МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
СХІДНОУКРАЇНСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ІМ. В. ДАЛЯ
ФАКУЛЬТЕТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ ТА ЕЛЕКТРОНІКИ
КАФЕДРА КОМП'ЮТЕРНИХ НАУК ТА ІНЖЕНЕРІЇ

До захисту допускається
Т.в.о. завідувача кафедри
_____ Сафонова С.О.
«____»_____ 2020р.

МАГІСТЕРСЬКА РОБОТА

НА ТЕМУ:

ДОСЛІДЖЕННЯ І РОЗРОБКА ANDROID-ДОДАТКІВ НАВІГАЦІЇ ПО МІСТУ

Освітньо-кваліфікаційний рівень «Магістр»
Спеціальність 122 «Комп'ютерні науки»

Науковий керівник роботи:	_____ (підпис)	Є.В. Щербаков (ініціали, прізвище)
Консультант з охорони праці:	_____ (підпис)	Я.О. Критська (ініціали, прізвище)
Студент:	_____ (підпис)	О.І. Федоряченко (ініціали, прізвище)
Група:		КН-18дм

Сєвєродонецьк 2020

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
СХІДНОУКРАЇНСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
ІМЕНІ ВОЛОДИМИРА ДАЛЯ

Факультет Інформаційних технологій та електроніки
Кафедра Комп'ютерних наук та інженерії
Освітньо-кваліфікаційний рівень магістр
Напрям підготовки _____
(шифр і назва)
Спеціальність 122 Комп'ютерні науки
(шифр і назва)

ЗАТВЕРДЖУЮ:

Т.в.о завідувача
кафедри _____
_____ С.О. Сафонова
«_____» _____20____р.

ЗАВДАННЯ
НА МАГІСТЕРСЬКУ РОБОТУ СТУДЕНТУ

Федоряченку Олексію Ігоровичу
(прізвище, ім'я, по батькові)

1. Тема роботи Дослідження і розробка Android-додатків навігації по місту

керівник проекту (роботи) Щербаков Євген Васильович, к.т.н., доцент
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

затверджені наказом вищого навчального закладу від " 11 " 10 2019 року № 135/15.15

2. Строк подання студентом проекту (роботи) 10.01.2020

3. Вихідні дані до роботи Матеріали науково-дослідної практики, мова програмування – Java, XML, середовище розробки – Android Studio

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити)
Теорико-методологічні аспекти
Методи і засоби навігації по місту
Аналіз методів і засобів оптимізації навігації по місту
Аналіз та дослідження існуючих додатків-навігаторів
Практична реалізація додатка-навігатора
Охорона праці та безпека в надзвичайних ситуаціях

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень)
Електронні плакати

6. Консультанти розділів проекту (роботи)

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Охорона праці	Критська Я.О. асистент		

7. Дата видачі завдання 02.09.2019

Керівник _____
(підпис)

Завдання прийняв до виконання _____
(підпис)

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломного проекту (роботи)	Строк виконання етапів проекту (роботи)	Примітка
1	Аналітичний огляд літератури за темою роботи	02.09.19-30.09.19	
2	Аналіз методів і засобів оптимізації навігації по місту	01.10.19-13.10.19	
3	Аналіз та дослідження існуючих додатків-навігаторів	14.10.19-27.10.19	
4	Дослідження методів розробки додатка-навігатора	28.10.19-10.11.19	
5	Практична реалізація додатка-навігатора	11.11.19-01.11.19	
6	Розробка заходів з охороні праці	02.12.19-22.12.19	
7	Оформлення пояснювальної записки	23.12.19-02.01.20	
8	Оформлення презентації роботи	03.01.20-10.01.20	

Студент _____ Федоряченко О.І.
(підпис) (прізвище та ініціали)

Науковий керівник _____ Щербаков Є.В.
(підпис) (прізвище та ініціали)

АНОТАЦІЯ

Федоряченко О.І. Дослідження і розробка Android-додатків навігації по місту.

Метою дипломної роботи є дослідження і розробка Android-додатка, який буде використовуватися в ролі зручного помічника при позиціонуванні в місті.

Розглянуто та досліджено методи і засоби оптимізації навігації по місту. Проведено аналіз та дослідження існуючих додатків-навігаторів, гідів окремих міст. Досліджені методи розробки додатків-навігаторів за допомогою мови розмітки XML, мови програмування Java, та інтегрованої системи розробки програм Android Studio. В результаті досліджень був розроблений програмний додаток-навігатор для мобільних пристроїв під управлінням ОС Android, який виконує функцію гіда по місту Сєвєродонецьк, працює з мапою Google, та сервісами Google API, має функцію швидкого пошуку на мапі, містить інформацію про заклади на мапі, прокладає маршрут до обраного місця, демонструє оточення користувача, показує прогноз погоди, інформацію про місто.

Ключові слова: Android, XML, Java, додаток, Google

АННОТАЦИЯ

Федоряченко А.И. Исследование и разработка Android-приложений навигации по городу.

Целью дипломной работы является исследование и разработка Android-приложения, которое будет использоваться в роли удобного помощника при позиционировании в городе.

Рассмотрены и исследованы методы и средства оптимизации навигации по городу. Проведен анализ и исследование существующих приложений-навигаторов, гидов отдельных городов. Исследованы методы разработки приложений-навигаторов с помощью языка разметки XML, языка программирования Java и интегрированной системы разработки приложений Android Studio. В результате исследований было разработано программное приложение-навигатор для мобильных устройств под управлением ОС Android, который выполняет фунцию гида по городу Северодонецк, работает с картой Google и сервисами Google API, имеет функцию быстрого поиска на карте, содержит информацию о заведениях на карте, прокладывает маршрут до выбранного места, демонстрирует окружение пользователя, показывает прогноз погоды, информацию о городе.

Ключевые слова: Android, XML, Java, приложение, Google

ABSTRACT

Fedoriachenko O. I. Research and development of Android applications for city navigation.

The purpose of the thesis is to research and develop an Android application that will be used as a convenient assistant when positioning in the city.

Methods and means of optimization of navigation in the city are considered and investigated. The analysis and research of existing applications-navigators, guides of individual cities. Methods for developing navigation applications using the XML markup language, the Java programming language, and the Android Studio integrated application development system are investigated. As a result of research, we developed a software application-Navigator for mobile devices running Android OS, which performs the function of a guide to the city of Severodonetsk, works with Google map and Google API services, has a quick search function on the map, contains information about institutions on the map, plots a route to the selected location, demonstrates the user's environment, shows the weather forecast, information about the city.

Keywords: Android, XML, Java, app, Google

ЗМІСТ

ВСТУП

Обґрунтування вибору теми дослідження. Ми живемо в такий час, коли майже у всіх людей є мобільний пристрій під управлінням ОС Android або IOS. Всі власники таких пристроїв користуються різноманітними додатками. Це дуже зручно, мати доступ до всього необхідного в одному пристрої. Додаток-навігатор з використанням мап - це один з найнеобхідніших додатків для кожного користувача. Він значно заощаджує час, якщо користувач знаходиться десь в дорозі, і йому треба потрапити туди де він ще ніколи не був. В такій ситуації це просто незамінний додаток. Також якщо у користувача є додаток навігатор, він може поділитися своїм місцезнаходженням з іншими людьми, і вони зможуть швидко зрозуміти де він, та за необхідності відшукати його.

Розробка Android-додатків навігації по місту дуже актуальна. Додаток повинен мати простий та зрозумілий інтерфейс, який не відштовхне потенційних нових користувачів, та виконувати всі основні функції навігації, такі як: прокладання маршруту до обраної точки на мапі, показувати оточення користувача, надавати інформацію про оточуючі об’єкти, показувати місцезнаходження та інше.

Тому обґрунтованою є тема магістерської роботи, у якій вирішується **науково-прикладне завдання** розроблення Android-додатків навігації по місту.

Об'єкт дослідження – проблеми навігації на місцевості.

Предмет дослідження – процеси забезпечення навігації по місту, за допомогою додатка-навігатора на ОС Android з використанням мови розмітки XML та програмування Java.

Мета і завдання дослідження. Метою роботи є дослідження і розробка Android-додатків навігації по місту.

Для досягнення мети дослідження необхідно вирішити такі **завдання:**

а) розглянути існуючі додатки-навігатори, засоби і методи навігації по місту та провести дослідження оптимізації навігації по місту з застосуванням Android-додатків для виявлення найбільш вдалих;

б) розглянути основні складності розробки Android-додатка навігації по місту.

в) розробити Android-додаток навігації по місту, який буде показувати оточуючі користувача об’єкти, відображати місця по категоріях, показувати оточення та інформацію про оточуючі об’єкти, прокладати маршрут, виступати в ролі гіда конкретного міста, та показувати погоду.

Методи дослідження. Проведені в роботі дослідження основані на методах зниження похибки визначення координат за допомогою датчиків і фільтрів, які використовуються в

Android-пристроях. Зменшення похибок вимірювання також досягається за рахунок клієнт-серверної архітектури додатка, в яких використовуються 3D-мапи і проводяться складні імовірнісні обчислення на GPS-даних, доступних через Android GNSS API.

Апробація матеріалів роботи. Основні методи, ідеї, які використовуються в магістерській роботі були представлені на форумі ІТ-Ідея 2019 - СНУ ім. В. Даля, м. Сєвєродонецьк.

Практичне значення отриманих результатів полягає в тому, що результати досліджень використані при розробленні додатка-навігатора на ОС Android для навігації по місту.

Публікації. За темою магістерської роботи опублікована одна наукова праця, це тези до збірника науково-практичних праць ІТ-Ідея 2019 - СНУ ім. В. Даля, м. Сєвєродонецьк, за темою: Використання системи GPS при розробці навігаційних Android-додатків.

Структура та обсяг роботи. Магістерська робота складається із вступу, 4 розділів, переліку посилань з 26 найменувань, висновків, 2 додатка на 17 сторінках. Загальний обсяг роботи складає 102 сторінки. Магістерська робота містить 68 рисунків та 4 таблиці.

1 СУЧАСНИЙ СТАН МЕТОДІВ І ЗАСОБІВ НАВІГАЦІЇ ПО МІСТУ. ПЕРСПЕКТИВИ ОПТИМІЗАЦІЇ НАВІГАЦІЇ ПО МІСТУ

1.1 Принцип роботи навігаторів на базі операційної системи Android

Поява електронного навігатора майже повністю витіснила топографічну мапу. Використовуючи його, можна не тільки ознайомитись з місцевістю, а й прокласти маршрут пересування.

В основі дії будь-якого навігатора на базі OS Android лежить принцип автономності, тобто використання багатофункціональної мапи місцевості без підключення до інтернету. Це стало можливим після відкриття системи глобального позиціонування (GPS) військовими міністерствами для цивільного використання.

Після запуску і виходу на орбіту першого штучного супутника Землі в 1957 році було помічено, що частота сигналу, що виходить від радянського літального апарату, змінюється в залежності від його наближення або віддалення. Це дозволило зробити припущення про те, що, знаючи положення і швидкість супутника, можна визначити власні координати на Землі.

На підставі цього в США була розроблена програма по створенню системи позиціонування, побудована на базі супутників. Спочатку вона називалася NavStar, а пізніше була перейменована в GPS (Global Positioning System - Систе́ма глоба́льного позиціонува́ння). Паралельно свої розробки такої системи велися і в СРСР - «ГЛОНАСС». Побудовою супутникової системи навігації в обох країнах займалося виключно військове відомство для своїх потреб.

Після того як в 1983 році рейс KE007, який летів з Нью-Йорка в Сеул, вторгнувся на територію СРСР і був збитий (причому причиною вторгнення була названа дезорієнтація екіпажу), президент Рональд Рейган підписав наказ про дозвіл використання системи навігації GPS в усьому світі для цивільних цілей. Слідом за американцями свою систему відкрила і Росія, як правонаступниця СРСР.

В даний час повне покриття забезпечують дві системи - GPS і «ГЛОНАСС», але разом з тим існує французька Doris і китайська BeiDou. До 2020 року планується до розгортання супутникова навігація Galileo - спільний проект Євросоюзу. Розвиток супутникових мереж і послужив поштовхом для створення GPS навігаторів на Андроїд, популярної і безкоштовної операційної системи.

Таким чином, для того щоб гаджет міг виконувати роль навігатора, необхідно виконання двох умов:

а) апаратна частина: наявність спеціального електронного пристрою, здатного отримувати сигнали від супутників;

б) програмне забезпечення: програма, здатна аналізувати отриманий сигнал і обчислювати координати по його характеристикам.

1.1.1 Апаратна частина

Точність позиціонування апарату з навігацією GPS залежить від апаратного виконання мікросхеми (чіпа), використовуваного для прийому і обробки сигналу. Але для того щоб приймач зміг визначити місце розташування, одного супутника мало, як мінімум необхідно чотири. Тому важливо, щоб мікросхема прийому підтримувала відразу кілька супутникових систем [1].

Найбільш поширеними виробниками GPS-чіпів, що встановлюються в гаджетах, є SiRF, Mediatek (MTK), Broadcom. Їх мікросхеми відрізняються високою чутливістю і, що важливо для смартфонів - автономністю.

Вивчаючи різні спеціалізовані видання і відгуки користувачів, можна сказати, що найкращим чіпом на сьогодні є Sirf III. У порівнянні з конкурентом він відстежує більше супутників при несприятливих умовах, таких як підвищена хмарність або густий ліс. Але при цьому мікросхема MTK виграє в ефективності використання отриманих вимірювань, особливо в статичних умовах.

В середині 2017 року корпорація Broadcom заявила про випуск новаторського чіпа GNSS з кодовою назвою BCM47755. За твердженнями компанії, її мікросхема здатна вивести позиціонування в смартфонах на новий якісний рівень. Точність повинна складати близько 30 см, при цьому чіпи конкурентів можуть її забезпечити в межах трьох-п'яти метрів.

Мікроконтролер BCM47755 для досягнення високої точності в своїй роботі використовує одночасне фіксування двох сигналів - L1 і L5. Перший тип містить інформацію про становище супутника і його ідентифікатор (ефімеріса), а другий - уточнює його орбіту. Це перший приймач, випущений для мобільних гаджетів з одночасною підтримкою майже всіх існуючих сигналів: GPS L1 C / A, GLONASS L1, Galileo (GAL) E1, jps L5, Galileo E5a, QZSS L5, QZSS L1, BeiDou (BDS) B1.

1.1.2 Програмна частина

Хоча апаратна частина і є важливою ланкою в роботі системи навігації, але, як якісно вона не була б зроблена, її робота без належного програмного забезпечення може бути марною. Тому функціонування GPS базується на трьох ключових ланках: супутник, приймач, програмне

забезпечення. Саме програма визначає інформативність після обробки отриманих даних. У більшості випадків такий софт можна встановити безкоштовно на планшет, навігатор або смартфон.

В системі Android існує також вбудоване ПО, пов'язане з підтримкою GPS. Це проміжна програма між приймачем і додатком навігатора. Вона не тільки повідомляє системі про тип чіпа, встановленого в гаджеті, а й керує «теплим» і «холодним» стартом.

При активації чіпа перший раз отримання координат займає досить тривалий час. Пов'язано це зі створенням альманаху - файлу, в якому містяться відомості про видимі в даних широтах супутниках. Причому в ньому ж вказується і ефемеріс, одержуваний від них. Цей альманах самовідновлюєтьсяся через кожен місяць. Такий запуск називають «холодним».

Наступні ж «теплі» старти вже не вимагають створення альманаху, а оновлюють тільки ефемеріс. Далі дані з альманаху зчитуються вже додатком, що містить мапи місцевості і володіє різними можливостями, що дозволяють зробити навігацію зручною. А також додатки для поліпшення визначення координат можуть додатково використовувати відомості, отримані від вишок GSM або WiFi.

Встановивши платний або безкоштовний навігатор для Android на свій пристрій, в більшості випадків користувач зможе виконувати наступний дії:

а) визначати своє місце розташування і бачити його на електронній мапі;

б) отримувати інформацію про об'єкти які знаходяться поруч, наприклад, магазини, комунальні будинки, аптеки;

в) прокласти точний маршрут з однієї точки в іншу з урахуванням транспортних розв'язок і ситуацій на дорогах;

г) попередньо розрахувати час, який знадобиться на подолання конкретного відстані;

ґ) використовувати можливості звукового навігатора, який буде не тільки вести по прокладеному маршруту, а й повідомляти про зміни на ньому.

1.2 Огляд популярних GPS навігаторів на мобільних пристроях з ОС Android

а) Google Maps, приклад роботи нижче на рис. 1.1;

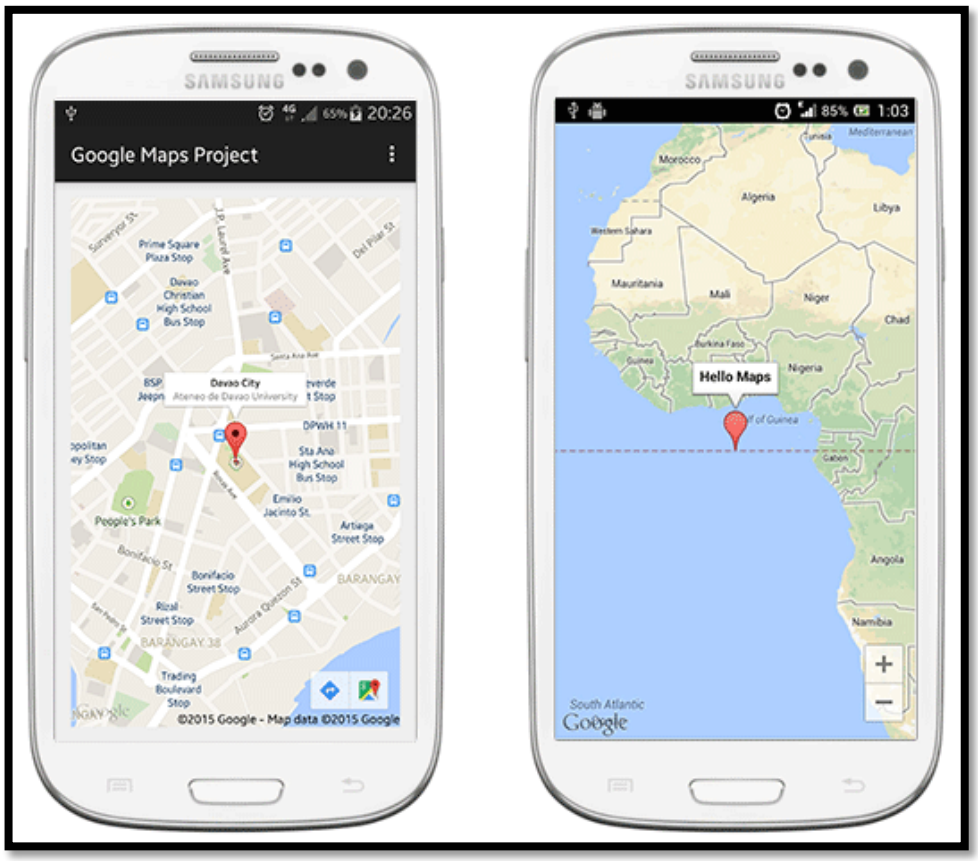

Рисунок 1.1 – Приклад роботи додатка Google Maps

Google Maps для Android – стандартний Android-навігатор. Функція для завантаження офлайн мап з'явилася нещодавно. Ця функція дозволяє по мапі та прокладати маршрут без Інтернет-підключення. Оффлайн-мапи в Google Maps поки ще сира функція, відносно оффлайн-мап у інших Android-навігаторів. Дуже великою незручністю є, неможливість завантаження мап для великих областей України та інших країн СНД. Оффлайн-мапи великих міст тільки у окремому доступі. Без онлайн-синхронізації при виїзді за межі міста буде дуже складно орієнтуватися [2].

Недоліки Google Maps:

1) має малий вибір пам’яток і локацій, для отримання детальнішої інформації доведеться користуватися сторонніми аплетами;

2) інформація про ДТП відсутня, не можливо додати місце аварії вручну;

3) немає попереджень про контроль швидкості, один голосовий асистент.

Переваги Google Maps:

1) навігатором зручно користуватись командами Окей Google, що дозволить увести запит у відведений пошуковий рядок;

2) мапи в додатку безкоштовні і є функція оффлайн;

3) не потрібно мучитись з активацією, установкою додаткових мап та іншого.

б) Яндекс.Навігатор, приклад роботи нижче на рис. 1.2;

Рисунок 1.2 – Приклад роботи додатка Яндекс.Навігатор

Додаток Яндекс.Навігатор – якісний, зручний і безкоштовний навігатор для Android. Він не вимагає багато апаратних ресурсів мобільного девайса. Навігатор легкий в управлінні і займає небагато місця на диску. Прокладає маршрути з урахуванням щільності трафіку, завантаженості доріг, аварій і перекриттів вулиць.

Інтерфейс у додатка зручний та наочний. Якщо по прокладеному маршруту пересуватися на авто, дає інформацію про час та відстань, які залишилися до прибуття в точку. Якщо попутно треба кудись заїхати, наприклад на заправку, автомийку, СТО, то на мапі можна вказати додатковий параметр, та маршрут буде перебудований оптимально з урахуванням цього параметру. Також можна увімкнути та вимкнути видимість пробок та напрямків натисканням на відповідну кнопку.

Однією з особливостей додатка є нічний GPS-режим, в якому відображення мапи відбувається в темних тонах, без відволікання водія в нічний час, та розслабляє стомлені очі від

нагрузки у дорозі. Доступних для пошуку об’єктів дуже багато, тут можна знайти і такі як: поштові відділення, різні салони, та багато іншого.

Також доступне голосове управління подібне Google Maps. Для роботи з оффлайн-мапами, можна скачати мапи для великих населених пунктів України та інших країн СНД. Використовуючи додаток в оффлайн-режимі, можна переглядати ці населені пункти, але прокласти маршрут, та шукати визначні місця можна тільки в онлайн-режимі під’єднавшись до Інтернету.

в) Навітел, приклад роботи нижче на рис. 1.3;

Навітел Навігатор – прооблений на відмінно Android-навігатор. Для того щоб користуватись мапами потрібно заплатити невелику суму, але також мапи можна скачати в Інтернеті.

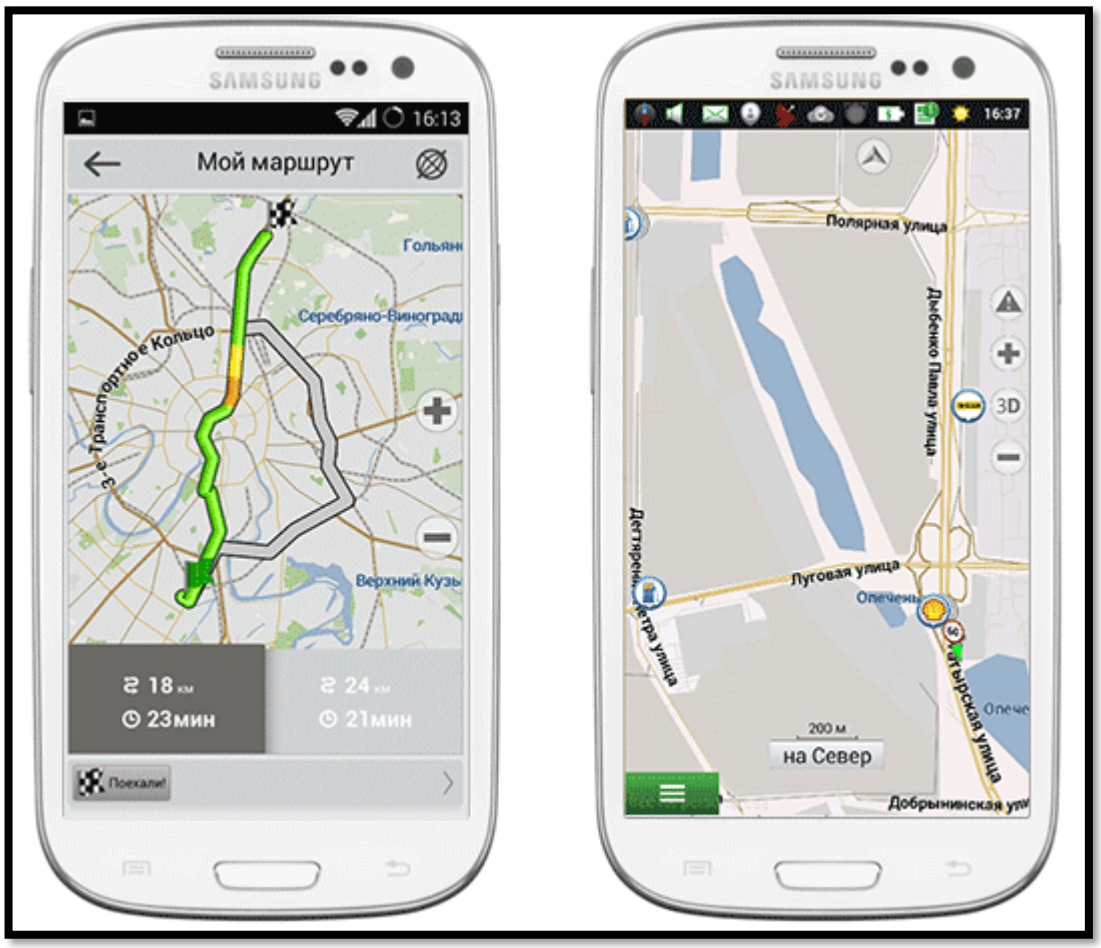

Рисунок 1.3 – Приклад роботи додатка Навітел

Переваги Навітел-навігатора:

1) можна перемикатить між двовимірним та тривимірним режимами;

2) присутність зручного інтерфейсу. Також є вибір між режимами (нічна тема, денна

тема);

3) присутність голосового управління;

4) можна перемикатись між супутниками, мапа з відображенням об’єктів доступна при будь-яких метеоумовах;

5) при побудовані маршруту, керувати мапою дуже просто та зручно, можна користуватися зумом, обертати мапу по необхідності в процесі руху користувача, перемикатися між видами транспорту: пішохід, велосипед, авто і т.д;

6) користувач в процесі руху отримує інформацію про швидкість, скільки залишилось до досягання точки призначення, повороти озвучуються.

г) Waze, приклад роботи нижче на рис. 1.4;

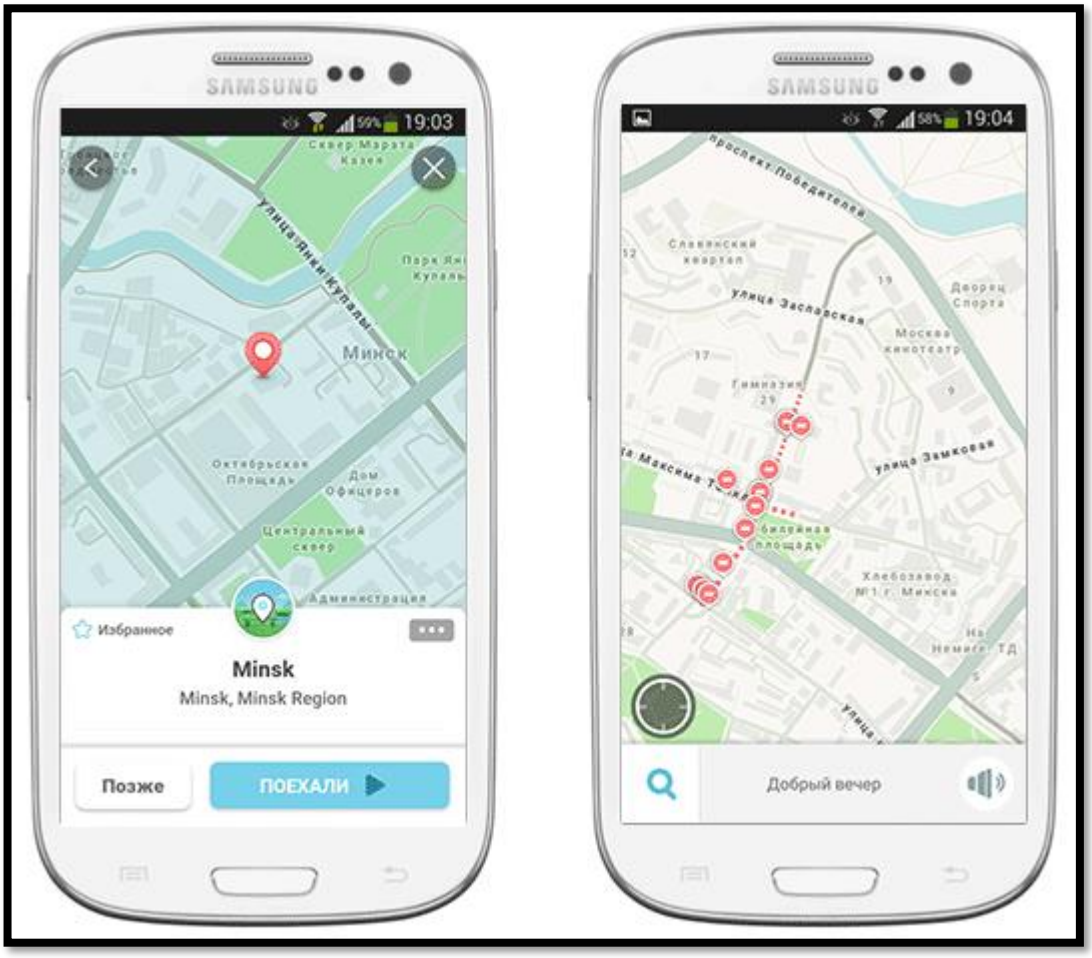

Рисунок 1.4 – Приклад роботи додатка Waze

Waze – безкоштовний навігатор-навігатор з відкритим кодом.

Переваги Waze:

1) користувачі можуть повідомляти про аварії, пробки, ремонт доріг, пости ДАІ та інші проблеми на дорозі, також збирається інформація з бази на сервері;

2) використання OSM-мап, вони більш точні і детальні;

3) при збої програми, аварійному вимкненні телефону автоматично робе перебудову маршруту, також спрощує перегляд мапи і маршруту.

г) OsmAnd, приклад роботи нижче на рис. 1.5:

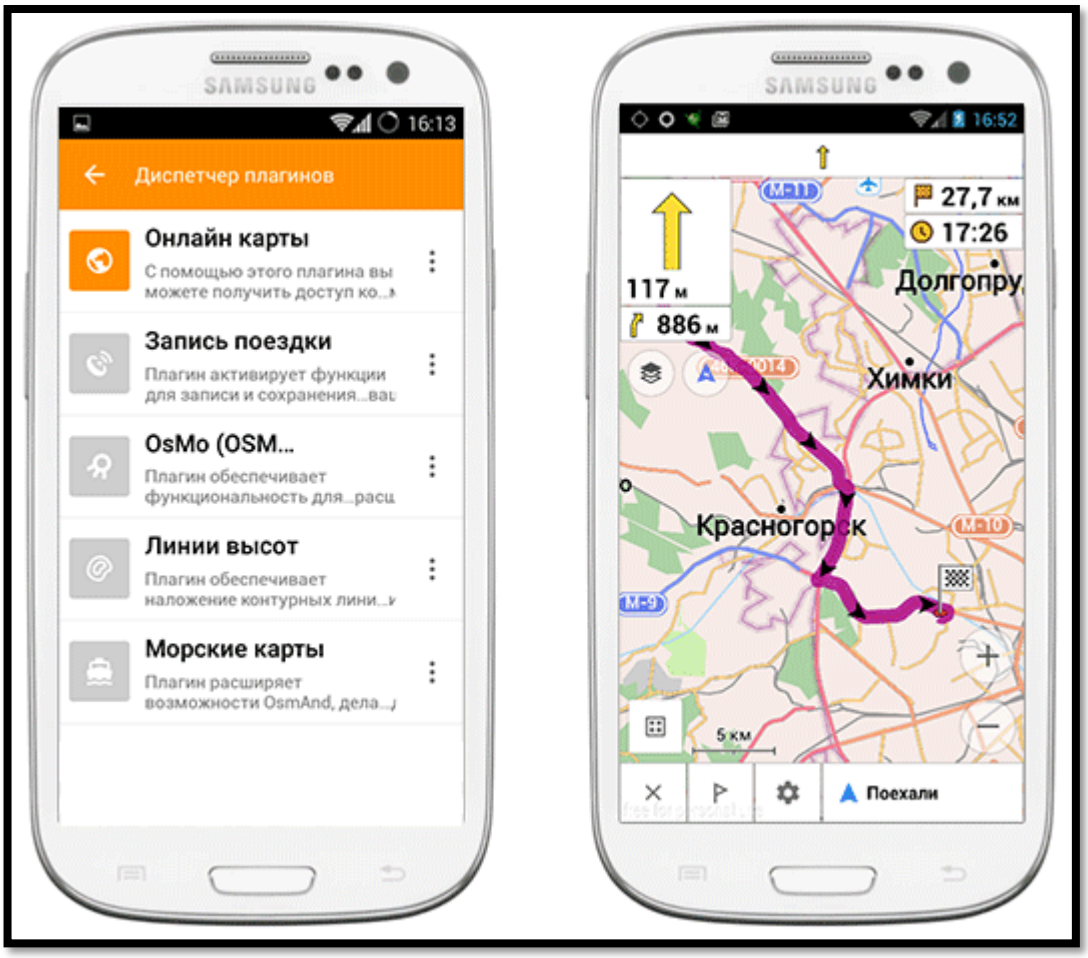

Рисунок 1.5 – Приклад роботи додатка OsmAnd

Переваги OsmAnd:

1) зручний в ролі оффлайн-навігатора. Для користувачів яким зручніше один раз завантажити мапу та бути в оффлайн режимі весь час;

2) використовуються растрові та векторні мапи, які складаються користувачами. Є можливість фільтрувати об’єкти, вибирати тип транспортного засобу. Маршрут можна оптимізувати, побудувати найкоротший, уникати автомагістралей, платних доріг і т.д.;

3) є підтримка плагінів, завдяки їм можна розширити функціональність, наприклад завантацити лижні або морські мапи, вірахувати дистанцію за допомогою GPX. Також користувачу можна редагувати мапи;

4) є голосовий асистент, присутнє управління голосом.

д) Maps.Me, приклад роботи нижче на рис. 1.6;

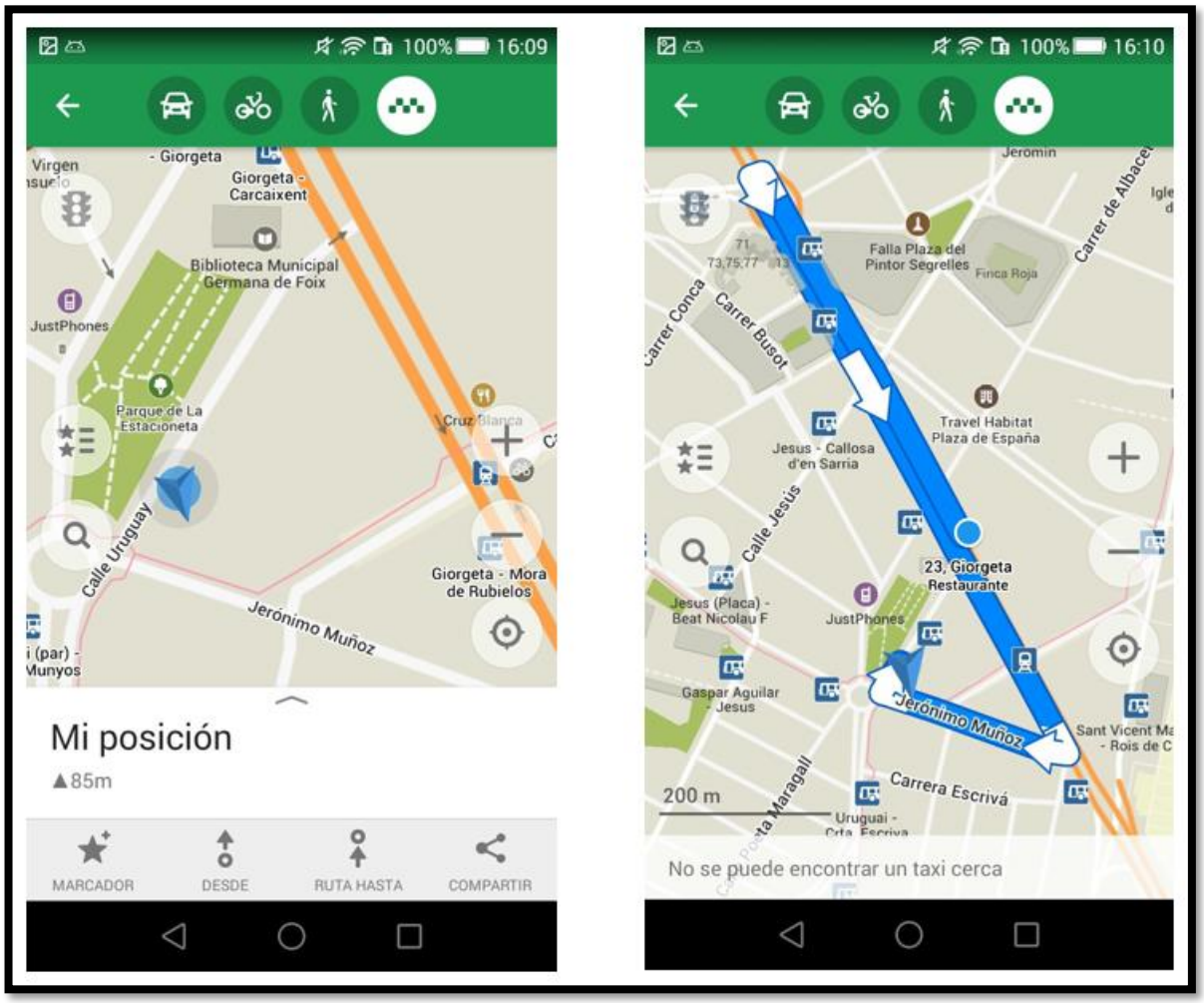

Рисунок 1.6 – Приклад роботи додатка Maps.Me

Переваги Maps.me :

1) присутні точні та деталізовані мапи. На мапах відображуються найдрібніші подробиці, серед нид можна знайти такі як: лавки, та стежки.

2) можна переглянути мапу метро;

3) багато налаштувань, можна змінити одиниці виміру, переглянути історію пересувань і т.д. Також можна налаштувати автомасштабування при необхідності.

4) є перемикання між видами транспорту: пішохід, велосипед, авто, громадський транспорт і т.д.;

5) також можна користуватися оффлайн-мапами, які скачані на пристрій, працюють повноцінно тільки при включеному Інтернеті.

Недоліки Maps.me :

1) навігація з обмеженнями для автомобілістів, маршрути не завжди коректно

будуються;

2) у додатку існують похибки, дані не завжди актуальні, особливо для глухих місць;

3) масштабування мапи може не спрацювати.

1.3 Огляд популярних в Україні GPS додатків-навігаторів та гідів по окремим містам на ОС Android

а) перший додаток-навігатор та гід міста Київ – ИнтересныйКиев;

В додатку є влаштована мапа Google, яка за допомогою позначок покаже туристам найцікавіші місця, також є мапа 1913 року, завдяки цьому туристи можуть подивитись як змінювалось місто з тих років.

Не зручно зроблений вибір категорій, якщо натиснути на активність – Місто, користувачеві буде представлений список проходячих картинних галерей та виставок, музичних концертів, майстер класів і т.д. Дуже багато новин вже застаріли за декілька років, та не зникли звідти, це схоже на сміттєсховище новин, нових новин майже нема, а якщо і є то декілька. Активність – пам'ятки також містить дуже багато не актуальних новин, це одразу відштовхує велику кількість користувачів. З активністю – Заклади де можна поїсти пренензій нема, до кожного кафе, ресторану і т.п. можна прокласти маршрут з використанням Google maps, приклад роботи показаний нижче на рис. 1.7 та 1.8:

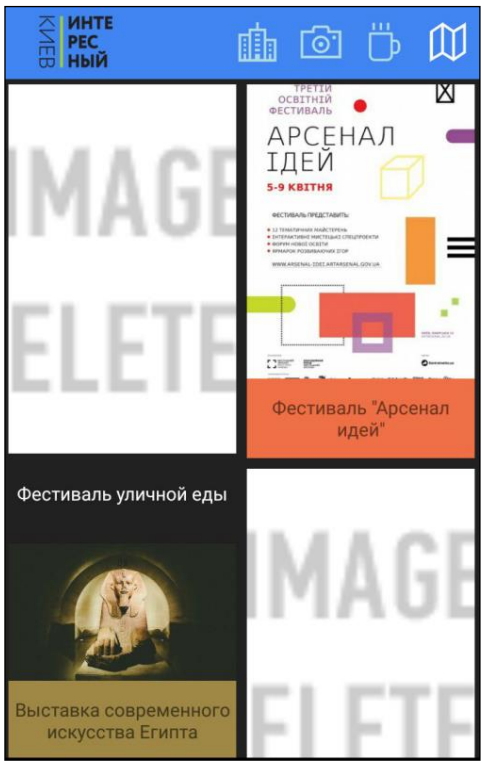

Рисунок 1.7 – Головна активність додатка-навігатора та гіда міста Київ – ИнтересныйКиев

Рисунок 1.8 – Один з застарілих заходів додатка-навігатора та гіда міста Київ – ИнтересныйКиев

б) другий додаток-навігатор та гід по місту Одеса Путеводитель по Одессе;

Додаток дає змогу побачити на мапі Google цікаві місця, з прокладкою маршруту до кожного з них. Також у додатку присутня невелика інформація про місто, його історія.

Кнопка – Места переводить користувача на сторінку з категоріями, серед яких можна знайти такі як: музеї, театри, пам’ятники, визначні місця. Серед них можна обрати одну категорію і переглянути список відповідний цій категорії. Якщо користувач захоче прокласти маршрут до вибраного місця, то в нього нічого не вийде, додаток не може знайти поточне місцезнаходження користувача, постійно йде завантаження координат, показано нижче на рис. 1.9. Увійти у Google всередині додатка зі своїм логіном взагалі неможливо, користувачу буде показаний білий екран, або завантажена сторінка браузеру.

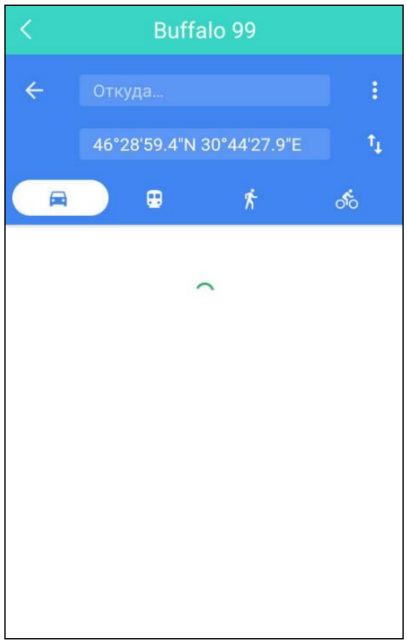

Рисунок 1.9 – Приклад прокладання маршруту до обраного закладу у додатку «Путеводитель по Одессе»

Якщо потрібно прокласти маршрут до місця, то це можливо лише за допомогою Google Maps, але тільки коли буде обрано пошук цікавих місць на мапі. Кнопка «Календарь» перенесе користувача додатка у активність де будуть показані усі діючі заходи. Кнопка – «Еще» покаже користувачеві категорії, з яких можна обрати: цікаві місця, харчування, розміщення, відпочинок, та ін. Приклад роботи додатка-навігатора та гіда міста Одеса показано нижче на рис. 1.10:

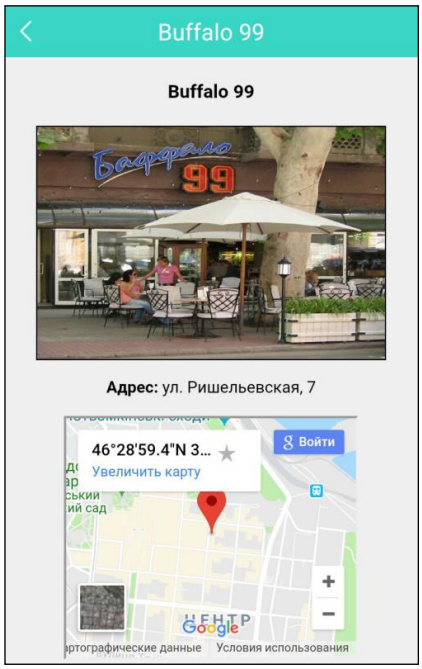

Рисунок 1.10 – Приклад одного з закладів у додатку-навігатору та гіду міста Одеса – «Путеводитель по Одессе»

1.4 Налаштування Android Studio і додавання у додаток мапи Google

Налаштування Android Studio. Для того щоб використовувати новий Maps API, потрібно надати Google ім'я пакета нашого застосування. Тому необхідно створити новий додаток в Android Studio і налаштувати деякі залежності для успішного підключення до Maps API [3].

Відкривши Android Studio треба створити новий проект, приклад нижче на рис. 1.11. На першому екрані налаштувань потрібно ввести дані, такі як ім'я проекту - тут пишемо MapApp, і домен компанії. Ім'я пакета додатка формується за замовчуванням з перевернутого доменного імені та імені проекту.

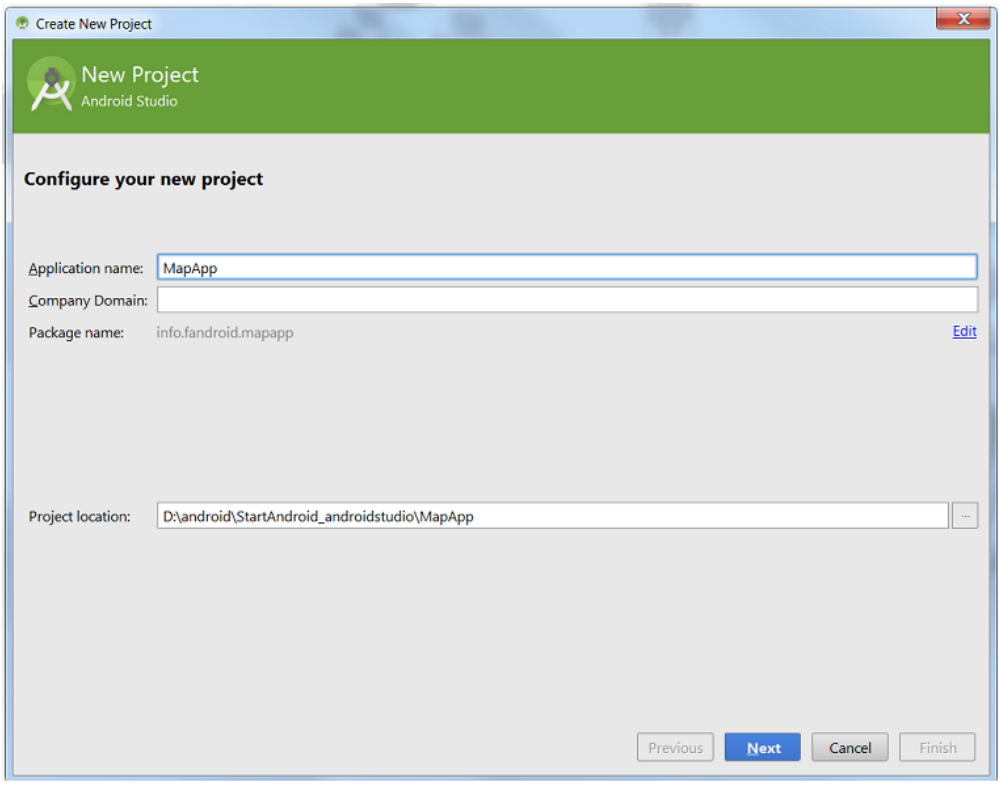

Рисунок 1.11 – Створення нового проекту в Android Studio

Треба обрати версію Android. Наступною дією треба татиснути кнопку «Далі», нічого не змінюючи, до кінця. Після створення проекту Android Studio буде індексувати його, це може зайняти деякий час в залежності від потужності машини. Можна побачити, що індексація закінчилася, коли кнопки у верхній частині екрану стануть активними.

Тепер, коли проект створений, треба переконатися, що є необхідні компоненти SDK, для підключення додатка до сервісів Google. У верхній панелі інструментів, обираємо кнопку SDK менеджера SDKManager. Відкриваємо папку Extras в SDK Manager і переконаємося, що встановлені наступні пакети: (Google Play Service; Google Repository).

Ще потрібно додати в файл build.gradle проекту залежності для Google Play Services. За замовчуванням є два файли build.gradle всередині структури проекту; один всередині папки модуля, позначений цифрою 1 на рис. 1.12 нижче, і один в папці проекту, на верхньому рівні, з цифрою 2 на рис. 1.12 нижче:

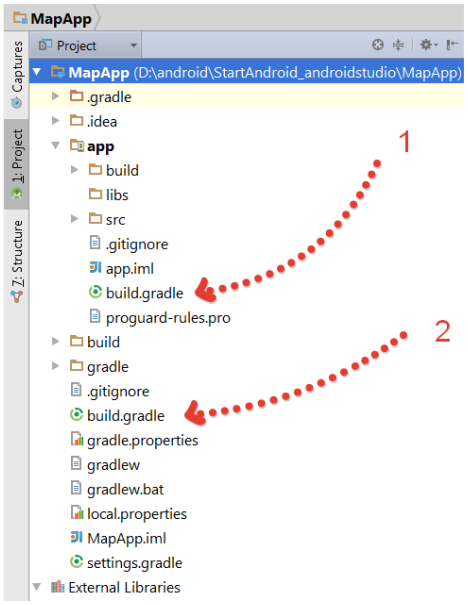

Рисунок 1.12 – Файли build.gradle в Android Studio

Треба відкрити файл в папці модуля, під цифрою 1 на рис. 1.12 вище. В розділ «dependencies» потрібно додати останню версію бібліотеки Google Play Services. У секцію dependencies потрібно додати рядок compile 'com.google.android.gms: play-services: ' і після двокрапки вписати версію. Після додавання рядка натисніть в правому верхньому куті посилання «Sync Now» для оновлення файлу збірки. Оновлення займе якийсь час.

1.5 Створення додатка з функціями мапи

Для створення програми з функцією мапи за допомогою ПО Android Studio, треба відкрити в Android Studio на панелі верхнього меню File (Файл)> New (Новий)> New Project (Новий проект), а потім вибрати шаблон активності "Google Maps Activity", як показано на рис. 1.13:

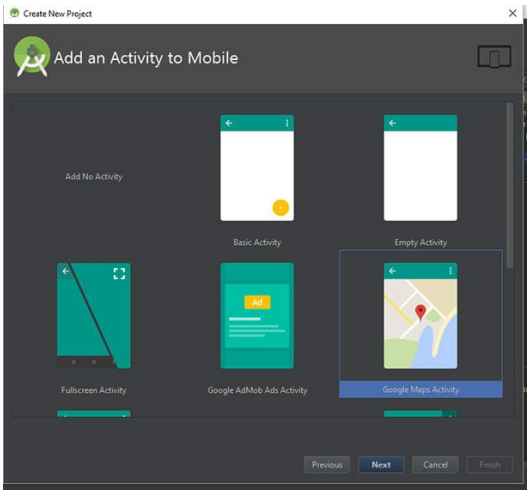

Рисунок 1.13 – Шаблон активності "Google Maps Activity"

Після цього треба натиснути Next (Далі) для відображення нової сторінки, на якій можна ввести ім'я активності (Maps Activity), ім'я для розкладки (Layout) і заголовок активності (Activity Title). Залишимо це значення за замовчуванням. На наступній сторінці можна вказати ім'я вашого застосування і пакета. Якщо дії закінчені, треба натиснути "Finish" (Закінчити), далі опиняємося на сторінці додатка [3].

Використовуємо раніше створений ключ API для цього додатка. На сторінці проекту програми треба відкрити файл _google_mapsapi.xml в папці / res / values. Потрібно замінити значення YOUR_KEY_HERE на ваш ключ API.

Після цього ви можна відкрити файл MapsActivity.java, який містить всю логіку створення додатка з мапою. Простий код всередині файлу виглядає наступним чином:

```
public class MapsActivity extends FragmentActivity implements OnMapReadyCallback {
private GoogleMap mMap;
@Override
protected void onCreate(Bundle savedInstanceState) {
super.onCreate(savedInstanceState);
setContentView(R.layout.activity_maps);
SupportMapFragment mapFragment = (SupportMapFragment) getSupportFragmentManager()
.findFragmentById(R.id.map); mapFragment.getMapAsync(this);}
@Override
public void onMapReady(GoogleMap googleMap) {
mMap = googleMap;
LatLng sydney = new LatLng(-34, 151);
mMap.addMarker(new MarkerOptions().position(sydney).title("Marker in Sydney"));
mMap.moveCamera(CameraUpdateFactory.newLatLng(sydney));}}
```

Цей код повинен створити екземпляр GoogleMap, який представляє мапу в додатку. Після цього він створить новий елемент Marker для розміщення у верхній частині мапи. Маркер використовує розташування з змінної LatLng, яка представляє собою координату (-34, 151). І нарешті, буде викликаний метод moveCamera (). Коли ми перший раз відкриємо додаток, він перемістить мапу на конкретну координату, яка наприклад знаходиться в Сіднеї. Якщо потрібно помістити координату в місце розташування користувача, то просто треба додати наступний рядок:

```
mMap.setMyLocationEnabled (true);
```

Далі представлений метод moveCamera (). Він дозволяє відобразити невелику кнопку "моє місцезнаходження" у верхньому правому куті мапи. Коли буде натиснута ця кнопка, мапа переміститься в поточне місце розташування користувача, однак для використання цієї функції необхідно включити на пристрої Android службу Location Services. Приклад додатка за мапою Google нижче на рис. 1.14:

Рисунок 1.14 – Додаток з мапою Google

1.6 Постановка наукової задачі та обґрунтування методики досліджень

Проведений аналіз популярних GPS навігаторів на мобільних пристроях з ОС Android показали, що невирішеними у окремих додатків є такі задачі, як: присутні лише окремі оффлайн мапи великих міст або регіонів, платне завантаження окремих мап, обмежена навігація для

автомобілістів, бувають похибки позиціонування, незручний режим показу мап, масштабування може не спрацьовувати.

При проведенні аналізу популярних в Україні GPS додатків-навігаторів та гідів на ОС Android по окремим містам було виявлено їх некоректну роботу з сервісами Google, некоректне, або неможливе прокладання маршрутів. Також розглянуті додатки мають незручний інтерфейс, в якому користувач може довго блукати у пошуках потрібної інформації, або зовсім відмовиться від цього одразу, не захотівши довго розбиратися. Вони вміщують багато марної, неактуальної інформації про заходи в місті, яка просто відлякує потенційних користувачів марно витраченим часом на пошуки.

Після проведення аналізу методів і засобів оптимізації навігації по місту, результати показали, що при використанні потрійної функції позиціонування A-GPS, фільтрів та датчиків, можна значно зменшити похибки позиціонування. А якщо використовувати клієнт-серверну архітектуру, яка використовує 3D-мапи і проводить складні імовірнісні обчислення на GPS даних доступних через Android GNSS API, то можна звести всі похибки позиціонування до мінімуму.

Для вирішення даної науково-практичної проблеми, в даному контексті слід визначити 3 задачі магістерської роботи:

а) аналіз сучасного стану методів і засобів навігації по місту;

б) аналіз та дослідження методів і засобів оптимізації навігації по місту;

в) практична реалізація додатка-навігатора для міста Сєвєродонецьк, який би поліпшив пошуки користувачам потрібного місця за допомогою мапи, та сервісів Google API, з урахуванням розглянутих методів і засобів оптимізації навігації по місту, та з урахуванням недоліків існуючих додатків-навігаторів, які були проаналізовані.

1.7 Висновки до розділу 1

В першому розділі були розглянуті принципи роботи навігаторів на базі операційної системи Android, їх апаратна та програмна частина. Проведений огляд та аналіз популярних GPS навігаторів на мобільних пристроях з операційною системою Android, також проведений огляд популярних в Україні GPS додатків-навігаторів та гідів по окремим містам з ОС Android. Розглянуто створення додатка з функціями мапи, та налаштування Android Studio і додавання у додаток мапи Google. Зроблена постановка наукової задачі та обґрунтування методики досліджень.

2 АНАЛІЗ МЕТОДІВ І ЗАСОБІВ ОПТИМІЗАЦІЇ НАВІГАЦІЇ ПО МІСТУ

2.1 Порівняння технологій GPS та A-GPS

GPS (Global Positioning System - система глобального позиціонування) - система супутникової навігації. Вона дозволяє визначити поточне місцезнаходження гаджета користувача на земній кулі, підбираючи сигнал із супутника.

A-GPS (Assisted - Global Positioning System – допоміжна система глобального позиціонування) - це не додаткова технологія, а надбудова до глобальної навігаційної системи. Вона додає до неї важливу функцію - потрійне позиціонування. Увімкнувши навігацію на телефоні, йому потрібно зв'язатися із супутником, щоб визначити своє поточне місцезнаходження, яке зазвичай називають - «холодний старт». Це може зайняти від 10-15 секунд до декількох хвилин залежно від місця розташування пристрою.

Використовуючи нову технологію, смартфон отримує координати при «холодному старті» не тільки з космосу. Дані можна отримати з базової станції вашого оператора або, найчастіше, через Інтернет - мобільний або Wi-Fi. Якщо використовуються вежі мобільного оператора, то спочатку можуть виникнути неточності в розташуванні пристрою. Вони виправляються, коли гаджет «додзвонюється» до супутника [4]. Позиціонування на трьох різних джерелах не тільки прискорює отримання координат, але й поєднує їх, щоб визначити місце розташування в межах кількох метрів. Нижче, на рис. 2.1 представлена структура технології A-GPS:

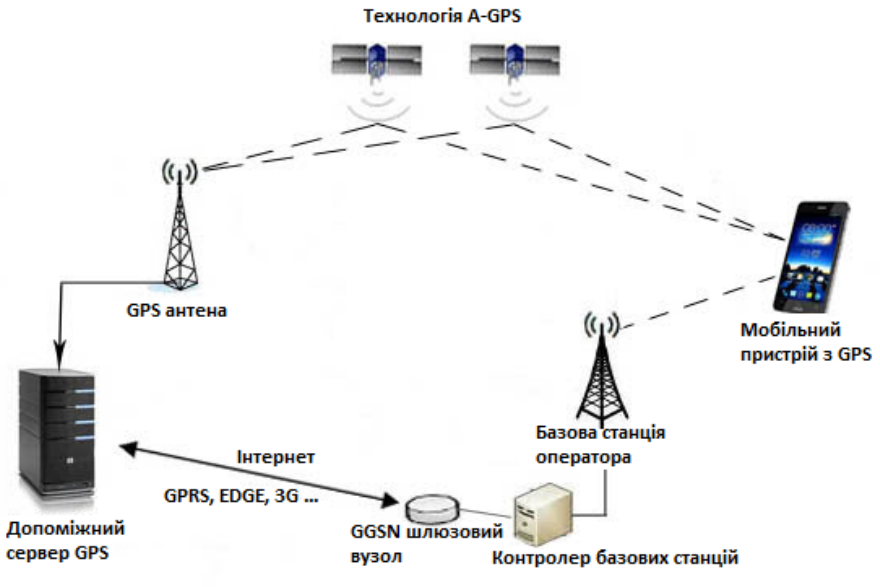

Рисунок 2.1 – Структура технології A-GPS

Ще одна важлива перевага технології - можливість приймати сигнал у так званих «мертвих зонах» - низинах, тунелях, густих лісах. Одне з трьох джерел сигналу всеодно якось "потрапить" у "мертву зону", і телефон отримає координати [5].

Незважаючи на очевидні переваги, у технології є кілька недоліків:

а) деякі приймачі такого типу потребують радіомодуль. Наприклад, якщо на вашому планшетному ПК немає SIM-карти, тобто радіомодуль (GSM) вимкнено, ви не зможете користуватися A-GPS;

б) поза покриттям мережі та без доступу до Інтернету функція просто не працює;

в) при використанні технології телефон або планшет передають певну інформацію через Інтернет. Обсяг інформації невеликий - всього кілька кілобайт. Але якщо ви в роумінгу, з вашого рахунку можуть стягнути чималу суму навіть за них.

2.2 Системи та методи локального позиціонування об’єктів

Місце розташування об'єкта може бути визначено як в системах глобального, так і локального позиціонування. На сьогоднішній день широкого поширення набули системи глобального позиціонування, такі як GPS і ГЛОНАСС [6].

Позиціонування - це автоматизоване визначення місця розташування об'єктів, передбачає: визначення координат об'єкту, формування повідомлень, що містять координатну інформацію, організацію обміну службовими повідомленнями, документування інформації про переміщення об'єктів, візуалізація інформації [7].

Системи позиціювання та моніторингу можна розділити в залежності від масштабів зони обслуговування системи на глобальні, регіональні, зональні і локальні:

а) глобальні системи використовують сигнали навігаційних супутників і призначені для визначення місця розташування по всій поверхні Землі. Для роботи таких систем необхідно наявність радіовидимості між супутниками і пристроями зі складу системи, тому функціонування таких систем можливо тільки на відкритому просторі. Прикладами такої системи є GPS і ГЛОНАСС;

б) регіональні системи здатні вирішувати завдання позиціонування на обмеженій території площею до 300 тисяч км^2, використовують такі засоби як:

1) мережі мобільного зв'язку;

2) системи цифрового рухомий транкінгового радіозв'язку (СЦПТР);

3) радіолокаційні станції (РЛС).

в) зональні системи призначені для визначення місцезнаходження об'єкта в межах якоїсь зони - території, площа якої лежить в межах від декількох гектарів (підприємство, невеликий

населений пункт) до декількох десятків квадратних кілометрів (промислова зона, аеропорт, місто та ін.);

Для визначення місцезнаходження такі системи можуть використовувати:

1) мережі мобільного зв'язку;

2) мережі мікросотового зв'язку DECT;

3) системи цифрового рухомого радіозв'язку;

4) радіомережі Wi-Fi;

5) ZigBee;

6) nanoLOC.

г) локальні системи моніторингу призначені для визначення місця розташування об'єктів на дуже невеликих обмежених відкритих територіях і закритих просторах;

Як правило, застосування таких систем як GPS / ГЛОНАСС ускладнене або неможливе не зважаючи на невисоку точність. Засобами, здатними забезпечити вирішення такого класу задач можуть бути:

1) мережі мікросотового зв'язку DECT;

2) радіомережі Wi-Fi;

3) радіомережі Bluetooth;

4) радіомережі ZigBee;

5) радіомережі nanoLOC;

6) радіомережі, що використовують технології та протоколи, адаптовані для вимірювання відстаней (ISO24730-2, CSS / ISO24730-5, NFER, UWB).

Також локальні системи позиціонування можна використовувати, як провідники в музях, торгових центрах, для оптимізації роботи складів. Використання систем глобальної навігації для вирішення завдань локального позиціонування ускладнене за рахунок збільшення вартості обладнання (наприклад, інтеграція GPS-приймачів) і наявності великої кількості перешкод.

Системи локального позиціонування передбачені для швидкого позиціонування об'єктів в закритих приміщеннях, що також можуть бути додатковими елементами у вже розроблених мережах. Такі системи повинні працювати в режимі реального часу - RTLS (Real Time Location Systems – система локального позиціонування в режимі реального часу).

Задача локального позиціонування користувача може бути вирішена на основі існуючої інфраструктури локальних безпровідних мереж (WLAN). У мережі WLAN входять точки доступу та обладнання користувача. Важливо, що всі точки доступу розміщуються стаціонарно, їх координати заздалегідь відомі, і всі вони є приймачами сигналу в деякій полосовій частоті.

Розташування користувача у таких мережах визначається на основі отриманої совокупної

інформації характеристик сигналів від точок доступу. Обмін координатної та службової інформації між інфраструктурою системи локального позиціонування та об'єктом проводиться за рахунок повідомлень. Користувачі мобільних пристроїв можуть визначити своє розташування на мапі або на плані будівлі за допомогою локальної безпровідної мережі. За допомогою прив'язки точок доступу до координат місцевості можливо відображення користувача в ГІС системах.

Для позиціонування об'єктів всередині приміщення в даний час широко використовуються методи, в основі яких лежить ідея позиціонування з використанням заздалегідь сформованої бази даних. У базі даних зберігаються відомості про значення деякої метрики для точок з відомими координатами. Сукупність таких точок утворює опорну сітку. Позиціонування проводиться шляхом порівняння обраної метрики для поточного положення об'єкта зі значеннями метрик з бази даних і вибору найближчій по метриці опорної точки в якості оцінки місцезнаходження об'єкта. Точність позиціонування в значній мірі визначається вибором метрики. Вибір метрики в даний час є актуальною проблемою.

Існують наступні методи визначення місцезнаходження і вимірювання координат об'єкта в радіомережах:

а) RSSI (Received Strength Signal Indication) - відстань до об'єкта оцінюється за потужністю сигналу. Метод грунтується на допущенні досить жорсткої залежності між ступенем загасання сигналу і пройденою відстанню. Основна перевага методу в простоті. При його використанні не потрібно нічого, крім фіксації на базових вузлах прийнятої потужності сигналу від позиціонуючого об'єкта. Даний метод добре працює на малих відстанях, але при збільшені дальності дає велику помилку за рахунок специфіки поширення радіосигналу;

б) AoA (Angle of Arrival) - місцезнаходження об'єкта визначається в межах площі трикутника, утвореного перетином осей діаграм спрямованості антен секторів трьох базових станцій (модифікований метод тріангуляції);

в) ToF (Time of Flight) - вимір часу проходження електромагнітної хвилі від мобільного пристрою (об'єкта) до точки доступу за рахунок використання сигналу з лінійно-частотної модуляцією;

г) ToA (Time of Arrival) - вимір часу проходження сигналу від мобільного терміналу до базової станції, при якому відстань до об'єкта розраховується виходячи з різниці часу відправки сигналу і його отримання. При цьому даний спосіб вимагає суворої синхронізації часу на відправника і одержувача, чого досягти досить складно. Помилка синхронізації може істотно вплинути на помилку дальнометрії;

ґ) TDoA (Time Difference of Arrival) - проводиться вимірювання різниці часу приходу сигналу від мобільного пристрою до декількох базових станцій. Потрібна сувора синхронізація

часу, тільки на базових станціях, до мобільного пристрою таких вимог не пред'являється;

д) RTT (Round TripTime) - базова станція відправляє сигнал на мобільний пристрій і чекає отримання відповідного сигналу, за різницею часу відправлення та отримання сигналу визначається час проходження сигналу в обох напрямках, а, отже, і відстань між об'єктами;

е) LPT (Location Patterning Techniques) - визначення місцезнаходження проводиться з використанням розпізнавання образів радіосигналів, засноване на виборці і записах радіо зразків поведінки сигналу в певному навколишньому середовищу.

Вибір методу залежить від багатьох умов. Це і необхідна точність позиціонування, і темп опитування, і число позиціонуючих об'єктів, можливості їх тимчасової синхронізації, наявність і характер перешкод та ін. По можливості, слід віддавати перевагу найбільш простим методам, наприклад, RSS, але при підвищених вимогах до точності і в нестабільних умовах цього може бути недостатньо. У складній, постійно мінливій обстановці великих приміщень, пов'язаної з постійним переміщенням великого числа людей, найбільш підходящими представляються далековимірні параметри, одержувані на основі двосторонньої оцінки ToA. Вони не вимагають жорсткої тимчасової синхронізації, більш точні, ніж RSS, і краще TDoA, так як абонент в абсолютній більшості ситуацій знаходиться всередині зони покриття.

Найбільш поширеним на сьогоднішній день способом побудови WLAN є Wi-Fi, це технологія передачі даних середнього радіусу дії, зазвичай покриває відстань до 100 метрів. Спочатку WiFi не призначалася для використання в якості технології локального позиціонування, стандартна мережа надає лише інформацію до точки доступу, тому при визначенні місця розташування використовуються спеціалізовані методи, найчастіше, такі як RSSI і TDoA. Основні переваги використання Wi-Fi технології - це широке поширення і низька вартість обладнання. До основних недоліків можна віднести то, що для підвищення точності потрібне збільшення щільності розташування базових станцій, ефір Wi-Fi досить завантажений.

В даний час для підвищення точності використовуються системи, в яких відбувається об'єднання різних технологій позиціонування.

2.3 Використання фільтрів та датчиків для зниження похибки GPS на ОС Android

На точність визначення системою GPS-координат (широти і довготи) можуть впливати розташовані поруч високі будівлі, дерева, різні погодні явища, розташування супутників і т. п. Цієї точності досить для визначення власного місцезнаходження, однак якщо використовувати потік геоданих для обчислення відстані і вартості поїздки для служби таксі, то помилки накопичуються, і отриманий результат виявляється значно гірше очікуваного. На бюджетних

смартфонах помилка може становити дуже високий процент, що веде до різного роду неприємних ситуацій і невдоволення клієнтів.

Це дуже актуальна проблема на смартфонах Android, так як через невисоку вартість їх використовує переважна більшість користувачів. Для цього потрібен сервіс, який буде обробляти маршрути, отримані від сторонніх сервісів.

Одним з найнадійніших способів підвищити точність позиціонування і підрахунку дистанцій - скласти векторну мапу доріг міста і відображати GPS-координати на найближчі дороги. Однак цей метод можна застосовувати, якщо існує актуальна і надійна база координат доріг. Її створення - складне завдання, що вимагає задіяння значних ресурсів для кожного міста, тому цей спосіб занадто дорогий. Можна скористатися API від Google, але в цьому сервісі є обмеження на кількість запитів в день, та не у всіх користувачів є можливість встановити сервіси Google Play. Так як усі ці методи не підходять, для максимального поліпшення точності GPS-координат, краще використовувати фільтри та датчики Android.

Для вирішення проблеми потрібен алгоритм для ОС Android, який вирішує такі завдання:

а) компенсувати похибки, пов’язані з тим, що пройдений маршрут поступово збільшується, навіть коли авто фактично стоїть на місці. Це відбувається через те, що GPS-координати приходять з похибкою і на мапі виглядають як кінцеві точки неправильної «зірки»;

б) фільтрувати різкі «стрибки» в точки, віддалені від реального маршруту на значну відстань (до 500 метрів);

в) відновлювати маршрут при короткочасних (приблизно 30-60 секунд) втратах зв'язку з GPS;

г) алгоритм не повинен інтенсивно витрачати заряд батареї, а також оперативну пам'ять.

Оптимальним рішенням проблем позиціонування і позбавлення шуму є використання фільтра Калмана.

Два найкращих метода для визначення положення пристрою в просторі на основі даних від акселерометра, магнітометра і гіроскопа - віртуальний сенсор ROTATION_VECTOR та фільтр Маджвіка. Кращі вони з кількох причин. Вони прості у використанні і для реалізації не потрібно писати дуже багато коду. При цьому вони не шумлять і дуже швидко працюють [8].

Заради поліпшення показників віртуальних датчиків розробники Android провели велику роботу, завдяки чому тепер не потрібно вивчати і реалізувати складні алгоритми, такі як: sensor fusion і AHRS. Можна скористатися датчиком linear_accelerometer, він видає прискорення без урахування сили гравітації. Для визначення орієнтації в просторі можна та рекомендується використовувати датчик rotation_vector, він використовує фільтр Калмана, та показує кращий результат.

Фільтр Маджвіка - це програмне забезпечення з відкритим вихідним кодом, розраховане, в першу чергу, на низьку обчислювальну потужність цільової системи. В якості вхідних даних він використовує покази акселерометра, гіроскопа і (опціонально) магнитометра. На виході отримується кватерніон, що описує стан пристрою в просторі. Він працює дійсно швидко і майже не витрачає ресурси, але є проблема у визначенні параметрів цього фільтра. З плюсів варто відзначити, що його легко перенести на будь-яку платформу.

Фільтр Калмана - це ефективний рекурсивний фільтр, що оцінює вектор стану динамічної системи, використовуючи ряд неповних і зашумлених вимірювань. GeoHash - функція для перетворення 2 координат виду 31.341503, 45.563991 (довгота, широта) у 1 рядок. Він схожий на алгоритм бінарного пошуку і працює дуже швидко, майже не витрачаючи обчислювальні ресурси.

Фільтром можна скористуватися для вирішення двох проблем. Треба якось об'єднати точки, які знаходяться поруч, для зниження потоку надлишкової інформації. Для цього можна вибрати який-небудь радіус і об'єднувати всі точки, що потрапляють в коло з цим радіусом. Обчислення відстані в сферичних координатах - дорога операція, тому краще було б використати GeoHash-функцію, що дозволяє дуже швидко визначити, чи відносяться точки до однієї області, чи ні. Ця функція видає хеш у вигляді рядка, довжина якого визначається користувачем і впливає на точність кодування. Чим більше довжина хеша, тим менше область і більша точність координат в одній області. Максимальна можлива довжина геохеш-рядка - 12 символів. Дуже хороший результат для даної задачі показує довжина хеша 7 або 8 символів.

Ще одне застосування цієї функції - визначення і фільтрація "стрибків". Будемо вважати, що якщо GPS приймач видав поспіль більше 3 (задається користувачем) координат з одним хешем, то ця точка правильна і її потрібно враховувати. Якщо ж менше - то, ймовірно, це якесь випадкове значення, яке не потрібно враховувати.

Приклади прокладання маршрутів без застосування фільтра на основі GeoHash і з його використанням показані нижче на рис. 2.2 і 2.3:

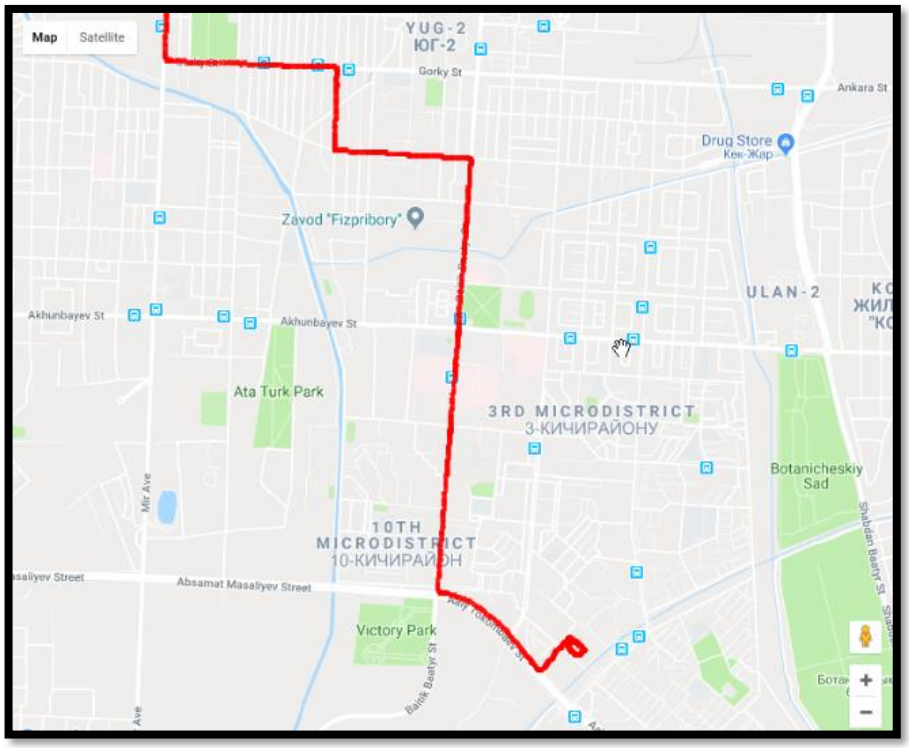

Рисунок 2.2 – Маршрут без застосування фільтра на основі GeoHash

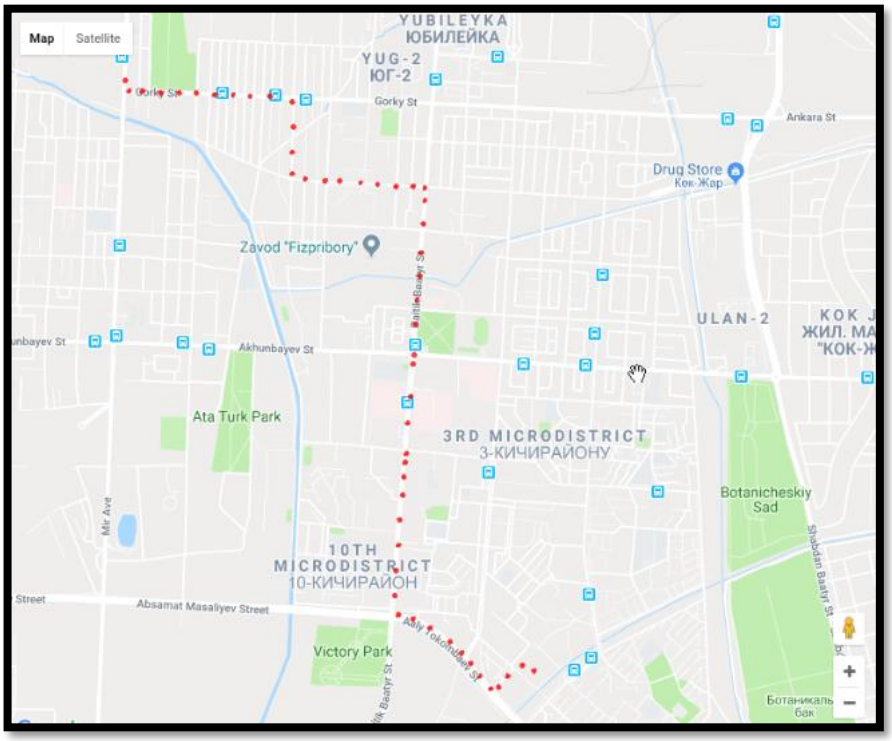

Рисунок 2.3 – Маршрут із застосуванням фільтра на основі GeoHash. Довжина рядка (precision) - 7. Мінімальна кількість точок з одним геохешем – 3

Як видно з наведених рисунків, GeoHash фільтр дозволяє сильно скоротити кількість оброблюваних точок. Це може бути критичним, якщо координати повинен обробляти сервер або

якщо планується зберігати маршрути в базі даних. Приклад процесу прокладання маршруту з використанням описаної вище методики приведено нижче, на рис. 2.4:

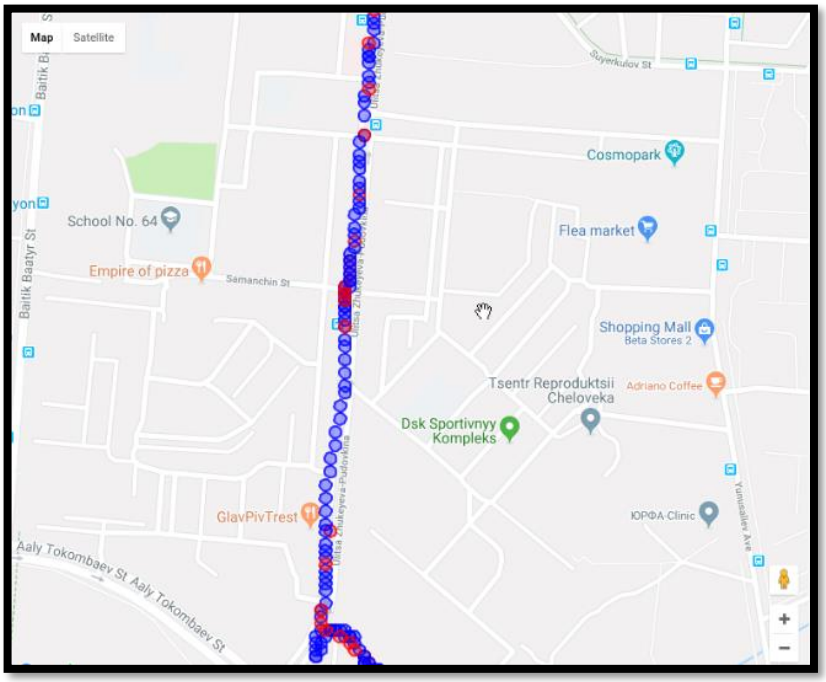

Рисунок 2.4 – Процес прокладання маршруту. Червоні точки - координати GPS, а сині - результат роботи фільтру. Як видно була невелика втрата зв'язку з GPS, але траєкторія успішно відновлена.

Як висновок з вищесказаного, знизити похибки GPS на ОС Android можна за допомогою написання Android-модулю та бібліотеки на мові C, які реалізують описані вище фільтри. Траекторії візуально стануть більш згладженими та зникає накручування довжини маршруту при відсутності руху.

Для підрахунку довжини маршруту використовуються два алгоритми. Перший - алгоритм Вінченті, він використовується всередині методу distanceBetween() класу Location. Тести показують, що результат майже не відрізняється при тому, що другий варіант вимагає значно менше ресурсів. Також є підозра, що на великих дистанціях похибка буде більшою. В межах міста другий алгоритм показує такий же результат, як і перший.

На даному етапі можна відновити траєкторію при втраті сигналу до 30-50 секунд. Далі помилка вимірювань акселерометра стає занадто великою. Також не завадить реалізувати відновлення траєкторії при короткочасній втраті зв'язку з GPS і максимально скоротити використання GPS.

2.4 Система позиціонування нового покоління в Uber

Для подолання проблеми похибки GPS, в Uber розроблений апгрейд для програмного забезпечення GPS на Android, що істотно поліпшило точність визначається позиції в міському оточенні за допомогою клієнт-серверної архітектури, яка використовує 3D-мапи і проводить складні імовірнісні обчислення на GPS даних доступних через Android GNSS API [9].

GPS це мережа з більш ніж 30 супутників керованих урядом США, що знаходяться на орбіті землі на висоті близько 20 тисяч кілометрів. (Більшість смартфонів в наші дні, так само можуть отримувати сигнал схожих Російських супутників "ГЛОНАСС".) Ці супутники посилають радіочастотні сигнали які GPS приймачі, як знаходяться в смартфонах, можуть фіксувати. Важливо, що ці супутники сповіщають про час, коли вони запустять ці сигнали.

Для кожного супутника, чий сигнал обробляє приймач, різниця між часом отримання і часом запуску, множиться на швидкість світла, отримана величина називається псевдодистанцією. Якщо годинник супутника і одержувача синхронізовані, і сигнал переміщається по лінії прямої видимості, тоді ця величина буде дорівнює реальної дистанції до супутника. Однак, годинники не синхронізовані, так що приймачу необхідно вирішити рівняння з чотирьох невідомих, його власні 3D координати на сфері, і відхилення годинників. Таким чином, необхідно мінімум чотири супутники для отримання цих чотирьох невідомих.

Якщо проігнорувати відхилення годиників, можемо інтуїтивно інтерпретувати наближення локації, виробленого GPS приймачем, перетином сфер з центром на супутниках і радіусом кожної сфери заданої псевдодистанцією. На практиці, приймач GPS обробляє сигнали від значно більшого числа супутників (до 20 GPS і ГЛОНАСС супутників видимих у відкритому полі), і отримання більшого, ніж мінімальна, число рівнянь надає додаткову стійкість до шуму, перешкодам і т.д. У доповненні до GPS і ГЛОНАСС, деякі нові / майбутні приймачі можуть / зможуть обробляти сигнали від інших супутникових систем. Деякі інші запускаються навігаційні супутникові системи Galileo, керований Європейським Союзом, IRNSS в Індії і BeiDou, керований Китаєм. Більш загальний термін GNSS (глобальна навігаційна супутникова система) охоплює ці системи.

GNSS локація неточна в міському середовищі. Дуже сильне твердження знаходиться за позиціонуванням заснованому на GNSS, то що приймач має лінію прямої видимості до кожного супутника, чию псевдодистанцію він обчислює. Цей механізм добре працює на відкритій місцевості, але в міському середовищі він працює погано, як показано нижче на рис. 2.5:

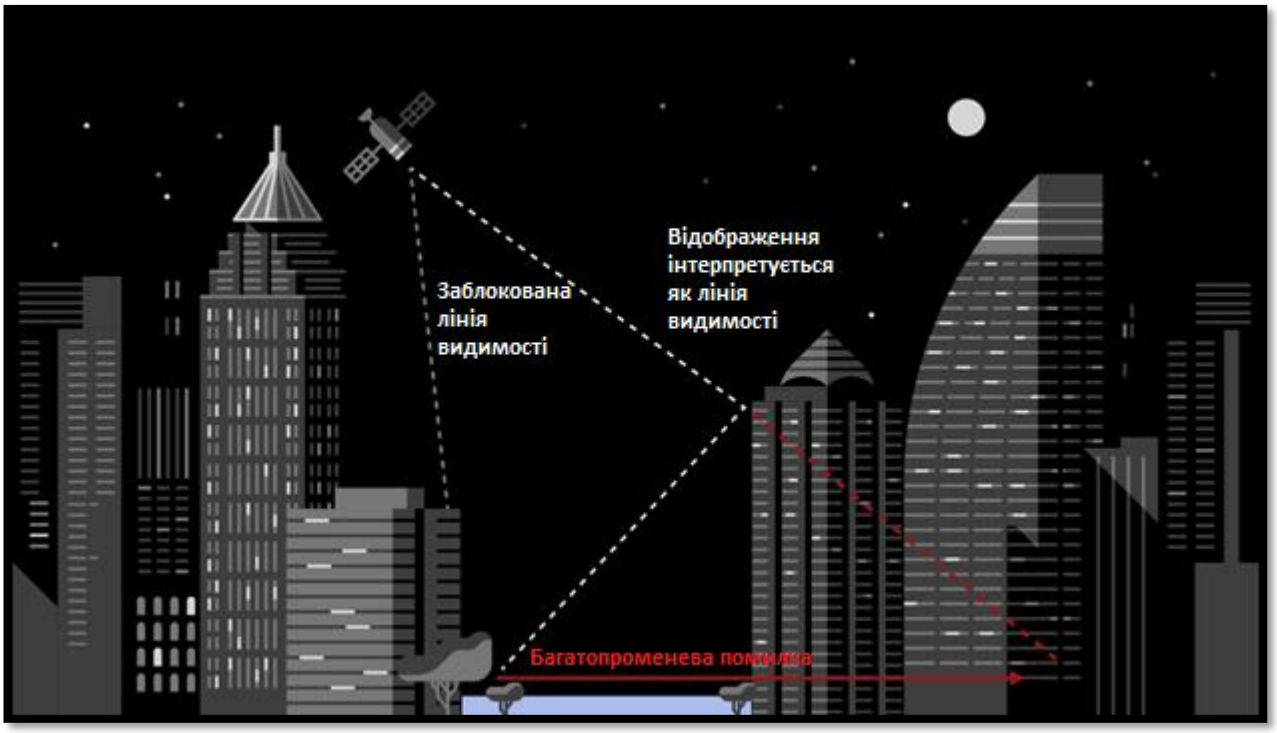

Рисунок 2.5 – Обмеження прямої видимості і сильне відображення може викликати великі похибки GPS.

Будинки дуже часто обмежують прямий огляд супутників, так що приймач обробляє сигнали відповідні відображенню від інших будівель. Суттєва неточність (позитивний зсув) в псевдодистанції одержувана з цього явища може вести до погрішностей в наближенні положення, які можуть досягати 50 або більше метрів в міських каньйонах. Той, хто пересувався пішки, або на машині, або замовляв Uber в великих містах відчував ці проблеми на собі.

Підхід Uber до збільшення точності визначення локації створює особливість з кожного обмеження GNSS сигналу, яке створює проблеми для стандартних приймачів. Для Android телефонів, LocationManager API надає не тільки приблизне положення телефону, але і показник сигнал-к-шуму (SNR) для кожного видимого GNSS супутника. Якщо зіставити інформацію про «силі сигналу» з 3D-мапами, тоді можна отримати дуже цінну інформацію про позиції. Нижче на рис. 2.6 показано спрощену версію того, як SNR супутників і 3D-мапи можуть бути використані для припущення на якій стороні вулиці ми знаходимося.

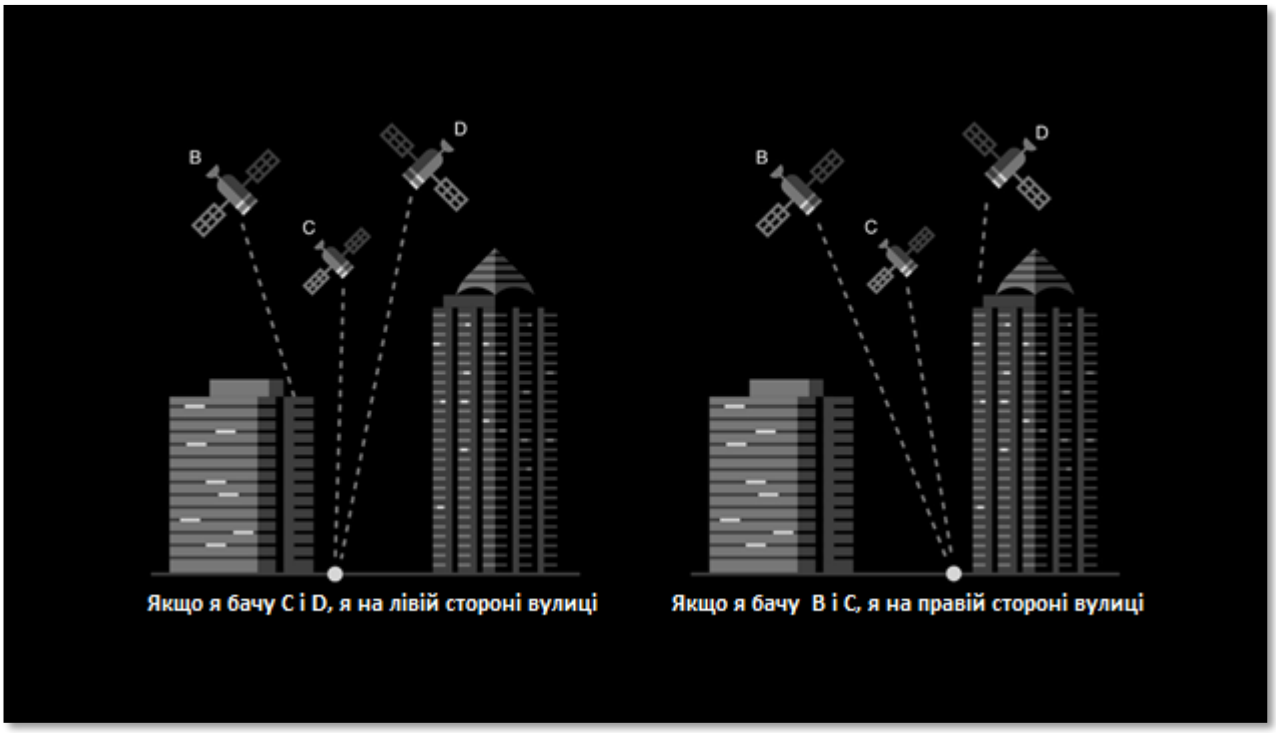

Рисунок 2.6 – Сила сигналу супутників, в поєднанні з 3D-мапами, надає дуже цінну інформацію про місцезнаходження.

Занурюючись в деталі, підхід Uber заснований на приміщення наступного припущення в математичні рамки: якщо SNR для супутника низький, тоді шлях лінії прямої видимості можливо обмежений або затінених; якщо SNR високий, тоді LOS (лінія прямої видимості) можливо чиста. Специфікатор «можливо» критичний тут: навіть якщо приймач знаходиться в тіньовий зоні, сильно відбиті сигнали, все ще можуть досягти його, і навіть якщо він знаходиться в чистій місцевості, отриманий сигнал може бути слабким (через руйнівної інтерференції між LOS і відбитими шляхами , явище відноситься до багатоколійні загасання). Так само в більшості випадків, 3D-мапа не до кінця точна, і безумовно не передає випадкові обмеження великими рухомими об'єктами не відбитими на мапі, як вантажівки. Це додає невизначеність в процес.

Розподіл усіх зіставлень тіней з використанням трасування променів. Незважаючи на те що інтуїтивне припущення, що сила сигналу супутників несе в собі корисну інформацію про місцезнаходження звучить добре, воно повинно бути конкретизовано з використанням імовірнісних рамок. Для будь-якого можливого положення приймача, можна перевіряти чи заблокований промінь з цього положення до супутника на нашій 3D-мапі. Тепер, використовуючи модель для розподілу ймовірності SNR під LOS і тіньових умов, визначаємо найбільш ймовірне значення SNR для цього супутника. Наприклад, якщо становище затінене, тоді ймовірність високого SNR мала. Загальна ймовірність заданої позиції, заснована на SNR супутників, цей добуток ймовірностей відноситься до різних супутників. Роблячи це на сітці

можливих положень, ми отримуємо вірогідну поверхню - чи теплову мапу можливих положень приймача, засновану тільки на силі сигналу супутників. Ми називаємо цю процедуру імовірнісним зіставленням тіней. Трасування променів від одного можливого місця розташування до кожного супутника для імовірнісного зіставлення тіней. Це робиться для тисяч ймовірних місць розташування, приклад приведено нижче на рис. 2.7:

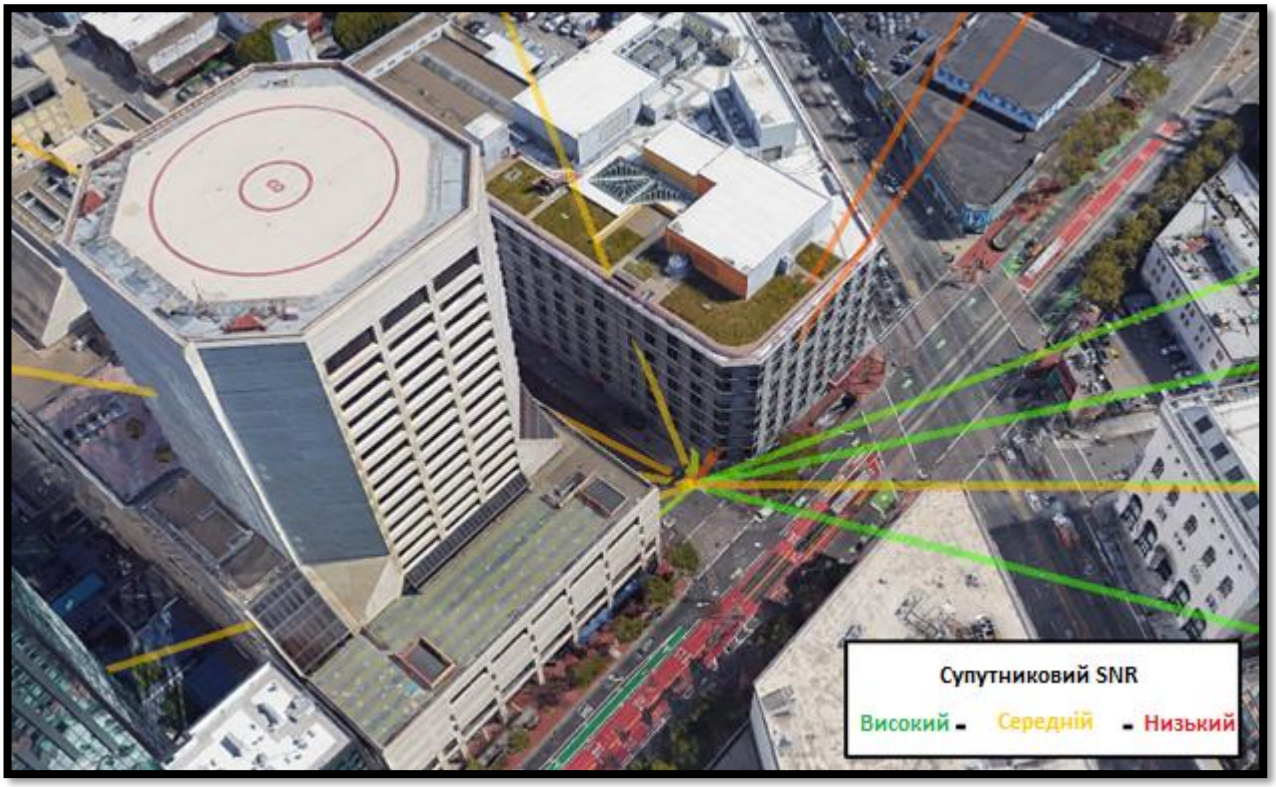

Рисунок 2.7 – Трасування променів від одного можливого місця розташування до кожного супутника для імовірнісного зіставлення тіней.

Імовірнісна поверхня або теплова мапа, від імовірнісного зіставлення тіней об'єднує інформацію від вимірювань SNR супутників. Як би там не було, як видно нижче, на рис. 2.8, ця теплова мапа може бути дуже складна. Вона може мати безліч відокремлених, сильно розділених гарячих точок (локальних максимумів) найчастіше відповідають заданій стороні вулиці, але іноді і неправильним позиціям (наприклад фантомів). Для того щоб звузити наше наближення позиції і уникнути наводку на фантомів, ми повинні з'єднати цю інформацію з ще більшою її кількістю.

Теплова мапа позицій обчислена з використанням сил сигналу супутників може мати безліч гарячих точок. У наведеному нижче прикладі на рис. 2.8, наше покращене наближення позиції (синій шлях, чорний еліпс з невизначеністю) слід за дійсним шляхом (жовтий шлях), в той час як звичайний GPS (червоний шлях, сірий еліпс з невизначеністю) неточний:

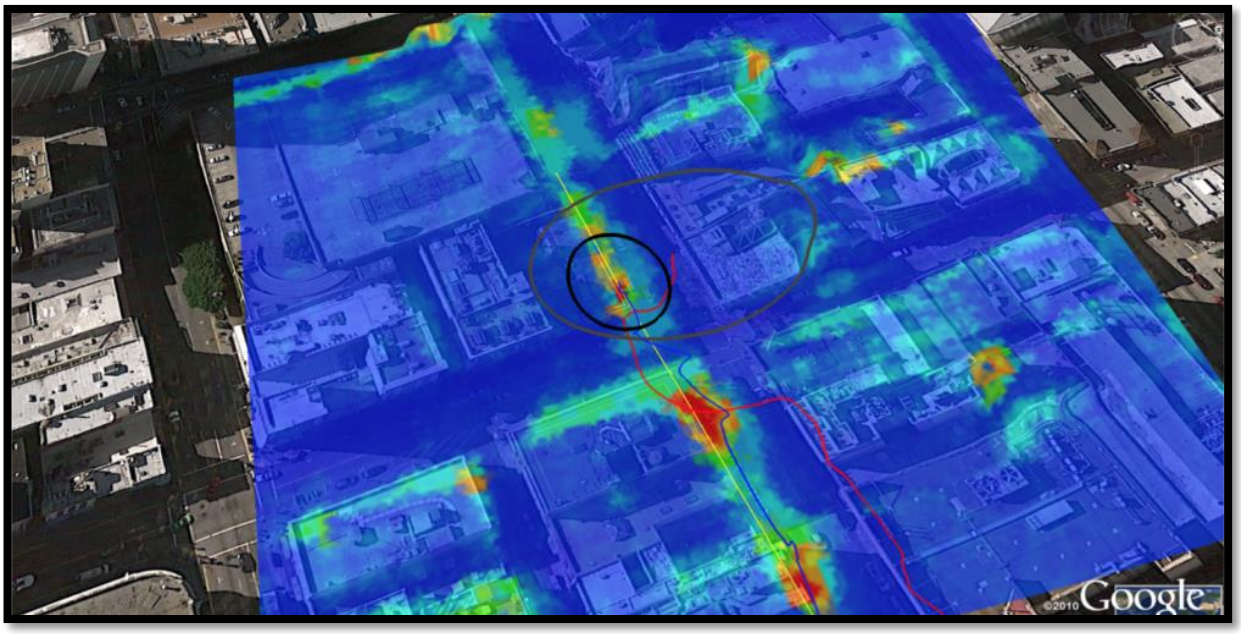

Рисунок 2.8 – Теплова мапа позицій обчислена з використанням сил сигналу супутників

Поєднання інформації через частковий фільтр. Для смартфонів з ОС Android , інформація яку ми використовуємо на додаток до сили сигналу супутників, це зазвичай стандартне виправлення позиції GNSS, але так само може бути і Android Fused позиція, яка може включати в себе засноване на Wi-Fi позиціювання. Оскільки дана локація може бути вельми неточна, одноразове миттєве поєднання стандартного виправлення GNSS з імовірнісним тіньовим зіставленням зазвичай веде до поганої продуктивності. Для того щоб отримати перевагу від інформації про силу сигналів супутників, ми довіряємо GPS менше в забудованих районах (сірий еліпс невизначеності GPS на рис. 2.8 це звичайна модель, яку ми використовуємо, а чорний еліпс невизначеності для поліпшеного GPS це результат нашого алгоритму). Потім ми використовуємо попередні вимірювання і накладаємо обмеження на зміни позиції з часом, використовуючи модель адаптовану для додатка (наприклад, пішохід проти автомобіля). Ми досягаємо цього з використанням часткового фільтра, який апроксимує ймовірності розподілу позицій приймача в будь-який заданий момент часу набором зважених часток. Іншими словами, ми припускаємо де знаходиться смартфон, використовуючи тисячі можливих локацій (тобто, частки).

Згодом, імовірнісні ваги та місця розташування частинок прогресує на основі вимірів і моделі руху. Оскільки теплова мапа від імовірнісного зіставлення тіней має так багато локальних максимумів і оскільки GNSS виправлення може мати такі великі викиди, ми не можемо використовувати звичайну техніку, як фільтр Калмана або розширений фільтр Калмана, який ґрунтується на відстеженні ймовірності розподілу добре наближеної Гауссовим розподілом, що мають форму дзвіночка. Частковий фільтр дозволяє наближати довільний розподіл, в обмін на більшу складність, і тут в гру вступає серверна архітектура Uber, приклад нижче на рис. 2.9.

Наближення розташування отримане, як зважений центр ваги гарячих точок, надане частковим фільтром часто виправляє дуже великі похибки GPS. Радіус неточності (білий круг) для поліпшеного GPS заснований на зрізі набору частинок, і часто є більш реалістичним виміром, ніж малий радіус неточності (чорне коло) зазвичай повертається чистим GPS навіть коли похибки позиції великі, показано на рис. 2.9:

Рисунок 2.9 – Радіуси неточності

Поєднання часткового фільтра і трасування променів додає складності до екосистеми бекенд серверів, з отриманням дуже stateful сервісів.

Покращена система GPS Uber складається з сервісу часткового фільтра, сервісу управління 3D-мап-тайлами, сервісу менеджера, Uber HTTP API, і хмарного сховища і інтегрується з іншими сервісами Uber, схема на рис. 2.10:

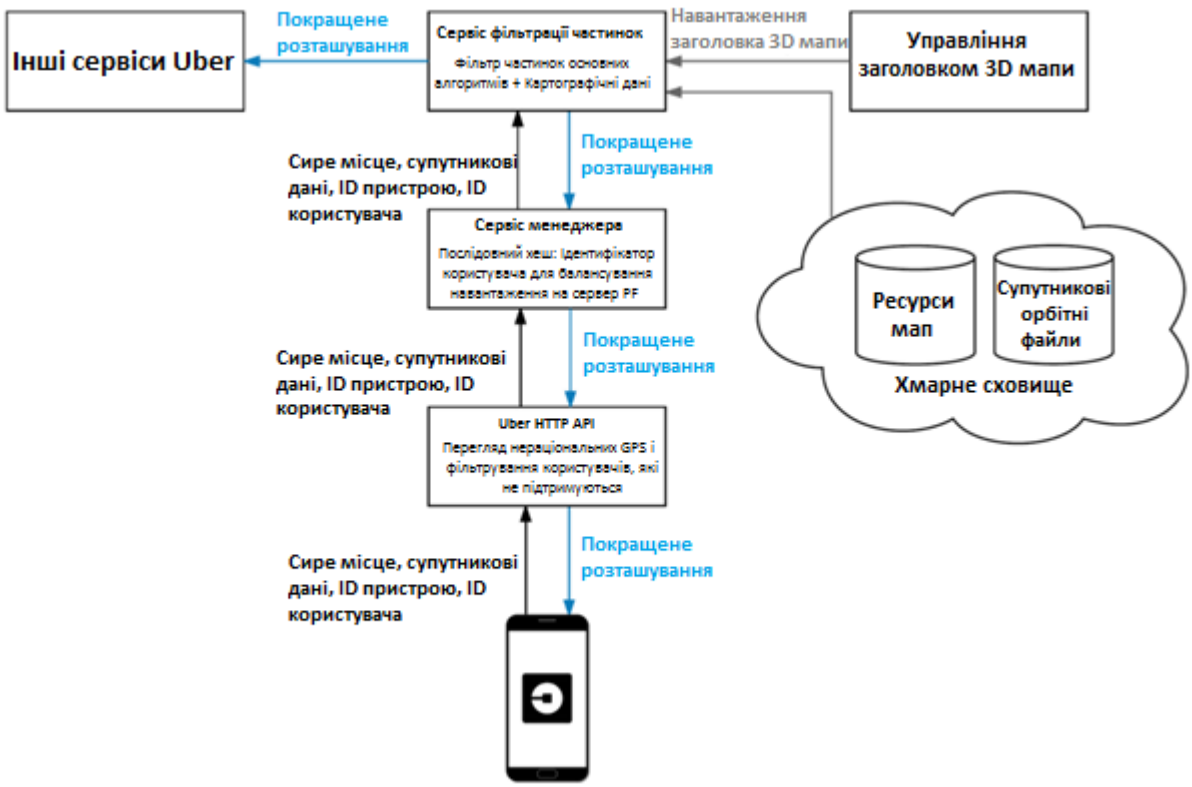

Рисунок 2.10 – Схема покращеної системи GPS Uber

Використовується два типи стану: стан часткового фільтра для кожного користувача і 3D-мапи, які використовуються для трасування променів, порегіонно. Використання часткового фільтра вимагає рівня серверної афінності. Кожен новий запит до сервісу повинен бути спрямований до того ж бекенд сервера для обробки, для того щоб оновити правильний частковий фільтр. Додатково, через великий розмір 3D-мап, кожен бекенд сервер може містити деяку малу кількість 3D світу в оперативній пам'яті.

Оскільки кожен сервер може містити тільки кілька квадратних кілометрів даних мап, не всі сервера здатні обслужити всіх користувачів. Реалізація бекенд системи для такого рішення зажадало створення шару маршрутизації сесій, який бере до уваги 3D-мапу сервера.

2.5 Визначення місцезнаходження GPS за допомогою пристрою Reach

Сучасні супутникові навігаційні технології забезпечують позиціонування з точністю близько до 10-15 метрів. У більшості випадків цього достатньо, проте в деяких випадках потрібно більше: наприклад, автономний безпілотний дрон, рухаючись досить швидко над земною поверхнею, буде відчувати себе некомфортно в хмарі координат із метрових помилок.

Для уточнення супутникових даних використовувались диференціальні системи та технології RTK (кінематика в реальному часі), але донедавна такі пристрої були дорогими та

громіздкими. Нещодавні досягнення в галузі цифрових технологій за допомогою мікрокомп'ютера Intel Edison допомогли вирішити цю проблему. Reach - це перший компактний високоточний GPS-приймач, дуже доступний в ціні [10].

Диференціальні системи навігації (DNSS) покращують точність позиціонування та швидкість переміщення користувачів, надаючи дані вимірювань або інформацію про корекцію з однієї або декількох базових станцій.

Координати кожної базової станції відомі з високою точністю, так що дані вимірювань станції служать для калібрування даних приймачів поблизу. Приймач може обчислити час поширення сигналу і теоретичну відстань між кожним супутником та собою та. Коли ці теоретичні значення порівнюються з даними спостереженнь, то різниці представляють помилки в прийнятих сигналах. З цих відмінностей виходить корекційна інформація (дані RTCM). Точність визначення координат за допомогою програми Reach наведена нижче на рис. 2.11:

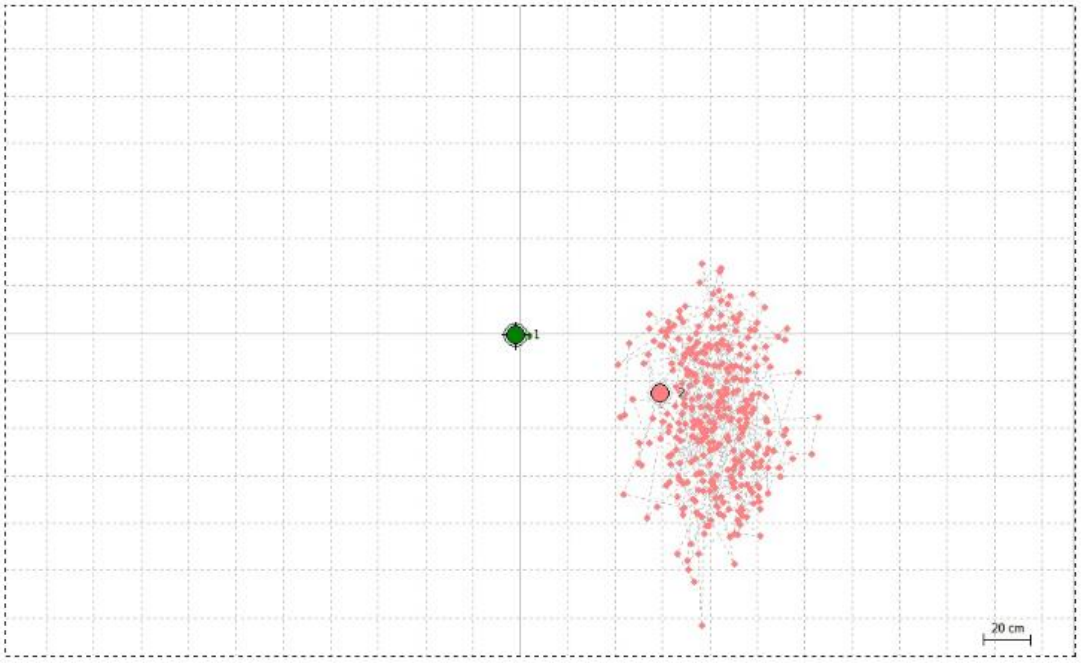

Рисунок 2.11 – Точність визначення координат за допомогою Reach.

Коригувальну інформацію можна отримати з пристрою Reach з двох джерел. По-перше, із загальнодоступної мережі базових станцій через Інтернет через NTRIP (мережевий транспорт RTCM через Інтернет-протокол), який реалізує описану вище ідею щодо глобальної комп'ютерної мережі. По-друге, за допомогою другого Reach він займає нерухоме положення біля першого і, таким чином, є базовою станцією з точки зору DNSS. Другий варіант кращий (точність DNSS сильно зменшується зі збільшенням відстані між приймачем та базовою станцією) - не випадково в рамках кампанії краудфандингу на сайті Indiegogo творці Reach пропонують придбати набір двох пристроїв у першу чергу.

Характеристики Reach:

а) супутниковий приймач: U-blox NEO-M8T — 72 channels, output rate up to 18Hz, supports GPS/QZSS L1 C/A, GLONASS L10F, BeiDou B1, SBAS L1 C/A: WAAS, EGNOS, MSAS, Galileo-ready E1B/C;

б) комп'ютерна платформа: Intel Edison — dual-core 500MHz;

в) інтерфейси: I2C, UART, GPIO, TimeStamp, OTG USB, Bluetooth, Wi-Fi, GNSS;

г) антена: Tallysman TW4721 Dual Feed GPS/BeiDou/Galileo/GLONASS;

ґ) розміри - 26x36 мм;

д) вага - 13 г.

Він складається з 3-х частин обладнання: комп’ютер Intel Edison під управлінням ОС Linux та програмне забезпечення RTKLIB RTK; GPS-приймач U-blox NEO-M8T та антена Tallysman TW4721. Приймач підтримує всі існуючі GPS системи, GLONASS, Beidou та QZSS. Всі ці програмні та апаратні компоненти забезпечують дивовижну точність визначення координат - до 2 см.

2.6 Дослідження завадостійкості, методик оцінки точності визначення координат та швидкості навігаційного приймача GPS і ГЛОНАСС

2.6.1 Методика оцінки точності визначення координат та швидкості навігаційного приймача

Імітатор навігаційних сигналів генерує сигнали систем GPS і ГЛОНАСС на частоті L1 за допомогою контролера з програмним забезпеченням SimGEN [11]. Проведення експерименту приведено нижче схематично на рис. 2.12:

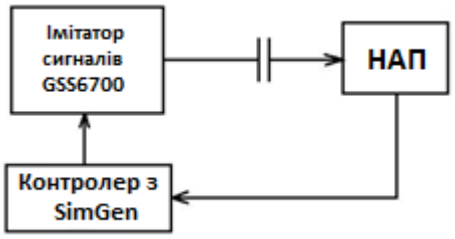

Рисунок 2.12 – Схема експерименту з імітатором навігаційних сигналів

Експеримент проводиться в кілька етапів:

1) у SimGEN пишеться сценарій руху по заданій траєкторії. Для цього за допомогою команди Reference задається точка старту, початкова швидкість і напрямок. Далі за допомогою

команди Combined змінюється швидкість, напрямок руху і висота. Такі зміни виробляються відповідно з траєкторією руху;

2) запускається імітація навігаційних сигналів. На виході імітатора генерується навігаційне поле;

3) навігаційний приймач приймає імітовані сигнали навігаційних супутників, вирішує навігаційну задачу і видає дані на персональний комп'ютер. Для відображення та накопичення даних від НАП використовується середовище u-Center;

4) координати і значення швидкості, виміряні НАП, зберігаються у вигляді таблиць;

5) обчислюється середнє відхилення координат імітованої траєкторії та отриманих від НАП з вихідними траєкторіями рухи за формулою (2.1):

$$X_{ОШ} = \frac{\sum_{i=1}^{n}(x_i - x_i^*)}{n}, \quad (2.1)$$

$X_{ош}$ – середня помилка вимірювання координати НАП; x_i – значення координати, виміряне НАП; x_i^* – координата вихідної траєкторії руху; n-кількість відліків. Наприклад, на рис. 2.13 представлені вихідна траєкторія руху і траєкторія, отримана від НАП. При обчисленні помилки за формулою (2.1) середня помилка становить 11 метрів.

6) обчислюється середнє відхилення реальної швидкості об'єкта від швидкості, що визначається навігаційним приймачем, за формулою (2.2):

$$V_{ОШ} = \frac{\sum_{i=1}^{n}(v_i - v_i^*)}{n}, \quad (2.2)$$

Vош – середня помилка вимірювання швидкості об'єкта навігаційним приймачем; V_i – значення швидкості, виміряне НАП; V_i* – значення реальної швидкості руху об'єкта; n – кількість відліків.

На рис. 2.13 нижче, на графіку показана отримана траєкторія польоту:

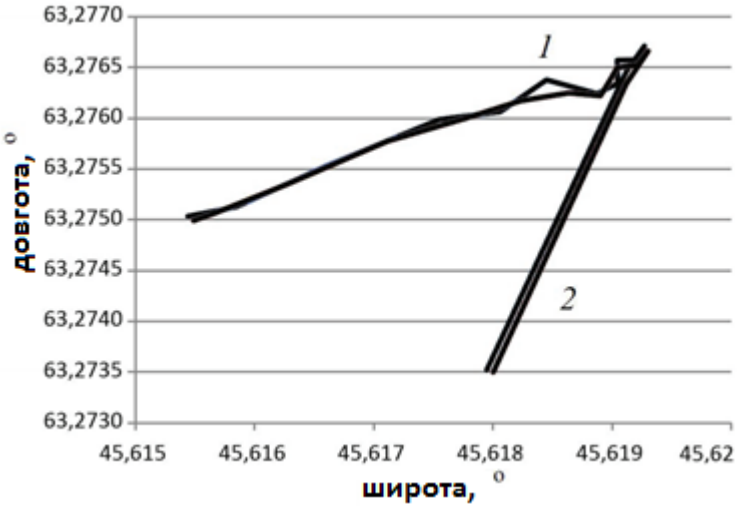

Рисунок 2.13 – Графік траєкторії польоту: 1 - вихідна траєкторія; 2 - траєкторія, отримана від навігаційного приймача

Нижче, на рис. 2.14 представлений графік вихідної залежності швидкості руху об'єкта від часу і швидкості, яка визначається тестованим навігаційним приймачем:

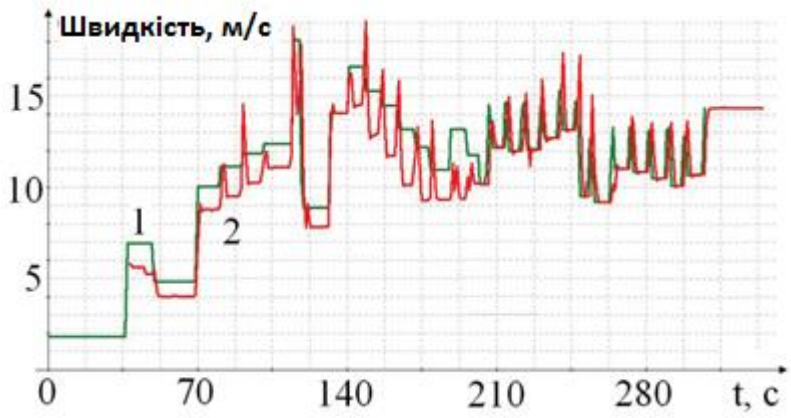

Рисунок 2.14 – Графік залежності швидкості від часу: 1 - залежність швидкості в імітованому сигналі; 2 - швидкість, що визначається навігаційним приймачем

Виходячи з отриманих даних, зробимо кількісну оцінку середньої помилки вимірювання швидкості навігаційним приймач. Використовуючи формулу (2.2), отримуємо, що середня помилка вимірювання швидкості в даному прикладі дорівнює 0,8 м/с.

2.6.2 Методика кількісної оцінки завадостійкості навігаційного приймача

Дослідження завадостійкості приймача НАП GPS і ГЛОНАСС виконується в певному порядку:

1) запускається імітація навігаційного сигналу однієї з систем GPS або ГЛОНАСС;

2) у Програмі U-Center очікується перехід НАП в режим стеження за навігаційними сигналами. Наявність режиму стеження показує значення 3D-параметра Fix Mode у вікні Data;

3) задається вид, виставляється несуча частота і потужність перешкоди і починається її генерація. Для цих цілей використовується векторний генератор Agilent N5272, який дозволяє створювати високочастотні коливання без модуляції і з амплітудною, частотною або фазовою модуляцією;

4) оцінка дії перешкоди ведеться за графіком залежності похибки вимірювання координат навігаційним приймачем від часу (параметр PACC3D у GPS і параметр PDOP у ГЛОНАСС в Програмі u-Center). Приклад даної залежності наведено на рис. 2.15;

5) з отриманого графіка фіксується значення похибки вимірювання координат, відповідне сталому режиму:

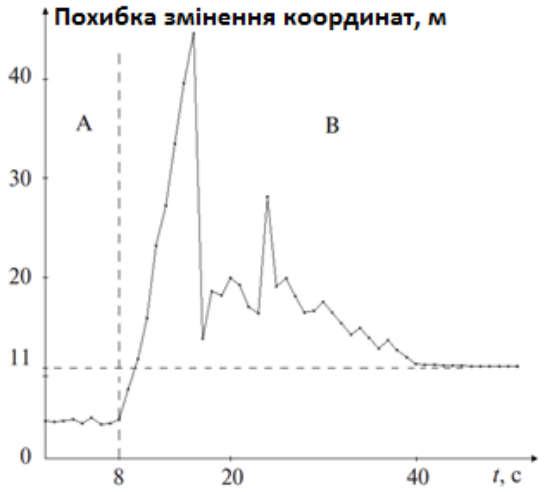

Рисунок 2.15 – Графік похибки вимірювання координат навігаційним приймачем від часу: область А – без перешкоди, область В – з перешкодою

6) змінюється рівень потужності перешкоди і визначається похибка вимірювання координат приймача НАП. Таким чином, знімається характеристика супроводу приймача при наявності перешкоди на вході.

7) в результаті знаходиться значення порогового відношення перешкода/сигнал по потужності, при якому відбувається зрив стеження за сигналами від навігаційних супутників. Зрив стеження показує значення No fix параметра Fix Mode у вікні Data. Кількісно

завадостійкість оцінюється коефіцієнтом приглушення, який чисельно дорівнює максимальному відношенню перешкода/сигнал, при якому НАП ще визначає свої координати.

2.6.3 Приклад дослідження завадостійкості навігаційного приймача GPS і ГЛОНАСС

Було проведено дослідження завадостійкості за описаною в попередньому пункті методикою. При дослідженні завадостійкості GPS-приймача розглядалися два типи перешкод: гармонійна на частоті 1575,42 МГц і частотно-модульована з центральною частотою 1575,42 МГц і девіацією частоти 1 МГц. При дослідженні завадостійкості приймача ГЛОНАСС розглядалися два типи перешкод: гармонійна на частоті 1602 МГц і частотно-модульовані з центральною частотою 1602 МГц і девіацією частоти 5 МГц. Отримані характеристики показані на рис. 2.16 для приймача GPS суцільною лінією, а для приймача ГЛОНАСС – пунктирний.

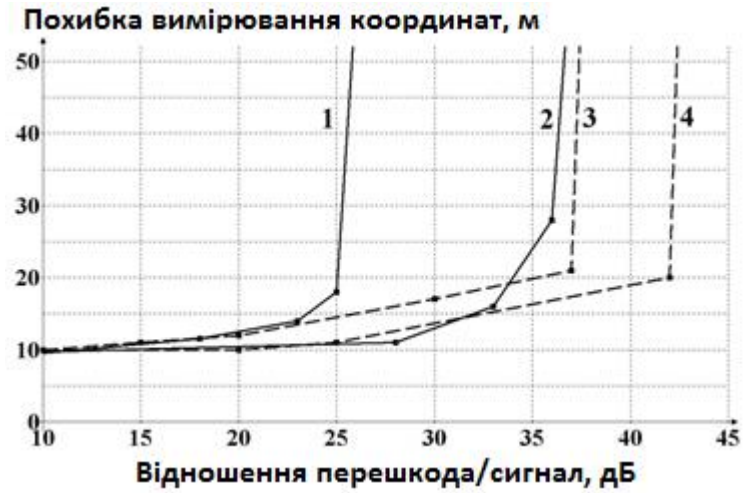

Рисунок 2.16 – Графік похибки вимірювання координат від відношення перешкода/сигнал: 1, 3 – частотно-модульована перешкода; 2, 4 – гармонійна перешкода

Кількісна оцінка перешкодостійкості GPS-приймача наступна: коефіцієнт приглушення для частотномодульованої перешкоди дорівнює 25 дБ, а для гармонійної – 36 дБ. Кількісна оцінка перешкодостійкості приймача ГЛОНАСС наступна: коефіцієнт придушення для частотно-модульованої перешкоди дорівнює 37 дБ, а для гармонійної – 42 дБ. Таким чином, широкосмугова частотно-модульована перешкода сильніше впливає на роботу навігаційних приймач. Завадостійкість приймачів системи ГЛОНАСС вище, ніж у нап GPS.

2.7 Вибір середовища розробки

Android Studio – інтегроване середовище розробки, виробництва Google, за допомогою якої розробникам стають доступні інструменти для створення додатків на платформі OS Android. Android Studio можна встановити на Windows, Mac і Linux. Обліковий запис розробника додатків в Google Play App Store коштує $ 25. Android Studio створювалася на базі IntelliJ IDEA [12].

IDE можна завантажити і користуватися безкоштовно. У ній присутні макети для створення UI, з чого зазвичай починається робота над додатком. В Studio містить інструменти для розробки рішень для смартфонів і планшетів, а також нові технологічні рішення для Android TV, Android Wear, Android Auto Glass і додаткові контекстуальні модулі.

Середа Android Studio призначена як для невеликих команд розробників мобільних додатків (навіть в кількості однієї людини), або ж великих міжнародних організацій з GIT або іншими подібними системами керування версіями. Досвідчені розробники зможуть вибрати інструменти, які більше підходять для масштабних проектів. Рішення для Android розробляються в Android Studio з використанням Java або C++. В основі робочого процесу Android Studio закладений концепт безперервної інтеграції, що дозволяє відразу ж виявляти наявні проблеми. Тривала перевірка коду забезпечує можливість ефективного зворотного зв'язку з розробниками. Така опція дозволяє швидше опублікувати версію мобільного додатку в Google Play App Store. Для цього є також підтримка інструментів LINT, Pro-Guard і App Signing [13].

З допомогою засобів оцінки продуктивності визначається стан файлу з пакетом прикладних програм. Візуалізація графіки дає можливість дізнатися, чи відповідає додаток орієнтиру Google в 16 мілісекунд. За допомогою інструменту для візуалізації пам'яті Розробник дізнається, коли його додаток буде використовувати занадто багато оперативної пам'яті і коли відбудеться «збірка сміття». Інструменти для аналізу батареї показують, яке навантаження припадає на пристрій.

Android Studio сумісна з платформою Google App Engine для швидкої інтеграції в хмарі нових API і функцій. У середовищі розробки ви знайдете різні API, такі як Google Play, Android Pay і Health. Присутня підтримка всіх платформ Android, починаючи з Android 1.6. Є варіанти Android, які істотно відрізняються від версії Google Android. Найпопулярніша з них — це Amazon Fire OS. В Android Studio можна створювати APK для цієї ОС.

2.8 Використання Java Development Kit для роботи з Android Studio

Для того щоб написану програму можна було скомпілювати і виконати, потрібно встановити на своєму комп'ютері комплект Java Development Kit, підтримуючий процес розробки

програм на Java. Цей комплект вільно надається компанією Oracle. Для роботи з Android Studio потрібна версія JDK 8, що застосовується на платформі Java SE 8, де SE-абревіатура від Standard Edition (стандартний випуск). Комплект JDK 8 містить чимало нових засобів, які не підтримуються в попередніх версіях Java, і тому для компіляції і виконання прикладів програм, рекомендується використовувати JDK 8. У разі використання більш ранніх версій програми, що містять нові засоби, не можуть бути скомпільовані [14].

Достатньо перейти на офіційну сторінку Oracle та завантаження слідуючи інструкціям для вашого типу комп'ютера і операційної системи. Після установки JDK можна компілювати і запускати програми. Серед багатьох програм, що входять в склад JDK, вам в першу чергу знадобляться дві: javac і java. Перша з них – це компілятор Java, а друга - стандартний інтерпретатор Java, який також називають модулем запуску додатків.

Слід, однак, мати на увазі, що програми, що входять до складу JDK, запускаються з командного рядка. Вони не є віконними додатками і не складають інтегроване середовище розробки (IDE).

2.9 Використання мови XML в Android Studio

Мова XML була створена для зберігання, транспортування та обміну даними, з його допомогою можна реалізувати обмін даними між різними системами. XML документ складається з частин, званих елемент. Елементи складають основу XML-документів. Вони утворюють структури, які можна обробляти програмно або за допомогою таблиць стилів. Елементи розміщують іменовані розділи інформація. Елементи будуються за допомогою тегів розмітки, позначають ім'я, початок і кінець елемента. Елементи можуть бути вкладені один в одного, на верхньому рівні знаходиться елемент, званий елементом документа або кореневим елементом в якому містяться інші елементи [15].

Документ XML може розташовуватися в одному або декількох файлах, причому деякі з них можуть перебувати на різних машинах. У XML використовується спеціальна розмітка для інтеграції вмісту різних файлів в один об'єкт, який можна охарактеризувати як логічну структуру. Завдяки тому, що документ не обмежений одним файлом, XML дозволяє створювати документ з частин, які можуть розташовуватися де завгодно.

Документ XML зазвичай містить наступні розділи:

а) xml-декларація;

б) пролог;

в) елемент;

г) атрибут;

ґ) коментар.

2.10 Висновки до розділу 2

В другому розділі було проведено огляд та порівняння технологій GPS та A-GPS, розглянуті та проаналізовані системи та методи локального позиціонування об’єктів. Також був досліджений метод зниження похибки GPS на ОС Android за допомогою фільтрів та датчиків. Розглянуто та досліджено систему позиціонування нового покоління в Uber. Проведено вибір середовища розробки, розглянуто використання Java Development Kit для роботи з Android Studio, та використання мови XML в Android Studio. Також розглянуто визначення місцезнаходження GPS за допомогою пристрою Reach, та проведені дослідження завадостійкості, методик оцінки точності визначення координат та швидкості навігаційного приймача GPS і ГЛОНАСС.

3 ПРАКТИЧНА РЕАЛІЗАЦІЯ ANDROID-ДОДАТКА НАВІГАТОРА

3.1 Робота з GPS в Android на Java

В Android SDK весь функціонал по роботі з навігаційними системами об'єднаний в пакет android.location. Ключові компоненти даного пакету, це:

а) LocationManager - (клас) забезпечує доступ до системної службі визначення місцезнаходження Android;

б) LocationListener - (інтерфейс) регламентує обробку додаток подій служби визначення місцезнаходження Android;

в) Location - (клас) представляє географічні координати отримані від навігаційної системи.

При написанні Android-додатка працючого з навігаційними системами на Java за допомогою Android SDK спочатку необхідно виконати ряд підготовчих операцій [16].

Це пов'язано з тим, що на відміну від Delphi, тут відсутні будь-які дозволи, що надаються за замовчуванням і немає готових компонентів, які повністю брали б на себе всю роботу по взаємодії з GPS приймачем. Всі необхідні дії потрібно виконати самостійно. Насамперед надаємо додатком необхідні дозволи в файлі маніфесту.

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

Далі створюємо в коді програми об'єкт LocationListener для обробки подій служби визначення місцезнаходження Android.

```
private LocationListener listener = new LocationListener() {
@Override
public void onLocationChanged(Location location) {
}
@Override
public void onStatusChanged(String provider, int status, Bundle extras) {
}
@Override
public void onProviderEnabled(String provider) {
}
@Override
public void onProviderDisabled(String provider) {
}
};
```

Призначення його методів - обробка відповідних подій, конкретно:

а) onLocationChanged - зміна місця розташування. Саме він використовується для визначення поточних географічних координат;

б) onStatusChanged - зміна стану постачальника даних про місцезнаходження. Зокрема приймача GPS;

в) onProviderEnabled - отримання доступу до постачальника даних про місцезнаходження;

г) onProviderDisabled - втрата доступу до постачальника даних про місцезнаходження.

На завершення зареєструємо створений нами об'єкт LocationListener для роботи з приймачем GPS. Для цього створюємо екземпляр класу LocationManager.

```
LocationManager manager = (LocationManager)
getSystemService(Context.LOCATION_SERVICE);
```

І виконуємо реєстрацію за допомогою методу requestLocationUpdates.

```
If (ActivityCompat.checkSelfPermission(this,
Manifest.permission.ACCESS_FINE_LOCATION)! = PackageManager.PERMISSION_GRANTED) {
// Перевірка наявності дозволів
// Якщо немає дозволу на використання відповідних дозволів виконуємо якісь дії
return;
}
manager.requestLocationUpdates (LocationManager.GPS_PROVIDER, 0, 0, listener);
```

Перед викликом методу requestLocationUpdates обов'язково необхідно перевірити наявність відповідних дозволів (оператор if). Якщо вони відсутні перед оператором return можна виконати деякі дії. Наприклад, записати повідомлення про помилку в журнал. Однак в будь-якому випадку при відсутності необхідних дозволів робота з навігаційною системою повинна бути завершена до реєстрації об'єкта LocationListener.

Метод requestLocationUpdates має кілька перевантажень. Найбільш часто використовувана з них приймає чотири параметри. Саме вона використана в прикладі вище.

а) постачальник даних про місцезнаходження. В даному прикладі використовується GPS;

б) мінімальний інтервал оновлення даних про місцеположення в мілісекундах. Значення «0» відповідає використанню мінімально можливого інтервалу часу для даного пристрою;

в) мінімальна відстань для поновлення даних про місцезнаходження в метрах. Значення «0» відповідає використанню мінімально можливої відстані для даного пристрою;

г) реєстрований об'єкт LocationListener.

Після реєстрації додаток зможе отримувати інформацію про місцезнаходження пристрою в міру його зміни. Якщо необхідно отримати її одноразово, необхідно замість методу requestLocationUpdates використовувати метод requestSingleUpdate, який також має кілька перевантажень. Найбільш затребувана з них приймає три параметри:

а) постачальник даних про місцезнаходження. В даному прикладі використовується GPS;

б) реєстрований об'єкт LocationListener;

в) об'єкт, який реалізує зворотний виклик. Необов'язковий параметр.

Приклад використання методу requestSingleUpdate:

```
manager.requestSingleUpdate (LocationManager.GPS_PROVIDER, listener, null);
```

Отримання географічних координат від GPS приймача можливо в подію onLocationChanged об'єкта LocationListener після його реєстрації.

Нижче наведено приклад виведення значень поточних широти і довготи на екран пристрою в два елементи TextView.

```
public void onLocationChanged (Location location) {
if (location! = null) {
lathitude.setText (String.valueOf (location.getLatitude ()));
longitude.setText (String.valueOf (location.getLongitude ()));
}
else {
lathitude.setText ("Sorry, location");
longitude.setText ("unavailable");
}
}
```

Якщо місце розташування визначити не вдалося (location == null), то в цих же елементах буде показано відповідне повідомлення.

Значення географічних координат повертаються в десятковому поданні. Тому для відображення в «звичайному» форматі (градуси-хвилини-секунди) необхідна додаткова обробка.

3.2 Проектування архітектури додатка-навігатора для ОС Android

Основними складовими частинами Android-додатка є активності, кожна активність відповідає одному з екранів додатка, наприклад: головне меню, пункт меню, мапа, та інше. У файлах xml - код розмітки, який вміщує опис кожної активності, розташування об’єктів

візуалізації у вигляді коду xml. Коли виконується запуск активності, автомамично розпізнається розмір екрану системою Android, та завдяки розмітці файлу xml виводить на екран відповідний контент, щоб активність могла виглядати однаково на різних пристроях, з різними діагоналями екрану [13].

При розробці дизайну додатка-навігатора основний акцент був поставлений на простоту та зручність використання новими користувачами додатка, при цьому функціонування знаходиться на високому рівні. З активностями додатка не прийдеться довго ознайомлюватись, вони несуть в собі всю основну необхідну інформацію та дуже легкі для пошуку необхідної інформації. Питання навігації дуже важливе по відношенню до розробки додатків, якщо інформація потрібна користувачеві була там, але він не зміг її знайти, то це велика проблема для розробника, це і погані відгуки, погані оцінки додатка, і це веде за собою відсіювання великої кількості потенційних користувачів, яких в жодному разі втрачати не можна.

Додаток-навігатор взаємодіє з сервісами Google API, які передають данні з Google за допомогою підключення сформованого API key. Для користування додатком, були підключені такі сервіси як: Places API, Google+ API, та їм подібні. Вони необхідні для отримання позначок на мапі, та інформації про них. Цей підхід дуже зручний, тому його активно використовують розробці подібних додатків к використанням мап та GPS.

3.3 Створення і підключення Google Maps Api Key

Ключ можна отримати безкоштовно в версії «free trial» версії на консолі Google Developer Console, як показано нижче на рис. 3.1:

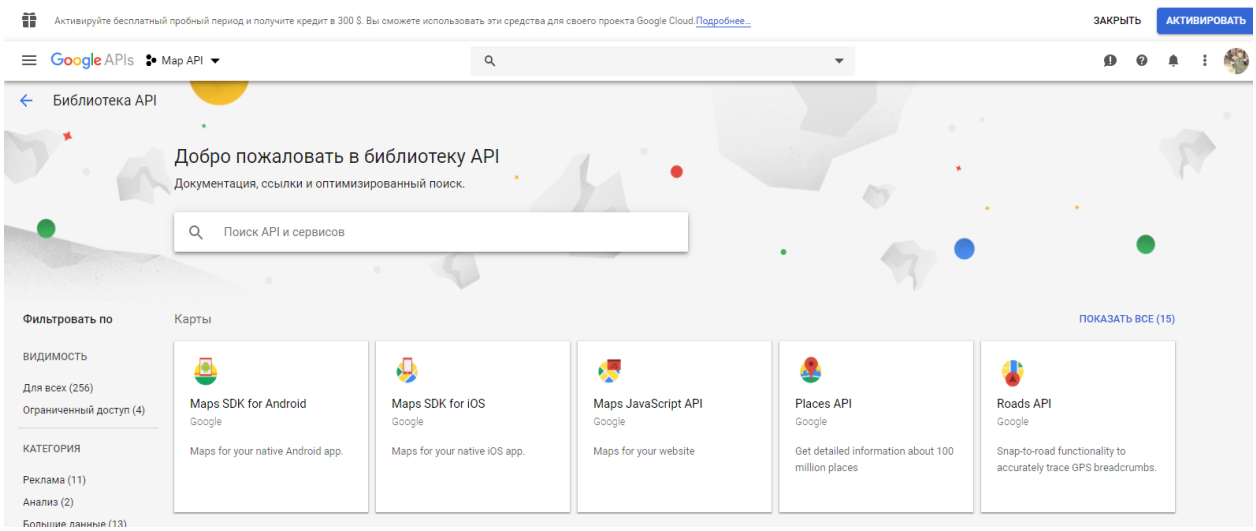

Рисунок 3.1 – Google Developer Console

Щоб розпочати треба натиснути > Create a new Project (Створити новий проект). Потрібно обрати сервіси, які вам потрібні, в нашому випадку це: Places API, Places SDK for Android, Google SDK Android API, Google+ API [3]. Після вибору всього необхідного можна створити облікові дані, вибравши у спливаючому вікні API key, як це показано в наступному прикладі на рис. 3.2:

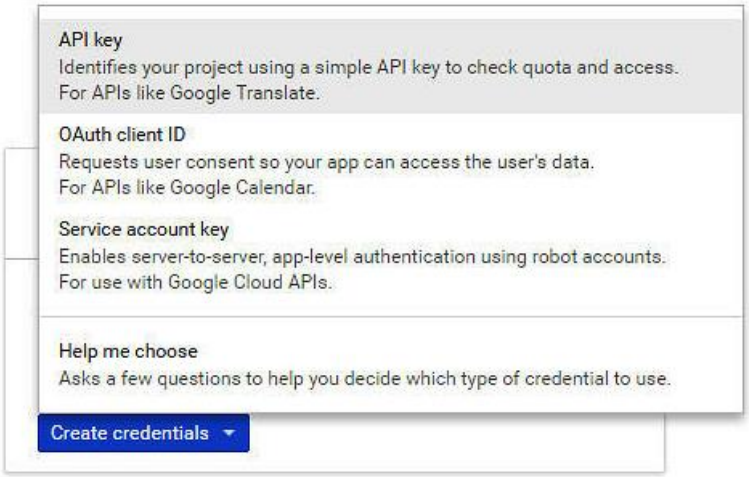

Рисунок 3.2 – Створення нового проекту API key

Після цього вам буде запропоновано ввести ім'я для вашого ключа прикладного інтерфейсу Android API, як це показано в наступному прикладі на рис. 3.3:

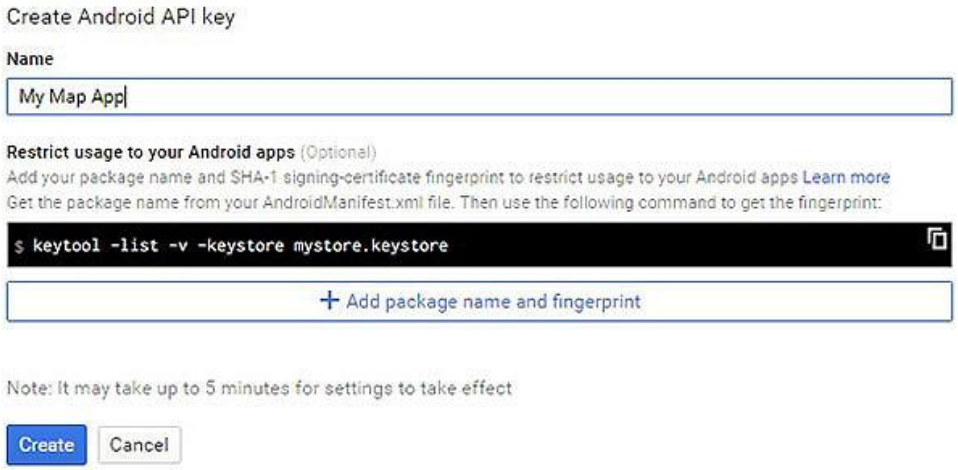

Рисунок 3.3 – Створення імені API key

Після завершення дій з призначенням імені ключа API натисніть кнопку> Create (Створити). Тепер у нас є ключ Android API, приклад нижче на рис. 3.4:

Рисунок 3.4 – Отриманий API key

Для роботи додатка отриманий API key потрібно вставити в файл strings.xml:

```
<string name="google_api_key">Отриманий_API_key</string>
```

3.4 Розробка програмного коду GooglePlaces.java

Приведений нижче фрагмент коду файлу GooglePlaces.java, описує роботу виконання запитів на відображення оточуючих місць з обраних категорій на мапі всередині додатка, завантажує маркери-позначки з результатами їх розміщення.

```
public class GooglePlaces {
private final LatLng mLocation;
public GooglePlaces(LatLng location) {
this.mLocation = location;
}
Set<Place> resultPlaces = new HashSet<Place>();
public void getPlacesNearby(NearbySearchQuery query, CallbackNearbyPlaces
callbackNearbyPlaces, Constants.PLACE_TYPES type)
throws JSONException, IOException, InterruptedException {
Log.i("Loaded_Markers", "Executed query = " + query.toString());
PlacesResult result = new
PlacesResult(NetworkFetcher.executeRequest(query.toString(), false));
callbackNearbyPlaces.onPlacesLoaded(result.getPlaces(), type);
Log.i("Loaded_Markers", "NextPageToken " + result.getNextPageToken());
}
```

3.5 Розробка програмного коду AdapterPlaces.java

Приведений нижче фрагмент коду файлу AdapterPlaces.java працює з розташуванням інформації про обрані з трьох категорій заклади: рівня цін, рейтинг закладу, фотографії закладу, телефон, мітки, та адреса закладу.

```
@Override
public int getItemCount() {
return mAdapterData.size();
}
public static class ViewHolder extends RecyclerView.ViewHolder {
private View mParentView;
private LinearLayout mPriceLevelStarsContainer;
private LinearLayout mRatingLevelStarsContainer;
private ImageView mImageView;
private TextView mTextViewTitle;
private TextView mTextViewPriceLevelValue;
private TextView mTextViewRatingLevelValue;
public ViewHolder(View itemView) {
super(itemView);
mParentView = itemView;
mImageView = (ImageView) itemView.findViewById(R.id.list_item_places_image);
mTextViewTitle = (TextView) itemView.findViewById(R.id.list_item_places_title);
mTextViewPriceLevelValue = (TextView)
itemView.findViewById(R.id.list_item_places_price_level_value);
mPriceLevelStarsContainer = (LinearLayout)
itemView.findViewById(R.id.list_item_places_price_level_stars);
mRatingLevelStarsContainer = (LinearLayout)
itemView.findViewById(R.id.list_item_places_rating_level_stars);
mTextViewRatingLevelValue = (TextView)
itemView.findViewById(R.id.list_item_places_rating_value);
}
}
private LayoutInflater getLayoutInflater() {
return LayoutInflater.from(mContext);
}
}
```

3.6 Розробка програмного коду MainActivity.java

Приведений нижче фрагмент коду файлу MainActivity.java відповідає роботу головної активності додатка: відображення оточення користувача, його місцезнаходження, відображення детальної інформації про місце, список місць, повернення на головну активність, прогноз погоди, відображення фотографій, оцінки місць.

```
@Override
```

```
public void run() {
NearbyPlacesManager.getInstance();
}
}, 1000);
new AboutDataLoader().loadData(this);
initAdmob();
}
public void showNearbyFragment(Constants.PLACE_TYPES type, Place placeDetail,
boolean showCurrentLocation) {
showScreen(FragmentNearby.newInstance(type, placeDetail, showCurrentLocation),
FragmentNearby.TAG, true, false);
}
public boolean isNearbyFragmentShowed() {
return getSupportFragmentManager().findFragmentByTag(FragmentNearby.TAG) != null;
}
public void showListFragment(Constants.PLACE_TYPES type) {
showScreen(FragmentListPlaces.newInstance(type), FragmentListPlaces.TAG, true,
false);
}
private void showHomeFragment() {
showScreen(FragmentHome.newInstance(), FragmentHome.TAG, false, false);
}
public void showForecastFragment() {
showScreen(FragmentForecast.newInstance(), FragmentForecast.TAG, true, false);
}
public void showAboutFragment() {
showScreen(FragmentAbout.newInstance(), FragmentAbout.TAG, true, false);
}
public void showLoadingDialog() {
DialogFragmentLoading.newInstance().show(getSupportFragmentManager(),
DialogFragmentLoading.TAG);
}
public void showPhotoDialog(Photo photo) {
DialogFragmentPhoto.newInstance(photo).show(getSupportFragmentManager(),
DialogFragmentPhoto.TAG);
}
public void showPlaceDetailFragment(Place place, Constants.PLACE_TYPES type) {
FragmentPlaceDetail fragmentPlaceDetail = FragmentPlaceDetail.newInstance(place,
type);
showScreen(fragmentPlaceDetail, FragmentPlaceDetail.TAG, true, false);
}
public void showReviewsScreen(PlaceDetails placeDetails, Constants.PLACE_TYPES
type) {
```

```
FragmentReviews fragmentReviews = FragmentReviews.newInstance(placeDetails, type);
showScreen(fragmentReviews, FragmentReviews.TAG, true, false);
}
public void hideLoadingDialog() {
Fragment loadingFragment =
getSupportFragmentManager().findFragmentByTag(DialogFragmentLoading.TAG);
if (loadingFragment != null) {
((DialogFragmentLoading) loadingFragment).dismiss();
}
}
```

3.7 Описання функціональності розробленого додатка-навігатора

Розроблений додаток-навігатор для ОС Andriod по місту Сєвєродонецьк. З його допомогою можна виявити своє оточення на мапі, додаток працює напряму з сервісами Google API, що дає змогу отримувати точну інформацію про місця які потрібно знайти на мапі. В додаток влаштована мапа Google, якщо користувачу потрібно прокласти маршрут до обраного місця на мапі всередині додатка, то натиснувши відповідну кнопку він буде перенаправлений в Google Maps, де обраний маршрут вже буде побудований. Додаток пропонує на головній активності вибір місць по трьох категоріях: розваги, харчування, відпочинок. Обравши потрібну категорію можна легко відсіявши всі зайві знайти наприклад кафе, та перекусити. Також можна відключати зайві категорії всередині додатка на мапі, категорії знаходяться в лівому куту мапи, вони підписані та позначені різними кольорами.

Інформацію про заклади, можна отримати натиснувши по ним на мапі, або обравши зі списку в категоріях.

Також у додатку є дві приємних активності, такі як «Погода», та «Інформація про місто». В активності «Погода» відображається: день неділі, місяць, рік, та звичайно сама погода. В активності «Інформація про місто» відображаються: фотографії значних місць міста, та невелика інформація про місто.

Структура додатка на ОС Android та схема обслуговування користувача додатка представлені нижче на рис. 3.5 та 3.6 відповідно:

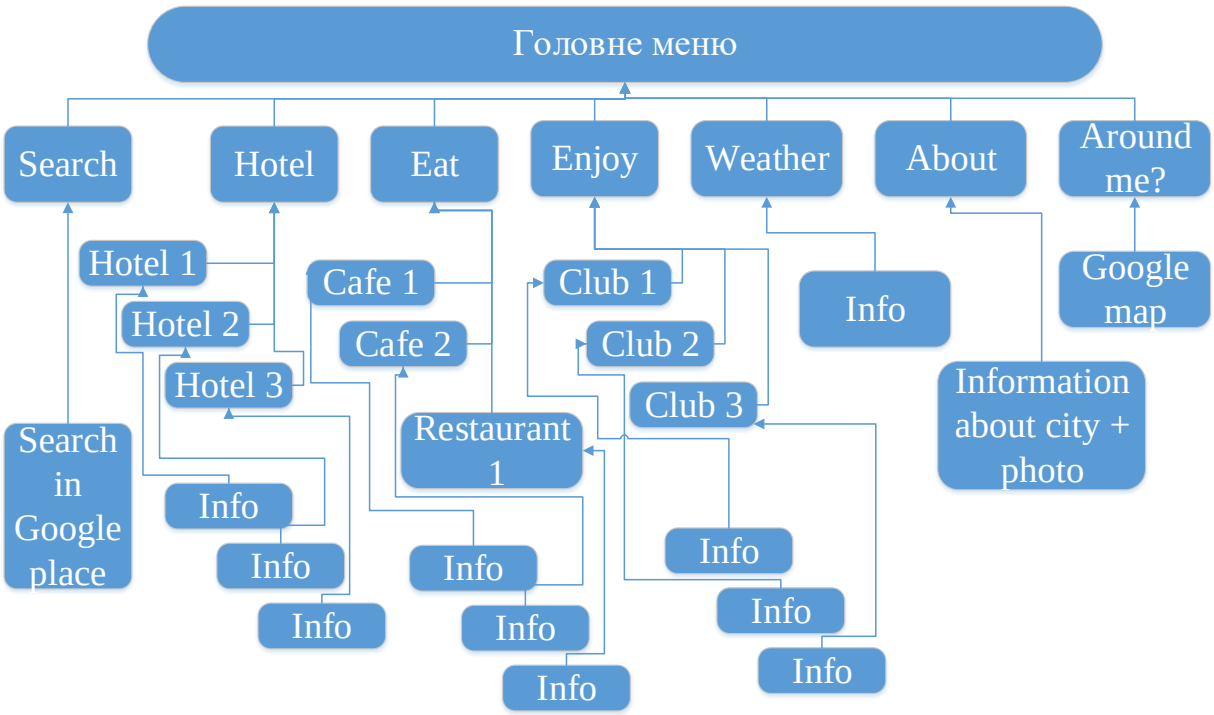

Рисунок 3.5 – Структура додатка на ОС Android

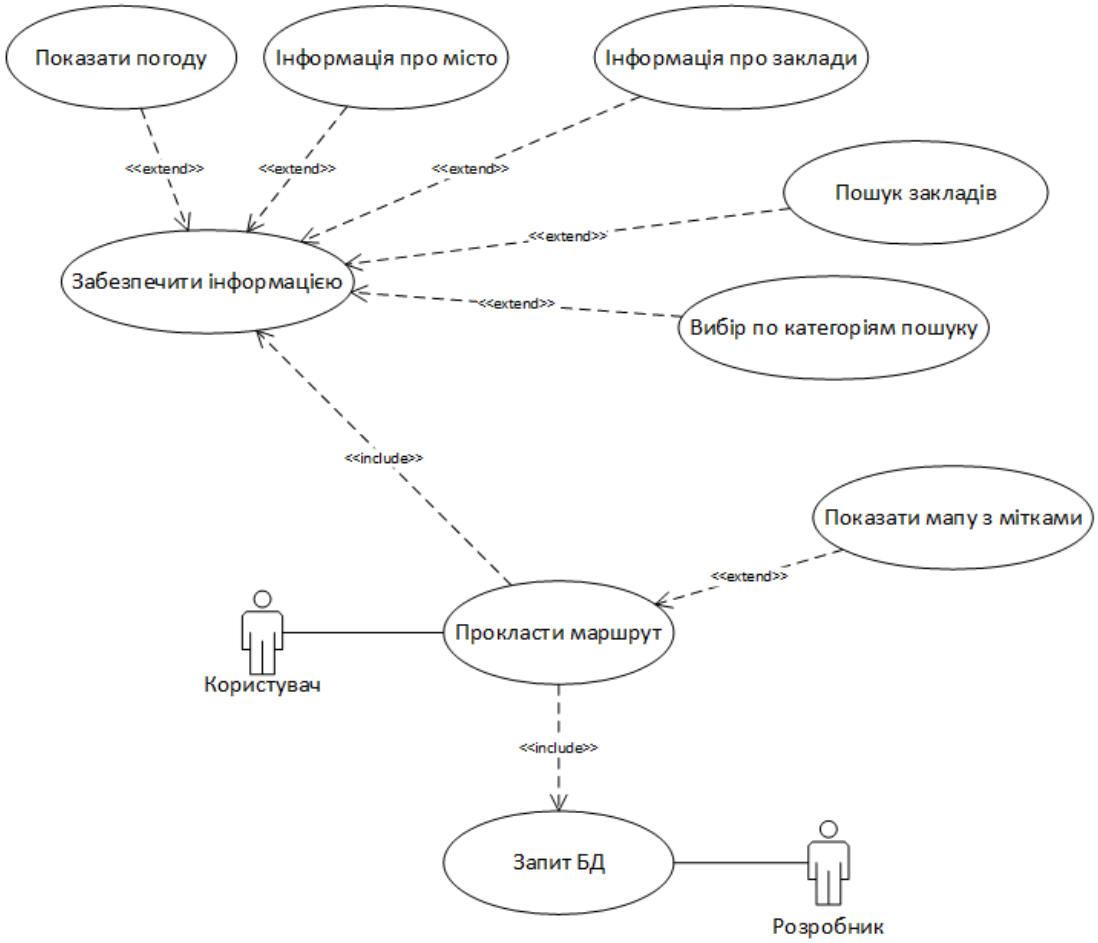

Рисунок 3.6 – Схема обслуговування користувача додатка

Нижче на рис. 3.7 приведений приклад головної активності додатка. Угорі можна бачити активність з рядком пошуку, увівши потрібне користувачеві місце, йому буде показаний випадаючий список, з відповідними закладами до його запиту, одразу з ними відображається їх рейтинг. Завдяки цій інформації користувач може обрати для себе самий вдалий для нього варіант, приклад на рис. 3.8. Також там можна відкрити активність з улюбленими місцями, які були туди раніше збережені. Ще є переміщення у Google Play Market, щоб поставити оцінку додатку, коли він буде доданий туди.

Нижче є три основні кнопки, що є категоріями: 1 – це відпочинок; 2 – харчування; 3 – розваги. При натисканні на кожну з цих кнопок користувач буде переведений в активність з відповідними цій категорії закладами.

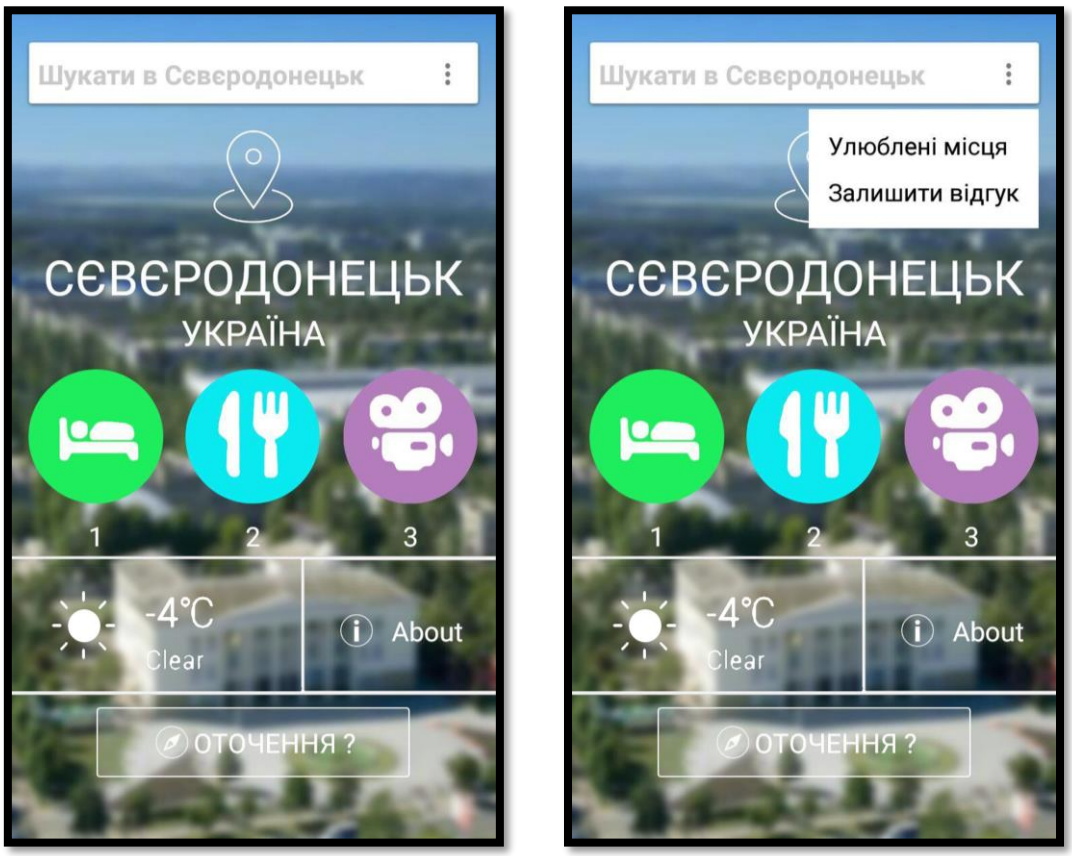

Рисунок 3.7 – Головна активність додатка

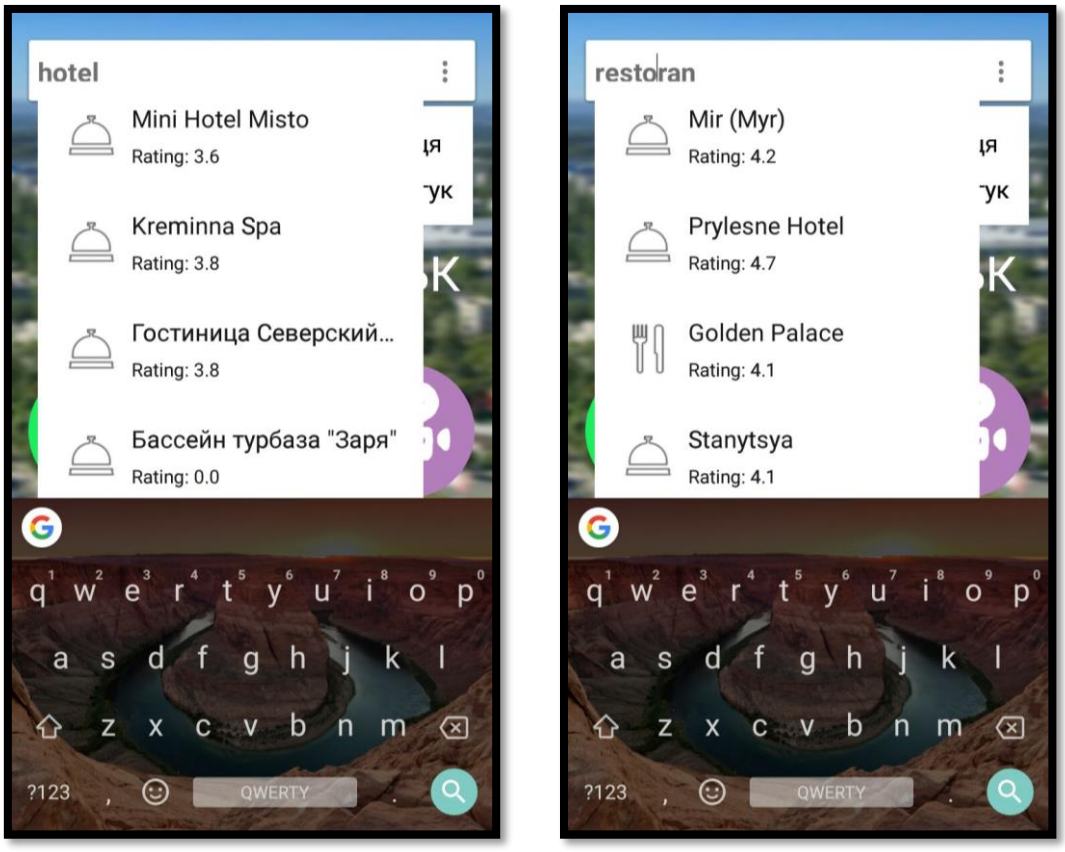

Рисунок 3.8 – Випадаючий список з результатами введеного запросу

Трохи нижче знаходиться кнопка погоди, на ній відображається яка саме зараз погода, та якщо натиснути на неї користувач буде переведений в активність, де зможе подивитися погоду на всю неділю. Користувачеві додатка буде показано: день неділі, число, рік, та погода на цей день. Приклад можна побачити нижче на рис. 3.9:

Рисунок 3.9 – Активність з погодою

Також поряд з кнопкою погоди, справа є кнопка About з інформацією про місто Сєвєродонецьк. Натиснувши її користувач буде переведений у активність з текстовою інформацією про місто та декількома фотографіями з значними місцями, що може бути приємним бонусом для приїжджих в це місто вперше. Приклад активності з інформацією, та фотогалереєю показано нижче на рис. 3.10 та 3.11:

Рисунок 3.10 – Активність з інформацією про місто, та фотогалереєю

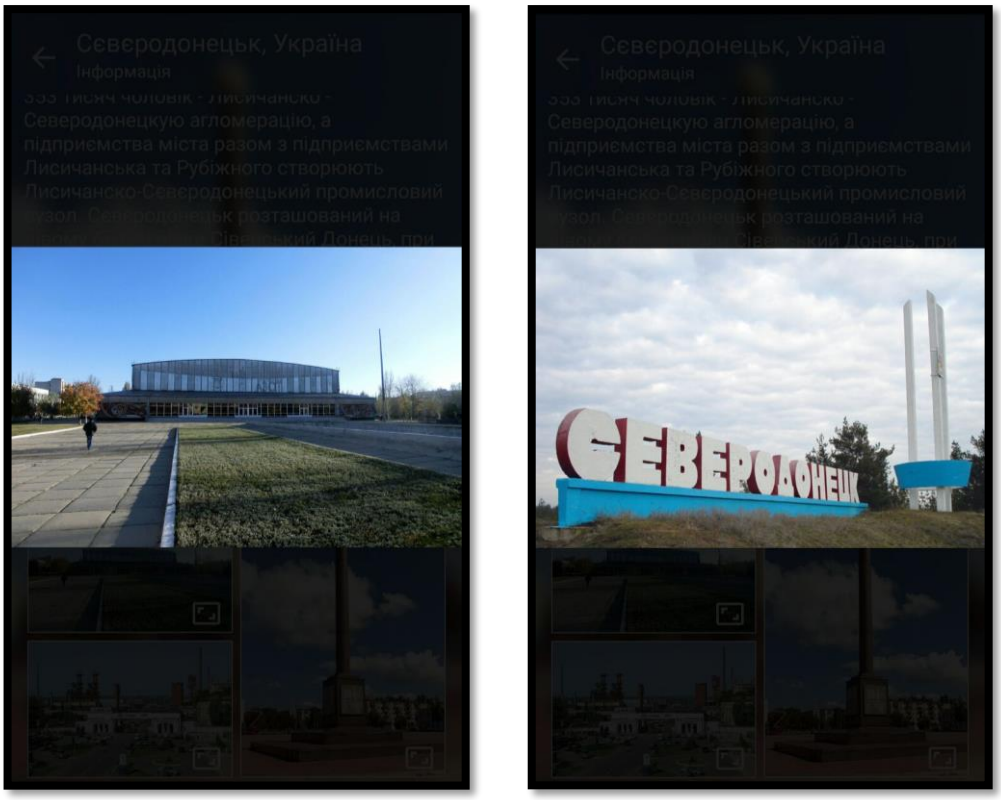

Рисунок 3.11 – Активність з інформацією про місто, та фотогалереєю. Приклад відкривання перших двох фотографій

Знизу знаходиться остання кнопка головної активності, вона має назву Around Me, при натисканні на неї користувач додатка може подивитись оточуючі його об’єкти на мапі убудованої всередині додатка. Приклад наведено нижче на рис. 3.12, де всі категорії активні, що видно зліва внизу:

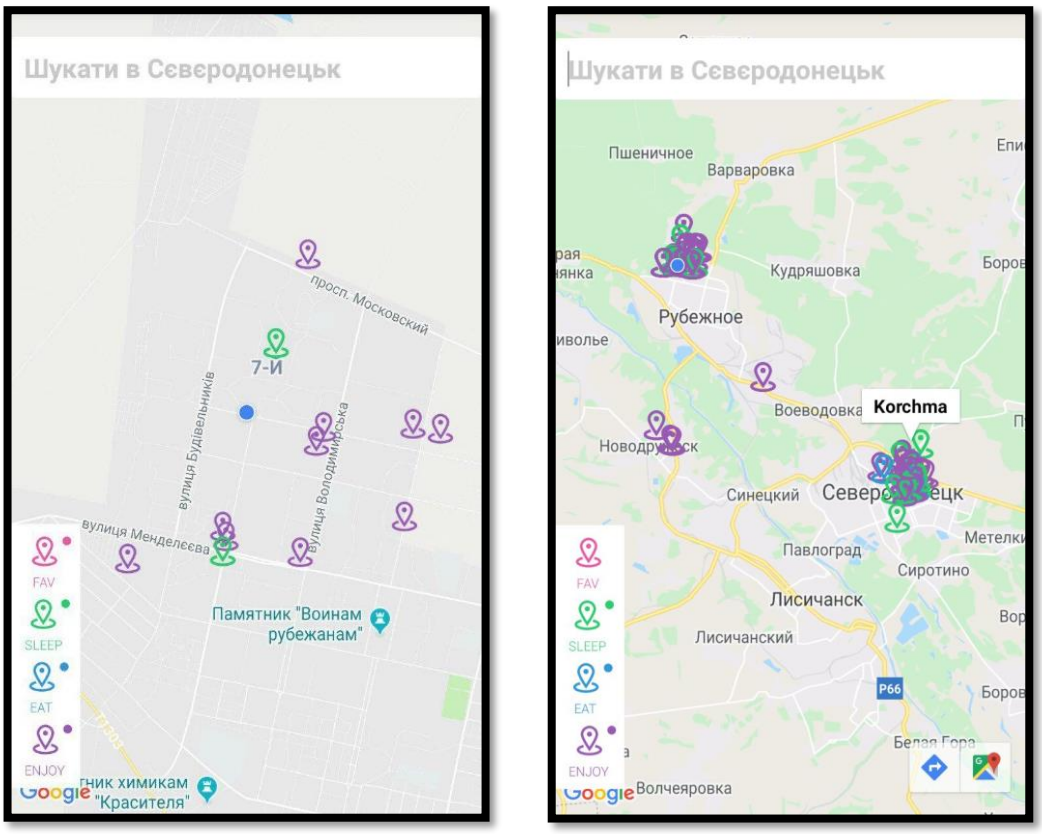

Рисунок 3.12 – Активність з оточенням користувача на мапі

В цій активності показаний приклад запуску кнопки категорії 1 – відпочинок. Користувач отримує список всіх готелів, хостелів, квартир, які здаються, та їх рейтинги.

При бажанні користувач додатка може скористатися швидким пошуком по назві закладу натиснувши на відповідну позначку лупи угорі, також поряд з нею позначка мапи, натиснувши на яку можна подивитися відповідні цій категорії місця на мапі, щоб подивитися наприклад як далеко заклади один від одного знаходяться. Приклад нижче на рис. 3.13:

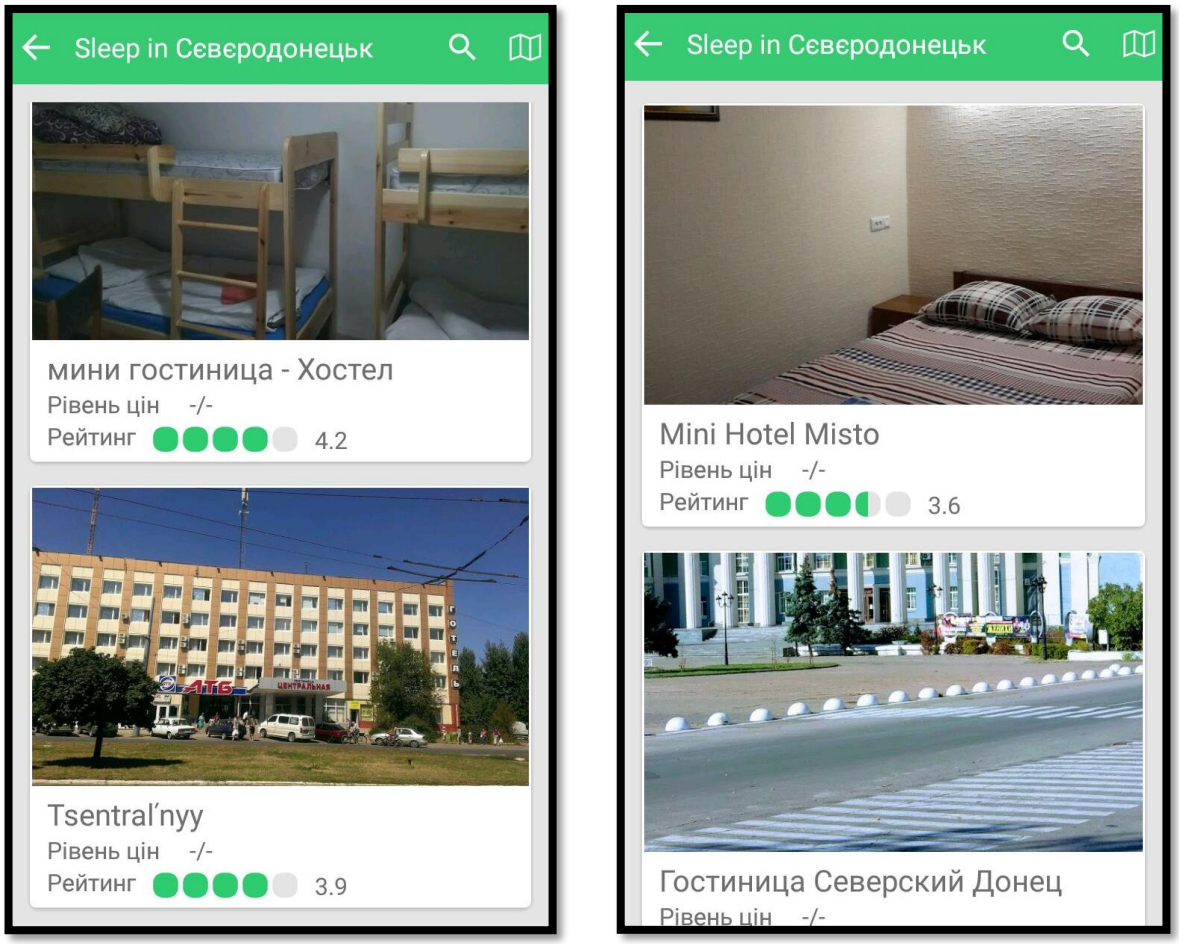

Рисунок 3.13 – Активність зі списком закладів для відпочинку

На наступній активності показаний приклад запуску кнопки категорії 2 – де показаний список закладів харчування. Користувач отримує список всіх кафе, ресторанів, барів, закусочних, столових, кав’ярень, піццерій, суші-барів та їх рейтинги. Також присутній швидкий пошук за назвою, та показ на мапі. Приклад нижче на рис. 3.14:

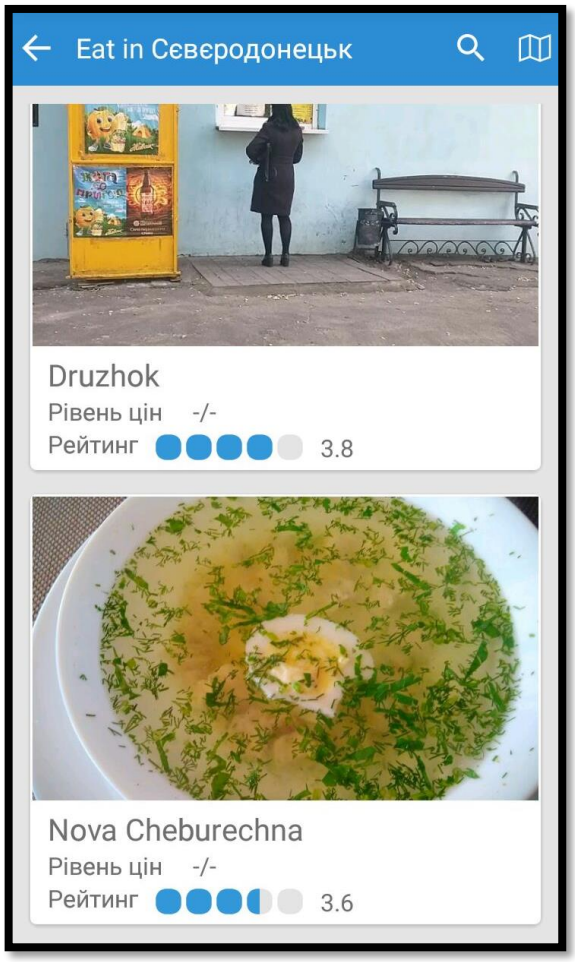

Рисунок 3.14 – Активність зі списком закладів харчування

На наступній активності показаний приклад запуску кнопки категорії 3 – де показаний список закладів для розваг. Користувач отримує список всіх розважальних центрів, кінотеатрів і т.п., та їх рейтинги, якщо данні з Google звичайно надають їх як розважальні, також користувач отримує список всіх барів, закусочних, кав’ярень та інше, та їх рейтинги, якщо в даних закладах міститься інформація від Google, що вони розважальні. Також можна користуватися швидким пошуком за назвою, та показом на мапі, як і в двох попередніх категоріях. Приклад нижче на рис. 3.15:

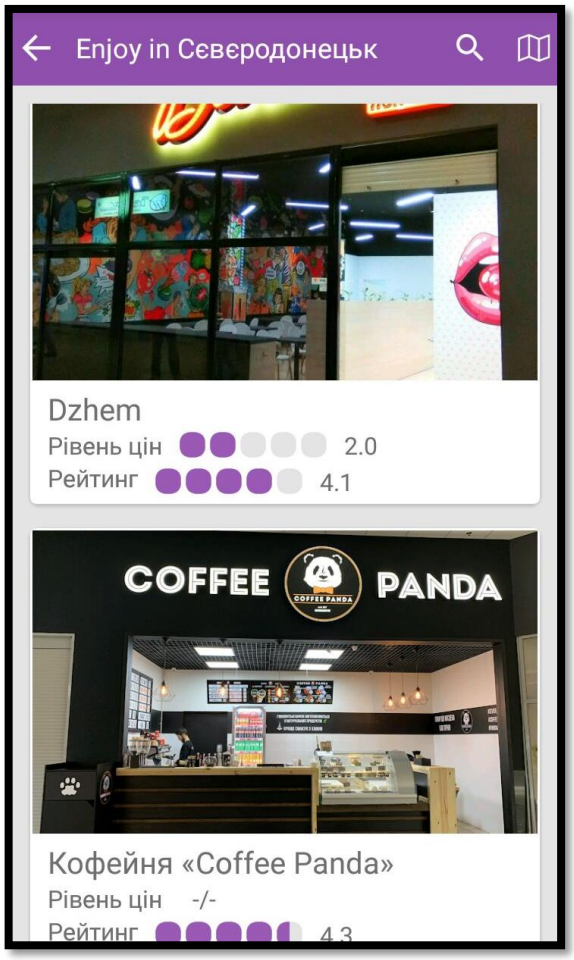

Рисунок 3.15 – Активність зі списком закладів де можна розважитись, та заклади де міститься інформація від Google, що вони розважальні, це можуть бути як приклад: кафе, кав’ярні

Далі розглянемо активність інформації про заклад на прикладі одного вибраного місця відпочинку, з категорії 1 – відпочинок. Можна переглянути фотоальбом конкретного місця, подивитися його адресу, поряд з адресою одразу можна подивитись його розташування на мапі, також можна дізнатись номер телефона, та зателефонувати використовуючи смартфон, як для звичайного виклику так і для виклику наприклад в мессенджері всередині самого додатка. Також можна переглянути мітки, рівень цін, та рейтинг, якщо вказано. Ще є приємна функція додавання закладів у обране, це значно економить час, при запуску додатка у головній активності можна одразу подивитися список своїх обраних закладів. Приклад нижче на рис. 3.16:

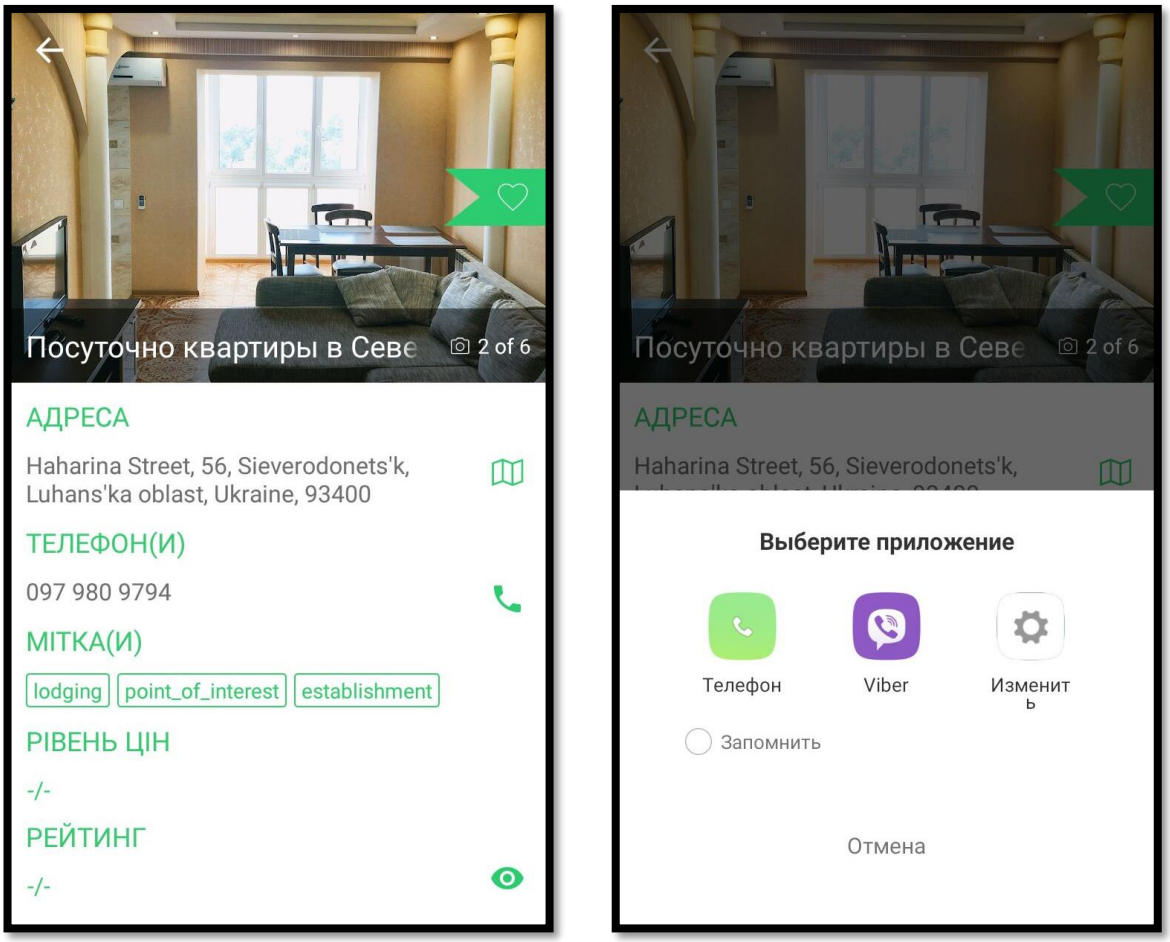

Рисунок 3.16 – Активність з інформацією про місце відпочинку, з категорії 1 – відпочинок

Для відображення на мапі закладів певної категорії, їх потрібно активувати знизу зліва на мапі всередині додатка, там є три категорії sleep - відпочинок, eat - харчування, enjoy - розваги, та додаткова fav – обрані заклади. Для зручності всі вони мають різні кольори. Коли заклад було обрано, прокласти маршрут до нього можна по відповідній кнопці справа внизу, спровокувавши перехід в Google Maps, де маршрут вже буде автоматично побудований. Приклад нижче на рис. 3.17:

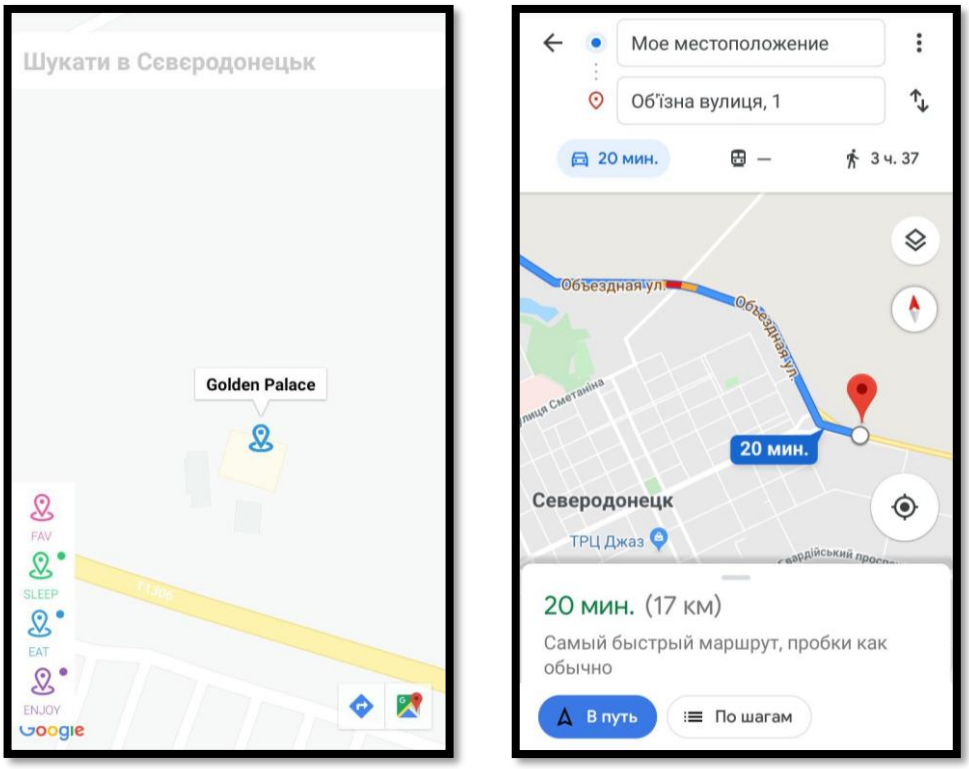

Рисунок 3.17 – Активність з мапою та обраними всіма категоріями на ній всіх закладів. Прокладений маршрут до обраного закладу за допомогою Google Maps

Точність позиціонування GPS можна побачити нижче на рис. 3.18:

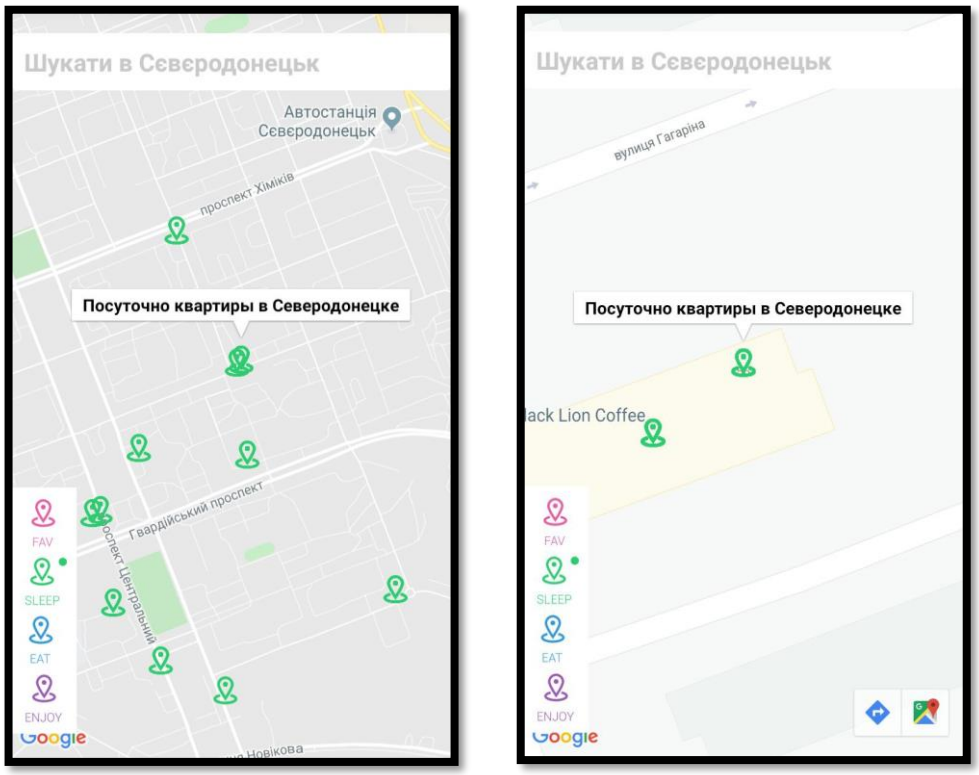

Рисунок 3.18 – Активність з мапою, для прикладу точності позиціонування GPS

Іконка для додатка розроблена в програмі Adobe Photoshop CS6, приведена нижче на рис. 3.19:

Рисунок 3.19 – Іконка для додатка

3.8 Висновки до розділу 3

Була спроектована архітектура додатка-навігатора, до складу якого входять 7 активностей. Був розроблений дизайн кожної активності цього додатка, кінцевою метою якого є зручне, інтуїтивне використання всіх можливостей додатка. Був розроблений і налагоджений програмний код, який забезпечує динамічне функціонування додатка-навігатора та його зв'язок із зовнішнім світом за допомогою системи GPS, з однієї сторони, та з користувачем - власником смартфона, за допомогою елементів управління відповідних активностей, з іншої сторони.

4 ОХОРОНА ПРАЦІ ТА БЕЗПЕКА В НАДЗВИЧАЙНИХ СИТУАЦІЯХ

В даному розділі проведено аналіз потенційних небезпечних та шкідливих виробничих факторів, причин пожеж. Розглянуті заходи, які дозволяють забезпечити гігієну праці і виробничу санітарію. На підставі аналізу розроблені заходи з техніки безпеки та рекомендації з пожежної профілактики.

4.1 Загальні питання з охорони праці

Умови праці на робочому місці, безпека технологічних процесів, машин, механізмів, устаткування та інших засобів виробництва, стан засобів колективного та індивідуального захисту, що використовуються працівником, а також санітарно-побутові умови повинні відповідати вимогам нормативних актів про охорону праці. В законі України «Про охорону праці» [17] визначається, що охорона праці - це система правових, соціально-економічних, організаційно-технічних, санітарно-гігієнічних і лікувально-профілактичних заходів та засобів, спрямованих на збереження життя, здоров'я і працездатності людини у процесі трудової діяльності.

При роботі з обчислювальною технікою змінюються фізичні і хімічні фактори навколишнього середовища: виникає статична електрика, електромагнітне випромінювання, змінюється температура і вологість, рівень вміст кисню і озону в повітрі.

4.2 Аналіз стану умов праці

Робота над дослідженнями і розробкою Android додатків навігації по місту проходитиме в робочому приміщенні. Для даної роботи достатньо однієї людини, для якої надано робоче місце зі стаціонарним комп’ютером.

Геометричні розміри приміщення зазначені в табл. 4.1. ті відповідають нормам згідно з ДСН 3.3.6.042-99 «Санітарні норми мікроклімату виробничих приміщень» [18].

Таблиця 4.1 – Розміри приміщення

Найменування	Значення
Довжина, м	5
Ширина, м	5
Висота, м	3
Площа, $м^2$	25
Об’єм, $м^3$	75

При порівнянні відповідності характеристик робочого місця нормативним основні вимоги до організації робочого місця за ДСанПіН 3.3.2.007-98 «Правила і норми роботи з візуальними дисплейними терміналами електронно-обчислювальних машин» [19] (табл. 4.2) і відповідними фактичними значеннями для робочого місця, констатуємо повну відповідність.

Таблиця 4.2 - Характеристики робочого місця

Найменування параметра	Фактичне значення	Нормативне значення
Висота робочої поверхні, мм	750	680 ÷ 800
Висота простору для ніг, мм	730	не менше 600
Ширина простору для ніг, мм	660	не менше 500
Глибина простору для ніг, мм	700	не менше 650
Висота поверхні сидіння, мм	470	400 ÷ 500
Ширина сидіння, мм	400	не менше 400
Глибина сидіння, мм	400	не менше 400
Висота поверхні спинки, мм	600	не менше 300
Ширина опорної поверхні спинки, мм	500	не менше 380
Радіус кривини спинки в горизонтальній площині, мм	400	400
Відстань від очей до екрану дисплея, мм	800	700 ÷ 800

4.2.1 Навантаження та напруженість процесу праці

Під час виконання робіт використовують ПК та периферійні пристрої (лазерні та струменеві), що призводить до навантаження на окремі системи організму. Такі перекоси у напруженні різних систем організму, що трапляються під час роботи з ПК, зокрема, значна напруженість зорового аналізатора і довготривале малорухоме положення перед екраном, не тільки не зменшують загального напруження, а навпаки, призводять до його посилення і появи стресових реакцій.

Роботу за дипломним проектом визнано, таку, що займає 50% часу робочого дня та за восьмигодинної робочої зміни рекомендовано встановити додаткові регламентовані перерви: (потрібне вибрати):

- для розробників програм тривалістю 15 хв через кожну годину роботи;
- для операторів персональних комп’ютерів тривалістю 15 хв через дві години роботи;
- для операторів комп’ютерного набору тривалістю 10 хв через кожну годину роботи.

4.3 Виробнича санітарія

Аналіз небезпечних та шкідливих виробничих факторів виконується у табличній формі (табл. 4.3). Роботу, пов'язану з ЕОП з ВДТ, у тому числі на тих, які мають робочі місця, обладнані ЕОМ з ВДТ і ПП, виконують із забезпеченням виконання НПАОП 0.00.-7.15-18 «Вимоги щодо безпеки та захисту здоров`я працівників під час роботи з екранними пристроями» [25].

Таблиця 4.3 – Аналіз небезпечних і шкідливих виробничих факторів

Небезпечні і шкідливі виробничі фактори	Джерела факторів (види робіт)	Кількісна оцінка	Нормативні документи
1	2	3	4
фізичні			
- підвищена температура поверхонь обладнання	експлуатація ЕОМ	2	[18]
- підвищена або знижена вологість повітря	-//-	2	[18]
- підвищений рівень електромагнітного випромінення	-//-	2	[23]
- підвищений рівень напруги електричної мережі, замикання якої може відбутися через тіло людини	-//-	4	[23] [24]
- підвищений рівень статичної електрики	-//-	2	[23]
- недостатність природного світла	порушення умов праці (вимог до приміщень)	2	[21]
психофізіологічні:			
- нервово-психічна перевантаження (розумове, перенапруження аналізаторів-зорових)	- пошук інформації для постановки теми; - пошук та аналіз аналогів і літератури; - пошук наявних технологій, моделювання та аналіз алгоритмів; - виконання роботи за темою диплома, тестування; - оформлення роботи	4	[25] [19]
- фізичні (статичне – сидіння)	порушення умов праці (організації місця праці-сидіння користувача,) та організації робочого часу - безперервна робота)	2	[25] [19]

4.3.1 Пожежна безпека

Для гасіння пожеж в робочому присміщені приміщенні пропонується використовувати порошкові або вуглекислотні вогнегасники, так як вони є універсальними.

Згідно НАПБ А. 01.001-2014 «Правила пожежної безпеки в Україні» [20] таке приміщення, площею 25 м^2, відноситься до категорії "В" (пожежонебезпечної). Відповідно до норм первинних засобів пожежогасінні пропонується використовувати:

- повсть 1 1 м^2, кошму 2×1,5 м2 в кількості 1 шт.

4.3.2 Електробезпека

На робочому місці виконуються наступні вимоги електробезпеки: ПК, периферійні пристрої та устаткування для обслуговування, електропроводи і кабелі за виконанням та ступенем захисту відповідають класу зони за ПУЕ (правила улаштування електроустановок), мають апаратуру захисту від струму короткого замикання та інших аварійних режимів. Лінія електромережі для живлення ПК, периферійних пристроїв і устаткування для обслуговування, виконана як окрема групова три-провідна мережа, шляхом прокладання фазового, нульового робочого та нульового захисного провідників. Нульовий захисний провідник використовується для заземлення (занулення) електроприймачів. Штепсельні з'єднання та електророзетки крім контактів фазового та нульового робочого провідників мають спеціальні контакти для підключення нульового захисного провідника. Електромережа штепсельних розеток для живлення персональних ПК, укладено по підлозі поруч зі стінами відповідно до затвердженого плану розміщення обладнання та технічних характеристик обладнання. Металеві труби та гнучкі металеві рукави заземлені. Захисне заземлення включає в себе заземлюючих пристроїв і провідник, який з'єднує заземлюючий пристрій з обладнанням, яке заземлюється - заземлюючий провідник.

4.4. Гігієнічні вимоги до параметрів виробничого середовища

4.4.1 Параметри мікроклімату

Мікроклімат робочих приміщень – це клімат внутрішнього середовища цих приміщень, що визначається діючої на організм людини з'єднанням температури, вологості, швидкості переміщення повітря. Отже оптимальні значення для температури, відносної вологості й рухливості повітря для зазначеного робочого місця відповідають ДСН 3.3.6.042-99 «Санітарні норми мікроклімату виробничих приміщень» [18] і наведені в табл. 4.4:

Таблиця 4.4 – Норми мікроклімату робочої зони об’єкту

Період року	**Категорія робіт**	**Температура C^0**	**Відносна вологість %**	**Швидкість руху повітря, м/с**
Холодна	легка-1 а	22 - 24	40 – 60	0,1
Тепла	легка-1 а	23 - 25	40 – 60	0,1

4.4.2 Освітлення

У приміщенні, де розташовані ЕОМ передбачається природне бічне освітлення, рівень якого відповідає ДБН В.2.5-28:2018 «Природне і штучне освітлення» [21]. Джерелом природного освітлення є сонячне світло. Регулярно повинен проводитися контроль освітленості, який підтверджує, що рівень освітленості задовольняє ДБН В.2.5-28:2018 [21] і для даного приміщення в світлий час доби достатньо природного освітлення.

Розрахунок освітлення.

Для виробничих та адміністративних приміщень світловий коефіцієнт приймається не менше -1/8, в побутових – 1/10:

$$S_b = \left(\frac{1}{5} \div \frac{1}{10}\right) \cdot S_n, \tag{4.1}$$

де S_b – площа віконних прорізів, $м^2$;

S_n – площа підлоги, $м^2$.

$S_n = a \cdot b = 5 \cdot 5 = 25$ $м^2$,

$S = 1/8 \cdot 25 = 3{,}125$ $м^2$.

Приймаємо 2 вікна площею $S=1{,}6$ $м^2$ кожне.

Розрахунок штучного освітлення виробляється по коефіцієнтах використання світлового потоку, яким визначається потік, необхідний для створення заданої освітленості при загальному рівномірному освітленні. Розрахунок кількості світильників n виробляється по формулі (4.2):

$$n = \frac{E \cdot S \cdot Z \cdot K}{F \cdot U \cdot M}, \tag{4.2}$$

де E – нормована освітленість робочої поверхні, визначається нормами – 300 лк;

S – освітлювана площа, $м^2$; $S = 25$ $м^2$;

Z – поправочний коефіцієнт світильника (Z = 1,15 для ламп розжарювання та ДРЛ; Z = 1,1 для люмінесцентних ламп) приймаємо рівним 1,1;

K – коефіцієнт запасу, що враховує зниження освітленості в процесі експлуатації – 1,5;

U – коефіцієнт використання, залежний від типу світильника, показника індексу приміщення і т.п. – 0,575

M – число люмінесцентних ламп в світильнику – 2;

F – світловий потік лампи – 5400лм (для ЛБ-80).

Підставивши числові значення у формулу (4.2), отримуємо:

$$n = \frac{300 \cdot 25 \cdot 1{,}1 \cdot 1{,}5}{5400 \cdot 0{,}575 \cdot 2} \approx 2{,}0$$

Приймаємо освітлювальну установку, яка складається з 2-х світильників, які складаються з двох люмінесцентних ламп загальною потужністю 160 Вт, напругою – 220 В.

4.4.3 Вентилювання

У приміщенні, де знаходяться ЕОМ, повітрообмін реалізується за допомогою природної організованої вентиляції (вентиляційні шахти), тобто при
V приміщення > 40 м^3 на одного працюючого допускається природна вентиляція. Цей метод забезпечує приток потрібної кількості свіжого повітря.

4.5 Заходи з організації виробничого середовища та попередження виникнення надзвичайних ситуацій

Відповідно до санітарно-гігієнічних нормативів та правил експлуатації обладнання наводимо приклади деяких заходів безпеки.

Розрахунок захисного заземлення (забезпечення електробезпеки будівлі).

Згідно з класифікацією приміщень за ступенем небезпеки ураження електричним струмом НПАОП 40.1-1.01-97 «Правила безпечної експлуатації електроустановок» [22], НПАОП 40.1-1.21-98 «Правила безпечної експлуатації електроустановок споживачів» [26] приміщення в якому проводяться всі роботи відноситься до першого класу (без підвищеної небезпеки). Під час роботи використовуються електроустановки з напругою живлення 36 В, 220 В, та 360 В. Опір контура заземлення повинен мати не більше 4 Ом.

Розрахунок проводять за допомогою методу коефіцієнта використання (екранування) електродів. Коефіцієнт використання групового заземлювача η – це відношення діючої провідності цього заземлювача до найбільш можливої його провідності за нескінченно великих відстаней між його електродами. Коефіцієнт використання вертикальних заземлювачів ηв в залежності від розміщення заземлювачів та їх кількості знаходиться в межах 0,4…0,99. Взаємну екрануючу дію горизонтального заземлювача (з'єднувальної смуги) враховують за допомогою коефіцієнта використання горизонтального заземлювача η_c.

Послідовність розрахунку.

а) Визначається необхідний опір штучних заземлювачів $R_{шт.з.}$:

$$R_{шт.з.} = \frac{R_д \cdot R_{пр.з.}}{R_{пр.з.} - R_д}, \qquad (4.3)$$

де $R_{пр.з.}$ – опір природних заземлювачів; $R_д$ – допустимий опір заземлення. Якщо природні заземлювачі відсутні, то $R_{шт.з.}=R_д$.

Підставивши числові значення у формулу (рис.4.3), отримуємо:

$$R_{шт.з.} = \frac{4 \cdot 40}{40 - 4} \approx 4 \text{ Ом}$$

б) опір заземлення в значній мірі залежить від питомого опору ґрунту ρ, Ом·м. Приблизне значення питомого опору глини приймаємо ρ=40 Ом•м (табличне значення);

в) розрахунковий питомий опір ґрунту, $\rho_{розр}$, Ом·м, визначається відповідно для вертикальних заземлювачів $\rho_{розр.в}$, і горизонтальних $\rho_{розр.г}$, Ом·м за формулою:

$$\rho_{розр..} = \psi \cdot \rho, \qquad (4.4)$$

де ψ – коефіцієнт сезонності для вертикальних заземлювачів I кліматичної зони з нормальною вологістю землі, приймається для вертикальних заземлювачів $\rho_{розр.в}=1,7$ і горизонтальних $\rho_{розр.г}=5,5$ Ом·м.

$\rho_{розр.в} = 1,7 \cdot 40 = 68$ Ом•м

$\rho_{розр.г} = 5,5 \cdot 40 = 220$ Ом•м

г) розраховується опір розтікання струму вертикального заземлювача $R_в$, Ом, за (4.5):

$$R_в = \frac{\rho_{розр.в}}{2 \cdot \pi \cdot l_в} \cdot \left(\ln \frac{2 \cdot l_в}{d_{ст}} + \frac{1}{2} \cdot \ln \frac{4 \cdot t + l_в}{4 \cdot t - l_в} \right), \quad (4.5)$$

де $l_в$ – довжина вертикального заземлювача (для труб - 2−3 м; $l_в$=3 м);

$d_{ст}$ – діаметр стержня (для труб - 0,03−0,05 м; $d_{ст}$=0,05 м);

t – відстань від поверхні землі до середини заземлювача, яка визначається за ф. (4.6):

$$t = h_в + \frac{l_в}{2}, \quad (4.6)$$

де $h_в$ – глибина закладання вертикальних заземлювачів (0,8 м); тоді $t = 0,8 + \frac{3}{2} = 2,3$ м

$$R_в = \frac{68}{2 \cdot \pi \cdot 3} \cdot \left(\ln \frac{2 \cdot 3}{0,05} + \frac{1}{2} \cdot \ln \frac{4 \cdot 2,3 + 3}{4 \cdot 2,3 - 3} \right) = 18,5 \text{ Ом}$$

ґ) визначається теоретична кількість вертикальних заземлювачів n штук, без урахування коефіцієнта використання $\eta_в$:

$$n = \frac{2 \cdot R_в}{R_д} = \frac{2 \cdot 18,5}{4} = 9,25 \quad (4.7)$$

І визначається коефіцієнт використання вертикальних електродів групового заземлювача без врахування впливу з'єднувальної стрічки $\eta_в$ =0,57 (табличне значення).

д) визначається необхідна кількість вертикальних заземлювачів з урахуванням коефіцієнта використання $n_в$, шт:

$$n_в = \frac{2 \cdot R_в}{R_д \cdot \eta_в} = \frac{2 \cdot 18,5}{4 \cdot 0,57} = 16,2 \approx 16 \quad (4.8)$$

е) визначається довжина з’єднувальної стрічки горизонтального заземлювача $l_с$, м:

$$l_с = 1,05 \cdot L_в \cdot (n_в - 1), \quad (4.9)$$

де $L_в$ – відстань між вертикальними заземлювачами, (прийняти за $L_в$ = 3м);

$n_в$ – необхідна кількість вертикальних заземлювачів.

$$l_с = 1,05 \cdot 3 \cdot (16 - 1) \approx 48 \text{ м}$$

є) визначається опір розтіканню струму горизонтального заземлювача (з’єднувальної стрічки) $R_г$, Ом:

$$R_г = \frac{\rho_{розр.г}}{2 \cdot \pi \cdot l_с} \cdot \ln \frac{2 \cdot l_с^{\,2}}{d_{см} \cdot h_г}, \quad (4.10)$$

де $d_{см}$ – еквівалентний діаметр смуги шириною b, $d_{см}$ =0,95b, b = 0,15 м;

$h_г$ – глибина закладання горизонтальних заземлювачів (0,5 м);

$l_с$ - довжина з’єднувальної стрічки горизонтального заземлювача $l_с$, м

$$R_г = \frac{220}{2 \cdot \pi \cdot 48} \cdot \ln \frac{2 \cdot 48^2}{0,95 \cdot 0,15 \cdot 0,5} = 8,1 \text{ Ом}$$

ж) визначається коефіцієнт використання горизонтального заземлювача $\eta_{с.}$ відповідно до необхідної кількості вертикальних заземлювачів $n_в$;

Коефіцієнт використання з'єднувальної смуги $\eta_с$=0,3 (табличне значення).

з) розраховується результуючий опір заземлювального електроду з урахуванням з’єднувальної смуги:

$$R_{заг} = \frac{R_в \cdot R_г}{R_в \cdot \eta_с + R_г \cdot n_в \cdot \eta_в} \le R_д. \quad (4.11)$$

Висновок: дане захисне заземлення буде забезпечувати електробезпеку будівлі, так як виконується умова: $R_{заг}$ <4 Ом, а саме:

$$R_{заг} = \frac{18,5 \cdot 8,1}{18,5 \cdot 0,3 + 8,1 \cdot 16 \cdot 0,57} = 1,9 \leq R_{д}$$

4.6 Охорона навколишнього природного середовища

Діяльність за темою магістерської роботи, а саме: робота за компьютером, в процесі її виконання є фактори що впливать на навколишнє природнє середовище.

Основним екологічним аспектом в процесі діяльності за даними спеціальностями є процеси впливу на атмосферне повітря та процеси поводження з відходами, які утворюються, збираються, розміщуються, передаються на відалення (знешкодження), утилізацію, тощо в ІТ галузі.

Вплив на атмосферне повітря при нормальних умовах праці не оказує, бо не має в приміщенні сканерів, принтерів та інших джерел викиду забруднюючих речовин в повітря робочої зони.

В процесі діяльності компьютера виникають процеси поводження з відходами ІТ галузі. Нижче надано перелік відходів, що утворюються в процесі роботи:

Відпрацьовані люмінесцентні лампи - I клас небезпеки

Батарейки та акумулятори (малі) -III клас небезпеки

Змінні носії інформації - IV клас небезпеки

Відпрацьований ізолюючий матеріал, дроти та кабелі - IV клас небезпеки

Макулатура - IV клас небезпеки

Побутові відходи - IV клас небезпеки

4.7 Висновки до розділу 4

В результаті проведеної роботи було зроблено аналіз умов праці, шкідливих та небезпечних чинників, з якими стикається робітник. Було визначено параметри і певні характеристики приміщення для роботи, описано, які заходи потрібно зробити для того, щоб дане приміщення відповідало необхідним нормам і було комфортним і безпечним для робітника. Приведені рекомендації щодо організації робочого місця, а також важливу інформацію щодо пожежної та електробезпеки. Були наведені розміри приміщення та наведено значення температури, вологості й рухливості повітря, необхідна кількість і потужність ламп та інші параметри, значення яких впливає на умови праці робітника.

А також визначені основні екологічні аспекти впливу на навколишнє природнє середовище та зазначені заходи щодо поводження з ними.

ВИСНОВКИ

Розробка Android-додатків навігації по місту є актуальною задачею. Додатки повинні мати простий та зрозумілий інтерфейс, якій не відштовхне потенційних користувачів, та виконувати всі основні функції навігації, такі як: прокладання маршруту до обраної точки на мапі, показувати оточення користувача, інформацію про оточуючі об’єкти, показувати місцезнаходження, та інше.

В ході роботи були розглянуті принципи роботи навігаторів на базі операційної системи Android, їх апаратна та програмна частина. Проведений огляд популярних GPS навігаторів на мобільних пристроях з операційною системою Android, проаналізовані їх переваги та недоліки, з метою запобігання подібного при розробці. Розглянуто створення додатка з функціями мапи та налаштування Android Studio для коректної роботи. Проведено огляд та порівняння технологій GPS та A-GPS, розглянуті та проаналізовані системи та методи локального позиціонування об’єктів. Також досліджено використання фільтрів та датчиків для зниження похибки GPS на ОС Android. Розглянуто та досліджено систему позиціонування нового покоління в Uber. Також розглянуто визначення місцезнаходження GPS за допомогою пристрою Reach. Проведені дослідження завадостійкості, досліджені методики оцінки точності визначення координат та швидкості, дослідження завадостійкості навігаційного приймача GPS і ГЛОНАСС.

Спроектована архітектура додатка-навігатора, до складу якого входять 7 активностей. Був розроблений дизайн кожної активності цього додатка, кінцевою метою якого є зручне, інтуїтивне використання всіх можливостей додатка. Був розроблений і налагоджений програмний код, який забезпечує динамічне функціонування додатка-навігатора та його зв'язок із зовнішнім світом за допомогою системи GPS, з однієї сторони, та з користувачем - власником смартфона, за допомогою елементів управління відповідних активностей, з іншої сторони.

Було проведено аналіз умов праці, шкідливих та небезпечних чинників, з якими стикається робітник. Було визначено параметри і певні характеристики приміщення для роботи над запропонованим проектом написаному в дипломній роботі, описано, які заходи потрібно зробити для того, щоб дане приміщення відповідало необхідним нормам і було комфортним і безпечним для робітника. Приведені рекомендації щодо організації робочого місця, а також важливу інформацію щодо пожежної та електробезпеки. Були наведені розміри приміщення та наведено значення температури, вологості й рухливості повітря, необхідна кількість і потужність ламп та інші параметри, значення яких впливає на умови праці робітника. А також визначені основні екологічні аспекти впливу на навколишнє природнє середовище та зазначені заходи щодо поводження з ними.

ПЕРЕЛІК ПОСИЛАНЬ

1. Навігатор для Андроїда: огляд найбільш популярних додатків [Електронний ресурс] // androidsfaq.com. – 2018. – Режим доступу до ресурсу: www. URL: https://androidsfaq.com/navigator-dlia-androida/#i.
2. Вибираємо кращий навігатор для Андроїд. Безкоштовні gps-навігатори [Електронний ресурс] // softdroid.net. – 2019. – Режим доступу до ресурсу: www. URL: https://softdroid.net/navigator-android-karty-marshruty.
3. Прості карти для Android. Частина 1. [Електронний ресурс] // software.intel.com. – 2016. – Режим доступу до ресурсу: www. URL: https://software.intel.com/ru-ru/articles/easy-android-maps-part-1.
4. GPS і A-GPS: у чому різниця і що краще? [Електронний ресурс] // ichip.ru. – 2019. – Режим доступу до ресурсу: www. URL: https://ichip.ru/tekhnologii/gps-i-a-gps-v-chem-raznitsa-i-chto-luchshe-597622.
5. У чому відмінність GPS від A-GPS? [Електронний ресурс] // mcgrp.ru. – 2017. – Режим доступу до ресурсу: www. URL: https://mcgrp.ru/article/1236-v-chem-otlichie-gps-ot-a-gps.
6. Мініахметов, Р.М. Огляд алгоритмів локального позиціонування для мобільних пристроїв / Р.М. Мініахметов, А.А. Рогов, М.Л. Цимблер // Вісник Південно-Уральського Державного університету. - 2013. - №2-2. - С. 83-95.
7. Онуфрієва, Т.А. Огляд автоматизованих систем позиціонування об'єктів / Т.А. Онуфрієва, Л.А. Щавелєв // Інноваційна наука. - 2017. - №3-1. - С. 71-73.
8. Знижуємо похибку GPS на Android за допомогою фільтра Калмана і акселерометра [Електронний ресурс] // blog.maddevs.io. – 2018. – Режим доступу до ресурсу: www. URL: https://blog.maddevs.io/ru-reduce-gps-data-error-on-android-with-kalman-filter-and-accelerometer-b81f1026e06c.
9. Переосмислення GPS: Розробка системи позиціонування нового покоління в Uber [Електронний ресурс] // habr.com. – 2018. – Режим доступу до ресурсу: www. URL: https://habr.com/ru/post/353978/.
10. Reach: GPS з точністю до сантиметра [Електронний ресурс] // habr.com. – 2015. – Режим доступу до ресурсу: www. URL: https://habr.com/ru/company/intel/blog/258779/.
11. Корнілов І. М. Тестування навігаційної апаратури споживача GPS / ГЛОНАСС (Навчально-методичний посібник) – Єкатеринбург: Видавництво Уральського університету, 2017. – 48 с.

12. Android Studio IDE від Google. Можливості Android Studio від Google [Електронний ресурс] // wnfx.ru – Режим доступу до ресурсу: www. URL: http://wnfx.ru/android-studio-ide-ot-google/.
13. Bill Fillips, K. Styuart, Kristin Marsikano Android Programming for professionals – «The Big Nerd Ranch Guide» – 2017 – 687стр.
14. Г. Шілдт. Java 8. Керівництво для початківців (6-е видання), 2015. - 712 с.
15. Одиночкіна С. В. Основи технологій XML (Навчальний посібник) / Санкт-Петербург: Національний дослідницький університет інформаційних технологій, механіки та оптики, 2013. – 57 с.
16. Працюємо з GPS в Android на Java [Електронний ресурс] // streletzcoder.ru. – 2016. – Режим доступу до ресурсу: www. URL: http://streletzcoder.ru/rabotaem-s-gps-v-android-na-java/.
17. Закон України «Про охорону праці». Вводиться в дію Постановою ВР № 2695-XII від 14.10.92, ВВР, 1992, № 49, ст.669. [Електронний ресурс] // zakon.rada.gov.ua – Режим доступу до ресурсу: www. URL: https://zakon.rada.gov.ua/laws/show/2694-12
18. ДСН 3.3.6.042-99 «Санітарні норми мікроклімату виробничих приміщень». Вводиться в дію Постановою ВР № 42 від 01.12.1999. [Електронний ресурс] // zakon.rada.gov.ua – Режим доступу до ресурсу: www. URL: https://zakon.rada.gov.ua/rada/show/va042282-99
19. ДСанПіН 3.3.2.007-98 «Державні санітарні правила і норми роботи з візуальними дисплейними терміналами електронно-обчислювальних машин». Вводиться в дію Постановою ВР № 7 від 10.12.1998. [Електронний ресурс] // zakon.rada.gov.ua – Режим доступу до ресурсу: www. URL: https://zakon.rada.gov.ua/rada/show/v0007282-98
20. НАПБ А. 01.001-2014 «Правила пожежної безпеки в Україні». Затверджено Наказом Міністерства внутрішніх справ України № 1417 від 30.12.2014. [Електронний ресурс] // zakon4.rada.gov.ua – Режим доступу до ресурсу: www. URL: https://zakon4.rada.gov.ua/laws/show/z0252-15
21. ДБН В.2.5-28:2018 «Природне і штучне освітлення». [Електронний ресурс] // minregion.gov.ua – Режим доступу до ресурсу: www. URL: http://www.minregion.gov.ua/wp-content/uploads/2018/12/V2528-1.pdf
22. НПАОП 40.1-1.01-97 «Правила безпечної експлуатації електроустановок». Затверджено наказом Державного комітету України по нагляду за охороною праці № 257 від 6 жовтня 1997 р. [Електронний ресурс] // zakon.rada.gov.ua – Режим доступу до ресурсу: www. URL: https://zakon.rada.gov.ua/laws/show/z0011-98
23. ДСТУ 7237:2011 «Система стандартів безпеки праці. Електробезпека. Загальні вимоги та номенклатура видів захисту». Затверджено Держспоживстандартом України № 37 від

02.02.2011. [Електронний ресурс] // online.budstandart.com – Режим доступу до ресурсу: www. URL: http://online.budstandart.com/ru/catalog/doc-page.html?id_doc=30045

24. ГОСТ 13109-97 «Електрична енергія. Сумістність технічних засобів електромагнітних». Дата введення 01.01.1999. [Електронний ресурс] // dnaop.com – Режим доступу до ресурсу: www. URL: https://dnaop.com/html/42313/doc-ГОСТ_13109-97
25. НПАОП 0.00-7.15-18 «Вимоги щодо безпеки та захисту здоров`я працівників під час роботи з екранними пристроями». Зареєстровано в Міністерстві юстиції України 25 квітня 2018 р. за № 508/31960. [Електронний ресурс] // zakon.rada.gov.ua – Режим доступу до ресурсу: www. URL: https://zakon.rada.gov.ua/laws/show/z0508-18
26. НПАОП 40.1-1.21-98 «Правила безпечної експлуатації електроустановок споживачів». Затверджено Наказом Держнаглядохоронприці України № 4 від 09.01.98 [Електронний ресурс] // zakon.rada.gov.ua – Режим доступу до ресурсу: www. URL: https://zakon.rada.gov.ua/laws/show/z0093-98

ДОДАТОК А
Лістинг програми

Лістинг коду MainActivity.java:

```
1 package com.aleksey.navigator;
2
3 import android.content.Context;
4 import android.os.Bundle;
5 import android.os.Handler;
6 import android.support.v4.app.Fragment;
7 import android.support.v4.app.FragmentActivity;
8 import android.support.v4.app.FragmentManager;
9 import android.support.v4.app.FragmentTransaction;
10 import android.view.View;
11 import android.widget.Toast;
12
13 import com.aleksey.navigator.about.AboutDataLoader;
14 import com.aleksey.navigator.about.Photo;
15 import com.aleksey.navigator.db.DatabaseManager;
16 import com.aleksey.navigator.fragment.DialogFragmentLoading;
17 import com.aleksey.navigator.fragment.DialogFragmentPhoto;
18 import com.aleksey.navigator.fragment.FragmentAbout;
19 import com.aleksey.navigator.fragment.FragmentForecast;
20 import com.aleksey.navigator.fragment.FragmentHome;
21 import com.aleksey.navigator.fragment.FragmentListPlaces;
22 import com.aleksey.navigator.fragment.FragmentNearby;
23 import com.aleksey.navigator.fragment.FragmentPlaceDetail;
24 import com.aleksey.navigator.fragment.FragmentReviews;
25 import com.aleksey.navigator.googleplaces.Constants;
26 import com.aleksey.navigator.googleplaces.NearbyPlacesManager;
27 import com.aleksey.navigator.googleplaces.models.Place;
28 import com.aleksey.navigator.googleplaces.models.PlaceDetails;
29 import com.aleksey.navigator.network.NetworkFetcher;
30 import com.aleksey.navigator.settings.AppSettings;
31 import com.aleksey.navigator.util.ImageOptionsBuilder;
32 import com.aleksey.navigator.util.ThemeUtil;
33 import com.google.android.gms.ads.AdRequest;
34 import com.google.android.gms.ads.AdView;
35 import com.nostra13.universalimageloader.core.ImageLoader;
36
37
38 public class MainActivity extends FragmentActivity {
```

```
39
40      public static float density;
41
42      public static Context context;
43
44      protected AdView mAdmobView;
45
46      public static String getApiKey() {
47          return context.getString(R.string.google_api_key);
48      }
49
50      @Override
51      protected void onCreate(Bundle savedInstanceState) {
52          super.onCreate(savedInstanceState);
53
54          context = this;
55
56          //Fabric.with(this, new Crashlytics());
57
58          ThemeUtil.setTranslucentTheme(this);
59
60          density = getResources().getDisplayMetrics().density;
61
62          setContentView(R.layout.activity_main);
63
64          DatabaseManager.initWithContext(this);
65
66          ImageLoader.getInstance().init(
67                  ImageOptionsBuilder.createImageLoaderConfiguration(this));
68
69                                                  Constants.DEFAULT_PHOTO_WIDTH     =
getResources().getDisplayMetrics().widthPixels;
70
71          showHomeFragment();
72
73          new Handler().postDelayed(new Runnable() {
74              @Override
75              public void run() {
76                  NearbyPlacesManager.getInstance();
77              }
78          }, 1000);
79
80          new AboutDataLoader().loadData(this);
```

```
81
82          initAdmob();
83      }
84
85      public void showNearbyFragment(Constants.PLACE_TYPES type, Place placeDetail,
boolean showCurrentLocation) {
86                          showScreen(FragmentNearby.newInstance(type,  placeDetail,
showCurrentLocation), FragmentNearby.TAG, true, false);
87      }
88
89      public boolean isNearbyFragmentShowed() {
90          return getSupportFragmentManager().findFragmentByTag(FragmentNearby.TAG)
!= null;
91      }
92
93      public void showListFragment(Constants.PLACE_TYPES type) {
94          showScreen(FragmentListPlaces.newInstance(type), FragmentListPlaces.TAG,
true, false);
95      }
96
97      private void showHomeFragment() {
98          showScreen(FragmentHome.newInstance(), FragmentHome.TAG, false, false);
99      }
100
101      public void showForecastFragment() {
102
103           showScreen(FragmentForecast.newInstance(), FragmentForecast.TAG, true,
false);
104      }
105
106      public void showAboutFragment() {
107
108          showScreen(FragmentAbout.newInstance(), FragmentAbout.TAG, true, false);
109      }
110
111      public void showLoadingDialog() {
112            DialogFragmentLoading.newInstance().show(getSupportFragmentManager(),
DialogFragmentLoading.TAG);
113      }
114
115      public void showPhotoDialog(Photo photo) {
116          DialogFragmentPhoto.newInstance(photo).show(getSupportFragmentManager(),
DialogFragmentPhoto.TAG);
```

```
117     }
118
119
120     public void showPlaceDetailFragment(Place place, Constants.PLACE_TYPES type)
{
121                                     FragmentPlaceDetail    fragmentPlaceDetail    =
FragmentPlaceDetail.newInstance(place, type);
122
123         showScreen(fragmentPlaceDetail, FragmentPlaceDetail.TAG, true, false);
124     }
125
126                 public    void    showReviewsScreen(PlaceDetails    placeDetails,
Constants.PLACE_TYPES type) {
127                                         FragmentReviews    fragmentReviews    =
FragmentReviews.newInstance(placeDetails, type);
128
129         showScreen(fragmentReviews, FragmentReviews.TAG, true, false);
130     }
131
132
133     public void hideLoadingDialog() {
134
135                                                 Fragment     loadingFragment     =
getSupportFragmentManager().findFragmentByTag(DialogFragmentLoading.TAG);
136
137         if (loadingFragment != null) {
138             ((DialogFragmentLoading) loadingFragment).dismiss();
139         }
140     }
141
142     protected void initAdmob() {
143         mAdmobView = (AdView) findViewById(R.id.home_admob);
144         if (AppSettings.ENABLE_ADMOB) {
145             mAdmobView.setVisibility(View.VISIBLE);
146             AdRequest.Builder builder =  new AdRequest.Builder();
147             AdRequest adRequest = builder.build();
148
149             // Start loading the ad in the background.
150             mAdmobView.loadAd(adRequest);
151
152         } else {
153             mAdmobView.setVisibility(View.GONE);
154         }
```

```
155     }
156
157     private void showScreen(Fragment content,
158                                String contentTag, boolean addToBackStack, boolean
clearBackStack) {
159
160                  if  (!NetworkFetcher.isNetworkConnected(MainActivity.this)  &&
!contentTag.equalsIgnoreCase(FragmentHome.TAG)) {
161                Toast.makeText(this, getString(R.string.no_internet_connection),
Toast.LENGTH_LONG).show();
162
163             return;
164         }
165
166         FragmentManager fm = getSupportFragmentManager();
167
168         FragmentTransaction ft = fm.beginTransaction();
169         ft.setCustomAnimations(R.anim.left_slide_in, R.anim.left_slide_out,
170                 R.anim.right_slide_in, R.anim.right_slide_out);
171
172
173         ft.replace(R.id.activity_main_content, content, contentTag);
174
175
176         if (clearBackStack) {
177             fm.popBackStackImmediate(null,
178                     FragmentManager.POP_BACK_STACK_INCLUSIVE);
179         }
180
181         if (addToBackStack) {
182            ft.addToBackStack(String.valueOf(System.identityHashCode(content)));
183         }
184
185         ft.commitAllowingStateLoss();
186         fm.executePendingTransactions();
187     }
188 }
```

Лістинг коду GooglePlaces.java:

```
1 package com.aleksey.navigator.googleplaces;
2 import android.util.Log;
```

```
3 import com.aleksey.navigator.MainActivity;
4 import com.aleksey.navigator.callbacks.CallbackNearbyPlaces;
5 import com.aleksey.navigator.db.DatabaseManager;
6 import com.aleksey.navigator.db.cmn.DbNetwork;
7 import com.aleksey.navigator.googleplaces.models.DetailsResult;
8 import com.aleksey.navigator.googleplaces.models.Place;
9 import com.aleksey.navigator.googleplaces.models.PlaceDetails;
10 import com.aleksey.navigator.googleplaces.models.PlaceReview;
11 import com.aleksey.navigator.googleplaces.models.PlacesResult;
12 import com.aleksey.navigator.googleplaces.query.DetailsQuery;
13 import com.aleksey.navigator.googleplaces.query.GooglePlusQuery;
14 import com.aleksey.navigator.googleplaces.query.NearbySearchQuery;
15 import com.aleksey.navigator.googleplaces.query.TextSearchQuery;
16 import com.aleksey.navigator.network.NetworkFetcher;
17 import com.aleksey.navigator.settings.AppSettings;
18 import com.google.android.gms.maps.model.LatLng;
19 import org.json.JSONException;
20 import org.json.JSONObject;
21 import java.io.IOException;
22 import java.util.HashSet;
23 import java.util.List;
24 import java.util.Set;
25 public class GooglePlaces {
26     private final LatLng mLocation;
27     public GooglePlaces(LatLng location) {
28         this.mLocation = location;
29     }
30     Set<Place> resultPlaces = new HashSet<Place>();
31     public void getPlacesNearby(NearbySearchQuery query, CallbackNearbyPlaces
callbackNearbyPlaces, Constants.PLACE_TYPES type)
32             throws JSONException, IOException, InterruptedException {
33         Log.i("Loaded_Markers", "Executed query = " + query.toString());
34         PlacesResult result = new
PlacesResult(NetworkFetcher.executeRequest(query.toString(), false));
35         callbackNearbyPlaces.onPlacesLoaded(result.getPlaces(), type);
36         Log.i("Loaded_Markers", "NextPageToken " + result.getNextPageToken());
37     }
38     public PlacesResult getPlacesSearch(String searchText, List<String> types)
39             throws JSONException, IOException {
40         TextSearchQuery query = new TextSearchQuery(searchText);
41         query.setLocation(mLocation.latitude, mLocation.longitude);
42         query.setRadius(AppSettings.GOOGLE_PLACES_SEARCH_RADIUS);
43         query.addTypes(types);
```

```
44          PlacesResult result = new
PlacesResult(NetworkFetcher.executeRequest(query.toString(), false));
45          return result;
46      }
47      public DetailsResult getPlaceDetails(String reference)
48              throws JSONException, IOException {
49          DbNetwork dbNetwork =
DatabaseManager.getInstance().findNetworkQuery(reference);
50          DetailsQuery query = new DetailsQuery(reference);
51          query.setKey(MainActivity.getApiKey());
52          JSONObject response = NetworkFetcher.executeRequest(query.toString(),
true);
53          DetailsResult result = new DetailsResult(response);
54          return result;
55      }
56      public void getPlaceDetailsReviews(PlaceDetails placeDetails)
57              throws JSONException, IOException {
58          try {
59              List<PlaceReview> reviews = placeDetails.getReviews();
60              for (int i = 0; i < reviews.size(); i++) {
61                  GooglePlusQuery googlePlusQuery = new GooglePlusQuery();
62
googlePlusQuery.setGooglePlacePersonId(reviews.get(i).getAuthorPhotoUrl());
63                  JSONObject googlePlusObject =
NetworkFetcher.executeRequest(googlePlusQuery.toString(), true);
64                  String actualPhotoUrl =
googlePlusObject.getJSONObject("image").getString("url");
65                  actualPhotoUrl = actualPhotoUrl.replace("50", "100");
66                  reviews.get(i).setAuthorPhotoUrl(actualPhotoUrl);
67              }
68          } catch (Exception e) {
69              Log.i("LOG_PLACE_DETAIL", "Error getting avatar image");
70          }
71      }
72      public NearbySearchQuery getDefaultNearbySearchQuery(List<String> types) {
73          NearbySearchQuery nearbySearchQuery = new
NearbySearchQuery(mLocation.latitude, mLocation.longitude);
74          nearbySearchQuery.setRadius(AppSettings.GOOGLE_PLACES_LOCATION_RADIUS);
75          nearbySearchQuery.addTypes(types);
76          nearbySearchQuery.setKey(MainActivity.getApiKey());
77          return nearbySearchQuery;
78      }
79  }
```

ДОДАТОК Б

Комп’ютерна презентація

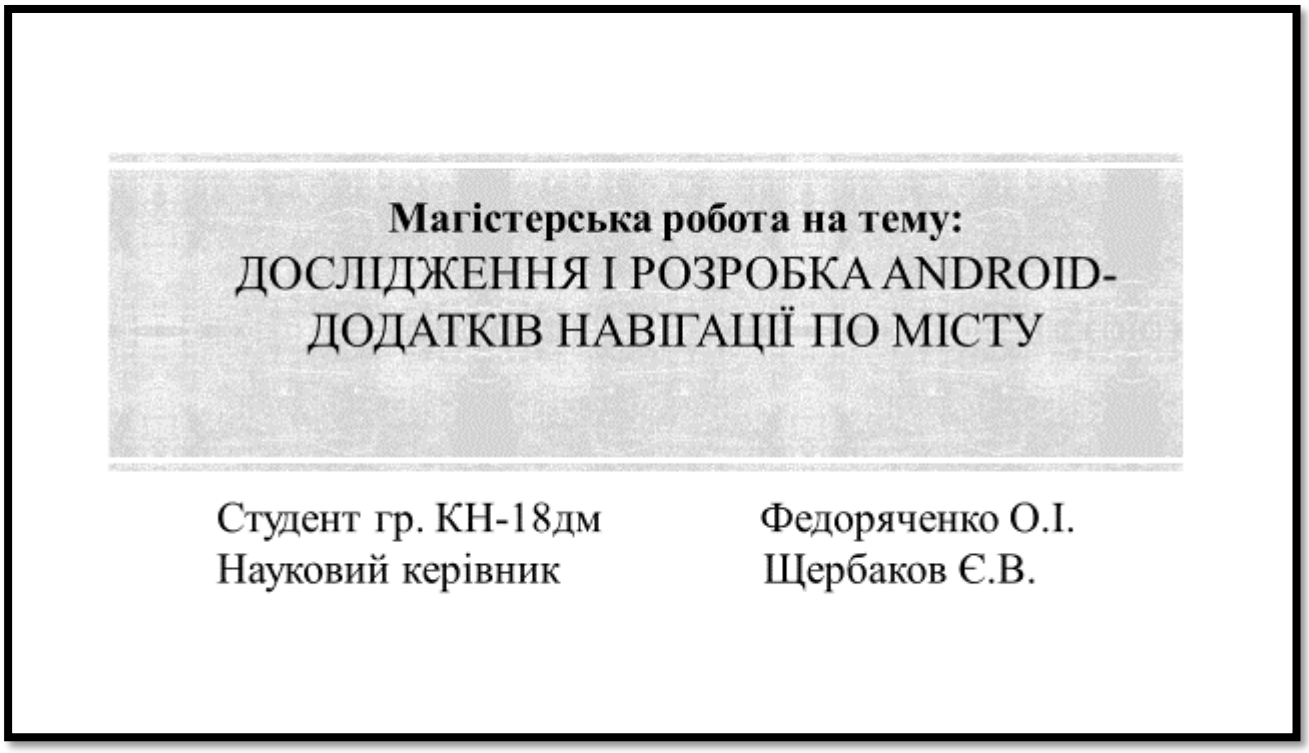

Рисунок Б.1 – Титульний лист

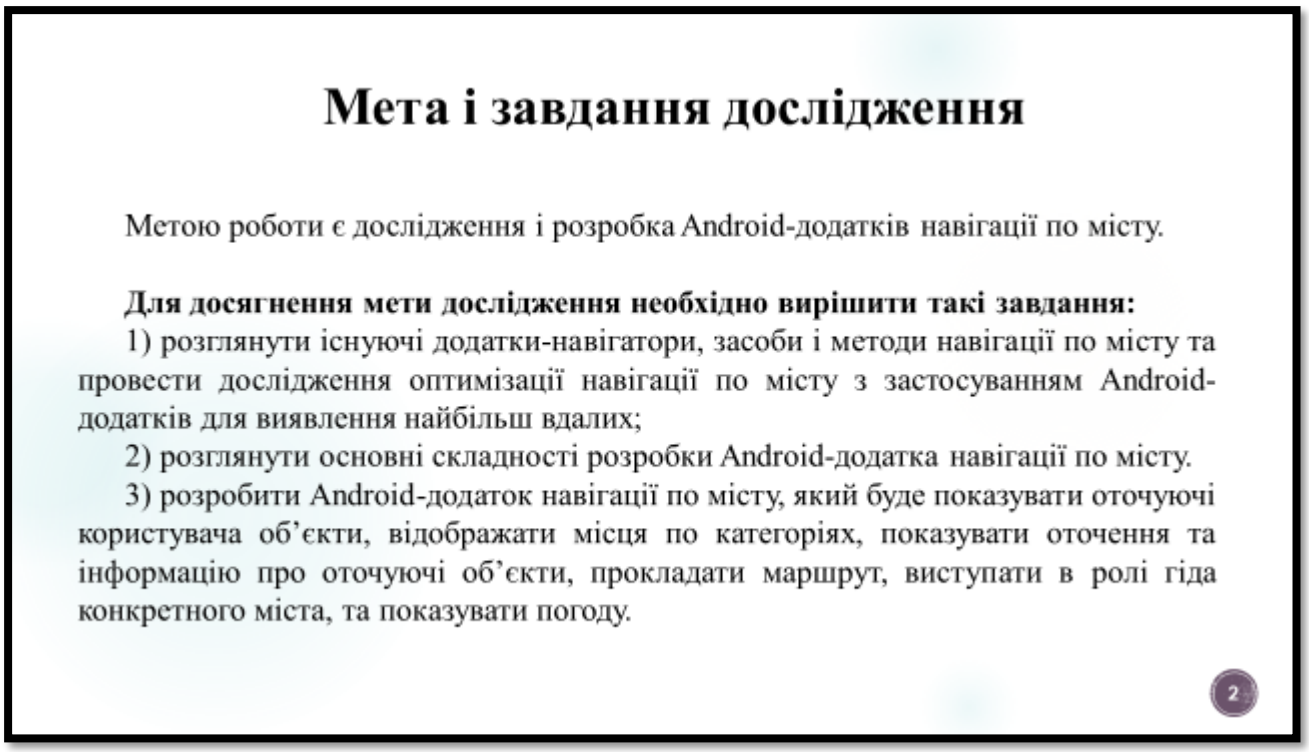

Рисунок Б.2 – Мета і завдання дослідження

Рисунок Б.3 – Приклади прокладання маршрутів без застосування фільтра на основі GeoHash і з його використанням

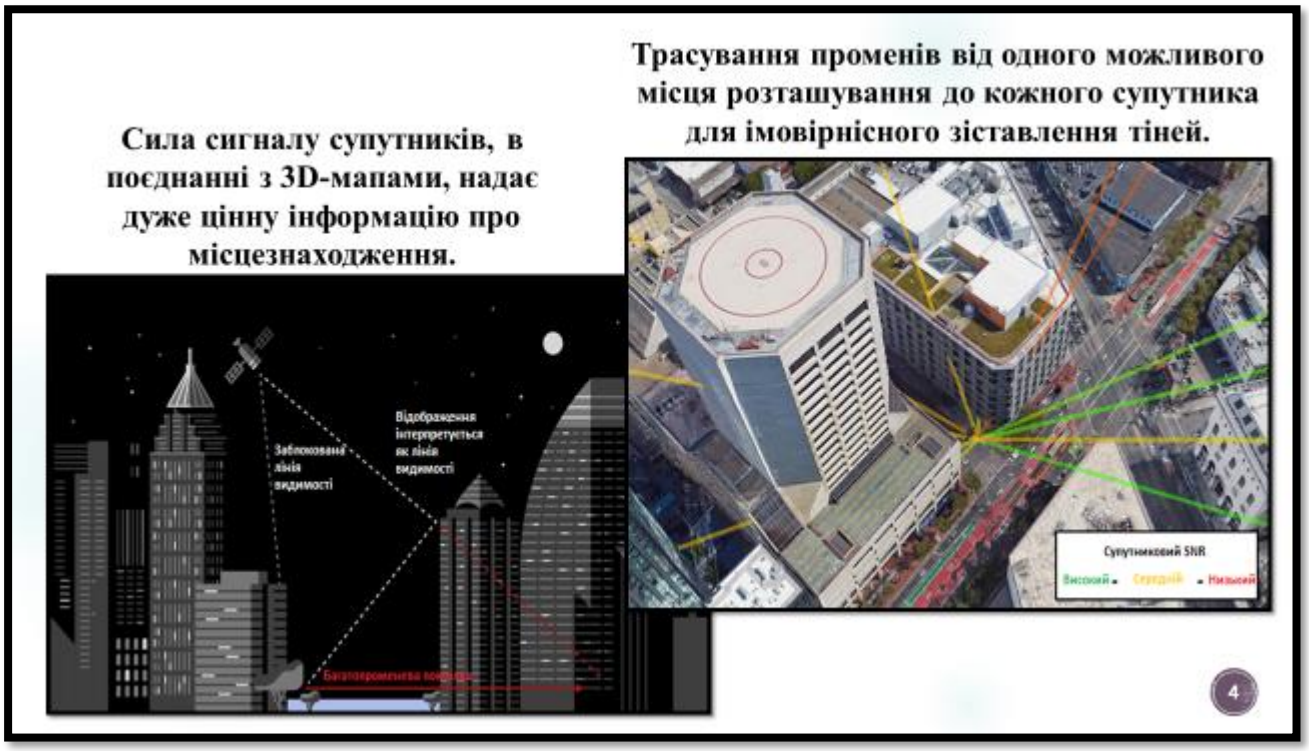

Рисунок Б.4 – Сила сигналу супутників, в поєднанні з 3D-мапами, надає дуже цінну інформацію про місцезнаходження. Трасування променів від одного можливого місця розташування до кожного супутника для імовірнісного зіставлення тіней.

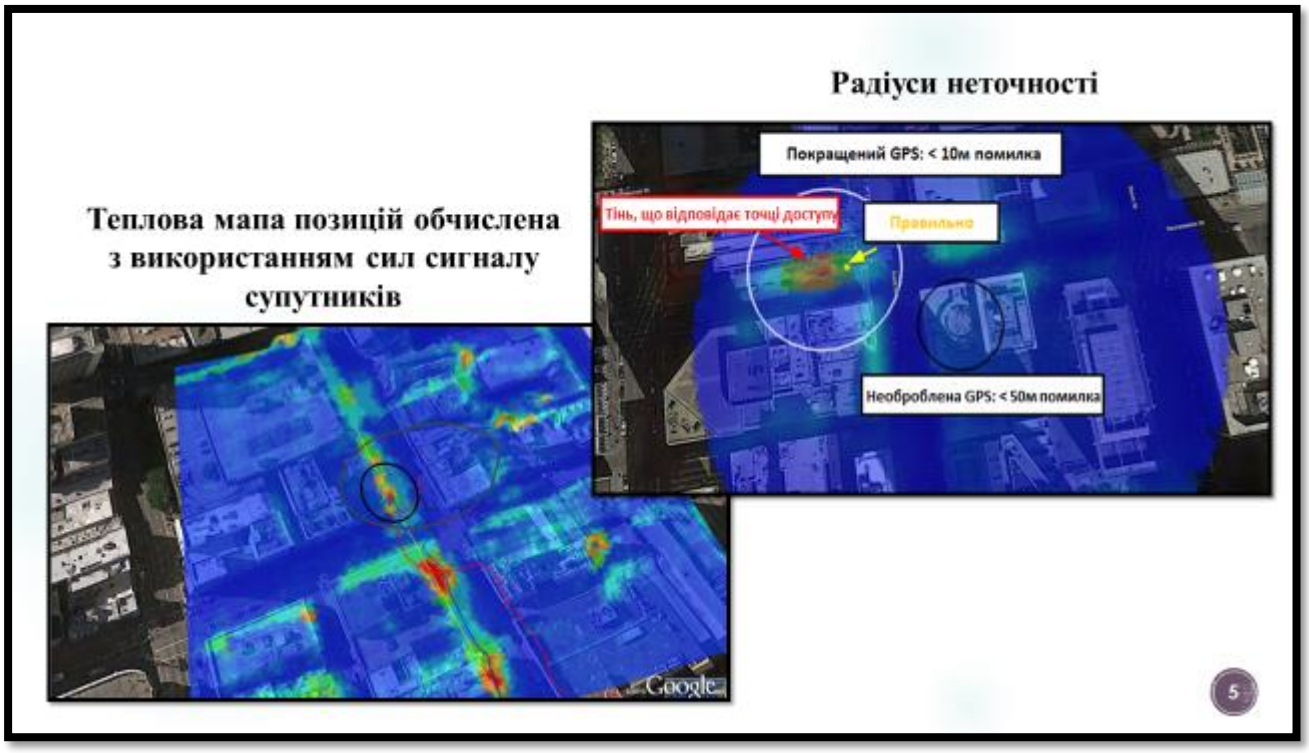

Рисунок Б.5 – Теплова мапа позицій обчислена з використанням сил сигналу супутників. Радіуси неточності

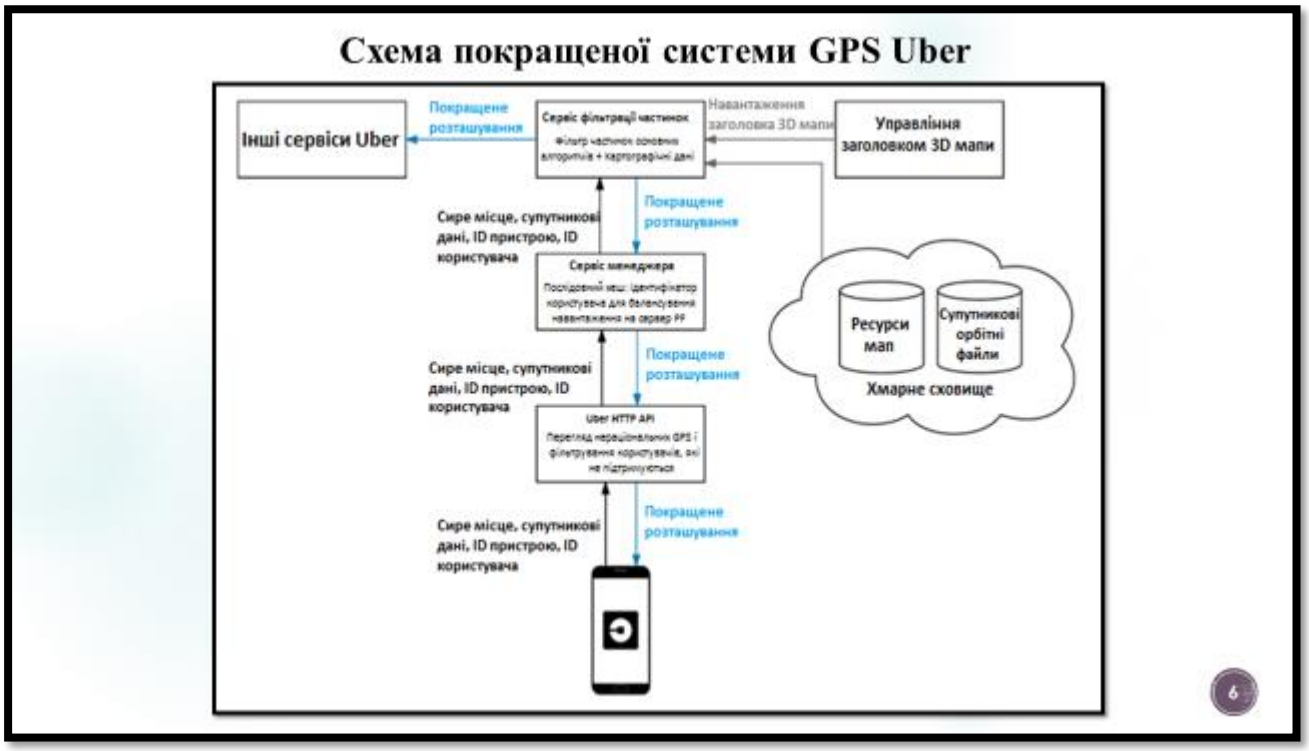

Рисунок Б.6 – Схема покращеної системи GPS Uber

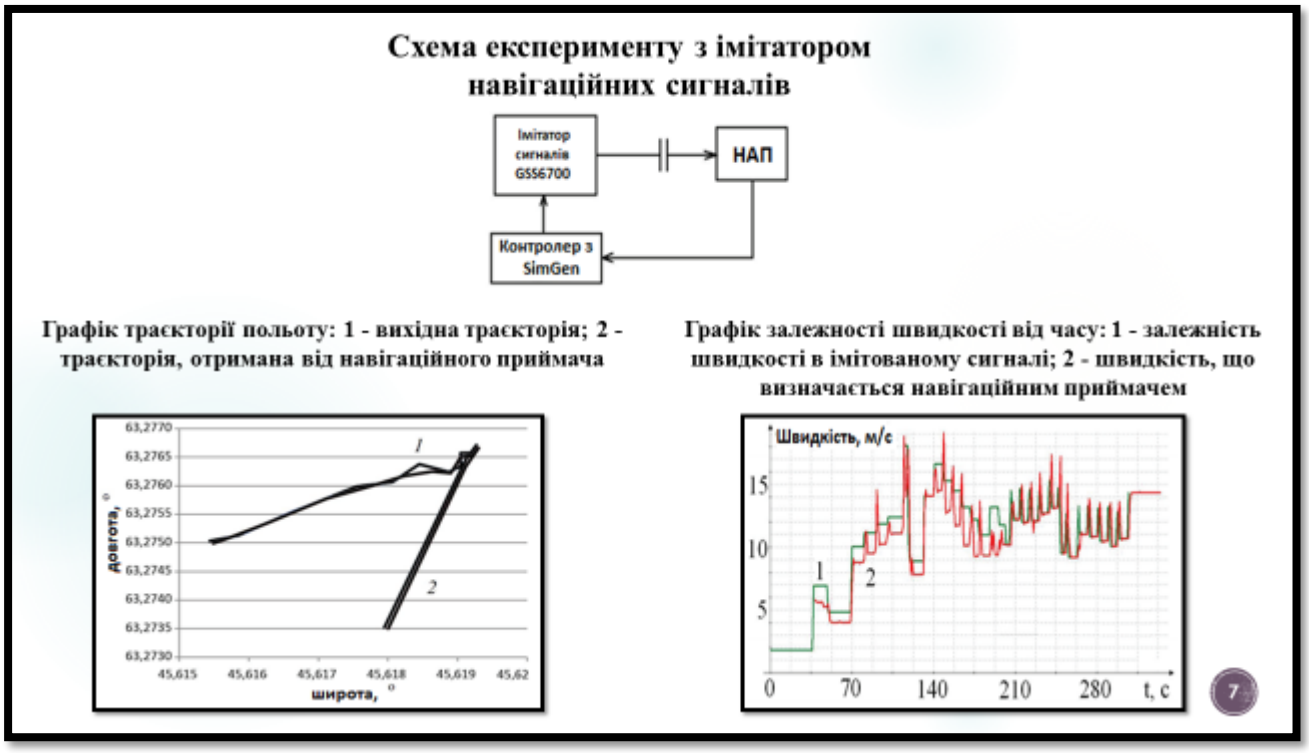

Рисунок Б.7 – Схема експерименту з імітатором навігаційних сигналів. Графік траєкторії польоту: 1 - вихідна траєкторія; 2 - траєкторія, отримана від навігаційного приймача. Графік залежності швидкості від часу: 1 - залежність швидкості в імітованому сигналі; 2 - швидкість, що визначається навігаційним приймачем

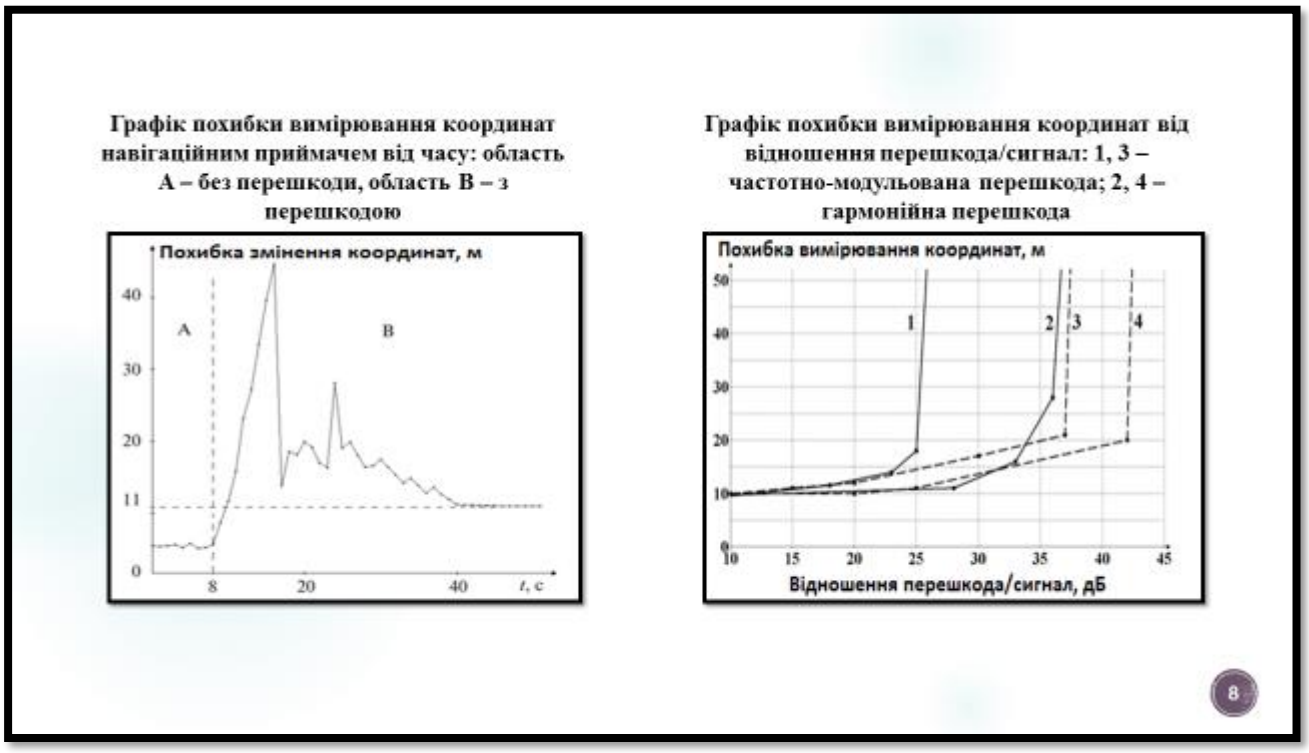

Рисунок Б.8 – Графік похибки вимірювання координат навігаційним приймачем від часу: область А – без перешкоди, область В – з перешкодою. Графік похибки

вимірювання координат від відношення перешкода/сигнал: 1, 3 – частотно-модульована перешкода; 2, 4 – гармонійна перешкода

Рисунок Б.9 – Схема обслуговування користувача додатка

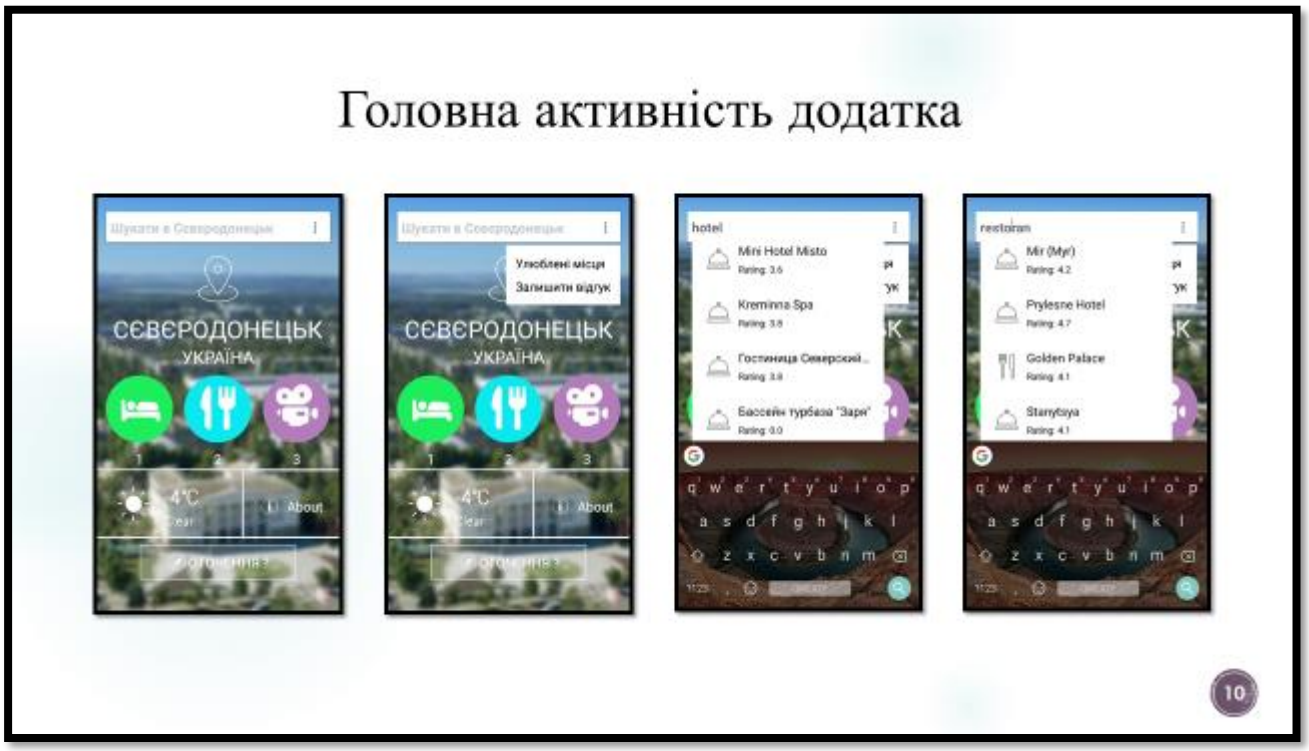

Рисунок Б.10 – Головна активність додатка

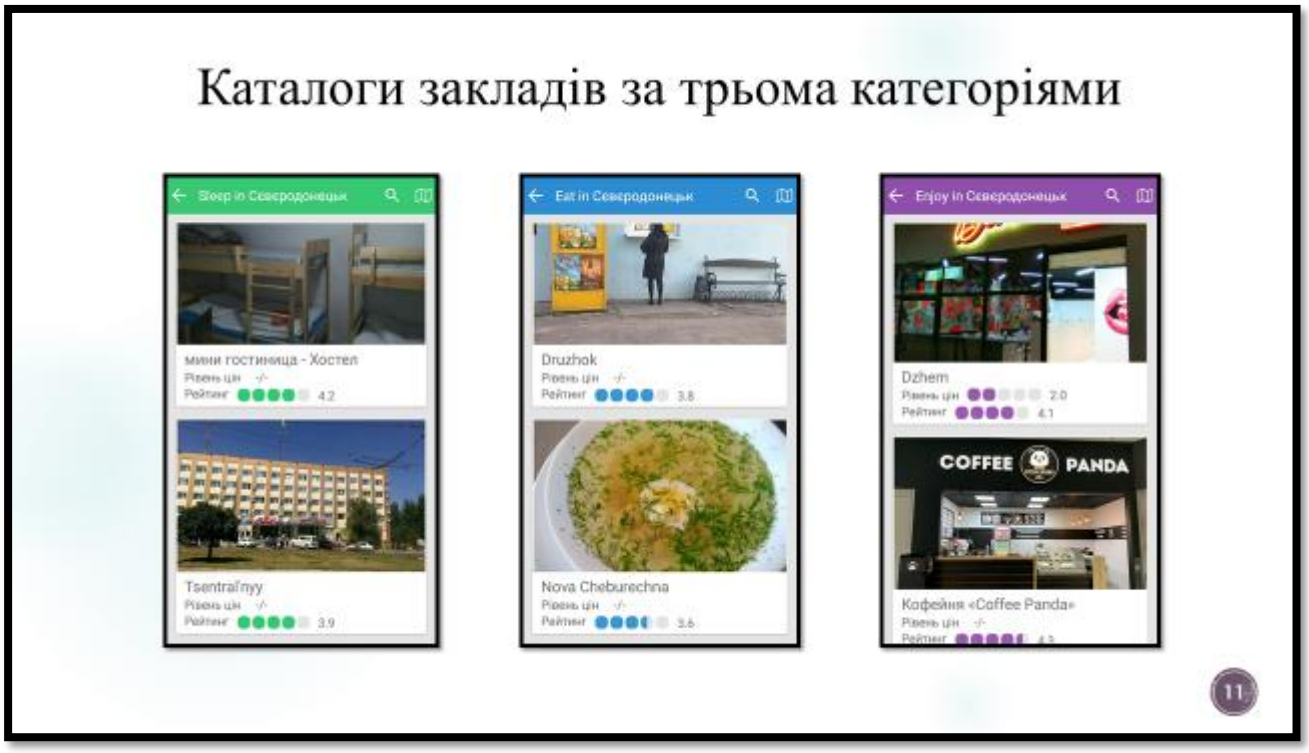

Рисунок Б.11 – Каталоги закладів за трьома категоріями

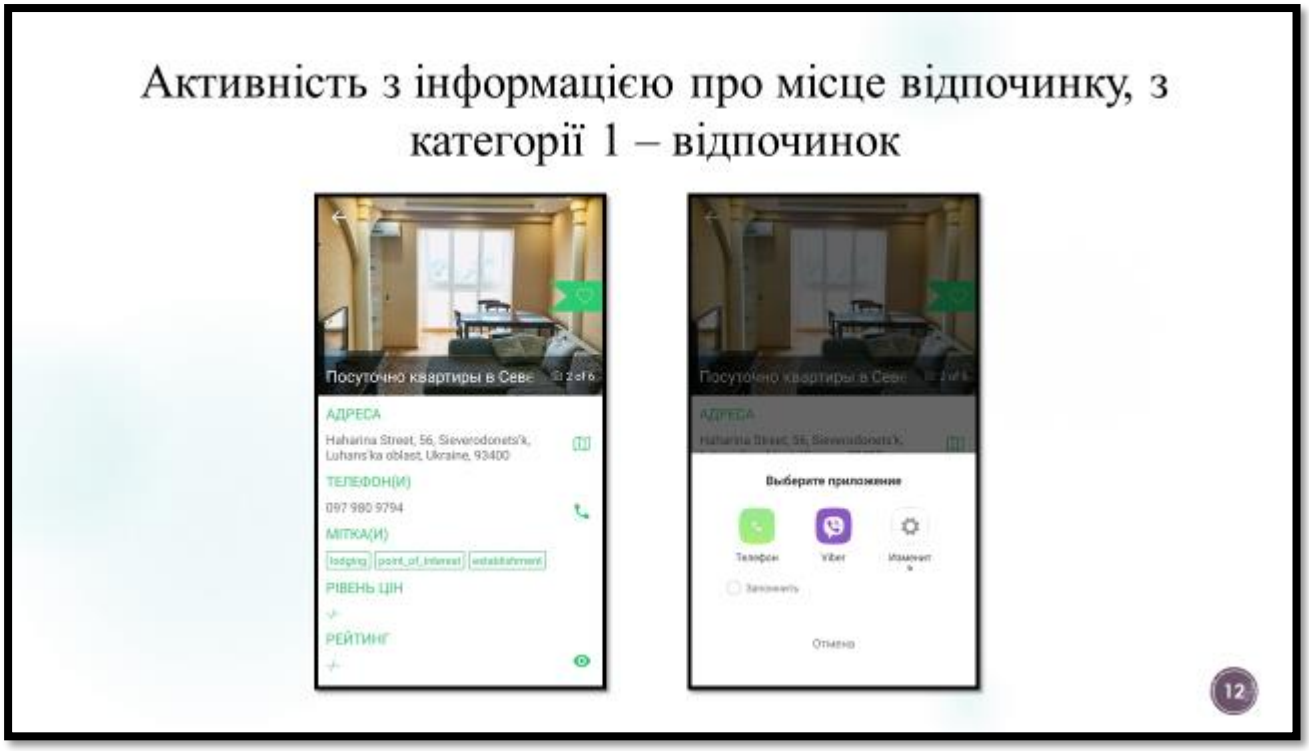

Рисунок Б.12 – Активність з інформацією про місце відпочинку, з категорії 1 – відпочинок

Рисунок Б.13 – Активність з інформацією про погоду

Рисунок Б.14 – Активність з інформацією про місто та фотоальбомом

Рисунок Б.15 – Активність з інформацією про оточення користувача

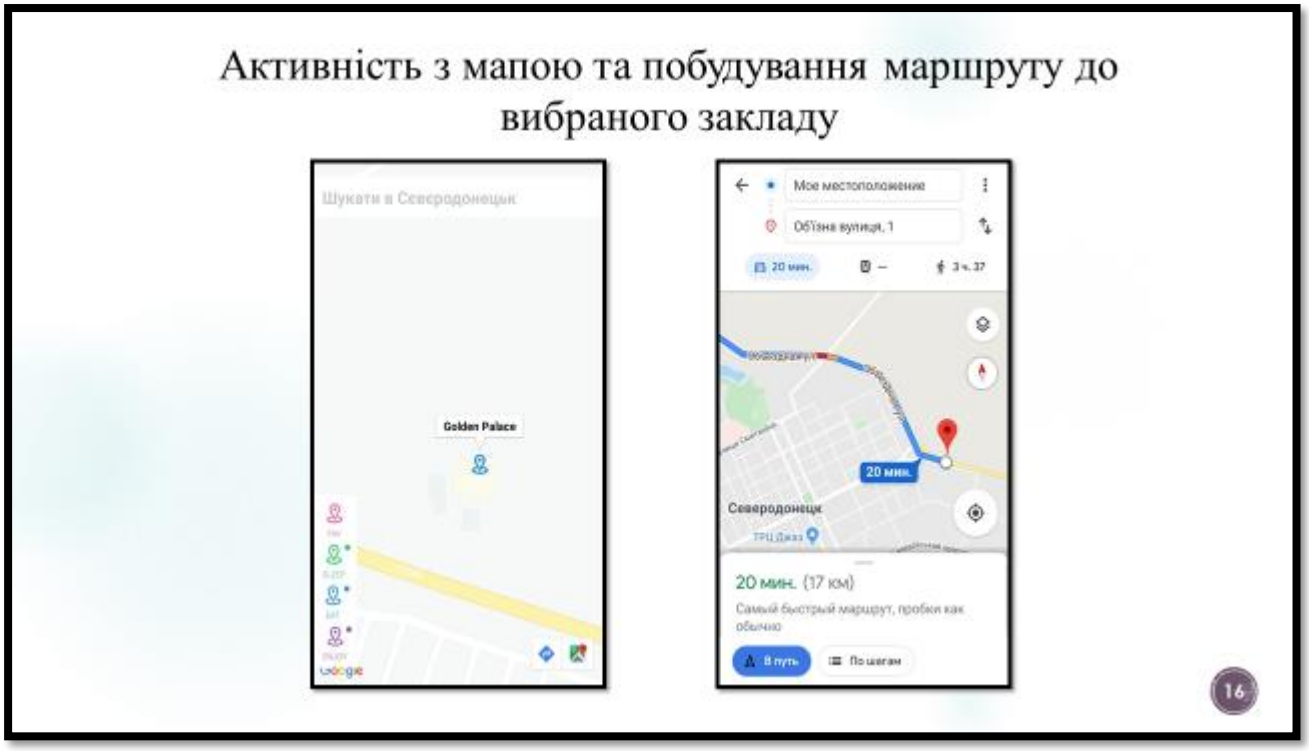

Рисунок Б.16 – Активність з мапою та побудування маршруту до вибраного закладу

Фрагмент коду AdapterPlaces.java

```
    @Override
public int getItemCount() {
return mAdapterData.size(); }
public static class ViewHolder extends RecyclerView.ViewHolder {
private View mParentView;
private LinearLayout mPriceLevelStarsContainer;
private LinearLayout mRatingLevelStarsContainer;
private ImageView mImageView;
private TextView mTextViewTitle;
private TextView mTextViewPriceLevelValue;
private TextView mTextViewRatingLevelValue;
public ViewHolder(View itemView) {
super(itemView);
mParentView = itemView;
mImageView = (ImageView) itemView.findViewById(R.id.list_item_places_image);
mTextViewTitle = (TextView) itemView.findViewById(R.id.list_item_places_title);
mTextViewPriceLevelValue = (TextView) itemView.findViewById(R.id.list_item_places_price_level_value);
mPriceLevelStarsContainer = (LinearLayout)
itemView.findViewById(R.id.list_item_places_price_level_stars);
mRatingLevelStarsContainer = (LinearLayout)
itemView.findViewById(R.id.list_item_places_rating_level_stars);
mTextViewRatingLevelValue = (TextView) itemView.findViewById(R.id.list_item_places_rating_value);} }
private LayoutInflater getLayoutInflater() {
return LayoutInflater.from(mContext); } }
```

17

Рисунок Б.17 – Фрагмент коду AdapterPlaces.java

Фрагмент коду GooglePlaces.java

```
public class GooglePlaces {
private final LatLng mLocation;
public GooglePlaces(LatLng location) {
this.mLocation = location;
}
Set<Place> resultPlaces = new HashSet<Place>();
public void getPlacesNearby(NearbySearchQuery query,
CallbackNearbyPlaces callbackNearbyPlaces, Constants.PLACE_TYPES type)
throws JSONException, IOException, InterruptedException {
Log.i("Loaded_Markers", "Executed query = " + query.toString());
PlacesResult result = new PlacesResult(NetworkFetcher.executeRequest(query.toString(), false));
callbackNearbyPlaces.onPlacesLoaded(result.getPlaces(), type);
Log.i("Loaded_Markers", "NextPageToken " + result.getNextPageToken());}
```

18

Рисунок Б.18 – Фрагмент коду GooglePlaces.java

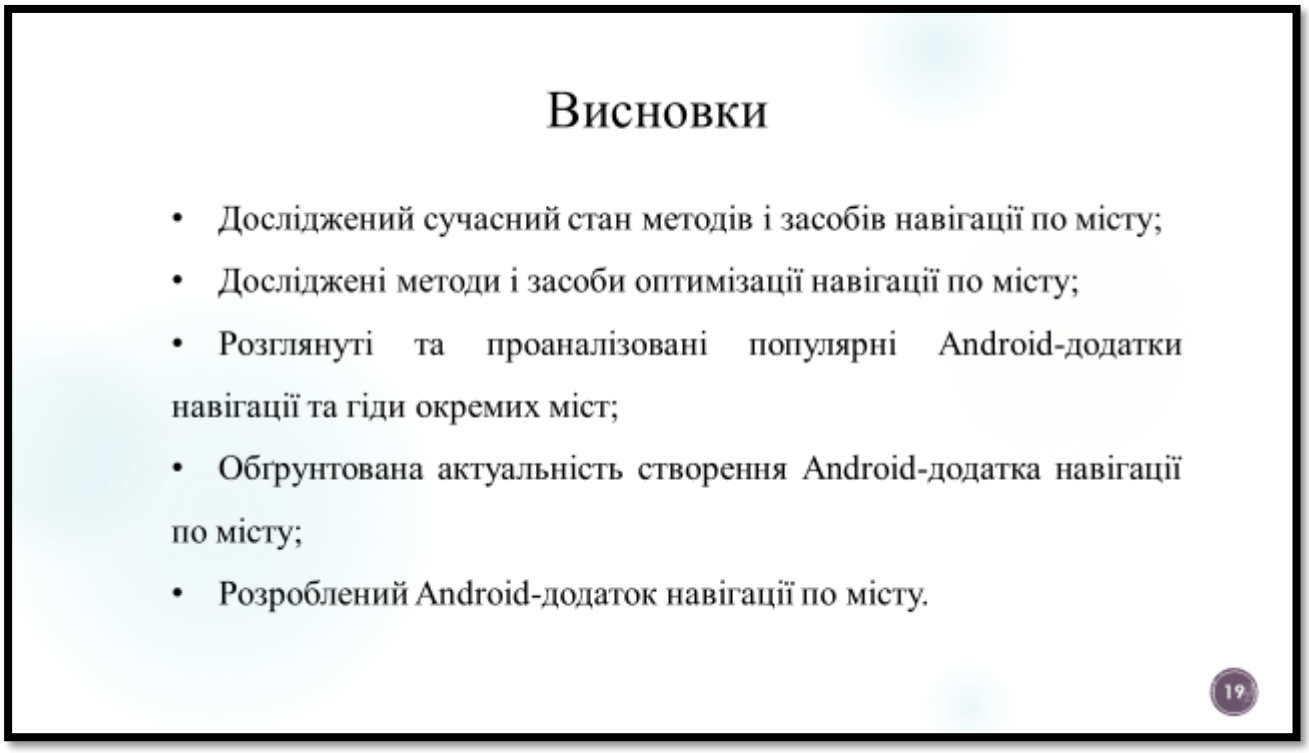

Рисунок Б.19 – Висновки