувача веб-сайту. Природно, якщо точність становить менше 50 відсотків, конфіденційність не є великою проблемою, тому веб-сайтам не потрібно запитувати дозвіл на ваше місцезнаходження під час його використання.

Комбінований збір даних

Звичайно, є застереження щодо використання будь-якого типу геолокації. Відвідувачам потрібно надати дозвіл на виявлення на основі пристрою, яка є найточнішою та найкраще підходить для інформації про місцезнаходження в конкретному місті. Виявлення на основі сервера, яке є найменш вразливим і найкраще підходить для інформації, що стосується конкретної країни, може повертати дані в обхід, якщо IP-адреса відвідувача маршрутизується через проксі-сервер (наприклад, VPN). У цьому випадку IP-адреса фактично зіставляється з розташуванням, яке стосується розташування сервера, а не відвідувача. Тому, оскільки будь-який тип збору даних може виявитися невдалим, веб-сайт інколи включатиме обидва типи як запасний варіант, вважаючи, що деякі дані краще, ніж жодні, для забезпечення найкращої взаємодії з користувачем [5].

3.3 Місцезнаходження та картографування в Android

Однією з найважливіших функцій смартфонів є їхня здатність визначати та відстежувати місцезнаходження. Ця функція надає розробникам багато можливостей для створення додатків, які покладаються на геолокаційні та картографічні сервіси для покращення користувацького досвіду. Визначення

місцезнаходження та картографія є невід'ємними функціями багатьох сучасних мобільних додатків. Вони дозволяють користувачам знаходити підприємства, послуги та події у своїй місцевості та переходити до них за допомогою покрокових інструкцій. Android надає ряд інструментів та API-інтерфейси.

Послуги на основі місцезнаходження (англ. Location-Based Services, LBS) - це використання інформації про місцезнаходження для надання послуг, пов'язаних з поточним місцезнаходженням користувача. Ці послуги реалізуються за допомогою поєднання апаратних, програмних і мережевих технологій. LBS можна умовно розділити на два типи: внутрішні та зовнішні. Внутрішні LBS використовують такі технології, як Wi-Fi, Bluetooth та iBeacon для визначення місцезнаходження користувачів у приміщеннях, наприклад, у будівлях. Зовнішні LBS, з іншого боку, використовують такі технології, як GPS і стільникові мережі для визначення місцезнаходження користувачів за межами будівель [6].

Для того, щоб додаток для Android міг використовувати послуги на основі визначення місцезнаходження, він повинен спочатку запитати у користувача дозвіл. Починаючи з Android 6.0 (Marshmallow), застосунки повинні запитувати дозвіл під час роботи, а не під час встановлення. Це означає, що користувачі можуть надавати або забороняти дозволи на визначення місцезнаходження для кожного додатка окремо. Розробники також можуть вказати рівень точності, необхідний для їхніх додатків, наприклад, висока точність для навігації і низька точність для фонового оновлення місцезнаходження.

3.4 Огляд картографічних служб

Картографічні служби використовуються для відображення географічних даних у візуальному форматі. Ці послуги надають різні

постачальники карт, такі як Google Maps, Mapbox і OpenStreetMap. Ці постачальники пропонують ряд інструментів і послуг, які розробники можуть використовувати для інтеграції функцій картографування у свої програми.

Mapbox

Mapbox заснована в 2010 році для постачання некомерційних екологічних і гуманітарних організацій картографічними даними та їх аналізу [7]. Надаючи користувацькі інструменти карти та дані про місцезнаходження великим гравцям, таким як Facebook, Snapchat і Foursquare, компанія також враховує потреби місцевих підприємців. І навіть незважаючи на те, що послуги вже не безкоштовні, Mapbox залишається вірним своєму витоку з відкритим кодом, випускаючи код і вносячи внесок у численні картографічні бібліотеки та програми.

Mapbox пропонує повний набір функцій та інструментів для інтеграції своїх картографічних служб у будь-який веб-сайт або мобільний додаток. За допомогою Mapbox Studio клієнти можуть створювати унікальні дизайни, щоб бездоганно поєднувати карти зі своїми продуктами.

Незалежно від того, чи потрібна візуалізація даних, покрокова навігація, контроль логістики чи функція пошуку магазинів, Mapbox дає повну можливість створити карту з правильним виглядом і відчуттям.

Співробітники Mapbox створювали свій продукт, орієнтуючись на налаштування та легку інтеграцію.

Переваги:

- Налаштування та гнучкість. Що стосується дизайну та розробки, то свободу, яку ви отримуєте з Mapbox Studio та Mapbox API, важко перевершити. Додавати об’єкти та вибирати шари для відображення, змінювати кольори та шрифти – усе це можна зробити кількома клацаннями миші, і отримана карта завжди привертає увагу.

- Швидке завантаження та висока продуктивність, особливо з великими масивами даних. Завдяки архітектурі набору фрагментів і

оптимізації Mapbox GL JS ви можете очікувати, що ваша інтегрована карта швидко завантажуватиметься та плавно відтворюватиметься, особливо якщо ви маєте справу зі складними наборами даних.

- Автономний режим в API. На відміну від деяких конкурентів, Mapbox пропонує повну підтримку своїх функцій на офлайн-картах, і немає обмеження на кількість плиток, які можна завантажити. Це надзвичайно корисно, коли неможливо встановити з'єднання для передачі даних або це невиправдано — наприклад, під час міжнародних подорожей, походів або просто коли потрібно оптимізувати швидкість завантаження карти.

- Підхід з відкритим кодом. Mapbox випускає свій код і заохочує спільноту перевіряти та вдосконалювати його. Ще одна прихована перевага полягає у використанні Mapbox GL Native – бібліотеки, яка дозволяє вставляти настроювані інтерактивні карти у рідні програми на iOS та Android.

- Стандартизована обробка даних. Робота з дуже суворими внутрішніми правилами керування даними Mapbox вимагає деякого звикання, але в довгостроковій перспективі це окупається.

Мінуси:

- Погане покриття карти в певних регіонах. Mapbox покладається на колективне картографування, а OpenStreetMap є основним джерелом даних, і це має наслідки. Незважаючи на те, що за останні десять років проект OSM значно зріс, його охоплення в таких регіонах, як Індія та Китай, ще потребує певної роботи.

- Mapbox API потребує певного навчання, і розробникам знадобиться деякий час, щоб навчитися особливостей середовища, зокрема потоку даних і стандартизації. Але це справедливо лише для тих, хто не має попереднього досвіду роботи з Mapbox.

Google Maps

GMP, або платформа Google Maps, може похвалитися покриттям 99% світу та понад 1 мільярдом активних користувачів щомісяця. Це вже давно галузевий стандарт для інтеграції карт, який використовують Bolt, Uber, Allianz та багато інших великих і менших компаній.

Використовуючи парк своїх супутників, автомобілів Street View і пристроїв Android, а також залучаючи місцевих співавторів, GMP надає величезну кількість глибоких і точних даних з оновленнями в реальному часі. А продукти — Карти, Маршрути та Місця — призначені для задоволення потреб будь-якої галузі, якщо бізнес може впоратися з ціною [8].

Існують причини, по яким Google Карти представлені найбільше на платформах і пристроях порівняно з іншими картографічними рішеннями:

- Миттєво впізнається. Більшість користувачів смартфонів і комп'ютерів знайомі з інтерфейсом Google Maps, який, ймовірно, створює відчуття внутрішньої довіри під час взаємодії з версією, вбудованою у вашу програму.

- Відмінна якість глобальних і локальних даних. За роки свого існування Google Maps завоювала безпрецедентну територію своїми картографічними службами та зібрала незбагненні обсяги інформації на місцевому рівні.

- Багатомовна підтримка. Наразі GMP підтримує понад 80 мов, і список постійно зростає.

- Перегляд вулиць. Ця функція, додану до мобільної версії Карт Google у вересні 2019 року, є унікальною пропозицією, якої немає в інших картографічних службах.

Мінуси:

- Кілька варіантів налаштування. Хоча нові спеціальні стилі розгортаються як бета-версія, Google Maps API наразі підтримує обмежені параметри для створення унікального вигляду ваших інтегрованих карт.

- Не є API з відкритим кодом.

- Непередбачувані зміни цін. Зростання цін на API Google Maps у 2018 році викликало хвилю обурення серед лояльних користувачів. Така відмінна послуга заслуговує на монетизацію та приносить прибуток, але довіра клієнтів — валюта, яку варто заощаджувати.

OpenStreetMap

OpenStreetMap – це проект спільноти, який надає численним веб-сайтам та програмам свої картографічні дані. Будучи картою з відкритим кодом, вона абсолютно безкоштовна для використання, але зберігає високий рівень точності та деталізації завдяки зусиллям місцевих ентузіастів карт та інженерів, які наповнюють її даними та підтримують її [9]. Насправді, навіть Mapbox використовує OSM як основу для своїх карт.

Переваги:

- API OpenStreetMap безкоштовний. Немає додаткових витрат для бізнесу.

- Карта з відкритим кодом. Велика кількість учасників, які захоплюються картографуванням, забезпечує постійне зростання бази даних.

- Мінуси:

- Вимагає створення додаткових сервісів. API було розроблено з метою оновлення та редагування карт і має базову функціональність. Доведеться або створити необхідну інфраструктуру на місці, або використовувати готові універсальні рішення (наприклад, Mapbox), які базуються на даних OSM.

- Обмежена кількість запитів. Через характер проекту надмірний обмін даними через API не вітається, і користувачі можуть бути заблоковані без попередження, якщо надсилають занадто багато запитів даних.

Хоча Google Maps і Mapbox використовують складні та часто подібні моделі ціноутворення, існує одна важлива відмінність у підході до виставлення рахунків: Google вимірює використання виключно в запитах, а Mapbox має опцію обліку окремих користувачів.

Залежно від вашого сценарію використання та методів, застосованих для інтеграції карт у вашу програму, Mapbox пропонує різні варіанти ціноутворення на вибір, тоді як Карти Google мають уніфікований підхід ціноутворення, який зосереджується на завантаженні сервера.

Обидві компанії надають щомісячні знижки: Google у вигляді кредиту в розмірі 200 доларів США, а Mapbox – на взаємодію з користувачем або запити API, з безкоштовними до 750 000 викликів API у деяких категоріях.

Оскільки API OpenStreetMap має дуже обмежену автономну функціональність для інтеграції з іншими програмами, краще звернути увагу на спільні риси Google Maps і Mapbox:

- Основні функції карти. Статичні та динамічні карти, точні та глибокі географічні та місцеві дані в різних режимах перегляду, можливість додавати маркери, полігони та зображення.

- Навігація. Напрямки в режимі реального часу з урахуванням трафіку, оптимізації маршруту та оцінки часу прибуття.

- Місцева інформація. Детальна інформація про місця та цікаві місця.

- Розширені параметри пошуку. Автоматичне передбачення та виправлення, пропозиції на основі розташування.

Основні відмінності між цими трьома API карт можна виразити через такі характеристики:

- Налаштування. Mapbox лідирує, Google Maps має трохи нижчий бал, а OSM на останньому місці.

- Операційні витрати. OpenStreetMap переконливо перемагає в цьому. З двома іншими вам потрібно бути дуже обережними, порівнюючи витрати для конкретного випадку використання.

- Легкість інтеграції. Це значною мірою залежатиме від досвіду роботи дизайнерів і розробників із відповідними API та їхніми SDK.

3.5 Основні принципи роботи зі службами визначення місцезнаходження

Процес розробки мобільних додатків із місцезнаходженням і картографуванням в Android можна розбити на кілька ключових етапів:

1. Визначення вимог до програми. Першим кроком у розробці програми з місцезнаходженням і картою є визначення вимог до програми. Це включає визначення типів послуг на основі місцезнаходження, які надаватиме додаток, наприклад геолокації, маршрутизації або локального пошуку.

2. Вибір правильних інструментів: після визначення вимог до програми потрібно вибрати правильні інструменти та API для впровадження цих функцій. Це може включати використання диспетчера розташування, геокодера та API карт, а також бібліотек і служб сторонніх розробників.

3. Розробка інтерфейсу користувача: інтерфейс користувача (UI) програми має вирішальне значення для її успіху. Розробникам потрібно розробити інтуїтивно зрозумілий і простий у користуванні інтерфейс користувача, який водночас забезпечує доступ до всіх функцій додатка на основі визначення місцезнаходження.

4. Впровадження функцій додатка: маючи вимоги до додатка, інструменти та інтерфейс користувача, розробники можуть розпочати впровадження функцій додатка на основі визначення місцезнаходження. Це може включати написання коду для доступу до диспетчера розташування, геокодера та API карт, а також інтеграцію сторонніх служб і бібліотек.

5. Перевірка програми: після впровадження програми її потрібно ретельно протестувати та налагодити, щоб переконатися, що вона працює належним чином. Це включає тестування функцій програми на основі визначення місцезнаходження в різних сценаріях реального світу.

6. Опублікування програми: її можна опублікувати в Google Play Store або інших магазинах програм, щоб користувачі могли завантажити та використовувати.

Розробляючи додатки зі службами визначення місцезнаходження та картографування, важливо дотримуватися найкращих практик, щоб забезпечити надійність і ефективність програми:

1. Тестування програми на кількох пристроях і в різних місцях, щоб переконатися, що вона працює правильно.

2. Використання служб визначення місцезнаходження лише за необхідності, оскільки вони можуть швидко розрядити батарею пристрою. Це може включати використання служб визначення місцезнаходження на основі мережі замість служб на основі GPS.

3. Використання кешування та багатопоточності для покращення продуктивності.

4. Належна обробка помилок для обробки неочікуваних ситуацій.

5. Надання чіткої політики конфіденційності. Важливо надати прозору політику конфіденційності, яка пояснює, як програма використовує дані про місцезнаходження та як ці дані захищені.

6. Надання чітких і лаконічних інструкцій щодо використання функцій визначення місцезнаходження та картографування програми.

Програми на основі визначення місцезнаходження стають все більш популярними серед користувачів. Платформа Android надає різні служби визначення місцезнаходження та картографічні служби, які розробники можуть використовувати для розробки додатків на основі визначення місця розташування. Розробники повинні враховувати різні чинники, такі як інтерфейс користувача, продуктивність і безпека під час розробки додатків на

основі визначення місцезнаходження. Враховуючи ці фактори, розробники можуть розробляти програми на основі визначення місцезнаходження, які забезпечують користувачам бездоганний досвід.

4 ПРОЕКТУВАННЯ МОБІЛЬНОГО ЗАСТОСУНКУ

4.1 Діаграма прецедентів

Діаграми прецедентів є одним з основних інструментів моделювання поведінки системи. Це діаграмне представлення взаємодії між користувачем і системою. Діаграми варіантів використання дозволяють описати функціональність системи в термінах дій, які можуть бути виконані користувачами та іншими системами.

Діаграма прецедентів складається з акторів і прецедентів. Актор - це будь-який зовнішній або внутрішній об'єкт, який взаємодіє з системою. Прецеденти - це описи дій, які виконує система для задоволення потреб користувачів та інших систем. Кожен прецедент пов'язаний з одним або декількома акторами, які його запускають [10].

Діаграми прецедентів допомагають розробникам уточнити вимоги користувачів і визначити функціональність системи. Розробники можуть легко візуалізувати взаємодію між користувачами та системою і легко визначити антецеденти, які необхідно реалізувати в системі.

Ключові елементи діаграми прецедентів:

- Актор - особа, група або система, що взаємодіє з системою.
- Прецедент - дія, що виконується системою для задоволення потреб користувача або іншої системи.
- Зв'язок актор-прецедент - показує взаємодію між акторами та прецедентами.

Система є загальним контейнером для всіх акторів і прецедентів.

Переваги використання діаграм варіантів використання:

- Допомагають визначити вимоги користувача та функціональність системи.

- Дозволяють легко візуалізувати взаємодію між користувачами та системою.

- Допомагають зрозуміти потреби користувачів і визначити, як система повинна взаємодіяти з користувачем.

- Фокусуються на функціональності системи і можуть прояснити, які прецеденти необхідно реалізувати.

- Легко зрозумілі для всіх зацікавлених сторін, включаючи розробників, менеджерів і користувачів.

В даному випадку акторами є користувач та адміністратор. Варіантами використання є, з боку користувача: перегляд мапи небезпечних об’єктів, перегляд інформації про знайдені об’єкти, додавання інформації про знайдений небезпечний об’єкт, налаштування сповіщень; адміністратора - перевірка інформації про додані об’єкти.

Розроблена діаграма варіантів використання зображена на рис. 4.1.

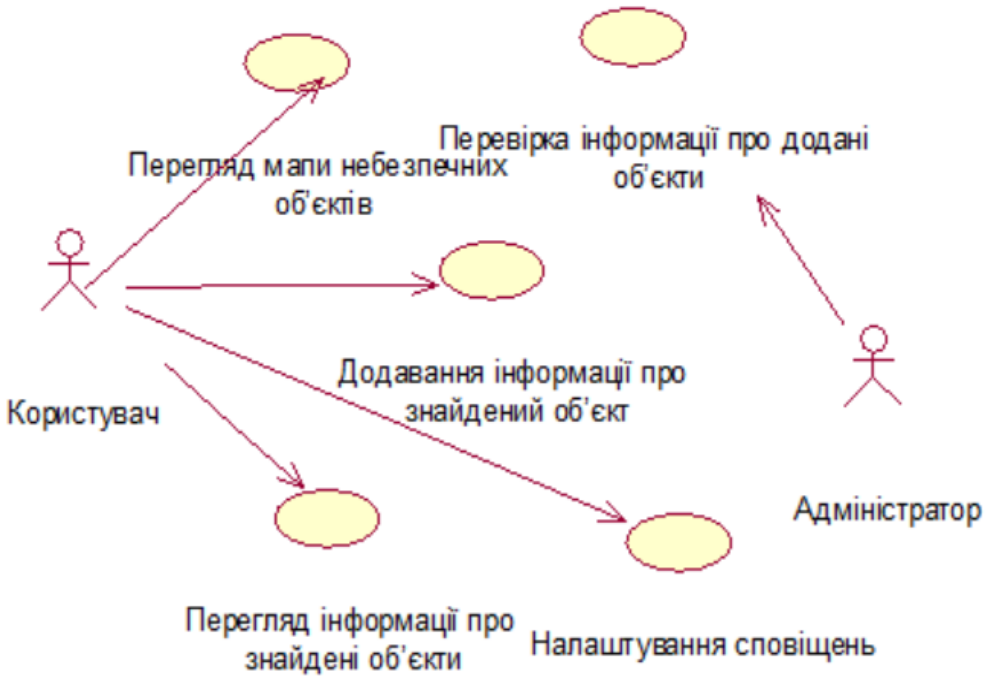

Рисунок 4.1 – Діаграма прецедентів

5 РЕАЛІЗАЦІЯ МОБІЛЬНОГО ЗАСТОСУНКУ

5.1. Використання бібліотеки osmdroid

Для роботи з картами OpenStreetMap було вирішено використовувати бібліотеку osmdroid.

Бібліотека osmdroid є популярним інструментом для розробки картографічних застосунків в середовищі Android. osmdroid базується на данних OpenStreetMap, що є відкритою та безкоштовною картографічною базою даних. Це дає доступ до широкого спектру географічних даних, які можна використовувати в застосунку без обмежень. osmdroid підтримує можливість завантаження карт і використання їх в офлайн режимі. Можна попередньо завантажити карти та використовувати їх без доступу до Інтернету, що дуже корисно у випадку обмеженої мережевої доступності або зменшення використання мобільних даних.

Також можна додавати різні шари на карту, такі як маркери, полігони, лінії, тайлові шари тощо. Це дозволяє створювати різноманітні картографічні візуалізації та інтерактивні функції.

osmdroid надає зручні можливості для обробки жестів та подій на карті, таких як зумування, перетягування, кліки тощо. Це дозволяє створювати інтерактивні застосунки з мультитач-жестами та навігаційними функціями. Також бібліотека надає гнучкі можливості для налаштування та стилізації карт. Можна змінювати розміри, кольори, типи шрифтів, шари тощо. osmdroid також дозволяє підключатися до різних тайлових серверів, що надають карти. Можна використовувати власний сервер або використовувати публічні тайлові сервіси, такі як MapQuest, Mapbox, CloudMade тощо.

osmdroid має велику та активну спільноту розробників, яка надає підтримку, відповідає на питання та вносить внески до подальшого розвитку бібліотеки [11].

Використання бібліотеки osmdroid дозволяє ефективно розробляти картографічні застосунки для Android, використовуючи безкоштовні дані OpenStreetMap та отримуючи доступ до різноманітних функцій та можливостей.

Для використання бібліотеки osmdroid у більшості випадків у файлі AndroidManifest.xml необхідно встановити наступні дозволи:

```
    <uses-permission
android:name="android.permission.ACCESS_FINE_LOCATION"/>
    <uses-permission  android:name="android.permission.INTERNET"
/>
    <uses-permission
android:name="android.permission.ACCESS_NETWORK_STATE"  />
    <uses-permission
android:name="android.permission.WRITE_EXTERNAL_STORAGE" />
```

В макеті сторінки застосунку, наприклад, у файлі src/main/res/layouts/activity_main.xml, необхідно додати елемент розмітки org.osmdroid.views.MapView:

```
    <org.osmdroid.views.MapView
        android:id="@+id/map"
        android:layout_width="fill_parent"
        android:layout_height="fill_parent" />
```

У відповідному класі (наприклад, MainActivity.java) необхідно прописати отримання дозволів та завантаження мапи.

```
    import static
android.Manifest.permission.ACCESS_FINE_LOCATION;
import static
android.Manifest.permission.WRITE_EXTERNAL_STORAGE;

public class MainActivity extends AppCompatActivity {

    private final int REQUEST_PERMISSIONS_REQUEST_CODE = 1;
    private MapView map = null;
```

```
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);

        Context ctx = getApplicationContext();
        Configuration.getInstance().load(ctx,
PreferenceManager.getDefaultSharedPreferences(ctx));

        setContentView(R.layout.activity_main);
        map = (MapView) findViewById(R.id.map);
             map.setTileSource(TileSourceFactory.MAPNIK);

        String[] perm = new String[]{ACCESS_FINE_LOCATION,
WRITE_EXTERNAL_STORAGE};
        requestPermissionsIfNecessary(perm);
    }

    @Override
    public void onResume() {
        super.onResume();
        map.onResume();
    }

    @Override
    public void onPause() {
        super.onPause();
        map.onPause();
    }

    @Override
    public void onRequestPermissionsResult(int requestCode,
@NotNull String[] permissions, @NotNull int[] grantResults) {
        super.onRequestPermissionsResult(requestCode,
permissions, grantResults);
```

```
        ArrayList<String> permissionsToRequest = new
ArrayList<>(Arrays.asList(permissions).subList(0,
grantResults.length));
        if (permissionsToRequest.size() > 0) {
            ActivityCompat.requestPermissions(
                    this,
                    permissionsToRequest.toArray(new String[0]),
                    REQUEST_PERMISSIONS_REQUEST_CODE);
        }
    }

    private void requestPermissionsIfNecessary(String[]
permissions) {
        ArrayList<String> permissionsToRequest = new
ArrayList<>();
        for (String permission : permissions) {
            if (this.checkSelfPermission(permission)!=
PackageManager.PERMISSION_GRANTED) {
                permissionsToRequest.add(permission);
            }
        }
        if (permissionsToRequest.size() > 0) {
            ActivityCompat.requestPermissions(
                    this,
                    permissionsToRequest.toArray(new String[0]),
                    REQUEST_PERMISSIONS_REQUEST_CODE);
        }
    }
}
```

Далі можна налаштувати відображення мапи та додати маркери.

```
map = findViewById(R.id.map);
map.setTileSource(TileSourceFactory.MAPNIK);
map.setBuiltInZoomControls(true);
map.setMultiTouchControls(true);
IMapController mapController = map.getController();
```

```
        mapController.setZoom(6.5);
        GeoPoint startPoint = new GeoPoint(48.89713,30.7148845);
        mapController.setCenter(startPoint);
        adiusMarkerClusterer poiMarkers = new
        RadiusMarkerClusterer(this);
        ArrayList<GeoPoint> obj = new ArrayList<GeoPoint>(){};
        // ...
        //додавання міток за координатами у базі
        map.getOverlays().add(poiMarkers);
        Drawable poiIcon =
getResources().getDrawable(R.drawable.outline_warning_amber_24);
        for (GeoPoint point:obj){
              Marker poiMarker = new Marker(map);
              poiMarker.setPosition(point);
              poiMarker.setIcon(poiIcon);
              poiMarkers.add(poiMarker);
        }
        map.invalidate();
```

За допомогою цього коду для об’єкта MapView з розмітки активності встановлюється джерело плиток (тайлів) для картографічного шару. У даному випадку використовується стандартне джерело плиток Mapnik, яке надає OpenStreetMap. Включаються вбудовані елементи керування масштабуванням на карті та можливість збільшувати, зменшувати та перетягувати карту за допомогою декількох пальців. Встановлюється початковий центр карти на задану початкову точку GeoPoint (координати приблизного центра України). Створюється та налаштовується об'єкт для кластеризації маркерів на карті.

5.2 Використання Firebase

Firebase Cloud – це комплексний набір хмарних інструментів і служб, які надає Google. Він пропонує розробникам широкий спектр функцій і

можливостей, які допоможуть швидко й ефективно створювати та масштабувати їхні мобільні та веб-додатки. Одним із основних компонентів Firebase Cloud є Firebase Realtime Database, яка спеціально розроблена для додатків Android.

Firebase Realtime Database — це хмарна база даних NoSQL, яка дозволяє розробникам зберігати та синхронізувати дані в режимі реального часу між кількома клієнтами. Вона надає гнучке та масштабоване рішення для керування даними програми та дозволяє безперебійну співпрацю між користувачами [12].

Використання Firebase як бази даних у додатку для Android має кілька переваг:

Синхронізація в реальному часі: Firebase забезпечує синхронізацію даних у реальному часі, що означає, що будь-які зміни, внесені в базу даних, миттєво відображаються на всіх підключених пристроях. Ця функція особливо корисна для додатків, які потребують оновлення в режимі реального часу, таких як програми для чату або інструменти для спільної роботи.

Підтримка в автономному режимі: Firebase має вбудовану підтримку в автономному режимі, що дозволяє користувачам отримувати доступ до даних і змінювати їх навіть у режимі офлайн. Коли пристрій повторно підключається до Інтернету, Firebase автоматично синхронізує локальні зміни з сервером, забезпечуючи узгодженість даних.

Масштабованість: Firebase — це хмарна база даних, що означає, що вона може легко обробляти великі обсяги даних і розраховувати на зростаючу базу користувачів. Він автоматично масштабується відповідно до вимог вашої програми, забезпечуючи плавну роботу та мінімальний час простою.

Легка інтеграція: Firebase надає повний набір API і SDK, які спрощують процес інтеграції з програмами Android. Він пропонує рідні бібліотеки для Android, що спрощує реалізацію функціональності Firebase і використання її функцій без тривалого програмування чи складних конфігурацій.

Автентифікація та безпека: Firebase пропонує вбудовані служби автентифікації, що дозволяє автентифікувати користувачів за допомогою різних методів, таких як електронна пошта/пароль, вхід із соціальних мереж (наприклад, Google, Facebook) тощо. Він також забезпечує надійні правила безпеки та механізми для контролю доступу до даних і захисту конфіденційної інформації.

Безсерверна архітектура: Firebase піклується про серверну інфраструктуру, обслуговування та масштабування, усуваючи потребу розробникам керувати серверами. Ця безсерверна архітектура дозволяє розробникам більше зосереджуватися на розробці додатків і зменшує операційні витрати.

Аналітика та моніторинг продуктивності: Firebase надає потужні інструменти аналітики для відстеження поведінки користувачів, вимірювання продуктивності програми та отримання цінної інформації. Він пропонує докладні звіти про залучення користувачів, утримання, звіти про збої та моніторинг продуктивності, що дозволяє розробникам оптимізувати свої програми на основі рішень, керованих даними.

Firebase використовує NoSQL базу даних. NoSQL (Not Only SQL) база даних - це тип бази даних, який відрізняється від традиційних реляційних баз даних тим, що не використовує структуровану мову запитів SQL і не використовує схему фіксованих таблиць зі зв'язками між ними.

Основні риси NoSQL баз даних:

Гнучкість у роботі з даними: NoSQL бази даних дозволяють зберігати і обробляти різноманітні типи даних, включаючи структуровані, напівструктуровані і невструктуровані дані. Це означає, що ви можете зберігати дані різної структури без потреби заздалегідь визначати схему таблиць.

Горизонтальне масштабування: NoSQL бази даних розроблені з орієнтацією на горизонтальне масштабування, що означає здатність розподіляти дані на кілька серверів для забезпечення високої доступності та

швидкодії. Це робить їх популярними в розподілених системах з великим обсягом даних.

Простота швидкого збереження та отримання даних: NoSQL бази даних зазвичай пропонують простий API для збереження та отримання даних, що спрощує розробку програм, особливо тих, що працюють з великим обсягом даних.

Гнучкість схеми: У NoSQL базах даних немає фіксованої схеми, що дозволяє додавати, змінювати та видаляти поля без необхідності проводити масштабні зміни схеми бази даних. Це робить їх більш гнучкими для розробки додатків, які вимагають частих змін у структурі даних.

Підтримка розподіленості: NoSQL бази даних добре підходять для розподілених середовищ, оскільки вони зазвичай підтримують реплікацію та шарування даних. Це дозволяє забезпечити високу доступність та надійність системи, а також підвищити продуктивність шляхом розподілу навантаження на кілька серверів.

Щоб використовувати Firebase Realtime Database у програмі Android, необхідно виконати такі дії:

Створити проект Firebase: перейти до консолі Firebase (console.firebase.google.com) і створити новий проект. Дати йому назву та вказати ім’я пакета програми.

Налаштувати застосунок: зареєструвати застосунок у Firebase, надавши назву пакета та інші відомості. Завантажити файл google-services.json і помістити його в папку модуля програми.

Додати Firebase SDK: у файл build.gradle застосунка додати залежності Firebase SDK для Realtime Database та синхронізувати проект після додавання залежностей.

Якщо програма вимагає автентифікації користувача, Firebase надає різні методи автентифікації, такі як електронна пошта/пароль, вхід у Google тощо.

Ініціалізувати Firebase у точці входу програми (наприклад, MainActivity) за допомогою методу FirebaseApp.initializeApp().

Використовувати Firebase Realtime Database API для взаємодії з базою даних. Читати та записувати дані можна за допомогою класу DatabaseReference. Наприклад, щоб записати дані, можна викликати setValue() для об’єкта DatabaseReference, а щоб прочитати дані, можна використовувати метод addValueEventListener().

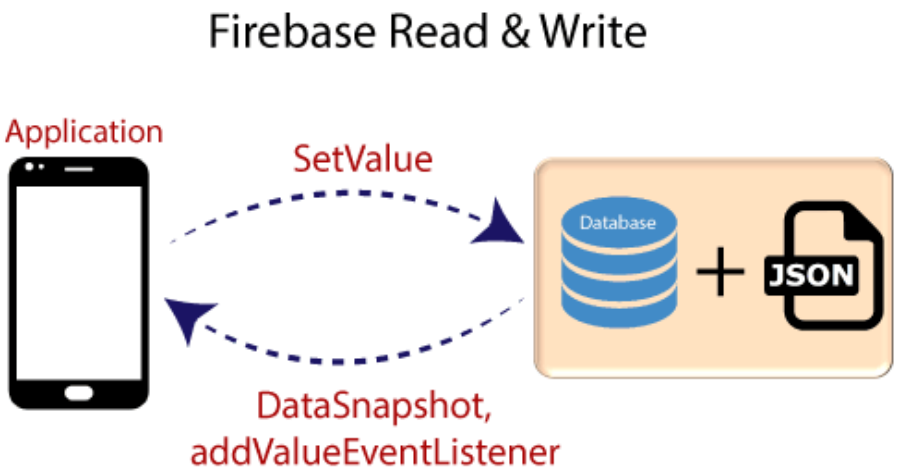

Рисунок 5.1 – Читання та запис даних у Firebase Realtime Database

Обробляти оновлення в реальному часі: Firebase Realtime Database забезпечує синхронізацію в реальному часі, дозволяючи отримувати оновлення щоразу, коли змінюються дані. Можна відстежувати зміни даних за допомогою різних прослуховувачів, таких як ValueEventListener або ChildEventListener, і відповідно оновлювати інтерфейс користувача.

Firebase дозволяє визначати правила безпеки, щоб обмежити доступ до бази даних. Можна визначити правила на основі автентифікації користувача, перевірки даних та інших умов для забезпечення безпеки та цілісності даних.

Firebase Realtime Database спрощує процес керування та синхронізації даних у програмах Android, надаючи потужне та масштабоване серверне рішення. Крім того, Firebase Cloud пропонує кілька інших служб, як-от

Firebase Cloud Messaging, Firebase Authentication, Firebase Storage тощо, які можна інтегрувати для покращення функціональності вашої програми.

5.3 Опис та тестування застосунку

Іконка застосунку зображена на рис. 5.1. При першому запуску або використанні функціональності, яка вимагає доступ до геолокації, Android показує діалогове вікно, в якому користувачеві пропонується надати або відхилити дозвіл. Можна вибрати точне або приблизне місцезнаходження. Для роботи застосунку потрібне підключення до мережі Інтернет та ввімкнена геолокація для коректного відображення мапи. Робота з застосунком починається з головної сторінки (рис. 5.2). На даній сторінці користувач може переглянути небезпечні об'єкти, які додані до бази. Ступінь кластеризації об'єктів залежить від масштабування карти. При натисканні на кожен об'єкт, можна переглянути детальну інформацію про нього.

Рисунок 5.1 – Іконка застосунку «Danger Alerts»

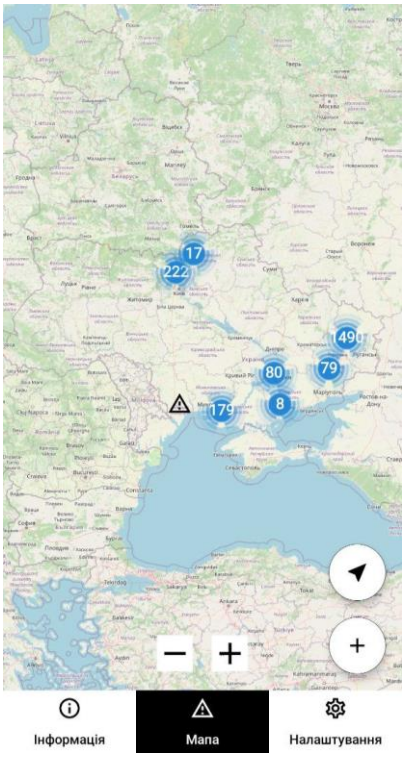

Рисунок 5.2 – Головна сторінка застосунку «Danger Alerts»

Рисунок 5.3 – Сторінка додавання нового об’єкта

Масштабування виконується або жестами, або за допомогою кнопок «+» і «-» на мапі. Кнопка «Моє місцезнаходження» центрує карту відносно положення користувача. Натиснувши на круглу кнопку «+», користувач може додати відомості про новий знайдений об’єкт. Для цього він повинен вказати всю потрібну інформацію на сторінці додавання об’єкта (рис. 5.3).

З головної сторінки користувач також може перейти до сторінки «Інформація» (рис. 5.4). Ця сторінка застосунку містить список видів небезпечних об'єктів, які можна знайти в результаті воєнних дій. Вона надає користувачам важливу інформацію про різні типи об'єктів, які можуть представляти загрозу після завершення бойових дій, такі як авіаційні бомби, ракети, артилерійські снаряди, балістичні ракети, дрони, гранати, мінометні, протипіхотні та протитанкові міни, ракети тощо. Вона сприяє поширенню обізнаності та забезпеченню безпеки при взаємодії з такими об'єктами, а також показує необхідні заходи обережності та захисту в умовах післяконфліктних територій.

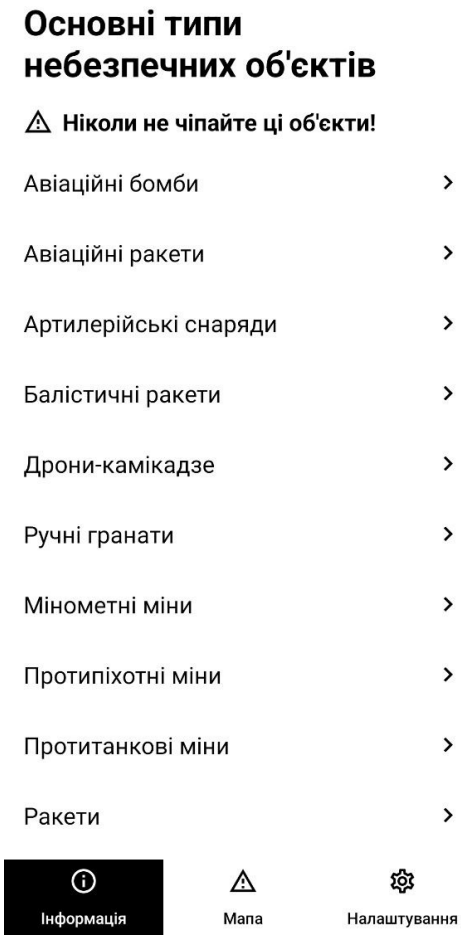

Рисунок 5.4 – Сторінка інформації про основні типи небезпечних об’єктів

На сторінці «Налаштування» користувач може авторизуватися: створити новий профіль або увійти до існуючого (рис. 5.5). Після цього він може повідомляти про нові об’єкти. Також на цій сторінці є можливість вказати бажаний радіус в кілометрах, в межах якого на пристрій будуть приходити сповіщення про знаходження нових небезпечних об’єктів. Для коректної роботи цієї функції на пристрої повинна бути ввімкнена геолокація та інтернет.

Рисунок 5.5 – Сторінка налаштувань застосунку

Після проведення тестування застосунку «Danger Alerts», він виявився зручним у використанні та мав зрозумілий інтерфейс для користувачів. Тестування пройшло без помилок, що свідчить про високу якість розробки та стабільність застосунку.

Користувачі легко знаходили потрібну інформацію про види небезпечних об'єктів. Застосунок надає детальні описи і характеристики кожного об'єкта, що дозволяє користувачам отримати необхідні знання для безпечного поводження з ними.

Застосунок має чітку структуру, що дозволяє користувачам швидко зорієнтуватися та знайти потрібну інформацію. Графічний дизайн із зручним

шрифтом та мінімалістичною кольоровою схемою також сприяє зручності використання застосунку.

Загальний результат тестування підтвердив, що застосунок «Danger Alerts» є надійним, зручним та ефективним джерелом інформації про небезпечні об'єкти під час та після воєнних дій. Він допомагає користувачам бути обізнаними та усвідомленими щодо потенційних небезпек і надає необхідні поради щодо безпечного поводження з цими об'єктами.

ВИСНОВКИ

У результаті даної кваліфікаційної роботи був розроблений застосунок на платформі Android, який дозволяє користувачам отримувати сповіщення про знаходження небезпечних об'єктів на території України.

Аналіз предметної області показав, що забруднення території України небезпечними об'єктами стало актуальною проблемою в результаті воєнного конфлікту. Це вимагає розробки ефективного інструменту для сповіщення користувачів про потенційні небезпеки у їхньому оточенні.

Обґрунтування вибору програмних засобів розробки та технологій було зроблено на основі їхньої популярності, широких можливостей та підтримки спільнотою розробників. Firebase було обрано для роботи з хмарною базою даних, оскільки воно забезпечує зручний доступ до даних з будь-якого пристрою та має високу масштабованість. Бібліотека osmdroid була обрана для роботи з мапою від OSM через свою гнучкість та можливість налаштування.

Проектування системи було проведено з урахуванням функціональних та нефункціональних вимог. Була розроблена структура бази даних, а також визначено основні компоненти застосунку. Реалізація застосунку була проведена з використанням зазначених програмних засобів. Було проведено тестування застосунку для перевірки правильності його роботи, а також для виявлення та усунення можливих помилок та недоліків.

Використання хмарної бази даних Firebase та мапи від OSM за допомогою бібліотеки osmdroid забезпечує зручний доступ до актуальних даних та точне відображення місцезнаходження об'єктів.

Застосунок успішно виконує свої завдання і може бути використаний для сповіщення користувачів про потенційні небезпеки, допомагаючи покращити безпеку та свідомість українців щодо проблеми забруднення країни небезпечними об'єктами.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Смертельні сюрпризи війни: як мінування території України впливає на екологію. URL: https://eco.rayon.in.ua/topics/550513-smertelni-syurprizi-viyni-yak-minuvannya-teritorii-ukraini-vplivae-na-ekologiyu (дата звернення 25.04.2023)

2. MineFree 2.0. https://www.minefree.info (дата звернення 26.04.2023)

3. Інтерактивна мапа територій, які потенційно можуть бути забруднені вибухонебезпечними предметами. https://mine.dsns.gov.ua (дата звернення 02.05.2023)

4. Android Studio. https://developer.android.com/studio (дата звернення 08.05.2023)

5. What is Geolocation? https://www.gravitatedesign.com/blog/what-is-geolocation (дата звернення 08.05.2023)

6. Location-Based Services. https://www.heavy.ai/technical-glossary/location-based-services (дата звернення 08.05.2023)

7. Maps and location for developers. URL: https://www.mapbox.com (дата звернення 09.05.2023)

8. Getting started with Google Maps Platform. https://developers.google.com/maps/get-started (дата звернення 09.05.2023)

9. OpenStreetMap API. https://wiki.openstreetmap.org/wiki/API (дата звернення 11.05.2023)

10. Діаграма прецедентів (Вікіпедія). https://uk.wikipedia.org/wiki/Діаграма_прецедентів (дата звернення 17.05.2023)

11. OpenStreetMap-Tools for Android. https://github.com/osmdroid/osmdroid (дата звернення 13.05.2023)

12. Firebase Realtime Database. https://firebase.google.com/docs/database (дата звернення 14.05.2023)