Вінницький національний технічний університет
Факультет інформаційних технологій та комп'ютерної інженерії
Кафедра програмного забезпечення

МАГІСТЕРСЬКА КВАЛІФІКАЦІЙНА РОБОТА

на тему:

«Розробка методу та програмного засобу для навігації із доповненою реальністю»

Виконав: студент II курсу
групи 3ПІ-22м спеціальності
121 – Інженерія програмного забезпечення
Кагальняк Р. Ю.

Керівник: к.т.н., доцент кафедри ПЗ
Бабюк Н. П.

« 18 » грудня 2023 р.

Опонент: к.т.н. доц.каф. ЗІ Маліновський В.І.

« 18 » грудня 2023 р.

Допущено до захисту
Завідувач кафедри ПЗ
д.т.н., проф. Романюк О.Н.
« 18 » грудня 2023 р.

ВНТУ – 2023

Вінницький національний технічний університет
Факультет інформаційних технологій та комп'ютерної інженерії
Кафедра програмного забезпечення
Рівень вищої освіти ІІ-й (магістерський)
Галузь знань 12 – Інформаційні технології
Спеціальність 121 – Інженерія програмного забезпечення
Освітньо-професійна програма – Інженерія програмного забезпечення

ЗАТВЕРДЖУЮ
Завідувач кафедри ПЗ
д.т.н., професор Романюк О. Н.
«19» вересня 2023 р.

З А В Д А Н Н Я
НА МАГІСТЕРСЬКУ КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТУ

Кагальняку Руслану Юрійовичу

1. Тема роботи – Розробка методу та програмного засобу для навігації із доповненою реальністю.
2. Керівник роботи: Бабюк Наталя Петрівна, к.т.н., доцент кафедри ПЗ, затверджені наказом вищого навчального закладу від «18» вересня 2023 р. №247.
   Строк подання студентом роботи 05.12.2023 р.
3. Вихідні дані до роботи: методи та алгоритми об'єктної доповненої реальності, візуальні діаграми системи, блок-схеми алгоритмів роботи програми, таблиці, що містять інформацію про оцінювання об'єктну доповнену реальність.
4. Зміст текстової частини: аналіз сучасного стану предметної області; порівняльний аналіз аналогів; постановка завдань для програмного засобу для навігації із доповненою реальністю; загальний опис методу для навігації із доповненою реальністю; удосконалення методу навігації з доповненою реальністю; розробка дизайну інтерфейсу і структурної схеми додатку; розробка алгоритму побудови маршруту за допомогою доповненої реальності; обґрунтування вибору засобів реалізації програми; реалізація методу для навігації із доповненою реальністю; результати роботи програмного засобу для навігації із доповненою реальністю; аналіз методів тестування програмного забезпечення; тестування розробленого програмного засобу; економічна частина.
5. Перелік ілюстративного матеріалу: мета роботи, об'єкт та предмет дослідження; основні задачі дослідження; порівняльний аналіз аналогів; розробка методу; алгоритм загальної роботи програми; вибір засобів для реалізації програмного засобу; реалізація методу навігації з доповненою реальністю; демонстрація вигляду розробленої програми; економічна доцільність розробки; висновки, публікації.

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
1-4	Бабюк Н. П., к.т.н., доц. каф. ПЗ	19.09.2023	05.12.2023
5	Причепа І.В., к.е.н., доц. каф. ЕПВМ	22.11.2023	01.12.2023

7. Дата видачі завдання 19.09.23

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів виконання магістерської роботи	Строк виконання етапів роботи	Примітка
1	Постановка задачі роботи	20.09.23 25.09.23	вик
2	Огляд існуючих методів та програмних засобів	25.09.23 01.10.23	вик
3	Аналіз існуючих методів навігації з доповненою рельністю	01.10.23 12.10.23	вик
4	Розробка методу навігації з доповненою рельністю	13.10.23 22.10.23	вик
5	Розробка інтерфейсу та тестування програми	23.10.23 05.11.23	вик
6	Підготовка матеріалів та опис розробки програмного засобу	05.11.23 10.11.23	вик
7	Розрахунок економічної частини роботи	20.11.23 01.12.23	вик
8	Оформлення пояснювальної записки та ілюстративного матеріалу	01.12.23 04.12.23	вик

Студент _____ **Кагальняк Р. Ю.**
(підпис) (прізвище та ініціали)

Керівник магістерської кваліфікаційної роботи _____ **Бабюк Н. П.**
(підпис) (прізвище та ініціали)

АНОТАЦІЯ

УДК 004.42

Кагальняк Р.Ю. Методи і програмні засоби для навігації із доповненою реальністю: магістерська кваліфікаційна робота зі спеціальності 121 – Інженерія програмного забезпечення, освітня програма – Інженерія програмного забезпечення. Вінниця: ВНТУ, 2023. 135 с.

На укр. мові. Бібліогр.: 21 назв; рис.: 52; табл. 12.

У магістерській роботі був проведений аналіз сучасного стану галузі мобільних навігаційних програм, визначено завдання та мету дослідження, а також чітко визначено предмет і об'єкт роботи. Проведено докладний порівняльний аналіз аналогів, що підтвердив актуальність обраного напрямку. Розроблена система надає користувачам навігаційні функції, використовуючи передову технологію доповненої реальності.

Розроблено алгоритми та блок-схеми для ключових операцій, а також адаптивний інтерфейс, сумісний з будь-яким Android смартфоном. Мобільний додаток написано з використанням мов програмування Kotlin і Java, інтегруючи Android SDK і Google Maps API. Основне середовище розробки – Android Studio. Результатом є мобільний додаток, який ефективно вирішує поставлені завдання та відповідає визначеній меті. Процес розробки включав перевірку швидкодії та коректності функціонування, що підтверджено успішними тестами.

ANNOTATION

UDC 004.42

Kahalniak R.Y. Methods and software tools for navigation with augmented reality: master's qualification thesis on specialty 121 - Software engineering, educational program - Software engineering. Vinnytsia: VNTU, 2023. 135 p.

In Ukrainian Bibliography: 21 titles; Fig.: 52; table 12.

In the master's thesis, an analysis of the current state of the mobile navigation applications sector was conducted, outlining the research objectives and purpose, as well as clearly defining the subject and object of the study. A comprehensive comparative analysis of analogs was carried out, confirming the relevance of the chosen direction. The developed system provides users with navigation functions utilizing augmented reality technology.

Algorithms and flowcharts were devised for key operations, along with an adaptive interface compatible with any Android smartphone. The mobile application was implemented using Kotlin and Java programming languages, integrating Android SDK and Google Maps API. The primary development environment chosen was Android Studio. The resulting mobile application effectively addresses the set tasks and aligns with the defined objectives. The development process included performance and functionality checks, validated through successful testing.

ЗМІСТ

ВСТУП

З інноваціями технологій ми постійно досліджуємо нові шляхи взаємодії з оточуючим нас середовищем. Один із найцікавіших та перспективних напрямків – це доповнена реальність (AR). Програми з AR набувають все більшої популярності, відкриваючи нові можливості для користувачів у різних сферах життя. В цій кваліфікаційній роботі було розглянуто актуальність застосунків розширеної реальності та їх вплив на сучасне суспільство.

Технологія доповненої реальності надає користувачам можливість розширення сприйняття навколишнього світу, додаючи віртуальні об'єкти та інформацію до реального оточення. Вони трансформують наш спосіб взаємодії з оточуючим середовищем, відкриваючи нові перспективи для освіти, розваг, маркетингу, медицини та інших сфер.

Технологія AR відкриває нові можливості для освіти і навчання, дозволяючи створювати інтерактивні уроки, де студенти можуть досліджувати різні предмети в новому форматі. Завдяки AR-додаткам можливо відобразити тривимірні моделі, історичні події або складні концепції, сприяючи кращому розумінню та запам'ятовуванню матеріалу. В розважальній індустрії вони стали невід'ємною частиною, надаючи можливість взаємодіяти з віртуальними персонажами, грати в ігри та розвивати логічне мислення.

AR-додатки також збагачують туристичний досвід, надаючи додаткову інформацію про визначні місця та культурні пам'ятки. У медичній галузі вони виявляються корисними для навчання студентів, симуляції хірургічних процедур та розробки нових методів лікування. Ці додатки дозволяють віртуально відтворити реальні умови, що сприяє набуттю лікарями досвіду та підвищує ефективність медичних процедур.

Застосунки розширеної реальності розкривають нові горизонти для сприйняття та взаємодії з світом, покращуючи процеси навчання, розваг та туризму, а також сприяючи розвитку медицини. Зростаюча популярність

AR-додатків свідчить про їхню актуальність та важливість у сучасному суспільстві. Розширена реальність перетворює наше сприйняття світу, і її потенціал ще далеко не вичерпаний.

Зв'язок роботи з науковими програмами, планами, темами. Робота виконувалася згідно плану виконання наукових досліджень на кафедрі програмного забезпечення.

Мета та завдання дослідження. Метою магістерської роботи є підвищення ефективності та спрощення процесу навігації, завдяки розробці навігатора з доповненою реальністю.

Основними задачами дослідження є:

– аналіз сучасного стану досліджуваної предметної області;

– проведення порівняльного аналізу аналогів;

– удосконалення методу об'єктної доповненої реальності, розробка алгоритмів реалізації навігації;

– реалізація програмного засобу для навігації з доповненою реальностю;

– тестування створеного програмного засобу;

– економічне обґрунтування доцільності наукового дослідження.

Об'єктом дослідження є процес розробки додатків із технологією доповненої реальності для платформи Android.

Предмет дослідження – розробка та оцінювання програмного засобу для навігації з доповненою реальністю.

Методи дослідження. Протягом дослідження було використано: методи розробки клієнтського інтерфейсу; методи навчання; методи оптимізації клієнтського інтерфейсу та мобільного додатка вцілому; методи алгоритмізації модулів програмної системи.

Наукова новизна отриманих результатів: Подальшого розвитку отримав метод нанесення навігаційних покажчиків за допомогою технології

AR, який відрізняється від досліджуваних зрозумілістю отриманого маршруту і, таким чином, полегшує процес навігації.

Практична цінність одержаних результатів полягає в тому, що запропонований метод та програмний засіб для навігації з доповненою реальністю дає можливість швидше та легше прокласти маршрут і зорієнтуватись у просторі .

Особистий внесок здобувача. Наукові результати, які містяться у магістерській кваліфікаційній роботі, були отримані індивідуально автором. У працях, опублікованих у співавторстві, автор має такі результати: архітектура клієнтських інтерфейсів; алгоритми розробки навігації з доповненою реальністю.

Апробація матеріалів магістерської кваліфікаційної роботи. Основні положення магістерської кваліфікаційної роботи доповідалися та обговорювалися на міжнародних та всеукраїнських конференціях: LI Науково-технічна конференція факультету інформаційних технологій та комп'ютерної інженерії (2022), Молодь в науці: дослідження, проблеми, перспективи (МН-2023).

Публікації.

Технічне завдання наведено в додатку А.

1 АНАЛІЗ ТА ОБҐРУНТУВАННЯ ВИБОРУ МЕТОДУ РОЗРОБКИ ТА ПОСТАНОВКА ЗАДАЧІ ДОСЛІДЖЕННЯ

1.1 Аналіз сучасного стану предметної області

Доповнена реальність (ДР) є передовою інформаційною та технологічною парадигмою, яка перетинає межі віртуального та фізичного світу, створюючи новий рівень взаємодії та сприйняття реальності [1]. Доповнена реальність є інтерактивною системою, що поєднує фізичний світ із віртуальним, додаючи до реального оточення віртуальні об'єкти, дані чи інформацію. Вона трансформує спосіб сприйняття користувача, розширюючи його чуттєві можливості та створюючи унікальний інтерфейс взаємодії. Доповнена реальність включає ключові компоненти, які визначають її функціональність, такі як сенсорні технології, віртуальну графіку та алгоритми розпізнавання. Сенсорні технології взаємодії, такі як камери, акселерометри, гіроскопи, надають системі можливість розпізнавати рухи та положення об'єктів в реальному часі. Використання віртуальної графіки дозволяє відтворити віртуальні об'єкти у реальному оточенні, створюючи ефект взаємодії з реальністю. Алгоритми розпізнавання об'єктів та маркерів грають ключову роль у визначенні положення та взаємодії віртуальних об'єктів в реальному просторі.

Доповнена реальність знаходить широке застосування в різних галузях, включаючи освіту, медицину, промисловість та розваги. Вона відкриває нові перспективи для створення інтерактивних інтерфейсів та оптимізації робочих процесів.

У сфері освіти використання доповненої реальності дозволяє створювати інтерактивні навчальні моделі. Наприклад, студенти можуть вивчати анатомію, використовуючи додаток, який проектує тривимірні моделі органів, дозволяючи їм ретельніше досліджувати будову та функції. Також доповнена

реальність надає можливість для віртуальних екскурсій. Туристи, учні чи студенти можуть досліджувати історичні пам'ятки, музеї та географічні особливості, використовуючи віртуальні проекції на своїх пристроях.

У медичній практиці доповнена реальність використовується для тренування майбутніх лікарів. Імітація хірургічних втручань на тривимірних моделях дозволяє медичним студентам набувати практичний досвід та розвивати навички.

Для покращення точності діагностики та лікування, доповнена реальність використовується для візуалізації медичних даних. Лікарі можуть переглядати тривимірні моделі органів та структур для більш ефективної роботи. У галузі розваг доповнена реальність перетворює ігровий простір. Гравці можуть взаємодіяти з віртуальними об'єктами, які інтегруються в реальний світ, створюючи інтерактивний геймплей.

Мистецтво та культура також використовують доповнену реальність для створення унікальних арт-проектів та виставок. Відвідувачі можуть розглядати віртуальні елементи, які доповнюють твори мистецтва чи інтер'єрні рішення.

У промисловості, де точність та безпека є пріоритетами, доповнена реальність використовується для тренування робітників та обслуговування складних механізмів. Вона надає віртуальні інструкції та симулює реальні умови роботи. У сфері проектування та будівництва доповнена реальність дозволяє інженерам та архітекторам візуалізувати проекти в реальному просторі. Це сприяє зручній взаємодії та вирішенню конструктивних завдань.

У сфері роздрібної торгівлі доповнена реальність може надавати віртуальні примірки товарів. Покупці можуть випробувати вироби в режимі реального часу, перш ніж здійснити покупку онлайн.

Ці приклади демонструють багатогранність та універсальність застосування технології доповненої реальності в різних сферах. Від освіти та медицини до розваг, промисловості та роздрібної торгівлі, доповнена

реальність виходить за межі технічних інновацій, стаючи важливим інструментом у сучасному суспільстві.

Проте, незважаючи на перераховані переваги доповненої реальності, існують також певні недоліки та обмеження. Одним із основних технічних недоліків доповненої реальності є висока енергозатратність. Застосування AR-технологій може призводити до швидкого розрядження акумуляторів мобільних пристроїв, що обмежує тривалість їх використання. Також для оптимальної роботи доповненої реальності необхідне стабільне та швидке Інтернет-з'єднання. Однак у деяких регіонах або умовах пересування може виникати проблема обмеженої швидкості передачі даних, що негативно впливає на досвід користувача.

Загалом, технологія доповненої реальності має великий потенціал у різних сферах нашого життя, відкриваючи нові перспективи для освіти, реклами, медицини та промисловості. З розвитком мобільних пристроїв та постійним удосконаленням технологій AR, можна очікувати подальший ріст та інновації в цьому напрямку, що суттєво змінить наше сприйняття та взаємодію з навколишнім світом.

Доповнена реальність існує у різних формах та типах, в залежності від способу її реалізації та взаємодії з користувачем.

Маркерна доповнена реальність (Marker-Based AR) – використовує спеціальні маркери або зображення для ідентифікації та розпізнавання місця встановлення віртуальних об'єктів. Коли камера пристрою визнає маркер, система може виводити віртуальні об'єкти на екран, пов'язані з цим маркером. Цей підхід використовується в різних сферах, включаючи розваги, маркетинг і освіту. Наприклад, додатки для музеїв можуть використовувати маркери на експонатах для надання додаткової інформації або віртуальних демонстрацій.

У безмаркерній доповненій реальності (Markerless AR) відсутні маркери, і вона базується на визначенні точного положення та розпізнаванні об'єктів в

реальному світі. Вона використовує характеристики зображення, сенсори та GPS [2]. Такий підхід застосовується у різних галузях, таких як навігація, віртуальні тестування об'єктів або створення інтерактивних просторів. Наприклад, додатки для навігації можуть використовувати цю технологію для розміщення віртуальних покажчиків чи інших об'єктів у реальному часі.

Проекційна AR використовує проектори для відображення віртуальних об'єктів на реальних поверхнях. Це може бути застосовано в різних областях, включаючи освіту, бізнес та розваги. Наприклад, додатки для бізнес-презентацій можуть проектувати важливі дані або графіки на конференц-столі під час наради.

Спрямована AR залежить від географічного положення користувача. GPS та компас використовуються для розміщення віртуальних об'єктів у реальному світі. Це застосовується у географічних іграх, туризмі та навігації. Наприклад, додатки для екскурсій можуть використовувати цю технологію для надання інформації про історичні місця в реальному часі.

Суміщена AR інтегрує віртуальні та реальні об'єкти, створюючи ілюзію, що вони існують у тому ж просторі. Це використовується у віртуальних візуалізаціях, тренуваннях та розвагах. Наприклад, в області медицини, ця технологія може допомагати хірургам переглядати віртуальні зображення під час операцій, щоб отримати додаткову інформацію.

Часто разом з доповненою реальністю розглядають віртуальну реальність. Обидві технології мають ефект імерсії, але між ними є сильні відмінності.

AR і VR різняться на рівні взаємодії з реальним світом. AR, зокрема, взаємодіє з реальним оточенням, додаючи віртуальні об'єкти чи інформацію до нашого реального поля зору. Це може бути відображення інструкцій на екрані смартфона або окуляра, відображення інформаційних шарів на карті чи додавання ігрових персонажів до реального світу через екран пристрою. Такий

підхід дозволяє користувачеві продовжувати взаємодіяти з оточуючим середовищем.

VR в свою чергу занурює нас у цілком віртуальне середовище, відокремлюючи від реального світу. Завдяки спеціальним гарнітурам, користувач поглиблюється у іншу реальність, де кожен аспект зорового сприйняття створюється штучно. Це використовується в іграх, тренуваннях та віртуальному туризмі.

У відношенні до технічних вимог, AR може бути реалізована на сучасних мобільних пристроях, таких як смартфони та планшети, або спеціальних AR-окулярах. Це зробило AR більш доступною для широкого кола користувачів. З іншого боку, VR вимагає потужних обчислювальних систем і спеціалізованих гарнітур, що робить його менш доступним для масового використання.

Сфери використання AR та VR розпливаються в різних галузях. AR знайшла застосування в освіті, медицині, рекламі та навігації, надаючи користувачам можливість взаємодіяти з додатковим контентом у реальному світі. Наприклад, AR-додатки на смартфонах можуть допомагати в навчанні, розширюючи можливості віртуального навчання.

З іншого боку, VR використовується в іншій категорії застосувань, таких як іммерсивні ігри, тренування, віртуальні прогулянки або терапія. Технологія VR дозволяє користувачеві потрапити в інший світ, що робить його ідеальним для ситуацій, де повна ізоляція від реальності бажана.

1.2 Порівняльний аналіз аналогів розроблюваного додатку

Додатки доповненої реальності (AR) вирізняються своєю здатністю поєднувати віртуальний контент з реальним оточенням, створюючи унікальні можливості для користувачів. Вони здатні відтворювати 3D-моделі, анімацію,

інтерактивні об'єкти та спеціальні ефекти, які інтегруються з реальним світом через камеру пристрою.

AR-додатки вміють виконувати відстеження об'єктів, розпізнавати маркери та взаємодіяти з реальними об'єктами, надаючи користувачам нові досвіди взаємодії з віртуальними об'єктами у реальному часі та просторі. Вони також можуть використовувати геолокацію, GPS та датчики пристрою для створення контекстної AR-інформації, такої як навігація чи інформація про місцевість.

З метою визначення актуальності розробки мобільного додатку, проводиться аналіз аналогів, що дозволяє зрозуміти доцільність розробки даного програмного засобу. Розглядаючи різні види мобільної розробки, включаючи розробку для Android, iOS та кросплатформенну розробку, аналізу піддаються додатки з усіх цих груп.

Google Maps (рис. 1.1) – один із найпопулярніших сервісів у цій галузі, який вирізняється своєю широкою функціональністю [3]. Розглянемо його основні переваги цього додатку. По-перше, Google Maps надає широкий функціонал для навігації. Користувач може прокладати маршрути для різних видів транспорту, користуватися пішохідними маршрутами та отримувати інформацію про громадський транспорт.

Оновлення в реальному часі – ще одна суттєва перевага. Google Maps надає актуальну інформацію про трафік та можливі затримки на маршрутах в режимі реального часу, що дозволяє обирати оптимальний шлях. Інтерактивна карта з режимами масштабування та повороту робить процес планування маршруту точним та зручним.

Інтеграція з іншими сервісами Google, такими як Street View, розширює можливості користувача для детального вивчення вулиць та місцевості. Також

велика кількість відгуків та оцінок дозволяє отримати рекомендації від інших користувачів.

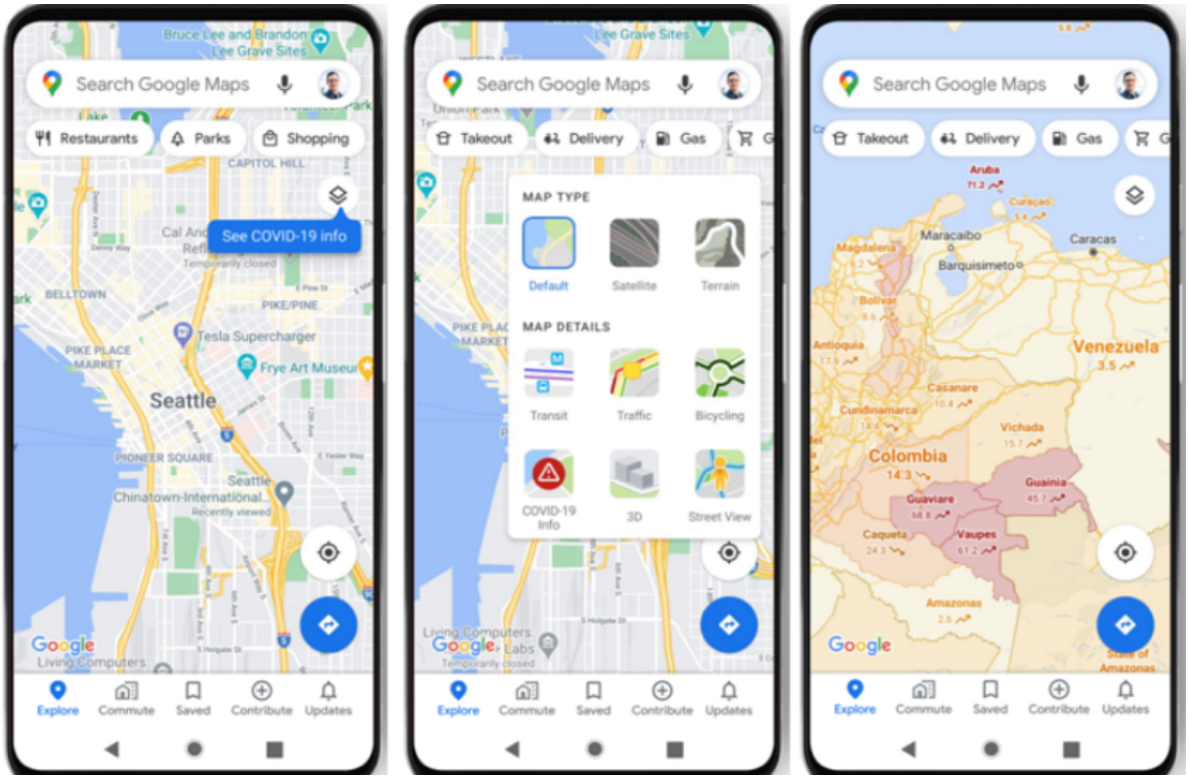

Рисунок 1.1 – Інтерфейс мобільного навігатора Google Maps

Проте не обійшлося і без недоліків. Високе завантаження ресурсів – один із головних мінусів. Велика кількість функцій та налаштувань може здатися новачкам неінтуїтивним. Залежність від стабільного інтернет-з'єднання може обмежити доступ до деяких функцій.

Apple Maps (рис. 1.2) – додаток, що входить до складу екосистеми Apple та використовується мільйонами користувачів. [4]. Додаток має звичні функції навігатора, такі як перегляд карт, визначення власного місцезнаходження, пошук локації за назвою та за координатами, а також, власне, прокладання маршруту. Однією з ключових переваг Apple Maps є інтеграція із екосистемою Apple, що сприяє відмінній синхронізації та взаємодії з іншими пристроями та сервісами компанії. Миттєві оновлення карт та адрес, які відбуваються

безпосередньо на пристрої, роблять використання додатку ефективним та зручним.

Apple Maps відомий своєю точністю та ретельністю картою країни, де враховуються навіть маленькі дорожні зміни. Важливим аспектом є також використання Siri для навігації, що забезпечує безпечну та комфортну поїздку без відволікання уваги від дороги.

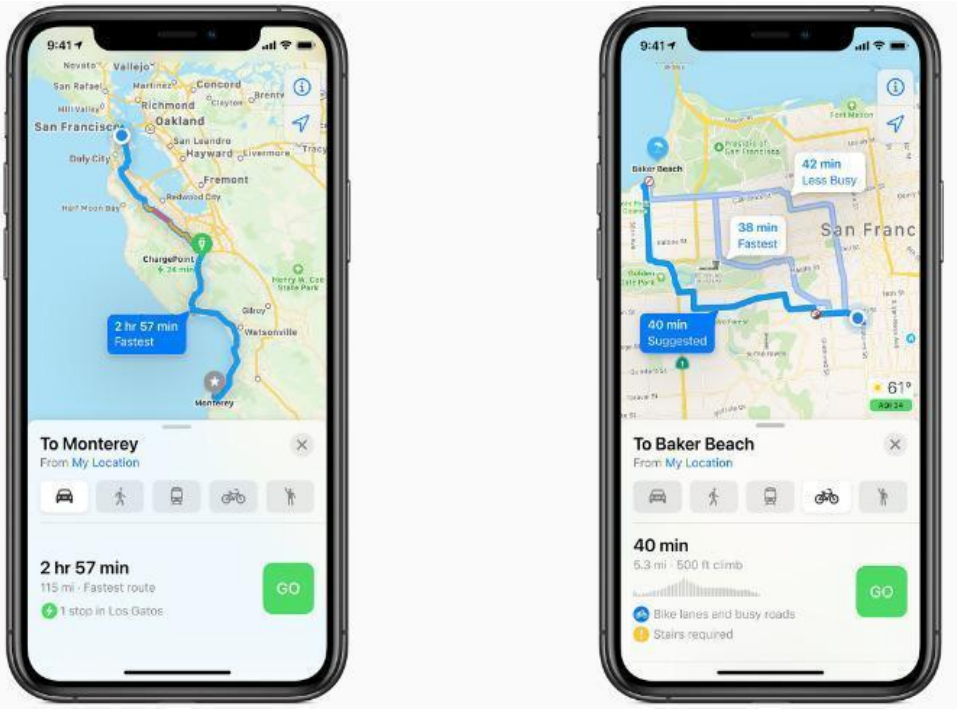

Рисунок 1.2 – Інтерфейс мобільного навігатора Apple Maps

Але також додаток Apple Maps має ряд недоліків. По-перше, додаток інтегрований лише з пристроями від Apple, що робить його менш універсальним порівняно з іншими навігаційними додатками. По-друге, в деяких регіонах відсутній повний функціонал, наприклад, відсутність деяких функцій громадського транспорту.

Також, як і в інших подібних додатках, Apple Maps вимагає стабільне інтернет-з'єднання для оптимальної роботи, що може бути проблематичним у віддалених районах або під час подорожі за кордон.

Насамкінець, додаток Apple Maps має не зовсім класичний інтерфейс в порівнянні з іншими навігаційними мобільними додатками, що робить його менш зручним для деяких користувачів.

ARCity (рис. 1.3) – мобільний додаток створений компанією Blippar для мобільних телефонів на основі операційної системи iOS [5]. Ідея створення ARCity виникла в 2018 році, коли команда молодих та амбіційних розробників, які мали значний досвід у галузі мобільних додатків та AR-технологій, об'єднали свої зусилля. Головною метою було створення навігаційного додатку, який не лише допомагав би користувачам знаходити шлях, але й робив би цей процес захоплюючим та відновлював зв'язок людей з навколишнім світом. Перші кроки були спрямовані на аналіз потреб користувачів та огляд існуючих навігаційних додатків. Команда розробників активно вивчала технологічні можливості та тенденції в галузі AR. Знання індустрії та внутрішній палкій бажання створити щось унікальне стали визначальними для подальших кроків. У 2019 році відбулося офіційне заснування компанії-розробника, "InnoAR Solutions." Велика увага була приділена вибору технічного стеку та створенню команди професіоналів, які мали великий досвід у розробці AR-додатків.

ARCity ставить особливий акцент на соціальний взаємодію користувачів, надаючи можливість обмінюватися геолокацією та розташуванням цікавих місць. Це робить додаток прекрасним інструментом для планування подорожей та зустрічей з друзями.

Ще однією перевагою є використання AR для відображення інформації про навколишні об'єкти в реальному часі. Відоме як "Augmented Reality

Browsing," ця функція дозволяє користувачам отримувати додаткові дані про навколишність за допомогою віртуальних об'єктів.

Рисунок 1.3 – Приклад роботи додатку ARCity

Sygic (рис. 1.4) – навігаційна система, розроблена однойменною словацькою компанією Sygic. Sygic була заснована в 2004 році в Словаччині, і з того часу стала лідером у галузі розробки навігаційних додатків для мобільних пристроїв. Початково компанія фокусувалася на створенні навігаційного програмного забезпечення для автомобілів, проте з плином часу їхні продукти розширились на інші платформи та види транспорту. Ця компанія вперше запропонувала навігаційні рішення для iPhone та стала другою, яка введенням подібних послуг підтримала платформу Android. Її програмне забезпечення працює на різних операційних системах, таких як Android, Android Auto, iOS, Windows Phone та Symbian, і надає картографічні

дані для понад 200 країн світу, обслуговує також понад 30 мов. Користувачі навігаційного застосунку Sygic GPS мають можливість завантажувати карти безпосередньо на свої пристрої та використовувати їх у випадках, коли їм необхідне навігаційне ведення, навіть у відсутності Інтернет-з'єднання. Компанія Sygic оптимізує обсяг даних, необхідних для завантаження, забезпечуючи користувачам змогу використовувати карти в автономному режимі і займати мінімальний обсяг пам'яті на їхніх пристроях.

Рисунок 1.4 – Мобільний додаток Sygic

Однією з ключових переваг є можливість використання додатку в офлайн-режимі. Це особливо важливо для подорожей у віддалені регіони без доступу до Інтернету. Sygic надає картографічні дані для більш ніж 200 країн світу, що робить його відмінним партнером для подорожей як в місцевих, так і у міжнародних масштабах. Користувачі можуть використовувати карти в автономному режимі, займаючи мінімальний обсяг пам'яті на своїх пристроях.

Проте безкоштовна версія додатку має обмеження, і для повного спектру функцій потрібно придбати платний план. Також у деяких регіонах можливі неточності у визначенні маршрутів, що може вплинути на точність навігації.

Результати аналізу і порівняння аналогів показано в табл. 1.1.

Таблиця 1.1 – Переваги та недоліки аналогів

Критерії	Google Maps	Apple Maps	ARCity	Sygic	Розроблюваний додаток
Кросплатформенність	+	-	-	+	-
Незалежність від персонального аккаунта	-	-	-	-	+
Прокладення маршрутів	+	+	+	+	+
Наявність AR технології	-	+	+	-	+

Дана таблиця містить 4 критеріїв, які характеризують дані аналоги, та створюваний у дипломній роботі мобільний додаток.

Отже, розробка мобільного навігатора з доповненою реальністю залишається актуальною у сучасному світі, оскільки міські площі постійно зростають, будуються нові міста, і потреба у знаходженні оптимальних маршрутів залишається необхідною. Програмне забезпечення має унікальні особливості, які виокремлюють його від подібних додатків, .

1.3 Аналіз методів розв’язання поставленої задачі

Створення мобільного додатку включає в себе кілька етапів, і цей процес може залежати від багатьох факторів, таких як тип додатку, платформа (iOS, Android або обидві), функціональність та інші вимоги. Було проаналізовано декілька основних підходів до розробки мобільних додатків:

1. Нативна розробка мобільних додатків включає створення програмного забезпечення спеціально для конкретної платформи, такої як iOS або Android. У цьому підході розробка ведеться мовами програмування та інструментами, які притаманні конкретній платформі. Для розробки додатків орієнтованих на платформу Android використовуються мови програмування Java – традиційна мова програмування для Android, використовується протягом багатьох років та Kotlin – сучасна мова, яка є офіційною мовою для розробки Android-додатків і рекомендується Google. Також використовується інтегроване середовище розробки Android Studio. Для розробки iOS-додатків використовують мови програмування Objective-C – традиційна мова, яка була основною для iOS-розробки до введення Swift і, відповідно, Swift – сучасна та ефективна мова програмування, розроблена Apple для створення програм для iOS яка прийшла на заміну Objective-C. Xcode є офіційним інтегрованим середовищем розробки для iOS-додатків. Містить редактор коду, інтерфейс розробки і інші корисні інструменти. Нативна розробка мобільних додатків дозволяє максимально використовувати функціональність конкретної платформи, забезпечує високу продуктивність і гарантує оптимальну роботу додатків в їхньому екосистемі.

2. Кросплатформерна розробка. Розробка кросплатформених мобільних додатків передбачає використання фреймворків та мов програмування, які дозволяють створювати додатки, що можуть працювати на різних операційних системах, таких як iOS та Android. Є чотири основні технології для розробки кросплатформерних додатків. Фреймворк React Native використовується для створення кросплатформених додатків з використанням бібліотеки React та мови програмування JavaScript (або TypeScript). Фреймворк Flutter, розроблений Google, дозволяє створювати красивий та високоефективний інтерфейс користувача. Для написання програмного коду

використовується мова програмування Dart. Фреймворк Xamarin дозволяє використовувати C# для розробки кросплатформених додатків, використовуючи різні API для доступу до функцій платформ. Фреймворк Ionic використовує веб-технології для створення кросплатформених додатків з використанням відомих фреймворків, таких як Angular або React. Для написання коду використовують HTML, CSS та JavaScript. Розробка кросплатформених додатків дозволяє зменшити зусилля та витрати на розробку, але потребує уваги до продуктивності та можливостей кожної платформи.

3. Прогресивні Веб-Додатки (PWA) - це веб-додатки, які використовують сучасні технології для створення користувацького досвіду, аналогічного тому, який можна отримати зі звичайних мобільних додатків чи десктопних програм. PWA об'єднують в собі переваги веб-сайтів та мобільних додатків, забезпечуючи високий рівень функціональності та доступність. Розробка PWA використовує веб-технології, такі як HTML, CSS та JavaScript.

Для розробки мобільного додатку для навігації з доповненою реальністю було вирішено використати методи нативної розробки мобільних застосунків, а саме мови програмування Java та Kotlin та інтегроване середовище розробки Android Studio. Також було вирішено використати можливості Google API для отримання даних необхідних для створення навігаційних маршрутів.

1.4 Постановка задач для програмного засобу - навігатора з доповненою реальністю

Після аналізу та дослідження за темою розробки програмного засобу для навігації з доповненою реальністю, було виділено задачі, які забезпечують якість та взаємодію модулів.

- визначити найбільш ефективний підхід до організації мобільного додатку;
- розробити алгоритм завантаження , опрацювання та відображення мапи та маршрутів;
- розробити графічний інтерфейс користувача для взаємодії із різними модулями мобільному додатку;
- розробити модуль для екрану з доповненою реальністю;
- здійснити тестування програмного модуля.
- здійснити економічне обґрунтування доцільності наукового дослідження

1.5 Висновки

У першому розділі МКР було розглянуто аналіз сучасного стану предметної області, визначено актуальність та важливість дослідження, адже доповнена реальність розвивається у зв’язку із розвитком індустрії мобільних телефонів. Проведено порівняльний аналіз аналогічних додатків для навігації з доповненою реальністю, охарактеризовано їх основні переваги та недоліки. Виконано постановку основних завдань на дослідження та розробку методу для програмного засобу для об’єктної доповненої реальності.

2 РОЗРОБКА АРХІТЕКТУРИ ТА АЛГОРИТМІВ ПРОГРАМНОГО ПРОДУКТУ

2.1 Загальний опис роботи систем доповненої реальності та опис методів побудови мап за допомогою систем доповненої реальності

На даний момент існує декілька видів AR систем, які використовують різні підходи до виявлення, розпізнавання та відстеження об’єктів за допомогою камери [6]. Наразі виділяють маркерну доповнену реальність, безмаркерну, локалізаційну.

Маркерна доповнена реальність (marker-based augmented reality) – це один з підходів до реалізації технології доповненої реальності, коли взаємодія між віртуальним та реальним світами базується на використанні маркерів або спеціальних об'єктів, які розпізнаються за допомогою камери пристрою або камери вбудованого пристрою (наприклад, смартфона чи планшета).

Маркери - це спеціальні зображення або об'єкти, які розпізнаються системою доповненої реальності. Це може бути QR-код, спеціальний символ чи зображення, в якому система впізнає особливості для визначення положення та орієнтації. Мобільний пристрій або камера підключеного пристрою фіксує маркери в реальному світі. Потім за допомогою алгоритмів комп'ютерного зору для визначається положення, орієнтація та розмір маркера. Після розпізнавання маркера, система розміщує віртуальний контент (графіка, анімація, текст) поверх маркера.

Користувач може взаємодіяти з віртуальним контентом, наприклад, використовуючи дотики, жести чи рухи пристрою. Маркерна доповнена реальність широко використовується в іграх, де маркери можуть служити як ключові точки для розташування віртуальних персонажів чи об'єктів. До

переваг маркерної доповненої реальності варто віднести високу точність визначення положення та орієнтації об’єктів та доволі легке впровадження, оскільки система вимагає лише розпізнавання спеціальних маркерів. Щодо недоліків – залежність від наявності маркера в області видимості та обмежений обсяг просторової взаємодії з віртуальним контентом.

Безмаркерна віртуальна реальність (markerless augmented reality) - це технологія, що дозволяє взаємодіяти з віртуальним контентом у реальному світі без використання спеціальних маркерів чи об'єктів. У цьому випадку система розпізнавання об'єктів використовує інші методи, такі як визначення положення та орієнтації пристрою, аналіз оточення, розпізнавання поверхонь і функцій реального світу.

Працює технологія за рахунок використання вбудованих сенсорів пристрою, таких як акселерометр, гіроскоп, магнітометр, для визначення положення та орієнтації, а також камери пристрою для аналізу оточуючого середовища та розпізнавання поверхонь, текстур та об'єктів. Також використовуються технології та алгоритми комп’ютерного зору для аналізу і розпізнавання об'єктів, контурів та інших характеристик зображення. Основне застосування це розробка ігор, де віртуальні об'єкти і персонажі взаємодіють з реальними об'єктами та поверхнями та візуалізація продуктів, меблів, інтер'єрів в реальному середовищі. Серед переваг безмаркерної доповненої реальності варто відзначити варіативність у виборі поверхонь для розміщення віртуального контенту, відсутність потреби у спеціальних маркерах або об'єктах та здатність взаємодіяти з реальним середовищем без додаткових обмежень.

Проте безмаркерна доповнена реальність зазвичай надає меншу точність визначення положення та орієнтації порівняно з маркерною технологією, а

також вимагає більше апаратних ресурсів пристрою для обробки великою кількості інформації з сенсорів та камери.

Локалізаційна доповнена реальність (location-based augmented reality, AR) - це технологія доповненої реальності, яка використовує геолокацію пристрою для розташування та інтеграції віртуального контенту в реальному часі на підставі фізичної локалізації об'єктів чи місць у реальному світі. Такий підхід дозволяє користувачам отримувати інформацію, взаємодіяти з віртуальними об'єктами та переживати доповнений контент, пов'язаний з конкретними місцями або об'єктами в їхньому навколишньому середовищі.

Для визначення місцезнаходження пристрою використовуються технології глобального позиціонування (GPS), супутникових систем та датчиків руху [7]. Для пошуку об’єктів проводиться аналіз навколишнього простору для визначення розташування фізичних об'єктів, які можуть бути підставою для віртуального контенту. Віртуальний контент, такий як текст, зображення, 3D-моделі чи інші елементи, розташовується в реальному просторі на підставі геолокаційних даних. Основні сфери використання локалізаційної доповненою реальності це допомога в навігації користувача в реальному часі, надаючи напрямки, вказівки та іншу інформацію, пов'язану з його місцезнаходженням, створення віртуальних екскурсій або турів, де користувачі можуть дізнатися більше про реальні місця, переглядати інтересні факти чи події.

Також ключову роль в побудові систем навігації в доповненій реальності відіграють алгоритми та технології позиціонування користувача. Основними технологіями у цьому напрямку є SLAM (Simultaneous Localization and Mapping) та VPS (Visual Positioning System) [8].

Технологія SLAM (Simultaneous Localization and Mapping), що українською перекладається як «одночасне визначення положення та побудова

карти», є ключовою для багатьох застосувань в галузі розширеної реальності (AR), робототехніки та комп'ютерного зору. SLAM вирішує завдання визначення положення об'єкта в просторі та побудови карти його оточення одночасно (рис. 2.1). Алгоритм SLAM складається з 3 компонентів: визначення положення (localization), побудова карти (mapping), та оптимізація і синхронізація. Для визначення точного об’єкта в просторі збирається інформація з сенсорів, таких як глобальні системи позиціонування (GPS), акселерометри, гіроскопи та камери. Побудова карти відбувається теж за допомогою сенсорів, а також включає розпізнавання рельєфу, об’єктів, текстур тощо. Насамкінець за допомогою алгоритмів оптимізації, SLAM коригує та покращує інформацію про положення та карту, враховуючи помилки та невідомі фактори.

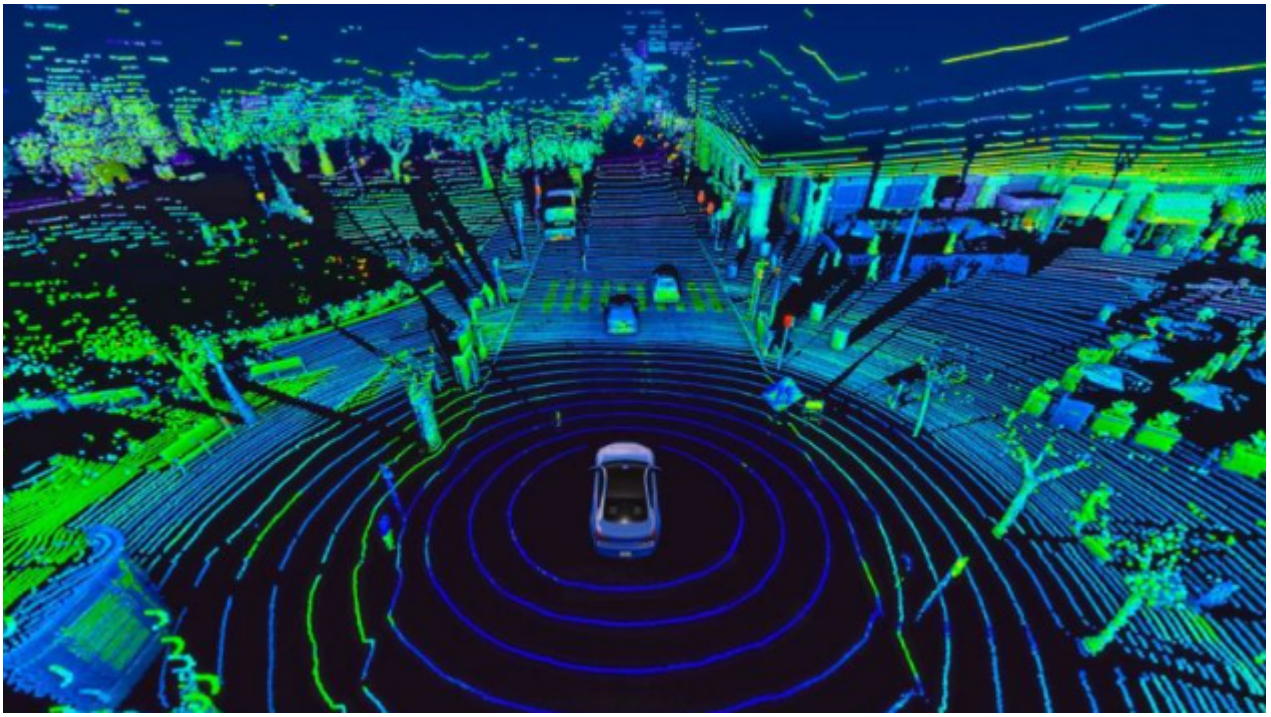

Рисунок 2.1 – Візуальне представлення карти оточення побудоване з використанням технології SLAM

SLAM розділяється на 4 етапи роботи: ініціалізація, пересування, оптимізація та корекція та використання карт. Під час ініціалізації визначається початкове положення та побудова перших елементів карти на основі даних від сенсорів. Потім коли об’єкт рухається в просторі SLAM визначає його нове положення та оновлює карту, використовуючи дані від сенсорів, під час чого також застосовуються алгоритми оптимізації для корекції помилок та вирішення неоднозначностей в даних про положення та карту. І, насамкінець, отримана карта використовується для навігації об'єкта в просторі та взаємодії з навколишнім середовищем [9].

Технологія SLAM застосовується для автономних роботів для навігації та визначення їхнього положення в невідомому оточенні, в AR-додатках використовується для відображення віртуальних об'єктів в реальному часі та прив'язки їх до конкретних місць, автономних автомобілях SLAM важливий для точної навігації та взаємодії з оточенням, а також в картографуванні для побудови карт об'єктів, місцевостей або споруд для подальшого використання [10].

Щодо технології VPS (Visual Positioning System) – вона використовується для точного визначення місцезнаходження за допомогою комп’ютерного зору. Вона працює шляхом порівняння зображень, зроблених камерою пристрою, з базою даних попередньо опрацьованих зображень і використання методів машинного навчання для оцінки положення й орієнтації пристрою на основі подібності та відмінностей між зробленими зображеннями та попередньо опрацьованими зображеннями [11]. VPS можна застосовувати разом з GPS, що значно підвищить точність локалізації пристрою (рис 2.2).

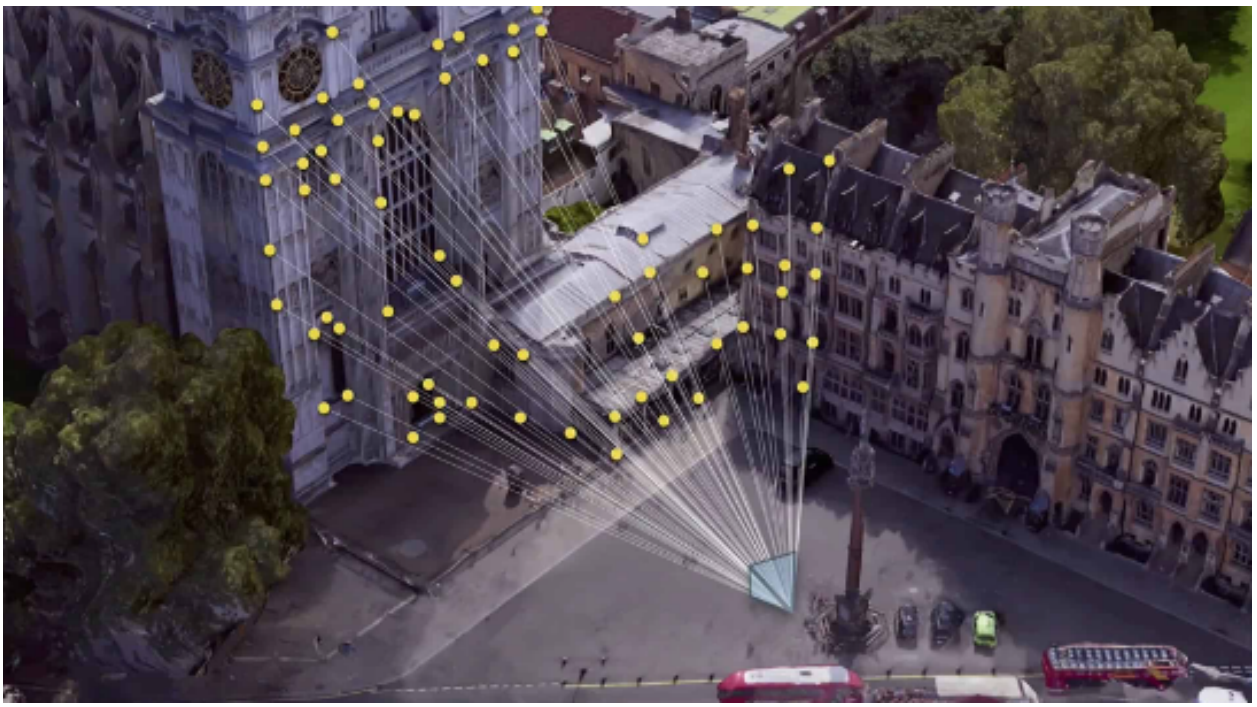

Рисунок 2.2 –Ілюстрація процесу локалізації пристрою за допомогою Visual Positioning System

Для обробки інформації з камери пристрою та побудови карти використовуються фільтр Калмана – для обробки даних з та корегування їх зображень з камери для отримання більш точних вимірів положення, розпізнавання образів (Image Recognition) для ідентифікації і розпізнавання конкретних зображень, оптичне потокове відстеження (Optical Flow Tracking) – для спостереження за рухом пікселів на зображенні для визначення швидкості та напрямку руху, методи глибинної карти (Depth Map Methods) – для визначення відстані до об’єктів.

2.2 Удосконалення методу навігації з доповненою реальністю

На даний час звичний метод навігації це відображення поточного місцезнаходження користувача, отриманого за допомогою GPS і рисування запропонованого маршруту до пункту призначення, отриманого за допомогою

доступу до API, на двовимірному зображенні карти на екрані пристрою. Дана методика використовується у більшості сучасних програм для карт та навігації. Однак цей метод вимагає від користувачів вміння читати карти, але зовсім не кожен користувач здатний зрозуміти ці позначення.

Для вирішення цієї проблеми запропоновано розробку навігаційної системи з доповненою реальністю. Найпростіший метод для імплементації системи є використання GPS та магнітометра пристрою (електронний компас) для розстановки навігаційних покажчиків у AR просторі. Проте, зважаючи на те що точність самої системи GPS не є ідеальною, а відповідно і навігаційні покажчики можуть відображатись у хибному напрямку (рис. 2.3). На ілюстрації зліва північ реальна та північ всередині AR простору збігаються, відповідно навігаційні вказівники відображаються на правильних ділянках зображення з камери, на ілюстрації справа північ в AR просторі не збігається з північчю реальною, тож ми можемо бачити що маркер відображений у хибному місці, і чим більший азимут і відстань до маркера, тим більш хибно він буде розташований в AR просторі. Тож при розробці таких навігаційних систем покладаються не лише на інформацію з GPS, а також використовують метод визначення поточної локації за допомогою маркерів та методи підтвердження орієнтирів на перетинах.

Окрім цього також впроваджують VPS з налаштуваннями машинного навчання для порівняння зображень з камери користувача з наявною базою зображень та визначення точного місцезнаходження та побудову подальшого маршруту для користувача. Використання декількох систем локалізації та позиціонування забезпечують надійну та точну побудову навігаційних вказівників.

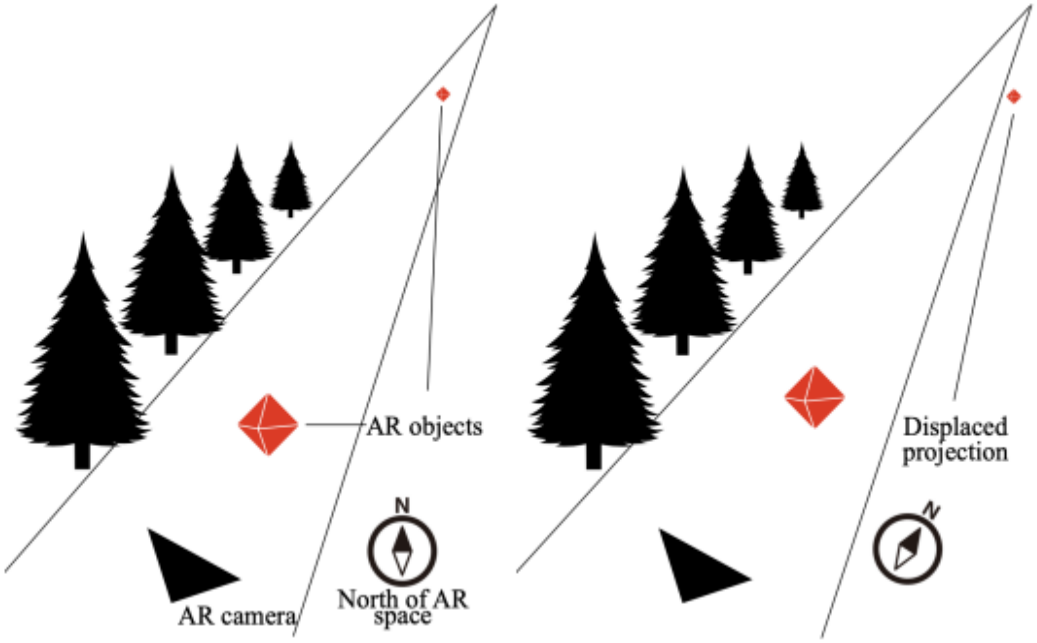

Рисунок 2.3 – Нанесення навігаційних вказівників у випадку з точним та зміщеним положенням півночі в AR просторі

Тож для покращення точності навігації в AR просторі необхідно підвищити якість визначення поточного місцезнаходження користувача. Окрім звичайного отримання локації через GPS-сервіс смартфона, можливо також задіяти інші датчики доступні у мобільних пристроях, такі як акселерометр, гіроскоп та магнітометр.

Акселерометр - це датчик, призначений для вимірювання прискорення. В мобільних телефонах акселерометр використовується для визначення змін у швидкості та орієнтації пристрою. Акселерометр влаштований за допомогою електромеханічних систем (MEMS). У MEMS-акселерометрах використовуються шкали, пружинки та конденсатори для вимірювання прискорення. Коли телефон рухається, маска або конструкція в середині акселерометра рухається відносно фіксованого каркасу, змінюючи ємність конденсаторів. Зміни ємності використовуються для визначення величини і

напрямку прискорення.

Гіроскоп у мобільних телефонах - це датчик, який вимірює або відстежує орієнтацію телефону та зміни його положення у просторі. Гіроскоп допомагає визначити кутову швидкість або обертання пристрою навколо трьох осей: вздовж, поперек і вертикально. У більшості мобільних телефонів гіроскоп використовується разом із звичайним акселерометром для отримання більш точної інформації про рух телефона в просторі. Разом ці два датчики надають телефону змогу визначати своє положення, орієнтацію і рух у тривимірному просторі. Гіроскопи також використовуються для покращення реалістичності ігральних додатків, віртуальної та доповненої реальності.

Магнітометр в мобільному пристрої - це датчик, призначений для вимірювання магнітного поля в околицях пристрою. Такий датчик може використовуватися для визначення напрямку магнітного північного полюса та, відповідно, для визначення орієнтації пристрою в просторі відносно магнітного північного напрямку. Магнітометри використовуються разом з іншими сенсорами, такими як акселерометр та гіроскоп, для отримання повної інформації про орієнтацію і рух пристрою в тривимірному просторі. Коли ці три типи датчиків комбінуються, вони надають можливість визначати не тільки положення, але і напрямок пристрою в просторі.

Проте як і будь який інший датчик, магнітометр, гіроскоп та акселерометр допускають «шум» в показаннях. "Шум" в показаннях датчиків вказує на випадкові або непередбачувані коливання вимірювань, які можуть виникати через різні фактори, такі як електричні перешкоди, теплові коливання, вібрації, електромагнітні впливи тощо. Шум може бути присутнім в сигналі датчика і впливати на точність вимірювань. В контексті сенсорів, таких як акселерометри, гіроскопи і магнітометри в мобільних телефонах, шум може впливати на якість інформації, яку ці сенсори надають. Наприклад, високий

рівень шуму може призвести до неточностей у визначенні орієнтації, руху чи інших параметрів пристрою.

Тож за рахунок цих трьох датчиків ми можемо однозначно вирахувати положення пристрою в просторі. Його можна виразити за допомогою матриці повороту. Матриця повороту — матриця переходу, яка зв'язує між собою координати векторів векторного простору при зміні системи координат.

В новій системі координат вектор x переходить у вектор x'. Між новими та старими координатами існує лінійний зв'язок

$$x' = R * x$$

Цей зв'язок визначається матрицею повороту R.

Також ми можемо отримати вектор прискорення з показань акселерометра.

Для того, щоб отримати прискорення в абсолютній системі координат (пов'язаної із Землею) необхідно помножити інвертовану матрицю повороту на вектор отриманого прискорення. Допустимо, у нас вже є матриця повороту R, яку ми отримали, скориставшись одним з алгоритмів AHRS (Attitude and heading reference system). В результаті вийде вектор із трьох елементів: прискорення на схід, на північ і вгору.

Є багато методів для визначення положення пристрою у просторі на основі даних від акселерометра, магнітометра та гіроскопа. З них усіх було вибрано 2 найкращих — віртуальний сенсор ROTATION_VECTOR та фільтр Маджвіка.

Фільтр Маджвіка - це алгоритм орієнтації, який використовується для визначення орієнтації об'єкта в просторі за допомогою вимірювань з акселерометра, гіроскопа та магнітометра. Цей алгоритм був розроблений Себастьяном Маджвіком і спеціально призначений для використання в невеликих пристроях, таких як мобільні телефони та інші пристрої з

обмеженим обсягом ресурсів.

Основна ідея за фільтром Маджвіка полягає в тому, щоб з комбінації даних акселерометра, гіроскопа та магнітометра отримати достовірне значення орієнтації об'єкта. Алгоритм враховує можливість помилок та шуму у вимірюваннях і намагається виправити ці неточності. Фільтр Маджвіка є популярним в галузі віртуальної реальності (VR), доповненої реальності (AR), навігації [12].

Тепер отримавши вектор прискорення в абсолютній системі координат можна приступити до фільтрації «шумів» в показаннях місцеположення. Для цього доцільно використати фільтр Калмана - ефективний рекурсивний фільтр, що оцінює вектор стану динамічної системи, використовуючи ряд неповних та зашумлених вимірів. рекурсивний математичний алгоритм, який використовується для обробки шумних вимірювань і намагається виділити справжні значення стану системи. Цей фільтр часто застосовується у сферах керування, обробки сигналів, навігації, фільтрації даних та інших областях, де важлива точність вимірювань.

Основна ідея фільтра Калмана полягає в тому, щоб комбінувати інформацію з двох джерел: прогноз системного стану на основі попередніх вимірювань та оновлення стану на основі нових вимірювань [13]. Фільтр враховує дисперсію (розподіл помилок) вимірювань і прогнозує оптимальне значення стану системи.

Головне завдання – визначити вектор стану системи, матрицю переходу, керуючий вектор та інші компоненти фільтра Калмана. Одне з найскладніших завдань - визначити матриці коваріаційні шуму процесу і шуму вимірювань. Матриці в поточному рішенні ґрунтуються на законі рівномірного руху, тільки в матричній формі (оскільки ми маємо два напрямки — схід і північ).

У такій формі досить легко розширити фільтр, щоб він враховував

прискорення вгору, але поки в цьому немає необхідності. А можна розбити фільтр на 2 або 3 для кожного напрямку. Майже нічого не зміниться, тільки розміри матриць зменшаться.

До параметрів фільтра відносяться:

- вектор стану системи:

$$X_k = [X\ Y\ X'\ Y']$$

де X і Y – координати, а X' і Y' - швидкості

- матриця еволюції системи:

$$F_k = [1.0\ 0.0\ dt\ 0.0\ 0.0\ 1.0\ 0.0\ 0.0\ dt\ 0.0\ 0.0\ 0.0\ 1.0\ 0.0\ 0.0\ 0.0\ 0.0\ 0.0\ 0.0\ 1.0]$$

- матриця управління:

$$B_k = \left[dt^2/2\ 0.0\ 0.0\ dt^2/2\ dt\ 0.0\ 0.0\ dt\right]$$

- вектор управління:

$$u_k = [X''\ Y'']$$

- матриця змін:

$$H_k = I$$

В досліджуваному випадку матриця змін дорівнює одиничній матриці, так як нам не потрібно проводити жодних конвертацій.

- коваріаційна матриця шуму процеса:

$$Q_k = \left[\sigma^2_{posX}\ 0.0\ \sigma_{posX} * \sigma_{velX}\ 0.0\ 0.0\ \sigma^2_{posY}\ 0.0\ \sigma_{posX} * \sigma_{velX}\ 0.0\ 0.0\ \sigma^2_{velX}\ 0.0\ 0.0\ 0.0\ 0.0\ 0.0\ 0.0\right.$$

де

$$\sigma_{vel} = \sigma_{acc} * dt$$

$$\sigma_{pos} = \frac{\sigma_{acc} * dt^2}{2}$$

Тепер, маючи параметри фільтра, необхідно привести вхідні данні до єдиної системи координат. Для цього було використано алгоритм геохешування (GeoHash) – алгоритм для конвертації 2 координат виду 12.345678, 98.765432 (довгота, широта) в 1 строку.

Умовно карта світу розбита на секції, чим більше символів використовується для позначення секції тим менший розмір цієї секції (рис. 2.4).

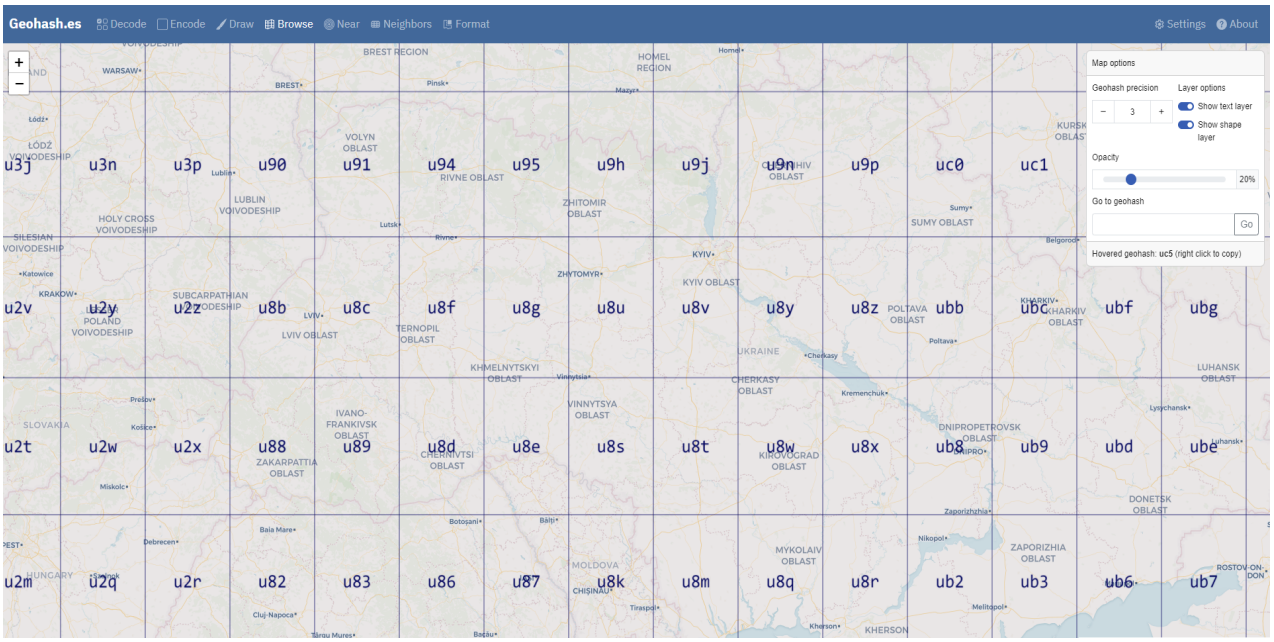

Рисунок 2.4 – Мапа світу розбита на секції відповідно до алгоритму GeoHash

У розроблюваному додатку даний алгоритм використовуватиметься для рішення двох задач: об’єднання точок які знаходяться в заданому радіусі, та виявлення і фільтрація недостовірних показань GPS. Метод геохешування видає хеш у виді строки, довжина якої задається користувачем, і чим більша довжина строки, тим меншу область вона описує і тим більша точність координат досягається. Максимально можлива довжина хешованої строки 12 символів. Для нашого випадку достатню точність видає строка довжиною 8 символів.

Використавши описані методи та алгоритми було досягнуто фільтрацію випадкових невірних показань з датчика GPS. Для демонстрації роботи алгоритмів було розроблено демо-проект який ілюструє роботу фільтру

Калмана (рис. 2.5). У даному проекті виконується запис локації пристрою і його відображення на карті. Для відображення карти було використано набір програмних засобів Google Maps. У додатку сигнали з датчику GPS позначені зеленим кольором, та будується лінія відповідно до цих показань. Фільтровані значення відображаються рожевим кольором. Для більш наочного прикладу було використано можливості debug режиму Android Studio та віртуальний емулятор Android-пристрою, для того щоб була змога задати показання з GPS програмно.

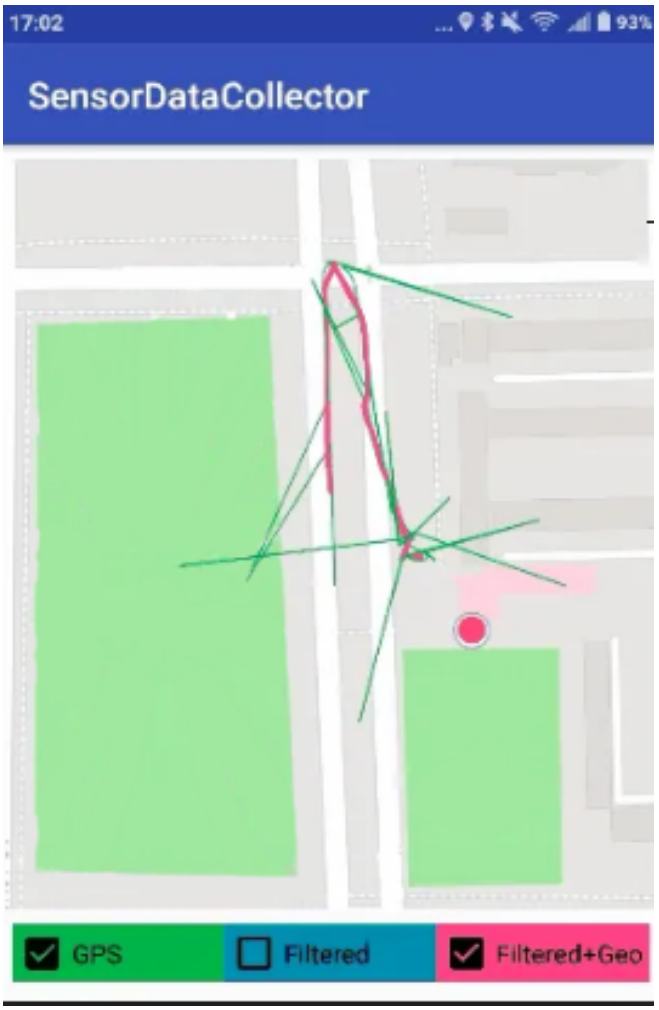

Рисунок 2.5 – Демонстрація роботи фільтру Калмана

За рахунок інтеграції описаних алгоритмів та фільтрів досягається згладження маршруту користувача, та фільтрація недостовірних «скачків» в показаннях датчика GPS. В подальшому це впливає на точність роботи додатку у режимі доповненої реальності.

2.3 Розробка структурних схем та інтерфейсу додатку

Для опису функціональних можливостей додатку було вирішено розробити UML діаграми [14]. Unified Modeling Language (UML) є стандартом для моделювання та документування програмних систем. UML надає графічні засоби для визначення структури і поведінки системи.

У UML існують різні види діаграм, які дозволяють моделювати різні аспекти системи. Наприклад, діаграми класів відображають структуру системи, а діаграми взаємодії показують взаємодію між об'єктами чи класами. Діаграми діяльності використовуються для моделювання процесів, тоді як діаграми станів описують стани об'єктів та їх переходи. У UML використовуються поняття класів, об'єктів, асоціацій та інших елементів для визначення структури програми. Відносини між класами можуть включати асоціації, агрегації та спадкування. Діаграми варіантів використання в UML допомагають моделювати взаємодію системи з її користувачами чи іншими системами. Загальною метою UML є спрощення розробки програмних систем, полегшення розуміння їх архітектури та забезпечення зручного інструменту для комунікації між учасниками проекту.

Спочатку було вирішено створити діаграму прецедентів для розроблюваного додатку. Діаграма прецедентів (Use Case Diagram) в Unified Modeling Language (UML) використовується для моделювання взаємодії між системою та її зовнішніми елементами, такими як користувачі чи інші

системи. Ця діаграма відображає функціональність системи та показує, як різні актори взаємодіють з системою через її прецеденти (рис. 2.6). Так як мобільний додаток є користувацьким програмним забезпечення, у додатку передбачений один актор – Користувач.

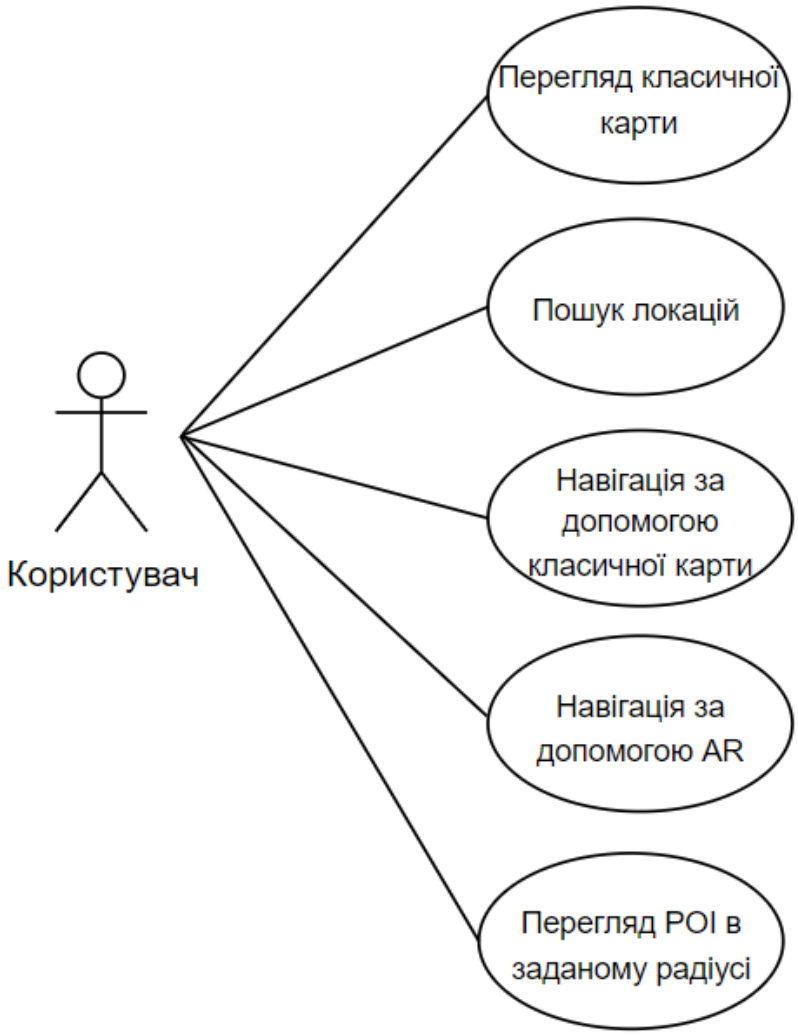

Рисунок 2.6 – Діаграма прецедентів мобільного додатку для навігації в доповненій реальності

Проаналізувавши функціональні можливості системи було виділено прецеденти, які продемонстровано на таблиці 2.1.

Таблиця 2.1 – Характеристика прецедентів

Прецедент	Короткий опис
Перегляд класичної карти	Завантаження карти місцевості на екран для подальшої взаємодії кристувача з картою. Користувач має змогу шукати локації та в подальшому використовувати карту для наступних дій.
Пошук локацій	Користувач може взаємодіяти з набором UI інстурментів для пошуку локацій. Для цього він може скористатись SearchView, вписавши в строку пошуку потрібний йому запит.
Навігація за допомогою класичної карти	Користувач створює маршрут між потрібними локаціями. Маршрут будується поверх карти, вказуючи напрямок руху. Для цього користувач повинен обрати дві локації та обрати пункт 2D навігації у меню. Після цього буде створено маршрут між обраними точками.
Навігація за допомогою AR	Користувач може обрати функцію навігації за допомогою системи доповненої реальності. Для цього йому потрібно обрати локації між якими здійснюватиметься навігація та обрати пункт меню «AR navigation». Користувач буде переключений на AR view на якому буде зображення з камери телефона та побудований маршрут поверх цього зображення.
Перегляд POI в заданому радіусі	Користувач переключається на AR view, на якому зображено локації доступні навколо користувача. Користувач може змінювати радіус пошуку локації.

Для прецеденту «Навігація за допомогою AR» було створено діаграму послідовності (рис. 2.7). Діаграма послідовності в UML слугує для візуалізації взаємодії між об'єктами у системі. У цій діаграмі об'єкти представлені в верхній частині, а їхні життєві цикли відображаються уздовж вертикальних ліній. Повідомлення, що передаються між об'єктами, показуються стрілками або лініями. Основні елементи включають акторів, які представляють ролі чи системи, що взаємодіють з системою. Також присутні лінії життєвого циклу, які показують, як триває життєвий цикл об'єкта в часі. Повідомлення між об'єктами показуються для визначення порядку взаємодії. Діаграма послідовності допомагає розуміти порядок та взаємодію об'єктів у конкретному сценарії використання, полегшуючи аналіз та визначення динаміки системи.

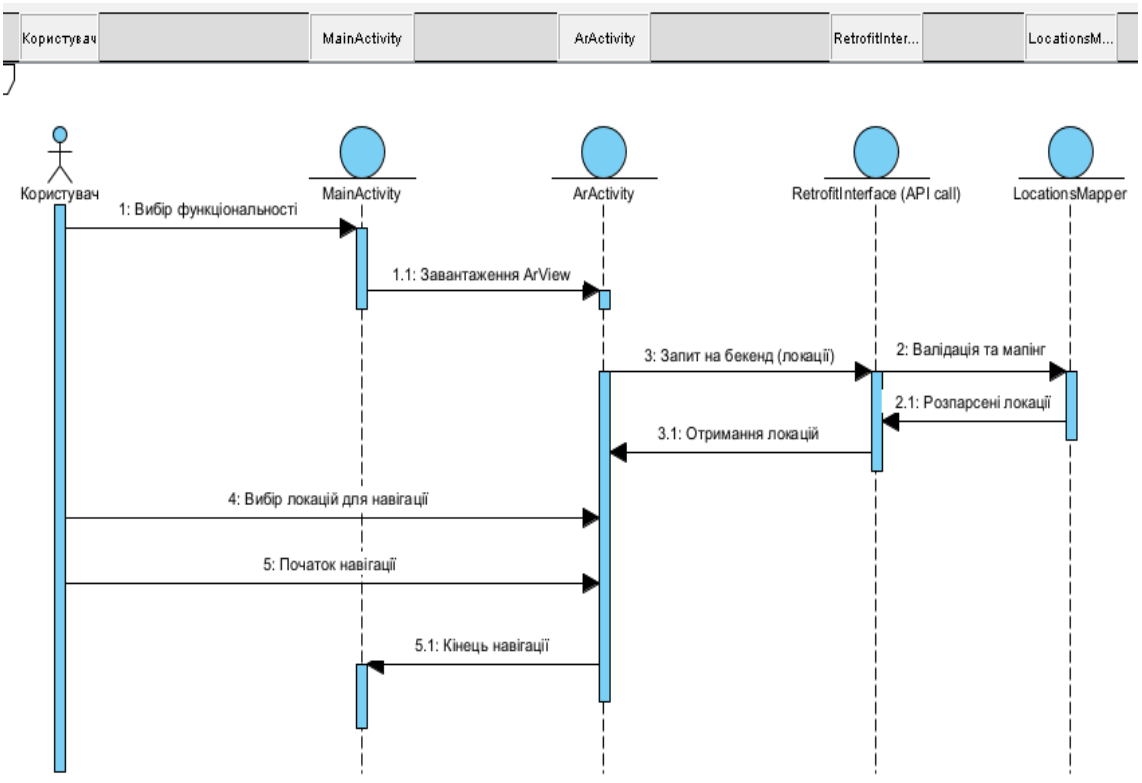

Рисунок 2.5 - Діаграма послідовності для прецеденту «Навігація за

допомогою AR»

Також для прецеденту «Навігація за допомогою AR» було створено діаграму діяльності (рис. 2.6). Діаграма діяльності в мові моделювання UML використовується для візуалізації потоку робіт в системі. Ця діаграма дозволяє моделювати послідовність дій та обмін даними між різними елементами системи. Вона широко використовується для аналізу бізнес-процесів та опису виконання алгоритмів. На діаграмі можна відобразити дії, які виконуються в системі, рішення, де потік може розділитися, вибори, що вказують на розділення потоку на різні напрямки, та об'єднання, де гілки об'єднуються. Також можна показати початок та завершення діаграми, а також об'єкти, які беруть участь у виконанні дій. Діаграми діяльності допомагають в розумінні та моделюванні потоків дій в системі, що може бути корисним на різних етапах розробки, від аналізу бізнес-процесів до проектування алгоритмів.

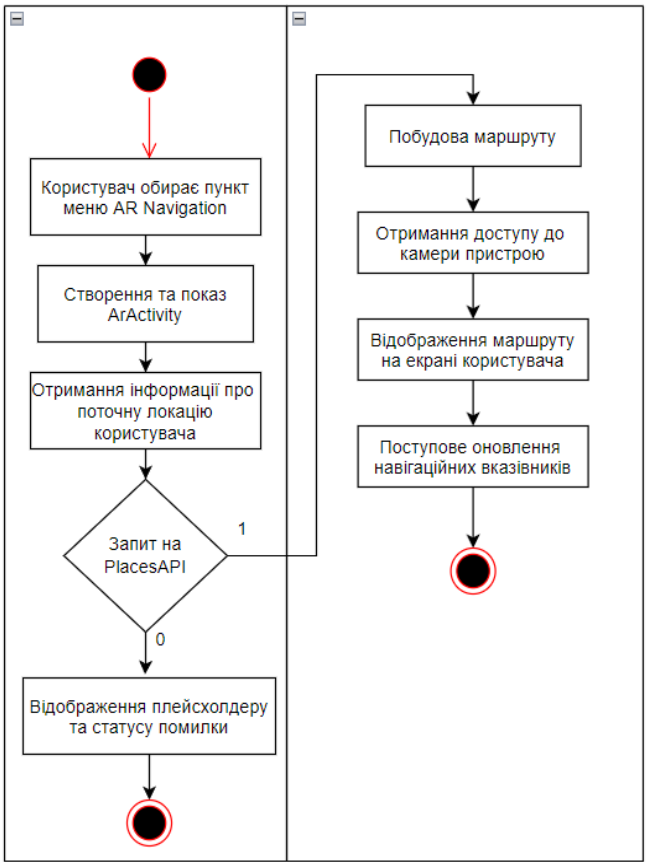

Рисунок 2.6 - Діаграма діяльності для потоку подій прецеденту «Навігація з доповненою реальністю»

Під час розробки мобільних додатків ключову роль відіграє графічний інтерфейс так як це те з чим безпосередньо взаємодіє користувач. В цілому концепція дизайнів мобільних додатків рухається в бік спрощення та полегшення екранів. Приділяється велика увага для того щоб створюваний інтерфейс був інтуітивно зрозумілий користувачу, не був перевантажений безліччю кнопок, та функцій і при цьому містив всі необхідні елементи керування.

Так як основна функція додатку це навігація за допомогою AR технології, було розроблено дизайн систему, яка описує основні елементи інтерфейсу додатку, та концепцію наповнення екранів.

Для розробки інтерфейсу користувача було вирішено використати бібліотеку компонентів MaterialDesign. Material Design - це дизайн система,

розроблена компанією Google, яка визначає стандарти дизайну для створення інтерфейсів користувача. Вона була представлена в 2014 році і призначена для забезпечення єдиної та зрозумілої естетики для веб-сайтів, мобільних додатків та інших платформ. Опис можливостей Material Design та методи імплементації рекомендованих компонентів описаний на офіційному сайті системи (рис. 2.7).

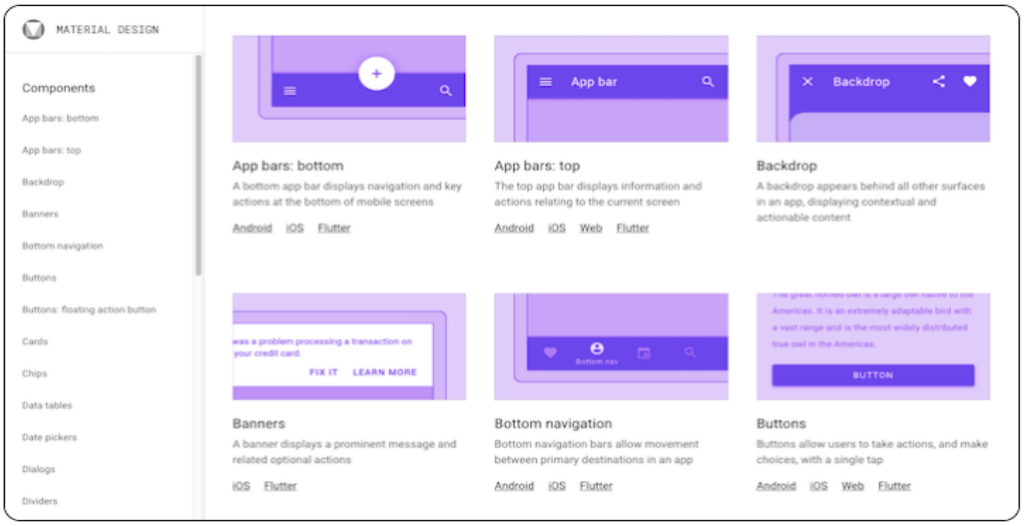

Рисунок 2.7 – Офіційна документація системи Material Design

Основні характеристики Material Design включають:

1. Матеріальна структура: В основі Material Design лежить концепція "матеріалу", що надає реалістичний вигляд і відчуття елементам інтерфейсу, натхненного фізичними об'єктами.

2. Карточки та тінь: Використання карточок (cards) для групування інформації та додавання тіней для створення вигляду шарів та глибини.

3. Анімація: Застосування плавних та динамічних анімацій для покращення взаємодії та реакції інтерфейсу на дії користувача.

4. Яскраві кольори: Використання насичених та яскравих кольорів для створення виразних та привабливих інтерфейсів.

5. Типографія: Використання чистих та чітких шрифтів для поліпшення читабельності.

6. Адаптивність: Розробка дизайну, який легко адаптується до різних розмірів екранів та пристроїв.

Material Design надає розробникам та дизайнерам зручні інструменти та рекомендації для створення сучасних та привабливих інтерфейсів, що допомагає створювати консистентні та ефективні веб- та мобільні додатки.

Виходячи з цього було розроблено структурні схеми екранів додатку, які думонструють розміщення елементів керування, та які описуватимуть workflow додатку. Перш за все було виконано дизайн головного екрану додатку який є точкою входу в застосунок (рис. 2.8).

На всій площині екрану розміщуватиметься класична карта місцевості яка показуватиме поточну локацію користувача. Користувач матиме змогу масштабувати та переміщати карту відповідно до своїх цілей. У верхній частині екрану розміщуватиметься елемент для введення запиту користувача. Відповідно після введення запиту (локації) і натискання на кнопку пошуку карта буде переміщена для демонстрації шуканої локації.

Рисунок 2.8 – Початковий екран додатку

В нижньому правому куті розміщується меню, при натисканні на яке на екрані з’являться додаткові елементи меню, які використовуватимуться для навігації користувача між функціями додатку: навігація за допомогою AR, пошук Points of interest та інформація про додаток. При натисканні на елемент меню «Навігація за допомогою AR» користувач буде перенаправлений на допоміжний екран для введення інформації для навігації (рис. 2.9).

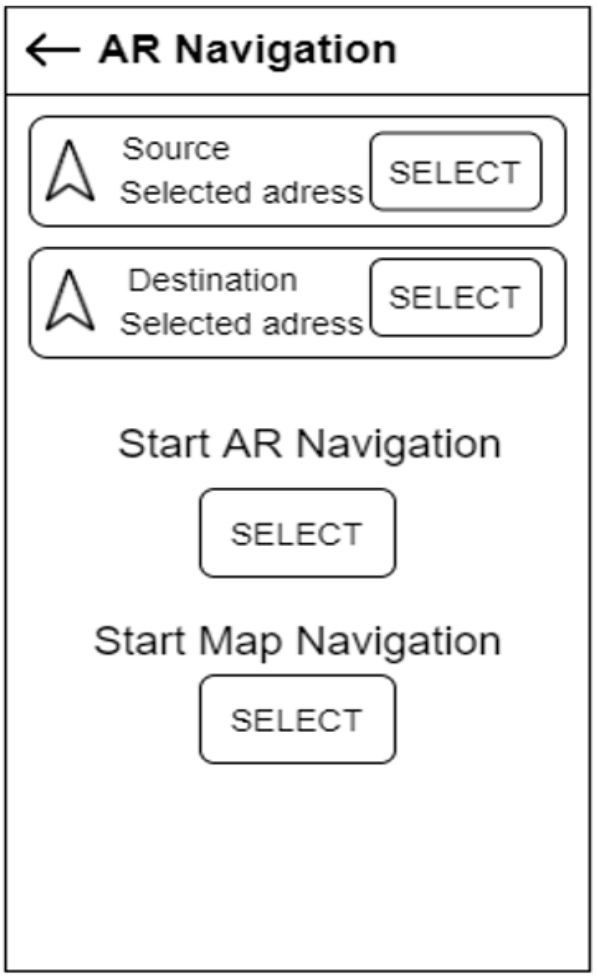

Рисунок 2.9 – Екран для введення інформації для навігації

В верхній частині екрану розміщений елемент інтерфейсу який містить назву обраного екрану та кнопку для навігації на попередній екран. Під ним розміщені дві підменю для вибору початкової локації та місцепризначення. При натисканні на кожну з них відкриється екран пошуку локації (рис. 2.10). У верхній частині екрані розміщена строка пошуку локації. Більшу частину екрану займає карта з розміщеними на ній локаціями та локацією користувача, з якою може взаємодіяти користувач для вибору локації. Під картою розміщений список знайдених локація які користувач може обрати.

Рисунок 2.10 – Екран вибору локацій

Після вибору початкової та кінцевої локації навігації і вибору пункту меню «Почати AR навігації» користувач буде перенаправлений безпосередньо на екран доповненої реальності з зображеним маршрутом з навігаційних вказівників від поточної локації до кінцевої (рис. 2.11).

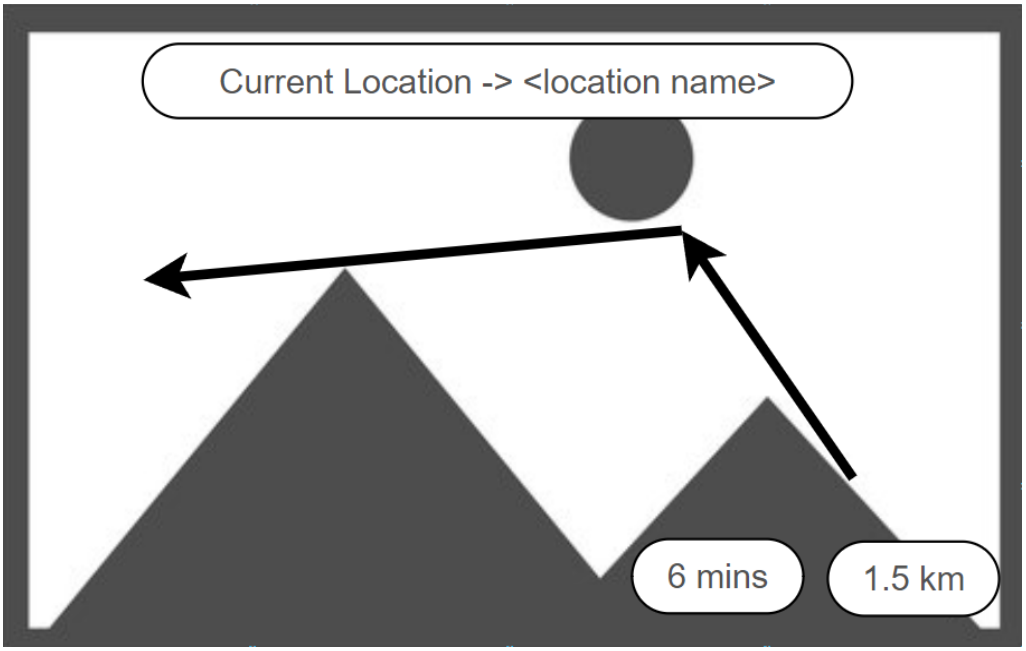

Рисунок 2.11 – Екран доповненої реальності з відображеним маршрутом між локаціями

У верхній частині екрану зображений елемент який вказує на обраний маршрут, тобто початкову локацію та кінцеву локацію. У правому нижньому куті розміщені два елемента які вказують на дистанцію маршруту та приблизно розрахований час на виконання маршруту. Данні в цих елементах оновлюватимуться відповідно до переміщення користувача.

2.4 Висновки

У другому розділі роботи було виконано загальний опис методу навігації з доповненою реальністю.

Удосконалено метод навігації з доповненою реальністю шляхом імплементації фільтру Калмана для фільтрації хибних показань з датчика GPS, забезпечивши достовірні дані для побудови маршрутів в доповненій реальності. Також використано алгоритм GeoHash для спрощення опису

координат, що додало швидкодії в обрахунках та вплинуло на загальну роботу додатку. Було досліджено технологію SLAM та запозичено використання фільтру Маджвіка для підвищення точності позиціонування. Було розглянуто особливості сучасних мобільних інтерфейсів, розроблено дизайн та структуру додатку.

3 РОЗРОБКА МЕТОДУ ТА ПРОГРАМНИХ ЗАСОБІВ ДЛЯ НАВІГАЦІЇ З ДОПОВНЕНОЮ РЕАЛЬНІСТЮ

3.1 Варіантний аналіз та обґрунтування вибору засобів реалізації програми

Існує кілька фреймворків та платформ для розробки мобільних додатків з доповненою реальністю (AR) [15]. Ось деякі з них:

1. ARKit – розроблений Apple для платформи iOS, ARKit надає потужні інструменти для розробки мобільних додатків з AR для iPhone та iPad [16].
2. ARCore – створений Google для платформи Android, ARCore надає розширений функціонал AR для Android-пристроїв.
3. Vuforia – це популярний фреймворк для розробки мобільних додатків з AR. Він підтримує як iOS, так і Android, і забезпечує інструменти для розпізнавання об'єктів, розміщення об'єктів у просторі та інші можливості.
4. Unity (з модулем AR Foundation) – це популярний мульти-платформений двигун для створення ігор та додатків. Модуль AR Foundation дозволяє розробникам створювати AR-додатки, які можуть працювати як на ARKit (iOS), так і на ARCore (Android).
5. Wikitude – це фреймворк для розробки AR-додатків, який підтримує як iOS, так і Android. Він надає різноманітні можливості, такі як розпізнавання об'єктів, відстеження маркерів, розміщення об'єктів у реальному світі тощо.
6. AR.js – це JavaScript-бібліотека, яка дозволяє розробляти AR-додатки прямо в браузері. AR.js базується на фреймворках Three.js та A-Frame.

7. BeyondAR – бібліотека для роботи з доповненою реальністю на пристроях з операційно системою Android написана мовою Java.

Обираючи фреймворк, слід враховувати особливості платформи, для якої ви розробляєте додаток, а також потреби вашого проекту та ваші власні навички та вподобання у роботі з конкретними інструментами.

Порівняно з більш сучасними фреймворками підходящими для інтеграції доповненої реальності в Android додатки, BeyondAR має значно вужчий функціонал, проте було вирішено використовувати саме його зважаючи на найменший розмір aar файлу та можливість повної конфігурації бібліотеки під свої потреби.

Для розробки додатку було вирішено використати інтегроване середовище розробки Android Studio [17]. Android Studio - це офіційне інтегроване середовище розробки (IDE) для платформи Android, яке використовується для створення мобільних додатків під управлінням Android (рис. 3.1).

Рисунок 3.1 – Логотип IDE Android Studio

Переваги Android Studio:

1. Офіційне інтегроване середовище – Android Studio є офіційним інструментом розробки від Google для Android, що забезпечує підтримку

нових функцій та оновлення.

2. Багатофункціональність – надає повний спектр інструментів для розробки, включаючи редактор коду, візуальні редактори макетів, інструменти для відлагодження та профілювання.

3. Підтримка Kotlin – однією з великих переваг є повна підтримка мови програмування Kotlin, яка стала офіційною мовою для розробки Android-додатків.

4. Вбудована система збірки та залежностей – використовує систему збірки Gradle для ефективного управління залежностями та побудови проектів.

5. Емулятор Android пристроїв – Android Studio постачається з швидким та функціональним емулятором Android для тестування додатків безпосередньо на комп'ютері.

6. Розширені інструменти відлагодження та профілювання – надає потужні інструменти для відлагодження коду та аналізу продуктивності додатків.

7. Оновлення та підтримка – постійно оновлюється з новими можливостями та підтримкою останніх версій Android.

Щодо недоліків середовища варто виділити великі вимоги до ресурсів системи особливо при розробці масштабних проектів, а також доволі складний інтерфейс для починаючих розробників.

Проте незважаючи на недоліки, Android Studio залишається ключовим інструментом для багатьох розробників Android завдяки своїм багатим функціоналом та офіційній підтримці від Google.

Під час розробки додатку використовуватимуться дві мови програмування – Java та Kotlin [18]. Java - це об'єктно-орієнтована мова програмування, створена компанією Sun Microsystems у 1995 році (рис. 3.2).

Рисунок 3.2 – Логотип мови програмування Java

Основні риси Java включають об'єктно-орієнтований підхід, платформонезалежність через використання віртуальної машини (JVM), безпеку, підтримку багатопоточності та широкий вибір бібліотек та фреймворків. Java базується на об'єктно-орієнтованому програмуванні, що сприяє створенню модульних та легких для розуміння програм. Все в Java є об'єктом, що полегшує реорганізацію коду та підтримку. Однією з ключових особливостей Java є платформонезалежність. Код, написаний на Java, компілюється в байт-код, який виконується на віртуальній машині (JVM). Це дозволяє використовувати Java на різних операційних системах без змін у вихідному коді. Вбудована підтримка багатопоточності в Java дозволяє створювати програми, які можуть виконувати багато завдань одночасно, що дуже важливо для сучасних додатків, особливо в контексті веб-розробки та мережевих застосувань. Java володіє великою кількістю бібліотек та фреймворків, таких як Spring, Hibernate, та Apache Struts. Це полегшує розробку та розширення функціоналу програм. Java використовується для розробки різноманітних додатків, від мобільних застосунків для Android до корпоративних інформаційних систем та веб-додатків. Її висока швидкість та надійність робить її популярною серед розробників для широкого спектру

завдань.

Kotlin - це мова програмування, розроблена компанією JetBrains, яка призначена для вдосконалення розробки на платформі Java. Вона поєднує в собі виразність та концизність з модернізованими можливостями. Однією з ключових особливостей Kotlin є спрощена синтаксична структура, яка дозволяє розробникам писати менше коду, зменшуючи ймовірність помилок та полегшуючи читання.

Рисунок 3.3 – Логотип мови програмування Kotlin

Kotlin, подібно до Java, може працювати на будь-якій платформі, що підтримує JVM. Однак, вона також надає можливість компіляції в JavaScript та використання в розробці мобільних додатків для Android. Kotlin включає в себе концепції функціонального програмування, такі як високорівневі функції, лямбда-вирази та інші. Це полегшує створення більш зручного та зрозумілого коду. Kotlin приділяє велику увагу безпеці, надаючи безпечні типи та покращуючи перевірку на етапі компіляції. Крім того, вона легко взаємодіє з існуючим Java-кодом, що полегшує поетапний перехід для розробників. Мова Kotlin має численні додаткові можливості, такі як вбудована підтримка нульових значень (nullable types), розширення функцій (extension functions), та короткий синтаксис для роботи з колекціями. Kotlin використовується для розробки різноманітних додатків, включаючи мобільні застосунки для Android,

веб-додатки та корпоративні системи. Її популярність зростає завдяки зручному синтаксису, покращеним можливостям та високій ступені інтероперабельності з Java.

Для збереження кодової бази було вирішено використати систему контролю версії Git та платформу Github.

Git є розподіленою системою контролю версій, розробленою Лінусом Торвальдсом. Вона забезпечує можливість відстежувати зміни в коді та сприяє спільній роботі розробників над проектами. Його особливості включають розподілену архітектуру, ефективність та здатність до відгалуження та злиття.

GitHub - це платформа для спільної роботи з проектами, що використовують Git. Забезпечує хмарне сховище для Git-репозиторіїв, що дозволяє розробникам спільно працювати над проектами. GitHub надає інструменти для відстеження проблем, обговорення коду та автоматичні можливості розгортання. Git дозволяє відстежувати кожен коміт, забезпечуючи повну історію змін у коді, розгалуження та злиття в Git дозволяють розробникам незалежно працювати над функціями та потім об'єднати свої зміни. GitHub дозволяє зберігати код в хмарі, що полегшує спільний доступ та зменшує втрату даних та інтегрується з різними іншими інструментами розробки, такими як CI, що полегшує автоматизацію тестування та розгортання. Загалом, Git та GitHub утворюють потужну комбінацію для ефективного управління та спільної роботи над програмним кодом.

3.2 Реалізація методу для навігації з доповненою реальністю

Для інтеграції бібліотеки для роботи з доповненою реальністю необхідно відповідно налаштувати Android проект. Для цього було створено пустий проект у середовищі розробки Android Studio, обрано початкові

налаштування та конфігурацію (рис. 3.4).

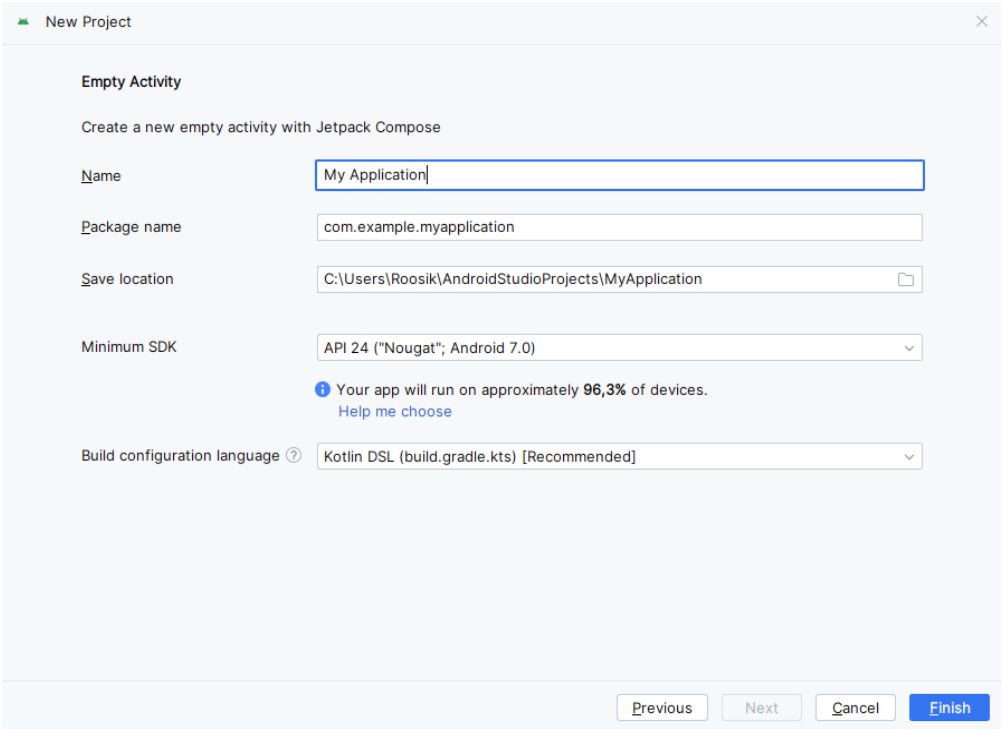

Рисунок 3.4 – Діалог створення нового проекту у середовищі Android Studio

Додаток було створено для підтримки усіх версій Android аж до версії Android SDK рівня 31, що відповідає 12 версії ОС Android.

Далі у конфігураційний файл Android проекту, build.gradle, було додано усі необхідні для розробки залежності, такі як бібліотека BeyondAR, Material Design, AppCompat, Google Play Services, Retrofit та інші [19].

Відповідно до політики використання SDK Android, розробник повинен імплементувати запити на використання певних функцій пристрою. Так як для роботи додатку необхідно мати можливість використання камери пристрою, мати доступ до показань GPS датчика а також мати можливість зберігати інформацію на пристрої користувача, було об’явлено використання даних дозволів у маніфесті додатку і використано механізм для отримання

переліченних дозволів під час роботи додатку (рис. 3.5).

```
public static void initialPermissionCheck(Context context, Activity activity) {

    if (ContextCompat.checkSelfPermission(context, Manifest.permission.CAMERA) !=
            PackageManager.PERMISSION_GRANTED) {
        ActivityCompat.requestPermissions(activity, new String[]{Manifest.permission.CAMERA}, requestCode: 10);
    }

    if (ContextCompat.checkSelfPermission
            (context, Manifest.permission.ACCESS_FINE_LOCATION) != PackageManager.PERMISSION_GRANTED) {
        ActivityCompat.requestPermissions(activity, new String[]{Manifest.permission.ACCESS_FINE_LOCATION}, requestCode: 20);

    }

    if (ContextCompat.checkSelfPermission(context, Manifest.permission.WRITE_EXTERNAL_STORAGE) !=
            PackageManager.PERMISSION_GRANTED){
        ActivityCompat.requestPermissions(activity,new String[]{Manifest.permission.WRITE_EXTERNAL_STORAGE}, requestCode: 30);
    }

    if (ContextCompat.checkSelfPermission(context, Manifest.permission.READ_EXTERNAL_STORAGE) !=
            PackageManager.PERMISSION_GRANTED){
        ActivityCompat.requestPermissions(activity,new String[]{Manifest.permission.READ_EXTERNAL_STORAGE}, requestCode: 40);
    }
}
```

Рисунок 3.5 – Метод для отримання необхідних дозволів від користувача

Для забезпечення можливості отримання інформації шляхом REST API було використано бібліотеки Retrofit, OkHttp та Gson.

Retrofit - це бібліотека для роботи з HTTP-запитами в Android-додатках. Розроблена компанією Square, вона спрощує використання HTTP-запитів та взаємодію з веб-службами в Android-додатках, реалізуючи RESTful API. Retrofit використовує аннотації для визначення параметрів запитів, URL-адрес та інших властивостей, що робить синтаксис чистим та зручним. Retrofit вбудовує можливості асинхронних запитів, використовуючи механізм CallBack або використовуючи бібліотеку RxJava для реактивного програмування. Retrofit дозволяє використовувати інтерсептори для модифікації або обробки запитів та відповідей, наприклад, для додавання заголовків або логування.

Gson - це бібліотека для перетворення об'єктів Java в JSON та навпаки. Розроблена компанією Google, вона дозволяє легко виконувати операції серіалізації та десеріалізації об'єктів у формат JSON, що є частою потребою в

роботі з мережевими запитами в Android-додатках. Gson підтримує серіалізацію та десеріалізацію вкладених об'єктів, списків, масивів та інших загальних структур даних. Gson пропонує простий та зрозумілий API для серіалізації та десеріалізації об'єктів, що робить процес зручним для розробників.

Для побудови навігаційних вказівників у доповненій реальності перш за все потрібно отримати маршрут між початковою та кінцевою локацією. Для отримання маршруту було використано сервіс Google Maps, виконавши запит на який отримаємо об’єкт DirectionsResponse, який містить опис кроків маршруту. Запит на дану API виконується шляхом імплементації методів Retrofit (рис. 3.6).

```
private void Directions_call() {
    HttpLoggingInterceptor interceptor = new HttpLoggingInterceptor();
    interceptor.setLevel(HttpLoggingInterceptor.Level.BODY);
    OkHttpClient client = new OkHttpClient.Builder().addInterceptor(interceptor).build();

    Retrofit retrofit = new Retrofit.Builder()
            .baseUrl("https://maps.googleapis.com/")
            .client(client)
            .addConverterFactory(GsonConverterFactory.create())
            .build();

    RetrofitInterface apiService = retrofit.create(RetrofitInterface.class);

    final Call<DirectionsResponse> call = apiService.getDirections(srcLatLng, destLatLng,
            getResources().getString(R.string.google_maps_key));
```

Рисунок 3.6 – Метод для отримання кроків маршруту з сервісу Google Maps

Після виконання запиту об’єкт DirectionsResponse парситься на окремі кроки для навігації, суцільний маршрут розділяється на окремі лінії, з яких отримується координати початку та кінця лінії. Таким чином отримується масив координат який описує кожен крок для побудови маршруту. Частина алгоритму відображення навігаційних вказівників наведено на рисунку 3.7.

```
for (int i = 0; i < steps.length; i++) {
    polylineLatLng.add(i, PolyUtil.decode(steps[i].getPolyline().getPoints()));

    String instructions = steps[i].getHtmlInstructions();

    if (i == 0) {
        GeoObject signObject = new GeoObject( id: 10000 + i);
        signObject.setImageResource(R.drawable.start);
        signObject.setGeoPosition(steps[i].getStartLocation().getLat(), steps[i].getStartLocation().getLng());
        world.addBeyondarObject(signObject);
        Log.d(TAG,  msg: "Configure_AR: START SIGN:" + i);
    }

    if (i == steps.length - 1) {
        GeoObject signObject = new GeoObject( id: 10000 + i);
        signObject.setImageResource(R.drawable.stop);
        LatLng latlng = SphericalUtil.computeOffset(
                new LatLng(steps[i].getEndLocation().getLat(), steps[i].getEndLocation().getLng()),
                distance: 4f, SphericalUtil.computeHeading(
                        new LatLng(steps[i].getStartLocation().getLat(), steps[i].getStartLocation().getLng()),
                        new LatLng(steps[i].getEndLocation().getLat(), steps[i].getEndLocation().getLng())));
        signObject.setGeoPosition(latlng.latitude, latlng.longitude);
        world.addBeyondarObject(signObject);
        Log.d(TAG,  msg: "Configure_AR: STOP SIGN:" + i);
    }

    if (instructions.contains("right")) {
        Log.d(TAG,  msg: "Configure_AR: " + instructions);
        GeoObject signObject = new GeoObject( id: 10000 + i);
        signObject.setImageResource(R.drawable.turn_right);
        signObject.setGeoPosition(steps[i].getStartLocation().getLat(), steps[i].getStartLocation().getLng());
        world.addBeyondarObject(signObject);
        Log.d(TAG,  msg: "Configure_AR: RIGHT SIGN:" + i);
    } else if (instructions.contains("left")) {
        Log.d(TAG,  msg: "Configure_AR: " + instructions);
        GeoObject signObject = new GeoObject( id: 10000 + i);
        signObject.setImageResource(R.drawable.turn_left);
        signObject.setGeoPosition(steps[i].getStartLocation().getLat(), steps[i].getStartLocation().getLng());
        world.addBeyondarObject(signObject);
        Log.d(TAG,  msg: "Configure_AR: LEFT SIGN:" + i);
    }
}
```

Рисунок 3.7 – Частина методу для позиціонування навігаційних вказівників у просторі доповненої реальності

Імплементовано цикл, який ітерується по отриманому масиву кроків маршруту і перевіряє кожен етап. Для першого елемента в просторі доповненої реальності відображається вказівник початку маршрута, з елемента масиву отримуються координати для точного позиціонування вказівника у відповідному місці. Аналогічно відображається кінцева точка маршруту. Після цього під час ітерації по масиву кроків навігації у простір доповненої реальності наносяться вказівники контрольних точок, тобто поворотів.

Наступна частина алгоритму додає проміжні вказівники між поворотами, таким чином досягається суцільний маршрут. Алгоритм ітерується по масиву координат між поворотами (рис. 3.8).

```
for (int i = 0; i < arObj_count; i++) {
    GeoObject inter_polyGeoObj = new GeoObject( id: 5000 + temp_inter_polycount++);
    if (i > 0 && k < polylineLatLng.get(j).size()) {
        increment_dist += 3f;
        tempLatLng = SphericalUtil.computeOffset(new LatLng(polylineLatLng.get(j).get(k).latitude,
                        polylineLatLng.get(j).get(k).longitude),
                increment_dist,
                SphericalUtil.computeHeading(new LatLng(polylineLatLng.get(j).get(k).latitude
                        , polylineLatLng.get(j).get(k).longitude), new LatLng(
                        polylineLatLng.get(j).get(k + 1).latitude
                        , polylineLatLng.get(j).get(k + 1).longitude)));
    }
    inter_polyGeoObj.setGeoPosition(tempLatLng.latitude, tempLatLng.longitude);
    inter_polyGeoObj.setImageResource(R.drawable.ar_sphere_default_125x);
    inter_polyGeoObj.setName("inter_arObj" + j + k + i);
    world.addBeyondarObject(inter_polyGeoObj);
}
```

Рисунок 3.8 – Цикл для побудови проміжних навігаційних вказівників

Для кожної точки перевіряються її координати – якщо точка знаходиться в радіусі 3 умовних одиниць (3 діаметра навігаційної сфери) до її абсолютного положення додається оффсет щоб запобігти створенню занадто щільних вказівників, так як надмірна кількість об’єктів негативно впливає на швидкодію та працездатність додатку.

Для вірного функціонування фільтру Маджвіка необхідно мати інформацію з магнітометра, акселерометра та гіроскопа. Для цього було використано об’єкт PackageManager. PackageManager у Android - це системна служба, яка надає інформацію про всі встановлені додатки на пристрої, а також управляє їхніми параметрами та властивостями. Цей клас є частиною Android SDK і надає доступ до різноманітних функцій, пов'язаних з управлінням

пакетами додатків. Так як ця служба має доступ до інформації про додатки, за її допомогою можна перевірити наявність дозволів додатку на використання магнітометра, акселерометра та гіроскопа. Метод для отримання даної інформації зображено на рисунку 3.9.

```
private void checkIfSensorsAvailable() {
    PackageManager pm = getActivity().getPackageManager();
    boolean compass = pm.hasSystemFeature(PackageManager.FEATURE_SENSOR_COMPASS);
    boolean accelerometer = pm.hasSystemFeature(PackageManager.FEATURE_SENSOR_ACCELEROMETER);
    if (!compass && !accelerometer) {
        throw new IllegalStateException(getClass().getName()
                + " can not run without the compass and the acelerometer sensors.");
    } else if (!compass) {
        throw new IllegalStateException(getClass().getName() + " can not run without the compass sensor.");
    } else if (!accelerometer) {
        throw new IllegalStateException(getClass().getName()
                + " can not run without the acelerometer sensor.");
    }
}
```

Рисунок 3.9 – Метод для отримання інформації про доступ додатку до магнітометра, акселерометра та гіроскопа

Для отримання поточної локації користувача розроблено окремий метод. Зважаючи на обмеження системи Android, забороняється доступ до показань GPS з пристрою користувача без окремо наданого користувачем дозволу, тож на початку метода перевіряється наявність дозволу у розробленого додатку. Якщо дозвіл надано отримується об’єкт LocationManager. LocationManager у Android - це клас, який дозволяє взаємодіяти з сервісами місцезнаходження на пристрої. Він надає доступ до інформації про місцезнаходження, такої як координати (широта та довгота), швидкість, висота і т. д. Цей клас є частиною Android SDK і дозволяє розробникам отримувати і використовувати дані про геолокацію у своїх додатках. Метод getLastLocation використовує у своїй реалізації фільтр Маджвіка за рахунок чого досягається підвищена точність

позиціонування, ніж при використанні «сирих» даних з датчика GPS (рис. 3.9).

```
public void onConnected(@Nullable Bundle bundle) {
    if (ActivityCompat.checkSelfPermission( context: this, Manifest.permission.ACCESS_FINE_LOCATION)
            != PackageManager.PERMISSION_GRANTED) {
        ActivityCompat.requestPermissions( activity: this,
                new String[]{Manifest.permission.ACCESS_FINE_LOCATION}, requestCode: 1);

    } else {
        locationManager = (LocationManager) this.getSystemService(Context.LOCATION_SERVICE);
        String locationProvider = LocationManager.NETWORK_PROVIDER;
        mLastLocation = LocationServices.FusedLocationApi.getLastLocation(mGoogleApiClient);

        if (mLastLocation != null) {
            try {
                Get_intent();
            } catch (Exception e) {
                Log.d(TAG, msg: "onCreate: Intent Error");
            }
        }
    }
    startLocationUpdates();
}
```

Рисунок 3.9 – Метод для отримання поточної позиції користувача

Основна частина лістингу буде знаходитись у додатку В.

3.3 Результати роботи програмного засобу для навігації в доповненій реальності

Після імплементації описаних алгоритмів та розробки інтерфейсу було отримано готовий продукт для навігації за допомогою доповненої реальності. Точкою входу в додаток є головний екран з мапою та елементами меню (рис. 3.10). Для відображення мапи в додатку використано GoogleMaps API. В додатку виконується запит до API в результаті якого отримується об’єкт карти і в подальшому відмальовується на заздалегідь розміченому екрані. Користувач

може шукати локації за допомогою Searchbar для подальшої навігації до об'єкту, або ж використати елементи меню для переходу на інші екрани додатку.

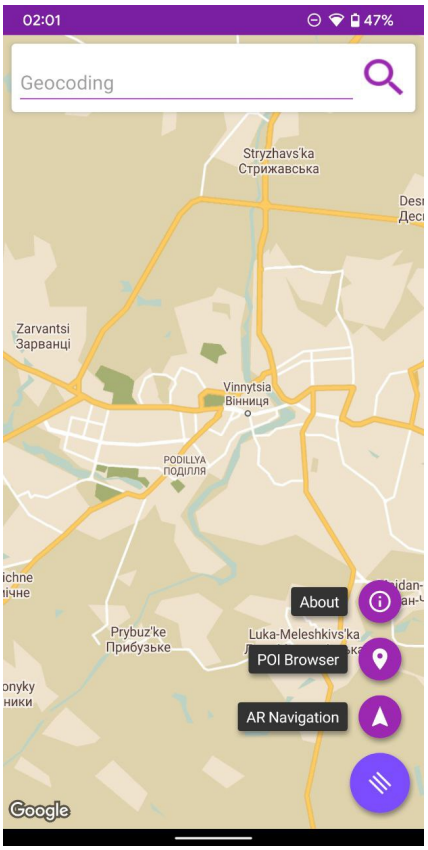

Рисунок 3.10 – Початковий екран додатку

Для доступу до навігації з доповненою реальністю користувачу потрібно обрати пункт меню «AR Navigation». Після цього користувач буде перенаправлений на екран вибору початкової та кінцевої локації.. Екран містить дві кнопки для вибору початкової та кінцевої локації, а також дві кнопки вибору варіанту навігації – класична чи за допомогою доповненої реальності. При натисканні на кнопки вибору локацій відкривається допоміжний екран вибору локацій (рис. 3.11) На екрані вибору локації також розміщена карта на якій відмічена локація користувача, розміщено маркери

усіх доступних локацій поряд, та покажчик обраної локації посередині екрану. Також в нижній частині екрану перелічений список локацій поблизу отриманих шляхом геокодингу об’єктів. Користувач повинен обрати будь яку локацію до якої створюватиметься навігація. Також користувач має змогу використати строку пошуку для швидкого пошуку заздалегідь відомої локації.

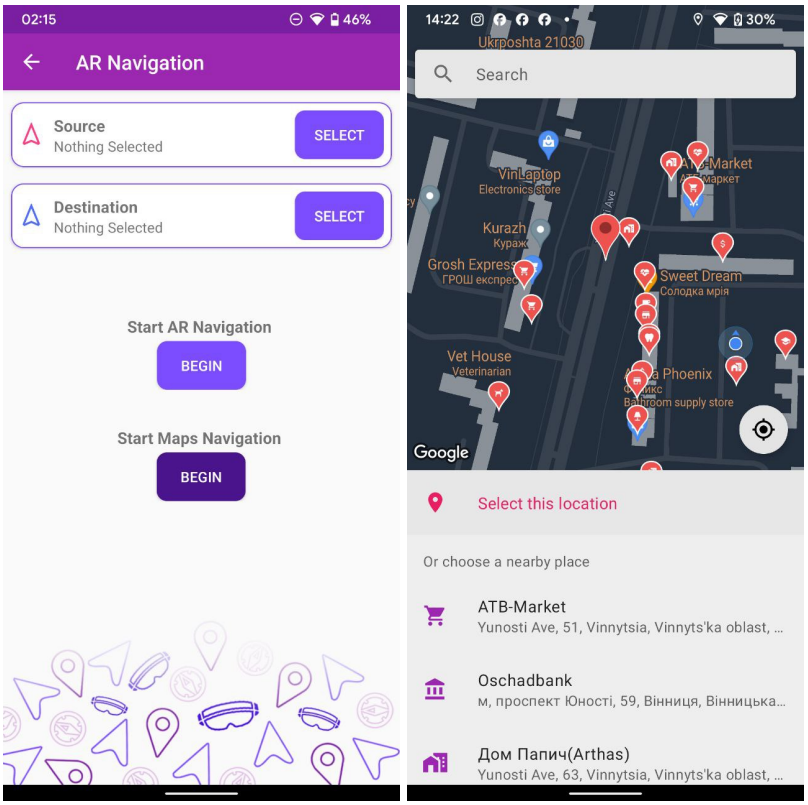

Рисунок 3.11 – Екрани вибору початкової та кінцевої локації

Після вибору початкової та кінцевої локації і вибору навігації з доповненою реальністю користувач буде направлений безпосередньо на екран доповненої реальності з відображеним маршрутом між обраними локаціями (рис. 3.12).

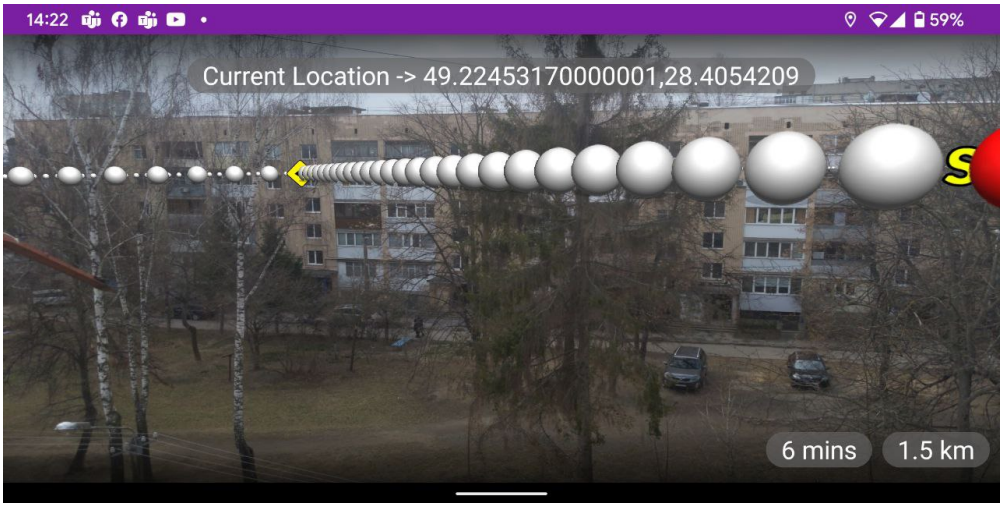

Рисунок 3.12 – Екран доповненої реальності з відображеним маршрутом між обраними локаціями

3.4 Висновок

У третьому розділі було виконано аналіз та обґрунтування вибору засобів реалізації програми.

Було вирішено використовувати мови програмування Java та Kotlin, середовище розробки Android Studio. Для інтеграції технології доповненої реальності було використано бібліотеку BeyondAR. Для отримання мапи, локацій, та маршрутів було використано сервіси Google.

Реалізовано алгоритм отримання усієї необхідної інформації для побудови маршруту, методи отримання обов’язкових дозволів на використання апаратних компонентів. Також реалізовано алгоритм обрахунку та відображення навігаційних вказівників відповідно до заданого маршруту. Розроблено додаток відповідно до попередньо створеного дизайну.

4 ТЕСТУВАННЯ ПРОГРАМИ

4.1 Аналіз методів тестування програмного забезпечення

Розробка високоякісних мобільних додатків - завдання, яке вимагає уваги до різноманітних аспектів, включаючи функціональність, інтерфейс, надійність, продуктивність, сумісність та безпеку. Якість програмного забезпечення (ПЗ) є ключовим елементом у досягненні задоволення користувачів та успіху додатка. Давайте розглянемо детальніше критерії оцінювання та методи тестування для мобільних додатків.

До критеріїв оцінювання відносяться:

- Функціональність. Визначає, наскільки ефективно додаток виконує свої основні завдання.
- Надійність. Надійність визначається стабільністю роботи додатка та його здатністю відновлювати роботу після помилок.
- Продуктивність. Продуктивність описує швидкодію додатка та його використання ресурсів.
- Сумісність. Визначає, як добре додаток взаємодіє з різними платформами та пристроями.
- Безпека. Здатність додатку до захищення важливих даних від несанкціонованого доступу, витоку чи інших інформаційних загроз.

Тестування програмного забезпечення (ПЗ) є процесом, який спрямований на перевірку та валідацію програми з метою визначення його якості та відповідності визначеним стандартам та вимогам. Основні цілі тестування включають:

1. Визначення відповідності вимогам: Тестування служить для перевірки, чи програма відповідає визначеним функціональним та нефункціональним вимогам, що були визначені на етапі розробки.

2. Виявлення помилок та дефектів: Одна з ключових цілей тестування - виявлення помилок та недоліків у програмі. Раннє виявлення дозволяє виправляти проблеми на ранніх етапах розробки, що допомагає зменшити вартість їх виправлення.

3. Вдосконалення якості: Тестування спрямоване на покращення якості програмного продукту, визначення його стабільності та оптимізацію продуктивності.

4. Верифікація та валідація: Ціль тестування включає в себе перевірку відповідності програми визначеним специфікаціям (верифікація) та переконання, що програма задовольняє потреби та очікування користувачів (валідація).

5. Забезпечення безпеки: Тестування спрямоване на виявлення вразливостей та забезпечення високого рівня безпеки програмного продукту.

6. Сумісність: Ціль тестування - перевірка працездатності програми на різних операційних системах, пристроях та в різних середовищах.

7. Тестування відмовостійкості: Спрямоване на перевірку стійкості програми до навантаження та її здатність відновлювати працездатність після відмов.

8. Забезпечення зручності використання: Тестування включає в себе оцінку користувацького інтерфейсу для забезпечення його зручності та інтуїтивності використання.

Загальна мета тестування полягає в забезпеченні високої якості та надійності програми, що відповідає вимогам та очікуванням користувачів. Тестування визначається не лише виявленням помилок, але і забезпеченням ефективності, безпеки та задоволення потреб користувачів.

Тестування програмного забезпечення включає в себе різні підходи, кожен з яких має свої цілі. Серед них:

1. Модульне тестування – фокусується на окремих модулях чи компонентах програми, перевіряючи їх коректність.

2. Інтеграційне тестування – спрямоване на взаємодію різних частин програми, визначаючи, як вони працюють у спільності.

3. Системне тестування – охоплює весь продукт, перевіряючи його відповідність вимогам та очікуванням користувачів.

4. Приймальне тестування – визначає, чи задовольняє програма потреби та очікування замовника.

5. Регресійне тестування – перевіряє, чи нові зміни не впливають на існуючий функціонал.

6. Стрес-тестування – випробовує програму під великим навантаженням, виявляючи її межі.

7. Тестування безпеки – перевіряє вразливості та забезпечує захист від потенційних атак.

8. Тестування сумісності – визначає, як програма працює на різних платформах та пристроях.

9. Тестування відмовостійкості – перевіряє стійкість до відмов та здатність до відновлення.

Ці різноманітні методи сприяють виявленню та виправленню помилок на різних етапах розробки, що сприяє стабільності та ефективності програмного продукту.

4.2 Тестування розробленого програмного продукту

Було виконано мануальне тестування додатку . Було перевірено роботу усіх функцій додатку, а саме: функціональність головного екрану, завантаження мапи, навігацію між екранами, роботу технології доповненої

реальності, навігацію в доповненій реальності, побудову навігаційних вказівників.

Спочатку було протестовано головну сторінку на відповідність елементів до раніше розроблених дизайнів та було протестовано функціональність розробленого екрану та відображення карти (рис. 4.1). Як ми можемо спостерігати мапа була успішно завантажена та відображена на екрані, всі інші UI елементи розміщені у відповідних місцях.

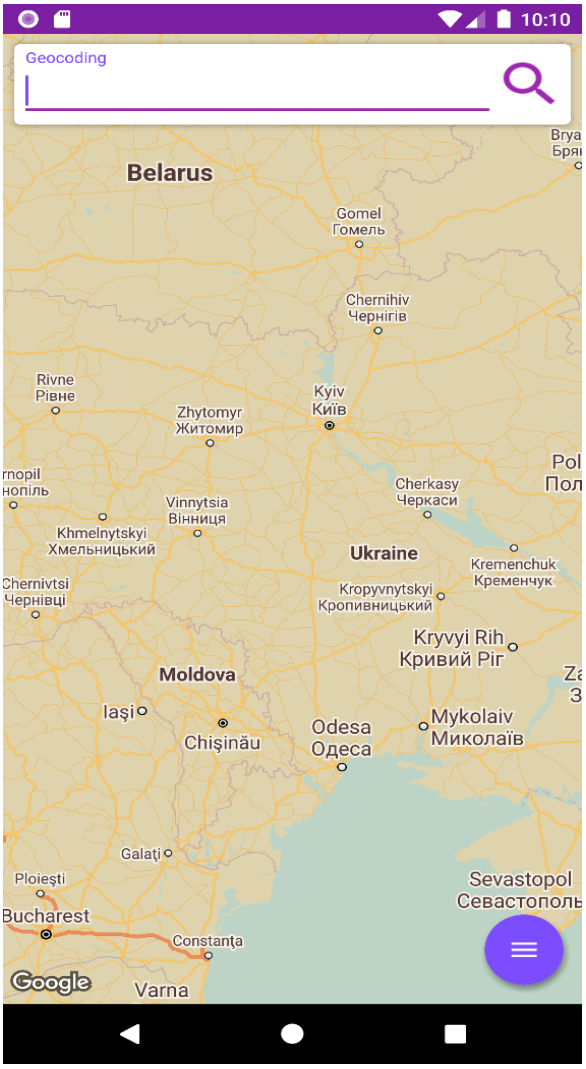

Рисунок 4.1 – Тестування функціональності головної сторінки

Далі було перевірено перехід з головного екрану на екран вибору початкової адреси і шуканого місця, а також перевірено працездатність

елементів для вибору користувача (рис. 4.2). Розмітка екрану відповідає попередньо створеному дизайну. Елементи вибору локацій працюють коректно, при кліку на елемент відкривається фрагмент вибору локації. Кнопки вибору методу навігації переводять користувача на відповідні екрани. Також було перевірено збереження введених даних під час згортання та розгортання додатку. За рахунок використання Android компоненту ViewModel, досягнуто зберігання введеної інформації, так як життєвий цикл ViewModel довший за життєвий цикл фрагменту чи активності.

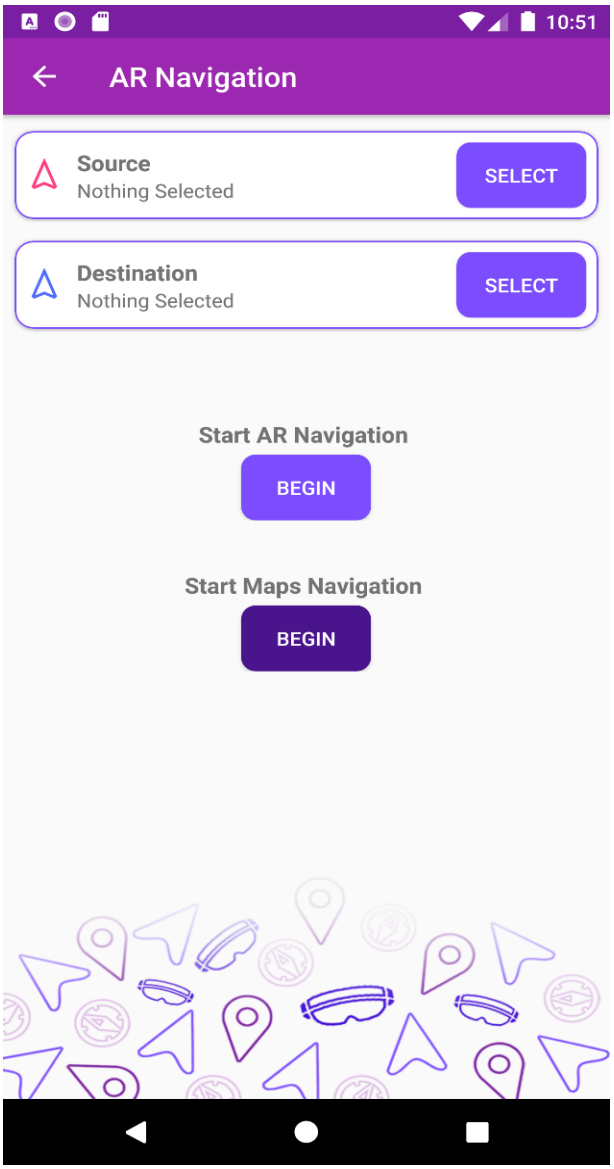

Рисунок 4.3 – Тестування екрану вибору локацій та режимів

Також було протестовано роботу функції «Start Map Navigation». При кліку на відповідну кнопку «BEGIN» повинна відкритись Google Map чи інша дефолтна навігаційна програма і прокладено маршрут між попередньо обраними точками (рис. 4.5).

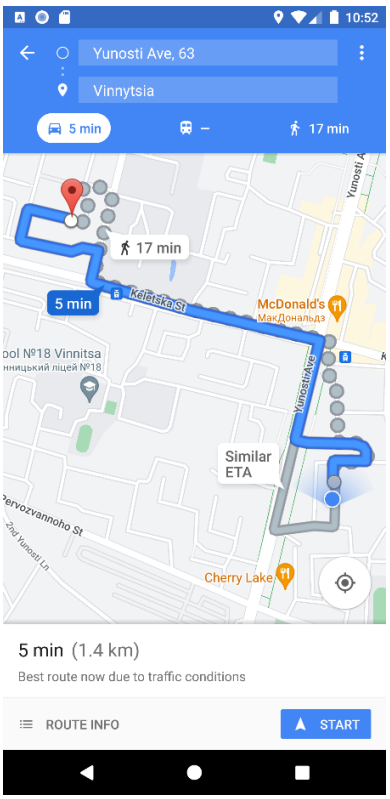

Рисунок 4.5 – Результат виконання функції «Start Map Navigation»

Також було протестовано роботу функції «Start AR Navigation». При кліку на відповідну кнопку «BEGIN» повинен відкритись AR-екран з навігацією між обраними точками. Тестування було проведено на емуляторі , з замоканими функціями камери та GPS щоб виключити ймовірність впливу на тест змінних величин. Для усіх проведених тестів було обрано статичну локацію девайса (рис. 4.6).

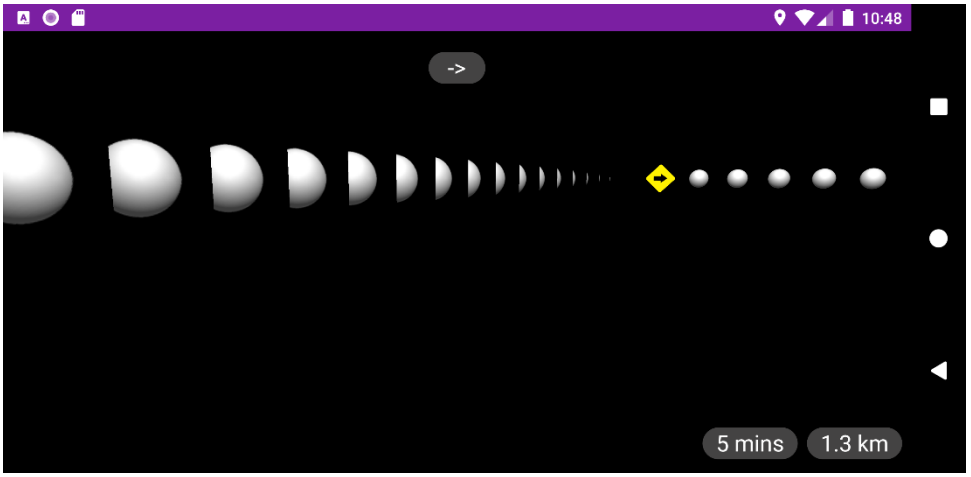

Рисунок 4.6 – Тестування функції «Start AR Navigation» з замоканими даними

Далі було протестовано екран з пошуком POI (Point of interest) об’єктів. Для тестування даного екрану також було використано емулятор , статичну локацію, та віртуальну камеру (рис. 4.7).

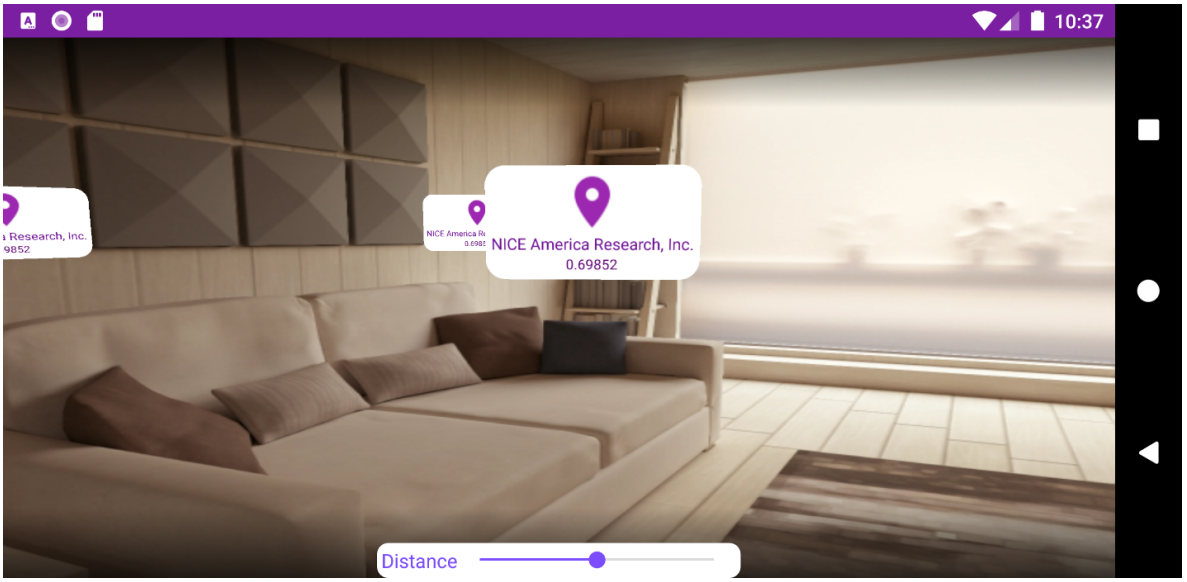

Рисунок 4.7 – Тестування екрану для пошуку POI об’єктів

Було проведено тестування опрацювання кліків користувача на view POI об’єкта. При кліку на відповідний об’єкт повинен відкритись діалог з детальною інформацією про об’єкт , фото та варіантами навігації (рис. 4.8).

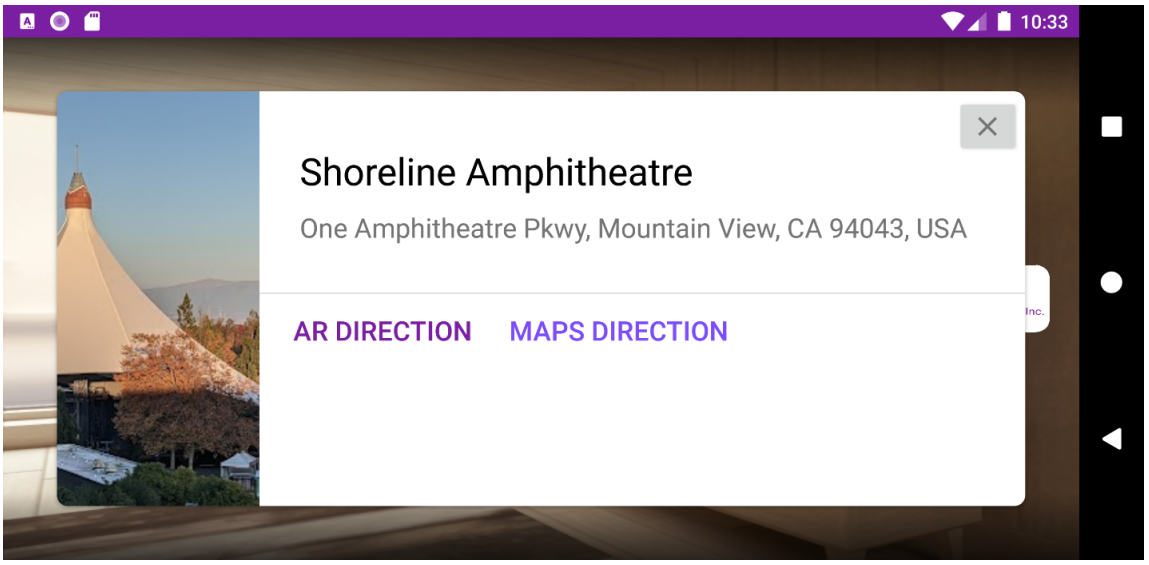

Рисунок 4.8 – Тестування показу діалогу з детальною інформацією

Під час тестування не було виявлено явних недоліків, додаток відповідає поставленим вимогам. Закладена у додаток функціональність працює правильно.

4.3 Розробка інструкцій користувача

Розробка інструкції також є важливим етапом при розробці будь якого програмного продукту, адже саме завдяки інструкції користувач дізнається про функціональність додатку та алгоритм роботи продукту.

Для початку роботи з додатком користувач повинен встановити його на свій мобільний пристрій. Додаток встановлюється шляхом відкриття apk файлу з програмою.

В якості інструкції користувача при першому відкритті програми було створено екрани онбордингу нового користувача, який містить опис та інформацію про основні функції додатку (рис. 4.9).

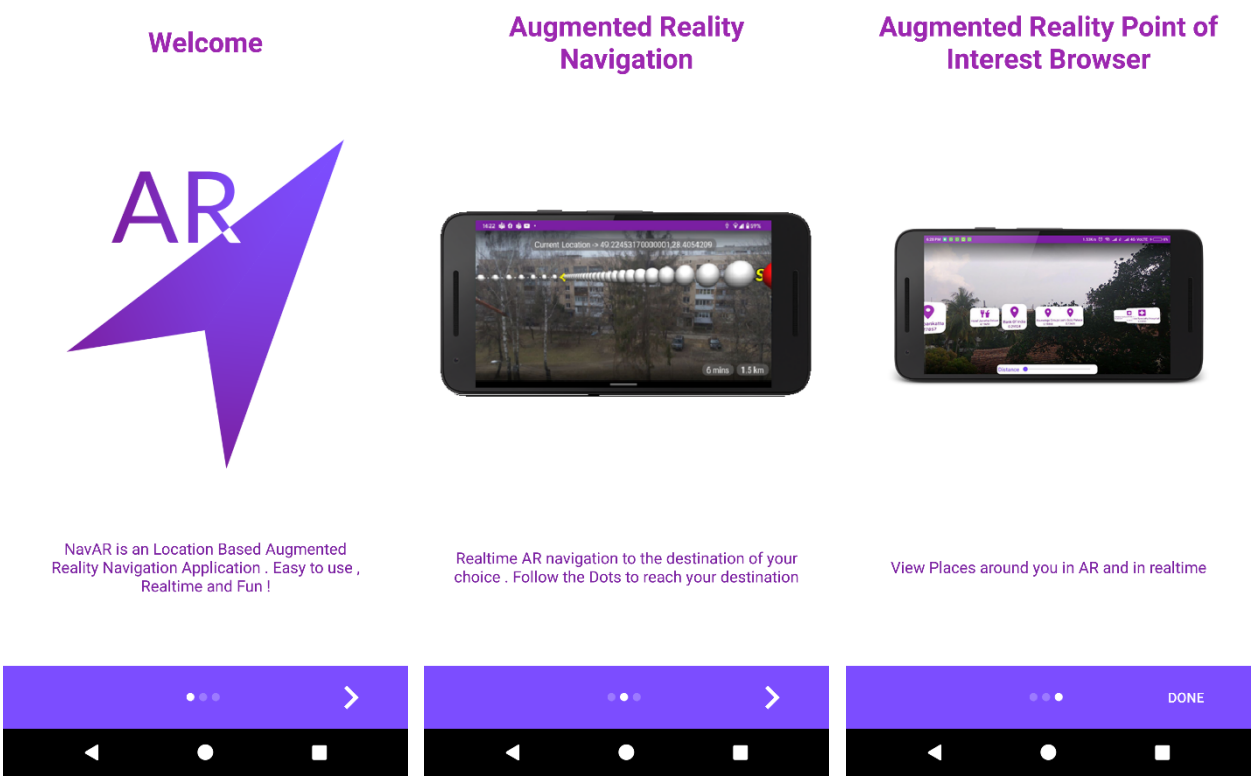

Рисунок 4.9 – Екрани з інструкцією для користувача

4.3 Висновки

Отже, в результаті було проведено тестування додатку методом «чорного ящика» і отримано позитивні результати. Розроблено інструкцію користувача з описом основних функціональних можливостей додатку.

5 ЕКОНОМІЧНА ЧАСТИНА

Науково-технічна розробка має право на існування та впровадження, якщо вона відповідає вимогам часу, як в напрямку науково-технічного прогресу та і в плані економіки. Тому для науково-дослідної роботи необхідно оцінювати економічну ефективність результатів виконаної роботи.

Магістерська кваліфікаційна робота за темою «Розробка методу та програмного засобу для навігації із доповненою реальністю» відноситься до науково-технічних робіт, які орієнтовані на виведення на ринок (або рішення про виведення науково-технічної розробки на ринок може бути прийнято у процесі проведення самої роботи), тобто коли відбувається так звана комерціалізація науково-технічної розробки. Цей напрямок є пріоритетним, оскільки результатами розробки можуть користуватися інші споживачі, отримуючи при цьому певний економічний ефект. Але для цього потрібно знайти потенційного інвестора, який би взявся за реалізацію цього проекту і переконати його в економічній доцільності такого кроку.

Для наведеного випадку нами мають бути виконані такі етапи робіт:

1) проведено комерційний аудит науково-технічної розробки, тобто встановлення її науково-технічного рівня та комерційного потенціалу;

2) розраховано витрати на здійснення науково-технічної розробки;

3) розрахована економічна ефективність науково-технічної розробки у випадку її впровадження і комерціалізації потенційним інвестором і проведено обґрунтування економічної доцільності комерціалізації потенційним інвестором.

5.1 Проведення комерційного та технологічного аудиту науково-технічної розробки

Метою проведення комерційного і технологічного аудиту дослідження за темою «Розробка методу та програмного засобу для навігації із доповненою

реальністю» є оцінювання науково-технічного рівня та рівня комерційного потенціалу розробки, створеної в результаті науково-технічної діяльності.

Оцінювання науково-технічного рівня розробки та її комерційного потенціалу рекомендується здійснювати із застосуванням 5-ти бальної системи оцінювання за 12-ма критеріями, наведеними в табл. 5.1 [20].

Таблиця 5.1 – Рекомендовані критерії оцінювання науково-технічного рівня і комерційного потенціалу розробки та бальна оцінка

Бали (за 5-ти бальною шкалою)					
	0	1	2	3	4
Технічна здійсненність концепції					
1	Достовірність концепції не підтверджена	Концепція підтверджена експертними висновками	Концепція підтверджена розрахунками	Концепція перевірена на практиці	Перевірено працездатність продукту в реальних умовах
Ринкові переваги (недоліки)					
2	Багато аналогів на малому ринку	Мало аналогів на малому ринку	Кілька аналогів на великому ринку	Один аналог на великому ринку	Продукт не має аналогів на великому ринку
3	Ціна продукту значно вища за ціни аналогів	Ціна продукту дещо вища за ціни аналогів	Ціна продукту приблизно дорівнює цінам аналогів	Ціна продукту дещо нижче за ціни аналогів	Ціна продукту значно нижче за ціни аналогів
4	Технічні та споживчі властивості продукту значно гірші, ніж в аналогів	Технічні та споживчі властивості продукту трохи гірші, ніж в аналогів	Технічні та споживчі властивості продукту на рівні аналогів	Технічні та споживчі властивості продукту трохи кращі, ніж в аналогів	Технічні та споживчі властивості продукту значно кращі, ніж в аналогів
5	Експлуатаційні витрати значно вищі, ніж в аналогів	Експлуатаційні витрати дещо вищі, ніж в аналогів	Експлуатаційні витрати на рівні експлуатаційних витрат аналогів	Експлуатаційні витрати трохи нижчі, ніж в аналогів	Експлуатаційні витрати значно нижчі, ніж в аналогів
Ринкові перспективи					
6	Ринок малий і не має позитивної динаміки	Ринок малий, але має позитивну динаміку	Середній ринок з позитивною динамікою	Великий стабільний ринок	Великий ринок з позитивною динамікою

7	Активна конкуренція великих компаній на ринку	Активна конкуренція	Помірна конкуренція	Незначна конкуренція	Конкурентів немає
Практична здійсненність					
8	Відсутні фахівці як з технічної, так і з комерційної реалізації ідеї	Необхідно наймати фахівців або витрачати значні кошти та час на навчання наявних фахівців	Необхідне незначне навчання фахівців та збільшення їх штату	Необхідне незначне навчання фахівців	Є фахівці з питань як з технічної, так і з комерційної реалізації ідеї
9	Потрібні значні фінансові ресурси, які відсутні. Джерела фінансування ідеї відсутні	Потрібні незначні фінансові ресурси. Джерела фінансування відсутні	Потрібні значні фінансові ресурси. Джерела фінансування є	Потрібні незначні фінансові ресурси. Джерела фінансування є	Не потребує додаткового фінансування
10	Необхідна розробка нових матеріалів	Потрібні матеріали, що використовуют ь ся у військово промисловому комплексі	Потрібні дорогі матеріали	Потрібні досяжні та дешеві матеріали	Всі матеріали для реалізації ідеї відомі та давно використовуют ься у виробництві
11	Термін реалізації ідеї більший за 10 років	Термін реалізації ідеї більший за 5 років. Термін окупності інвестицій більше 10-ти років	Термін реалізації ідеї від 3-х до 5-ти років. Термін окупності інвестицій більше 5-ти років	Термін реалізації ідеї менше 3-х років. Термін окупності інвестицій від 3-х до 5-ти років	Термін реалізації ідеї менше 3-х років. Термін окупності інвестицій менше 3-х років
12	Необхідна розробка регламентних документів та отримання великої кількості дозвільних документів на виробництво та реалізацію продукту	Необхідно отримання великої кількості дозвільних документів на виробництво та реалізацію продукту, що вимагає значних коштів та часу	Процедура отримання дозвільних документів для виробництва та реалізації продукту вимагає незначних коштів та часу	Необхідно тільки повідомлення відповідним органам про виробництво та реалізацію продукту	Відсутні будь-які регламентні обмеження на виробництво та реалізацію продукту

Результати оцінювання науково-технічного рівня та комерційного потенціалу науково-технічної розробки потрібно звести до таблиці.

Таблиця 5.2 – Результати оцінювання науково-технічного рівня і комерційного потенціалу розробки експертами

Критерії	Експерт (ПІБ, посада)		
	1	2	3
	Бали:		
1. Технічна здійсненність концепції	4	3	4
2. Ринкові переваги (наявність аналогів)	4	3	4
3. Ринкові переваги (ціна продукту)	4	4	3
4. Ринкові переваги (технічні властивості)	4	3	4
5. Ринкові переваги (експлуатаційні витрати)	3	3	3
6. Ринкові перспективи (розмір ринку)	3	4	3
7. Ринкові перспективи (конкуренція)	4	3	4
8. Практична здійсненність (наявність фахівців)	3	3	3
9. Практична здійсненність (наявність фінансів)	3	3	4
10. Практична здійсненність (необхідність нових матеріалів)	3	3	3
11. Практична здійсненність (термін реалізації)	3	3	4
12. Практична здійсненність (розробка документів)	3	3	3
Сума балів	41	38	42
Середньоарифметична сума балів $СБ_с$	40,3		

За результатами розрахунків, наведених в таблиці 5.2, зробимо висновок щодо науково-технічного рівня і рівня комерційного потенціалу розробки. При цьому використаємо рекомендації, наведені в табл. 5.3.

Таблиця 5.3 – Науково-технічні рівні та комерційні потенціали розробки

Середньоарифметична сума балів СБ , розрахована на основі висновків експертів	Науково-технічний рівень та комерційний потенціал розробки
41…48	Високий
31…40	Вище середнього
21…30	Середній
11…20	Нижче середнього

0…10	Низький

Згідно проведених досліджень рівень комерційного потенціалу розробки за темою «Розробка методу та програмного засобу для навігації із доповненою реальністю» становить 40,3 бали, що, відповідно до таблиці 5.3, свідчить про комерційну важливість проведення даних досліджень (рівень комерційного потенціалу розробки вище середнього).

5.2 Розрахунок витрат на проведення науково-дослідної роботи

Витрати, пов’язані з проведенням науково-дослідної роботи на тему «Розробка методу та програмного засобу для навігації із доповненою реальністю», під час планування, обліку і калькулювання собівартості науково-дослідної роботи групуємо за відповідними статтями.

5.2.1 Витрати на оплату праці

До статті «Витрати на оплату праці» належать витрати на виплату основної та додаткової заробітної плати керівникам відділів, лабораторій, секторів і груп, науковим, інженерно-технічним працівникам, конструкторам, технологам, креслярам, копіювальникам, лаборантам, робітникам, студентам, аспірантам та іншим працівникам, безпосередньо зайнятим виконанням конкретної теми, обчисленої за посадовими окладами, відрядними розцінками, тарифними ставками згідно з чинними в організаціях системами оплати праці.

Основна заробітна плата дослідників

Витрати на основну заробітну плату дослідників ($З_о$) розраховуємо у відповідності до посадових окладів працівників, за формулою [21]:

$$З_о = \sum_{i=1}^{k} \frac{M_{ni} \cdot t_i}{T_p}, \qquad (5.1)$$

де k – кількість посад дослідників залучених до процесу досліджень;

M_{ni} – місячний посадовий оклад конкретного дослідника, грн;

t_i – число днів роботи конкретного дослідника, дні;

T_p – середнє число робочих днів в місяці, T_p=21 день.

$$З_о = 25000{,}00 \cdot 50 / 21 = 59523{,}81 \text{ грн.}$$

Проведені розрахунки зведемо до таблиці.

Таблиця 5.4 – Витрати на заробітну плату дослідників

Найменування посади	Місячний посадовий оклад, грн	Оплата за робочий день, грн	Число днів роботи	Витрати на заробітну плату, грн
Керівник проекту	25000	1190,48	50	59523,81
Інженер-розробник програмного забезпечення	22000	1047,62	50	52380,95
Консультант	15000	714,29	40	35714,29
Всього				147619,05

Основна заробітна плата робітників

Витрати на основну заробітну плату робітників ($З_р$) за відповідними найменуваннями робіт НДР розраховуємо за формулою:

$$З_р = \sum_{i=1}^{n} C_i \cdot t_i \quad , \tag{5.2}$$

де C_i – погодинна тарифна ставка робітника відповідного розряду, за виконану відповідну роботу, грн/год;

t_i – час роботи робітника при виконанні визначеної роботи, год.

Погодинну тарифну ставку робітника відповідного розряду C_i можна визначити за формулою:

$$C_i = \frac{M_M \cdot K_i \cdot K_c}{T_p \cdot t_{зм}} \quad , \tag{5.3}$$

де $М_М$ – розмір прожиткового мінімуму працездатної особи, або мінімальної місячної заробітної плати (в залежності від діючого законодавства), приймемо *Мм*=6700,00 грн;

$К_і$ – коефіцієнт міжкваліфікаційного співвідношення для встановлення тарифної ставки робітнику відповідного розряду (табл. Б.2, додаток Б);

$К_с$ – мінімальний коефіцієнт співвідношень місячних тарифних ставок робітників першого розряду з нормальними умовами праці виробничих об'єднань і підприємств до законодавчо встановленого розміру мінімальної заробітної плати.

$Т_р$ – середнє число робочих днів в місяці, приблизно $Т_р$ = 22 дні;

$t_{зм}$ – тривалість зміни, год.

$С_1 = 6700,00 \cdot 1,10 \cdot 1,65 / (21 \cdot 8) = 72,38$ грн.
$З_{р1} = 72,38 \cdot 10,00 = 723,84$ грн.

Таблиця 5.5 – Величина витрат на основну заробітну плату робітників

Найменування робіт	Тривалість роботи, год	Розряд роботи	Тарифний коефіцієнт	Погодинна тарифна ставка, грн	Величина оплати на робітника грн
Встановлення допоміжного обладнання	10	2	1,1	72,38	723,84
Підготовка робочого місця дослідника	4	2	1,1	72,38	289,54
Інсталяція програмного забезпечення	6	5	1,7	111,87	671,20
Всього					1684,57

Додаткова заробітна плата дослідників та робітників

Додаткову заробітну плату розраховуємо як 10 … 12% від суми основної заробітної плати дослідників та робітників за формулою:

$$З_{дод} = (З_о + З_р) \cdot \frac{Н_{дод}}{100\%}, \quad (5.4)$$

де $Н_{дод}$ – норма нарахування додаткової заробітної плати. Приймемо 12%.

$З_{дод} = (147619,05+ 1684,57) \cdot 12 / 100\% = 17916,43$грн.

5.2.2 Відрахування на соціальні заходи

Нарахування на заробітну плату дослідників та робітників розраховуємо як 22% від суми основної та додаткової заробітної плати дослідників і робітників за формулою:

$$З_н = (З_о + З_р + З_{дод}) \cdot \frac{Н_{зп}}{100\%} \quad (5.5)$$

де $Н_{зп}$ – норма нарахування на заробітну плату. Приймаємо 22%.

$Зн = (147619,05+ 1684,57+ 17916,43) \cdot 22 / 100\% = 36788,41$грн.

5.2.3 Сировина та матеріали

До статті «Сировина та матеріали» належать витрати на сировину, основні та допоміжні матеріали, інструменти, пристрої та інші засоби і предмети праці, які придбані у сторонніх підприємств, установ і організацій та витрачені на проведення досліджень.

Витрати на матеріали (*М*), у вартісному вираженні розраховуються окремо по кожному виду матеріалів за формулою:

$$M = \sum_{j=1}^{n} Н_{вj} \cdot Ц_j \cdot К_j - \sum_{j=1}^{n} В_j \cdot Ц, \quad (5.6)$$

де H_j – норма витрат матеріалу j-го найменування, кг;

n – кількість видів матеріалів;

$Ц_j$ – вартість матеріалу j-го найменування, грн/кг;

$К_j$ – коефіцієнт транспортних витрат, ($К_j$ = *1,1 ... 1,15*);

$В_j$ – маса відходів j-го найменування, кг;

$Ц_{вj}$ – вартість відходів j-го найменування, грн/кг.

$М_1 = 2 \cdot 190{,}00 \cdot 1{,}1 - 0{,}000 \cdot 0{,}00 = 418$ грн.

Проведені розрахунки зведемо до таблиці.

Таблиця 5.6 – Витрати на матеріали

Найменування матеріалу, марка, тип, сорт	Ціна за од	Норма витрат, од	Величина відходів, кг	Ціна відходів, грн/кг	Вартість витраченого матеріалу, грн
Папір канцелярський офісний	190	2	0	0	418
Канцелярське приладдя (набір офісного працівника)	150	3	0	0	495
Всього					913

5.2.4 Розрахунок витрат на комплектуючі

Витрати на комплектуючі *($К_в$)*, які використовують при проведенні НДР на тему «Розробка методу та програмного засобу для навігації із доповненою реальністю» відсутні.

5.2.5 Спецустаткування для наукових (експериментальних) робіт

До статті «Спецустаткування для наукових (експериментальних) робіт» належать витрати на виготовлення та придбання спецустаткування необхідного для проведення досліджень, також витрати на їх проектування, виготовлення, транспортування, монтаж та встановлення.

Балансову вартість спецустаткування розраховуємо за формулою:

$$B_{спец} = \sum_{i=1}^{k} Ц_i \cdot C_{пр.i} \cdot K_i \quad , \qquad (5.7)$$

де $Ц_i$ – ціна придбання одиниці спецустаткування даного виду, марки, грн;

$C_{пр.i}$ –кількість одиниць устаткування відповідного найменування, які придбані для проведення досліджень, шт.;

K_i – коефіцієнт, що враховує доставку, монтаж, налагодження устаткування тощо, (K_i = *1,10…1,12);*

k – кількість найменувань устаткування.

$B_{спец}$ *= 5000· 1 · 1,1 = 5500 грн.*

Отримані результати зведемо до таблиці:

Таблиця 5.7 – Витрати на придбання спецустаткування по кожному виду

Найменування устаткування	Кількість, шт	Ціна за одиницю, грн	Вартість, грн
Тестовий мобільний телефон	1	5 000	5500
Персональний компютер	1	25 000	27500
Персональний компютер	1	30 000	33000
Персональний компютер	1	27 000	29700
Всього			95700

5.2.6 Програмне забезпечення для наукових (експериментальних) робіт

До статті «Програмне забезпечення для наукових (експериментальних) робіт» належать витрати на розробку та придбання спеціальних програмних засобів і програмного забезпечення, (програм, алгоритмів, баз даних) необхідних для проведення досліджень, також витрати на їх проектування, формування та встановлення.

Балансову вартість програмного забезпечення розраховуємо за формулою:

$$B_{прг} = \sum_{i=1}^{k} Ц_{iпрг} \cdot C_{прг.i} \cdot K_i \quad , \tag{5.8}$$

де $Ц_{iпрг}$ – ціна придбання одиниці програмного засобу даного виду, грн;

$C_{прг.i}$ – кількість одиниць програмного забезпечення відповідного найменування, які придбані для проведення досліджень, шт.;

K_i – коефіцієнт, що враховує інсталяцію, налагодження програмного засобу тощо, ($K_i = 1,10...1,12$);

k – кількість найменувань програмних засобів.

$B_{прг} = 10000 \cdot 3 \cdot 1,1 = 33600$ *грн.*

Отримані результати зведемо до таблиці:

Таблиця 5.8 – Витрати на придбання програмних засобів по кожному виду

Найменування програмного засобу	Кількість, шт	Ціна за одиницю, грн	Вартість, грн
ОС Windows 11	3	10000	33600
Прикладний пакет Microsoft Office 2019	3	5000	16800
Всього			50400

5.2.7 Амортизація обладнання, програмних засобів та приміщень

В спрощеному вигляді амортизаційні відрахування по кожному виду обладнання, приміщень та програмному забезпеченню тощо, розраховуємо з використанням прямолінійного методу амортизації за формулою:

$$A_{обл} = \frac{Ц_б}{T_в} \cdot \frac{t_{вик}}{12}, \quad (5.9)$$

де $Ц_б$ – балансова вартість обладнання, програмних засобів, приміщень тощо, які використовувались для проведення досліджень, грн;

$t_{вик}$ – термін використання обладнання, програмних засобів, приміщень під час досліджень, місяців;

$T_в$ – строк корисного використання обладнання, програмних засобів, приміщень тощо, років.

$A_{обл} = (33000{,}00 \cdot 3) / (5 \cdot 12) = 1650грн.$

Проведені розрахунки зведемо до таблиці.

Таблиця 5.9 – Амортизаційні відрахування по кожному виду обладнання

Найменування обладнання	Балансова вартість, грн	Строк корисного використання, років	Термін використання обладнання, місяців	Амортизаційні відрахування, грн
Персональний компютер	27 500	5	3	1375,00
Персональний компютер	33 000	5	3	1650,00
Персональний компютер	29 700	5	3	1485,00
Робоче місце дослідника	20000	5	3	1000,00
Оргтехніка	7000	4	3	437,50
Приміщення лабораторії	250000	20	3	3125,00
ОС Windows 11	33600	3	3	2800,00

Прикладний пакет Microsoft Office 2019	16800	3	3	1400,00
				13272,50

5.2.8 Паливо та енергія для науково-виробничих цілей

Витрати на силову електроенергію (B_e) розраховуємо за формулою:

$$B_e = \sum_{i=1}^{n} \frac{W_{yi} \cdot t_i \cdot Ц_e \cdot K_{вni}}{\eta_i}, \quad (5.10)$$

де W_{yi} – встановлена потужність обладнання на визначеному етапі розробки, кВт;

t_i – тривалість роботи обладнання на етапі дослідження, год;

$Ц_e$ – вартість 1 кВт-години електроенергії, грн; (вартість електроенергії визначається за даними енергопостачальної компанії), приймемо $Ц_e$ = 7,50 грн;

$K_{вni}$ – коефіцієнт, що враховує використання потужності, $K_{вni} < 1$;

η_i – коефіцієнт корисної дії обладнання, $\eta_i < 1$.

$B_e = 0,3 \cdot 400,0 \cdot 7,50 \cdot 0,95 / 0,97 = 881,44$*грн.*

Проведені розрахунки зведемо до таблиці.

Таблиця 5.10 – Витрати на електроенергію

Найменування обладнання	Встановлена потужність, кВт	Тривалість роботи, год	Сума, грн
Персональний комп’ютер	0,3	400	881,44

Персональний комп’ютер	0,3	400	881,44
Персональний комп’ютер	0,3	400	881,44
Робоче місце дослідника	0,15	400	440,72
Оргтехніка	0,45	10	33,05
Всього			3118,11

5.2.9 Службові відрядження

До статті «Службові відрядження» дослідної роботи належать витрати на відрядження штатних працівників, працівників організацій, які працюють за договорами цивільно-правового характеру, аспірантів, зайнятих розробленням досліджень, відрядження, пов’язані з проведенням випробувань машин та приладів, а також витрати на відрядження на наукові з’їзди, конференції, наради, пов’язані з виконанням конкретних досліджень.

Витрати за статтею «Службові відрядження» розраховуємо як 20…25% від суми основної заробітної плати дослідників та робітників за формулою:

$$B_{св} = (З_о + З_р) \cdot \frac{H_{св}}{100\%}, \qquad (5.11)$$

де $H_{св}$ – норма нарахування за статтею «Службові відрядження», приймемо $Н_{св} = 20\%$.

$B_{св}$ = (147619,05+ 1684,57) · 20 / 100% = 29860,72грн.

5.2.10 Витрати на роботи, які виконують сторонні підприємства, установи і організації

Витрати за статтею «Витрати на роботи, які виконують сторонні підприємства, установи і організації» розраховуємо як 30…45% від суми основної заробітної плати дослідників та робітників за формулою:

$$B_{сп} = (З_о + З_р) \cdot \frac{Н_{сп}}{100\%}, \quad (5.12)$$

де $Н_{сп}$ – норма нарахування за статтею «Витрати на роботи, які виконують сторонні підприємства, установи і організації», приймемо $Н_{сп}$= 35%.

$B_{сп}$ = (147619,05+ 1684,57) · 35 / 100% = 52256,27грн.

5.2.11 Інші витрати

До статті «Інші витрати» належать витрати, які не знайшли відображення у зазначених статтях витрат і можуть бути віднесені безпосередньо на собівартість досліджень за прямими ознаками.

Витрати за статтею «Інші витрати» розраховуємо як 50…100% від суми основної заробітної плати дослідників та робітників за формулою:

$$I_в = (З_о + З_р) \cdot \frac{Н_{ів}}{100\%}, \quad (5.13)$$

де $Н_{ів}$ – норма нарахування за статтею «Інші витрати», приймемо $Н_{ів}$ = 50%.

$I_в$ = (147619,05+ 1684,57) · 50 / 100% = 74651,81грн.

5.2.12 Накладні (загальновиробничі) витрати

До статті «Накладні (загальновиробничі) витрати» належать: витрати, пов'язані з управлінням організацією; витрати на винахідництво та раціоналізацію; витрати на підготовку (перепідготовку) та навчання кадрів; витрати, пов'язані з набором робочої сили; витрати на оплату послуг банків; витрати, пов'язані з освоєнням виробництва продукції; витрати на науково-технічну інформацію та рекламу та ін.

Витрати за статтею «Накладні (загальновиробничі) витрати» розраховуємо як 100…150% від суми основної заробітної плати дослідників та робітників за формулою:

$$B_{нзв} = (З_о + З_р) \cdot \frac{Н_{нзв}}{100\%}, \quad (5.14)$$

де $Н_{нзв}$ – норма нарахування за статтею «Накладні (загальновиробничі) витрати», приймемо $Н_{нзв}$ = 120%.

$B_{нзв}$ = (147619,05+ 1684,57) · 120 / 100% = 179 164,34грн.

Витрати на проведення науково-дослідної роботи розраховуємо як суму всіх попередніх статей витрат за формулою:

$$B_{заг} = З_о + З_р + З_{дод} + З_н + М + К_в + В_{спец} + В_{прг} + А_{обл} + В_е + В_{св} + В_{сп} + І_в + В_{нзв}. \quad (5.15)$$

$B_{заг}$=147619,05+1684,57+17916,43+36788,41173+913+0,00+95700+50400+ 13272,50+ 3118,11+29860,72+52256,27+74651,81+179 164,34= 703345,21 грн.

Загальні витрати *ЗВ* на завершення науково-дослідної (науково-технічної) роботи та оформлення її результатів розраховується за формулою:

$$ЗВ = \frac{B_{заг}}{\eta}, \quad (5.16)$$

де η - коефіцієнт, який характеризує етап (стадію) виконання науково-дослідної роботи, приймемо η =0,9.

$$ЗВ = 703345,21 / 0,9 = 781494,68грн.$$

5.3 Розрахунок економічної ефективності науково-технічної розробки при її можливій комерціалізації потенційним інвестором

В ринкових умовах узагальнюючим позитивним результатом, що його може отримати потенційний інвестор від можливого впровадження результатів тієї чи іншої науково-технічної розробки, є збільшення у потенційного інвестора величини чистого прибутку.

Результати дослідження передбачають комерціалізацію протягом 3-х років реалізації на ринку.

В цьому випадку майбутній економічний ефект буде формуватися на основі таких даних:

ΔN – збільшення кількості споживачів продукту, у періоди часу, що аналізуються, від покращення його певних характеристик;

1-й рік – 20000 користувачів;

2-й рік – 30000 користувачів;

3-й рік – 25000 користувачів.

N – кількість споживачів які використовували аналогічний продукт у році до впровадження результатів нової науково-технічної розробки, приймемо 100000 користувачів;

$Ц_б$ – вартість програмного продукту у році до впровадження результатів розробки, приймемо 1400 грн за річну підписку;

$\pm\Delta Ц_о$ – зміна вартості програмного продукту від впровадження результатів науково-технічної розробки, приймемо 200,00 грн.

Можливе збільшення чистого прибутку у потенційного інвестора $\Delta П_i$ для кожного із 3-х років, протягом яких очікується отримання позитивних результатів від можливого впровадження та комерціалізації науково-технічної розробки, розраховуємо за формулою:

$$\Delta П_i = (\pm\Delta Ц_о \cdot N + Ц_о \cdot \Delta N)_i \cdot \lambda \cdot \rho \cdot (1 - \frac{\vartheta}{100}), \quad (5.17)$$

де λ – коефіцієнт, який враховує сплату потенційним інвестором податку на додану вартість. У 2023 році ставка податку на додану вартість складає 20%, а коефіцієнт $\lambda = 0,8333$;

ρ – коефіцієнт, який враховує рентабельність інноваційного продукту. Приймемо ρ =30%;

ϑ – ставка податку на прибуток, який має сплачувати потенційний інвестор, у 2023 році $\vartheta = 18\%$;

Збільшення чистого прибутку 1-го року:

$\Delta П_1$ = (100000·200,00+1600·20000)·0,83·0,3·(1-0,18/100%)= 1690573,5 грн.

Збільшення чистого прибутку 2-го року:

$\Delta П_2 =$ (100000·200,00+1600·(20000+35000))·0,83·0,3·(1-0,18/100%)= 3329917,5 грн.

Збільшення чистого прибутку 3-го року:

$\Delta П_3 =$ (10000·200,00+1600·(20000+35000+25000))·0,83·0,3·(1-0,18/100%)= 5788933,5 грн.

Приведена вартість збільшення всіх чистих прибутків *ПП*, що їх може отримати потенційний інвестор від можливого впровадження та комерціалізації науково-технічної розробки:

$$ПП = \sum_{i=1}^{T} \frac{\Delta П_i}{(1+\tau)^t}, \qquad (5.18)$$

де $\Delta П_i$ – збільшення чистого прибутку у кожному з років, протягом яких виявляються результати впровадження науково-технічної розробки, грн;

T – період часу, протягом якого очікується отримання позитивних результатів від впровадження та комерціалізації науково-технічної розробки, роки;

τ – ставка дисконтування, за яку можна взяти щорічний прогнозований рівень інфляції в країні, τ =0,25;

t – період часу (в роках) від моменту початку впровадження науково-технічної розробки до моменту отримання потенційним інвестором додаткових чистих прибутків у цьому році.

ПП = 10655736/(1+0,25)^1+ 20491800/(1+0,25)^2+ 28688520/(1+0,25)^3= 36327863,04грн.

Величина початкових інвестицій PV, які потенційний інвестор має вкласти для впровадження і комерціалізації науково-технічної розробки:

$$PV = k_{інв} \cdot ЗВ, \quad (5.19)$$

де $k_{інв}$ – коефіцієнт, що враховує витрати інвестора на впровадження науково-технічної розробки та її комерціалізацію, приймаємо $k_{інв}$ =2;

$ЗВ$ – загальні витрати на проведення науково-технічної розробки та оформлення її результатів, приймаємо 781494,68грн.

$PV = k_{інв} \cdot ЗВ$ = 2· 781494,68= 1562989,364грн.

Абсолютний економічний ефект $E_{абс}$ для потенційного інвестора від можливого впровадження та комерціалізації науково-технічної розробки становитиме:

$$E_{абс} = ПП - PV \quad (5.20)$$

де $ПП$ – приведена вартість зростання всіх чистих прибутків від можливого впровадження та комерціалізації науково-технічної розробки, 36327863,04грн;

PV – теперішня вартість початкових інвестицій, 1562989,364 грн.

$E_{абс} = ПП - PV$ = 36327863,04- 1562989,364= 34764873,68грн.

Внутрішня економічна дохідність інвестицій $E_в$, які можуть бути вкладені потенційним інвестором у впровадження та комерціалізацію науково-технічної розробки:

$$E_в = \sqrt[Tж]{1 + \frac{E_{абс}}{PV}} - 1, \quad (5.21)$$

де $E_{абс}$ – абсолютний економічний ефект вкладених інвестицій, 34764873,68грн;

PV – теперішня вартість початкових інвестицій, 1562989,364 грн;

$T_ж$ – життєвий цикл науково-технічної розробки, тобто час від початку її розробки до закінчення отримування позитивних результатів від її впровадження, 3 роки.

$$E_в = \sqrt[Tж]{1 + \frac{E_{абс}}{PV}} - 1 = (1+34764873,68/\ 1562989,364)^{1/3} = 1,85.$$

Мінімальна внутрішня економічна дохідність вкладених інвестицій $\tau_{мін}$:

$$\tau_{мін} = d + f, \quad (5.22)$$

де d – середньозважена ставка за депозитними операціями в комерційних банках; в 2023 році в Україні d =0,1;

f – показник, що характеризує ризикованість вкладення інвестицій, приймемо 0,25.

$\tau_{мін}$ = 0,11+0,25 = 0,36 < 1,85 свідчить про те, що внутрішня економічна дохідність інвестицій $E_в$, які можуть бути вкладені потенційним інвестором у впровадження та комерціалізацію науково-технічної розробки вища мінімальної внутрішньої дохідності. Тобто інвестувати в науково-дослідну роботу за темою «Розробка методу та програмного засобу для навігації із доповненою реальністю» доцільно.

Період окупності інвестицій $T_{ок}$ які можуть бути вкладені потенційним інвестором у впровадження та комерціалізацію науково-технічної розробки:

$$T_{ок} = \frac{1}{E_в}, \tag{5.23}$$

де $E_в$ – внутрішня економічна дохідність вкладених інвестицій.

$T_{ок}$ = 1 / 1,85 = 0,54 року.

$T_{ок} <$ 3-х років, що свідчить про комерційну привабливість науково-технічної розробки і може спонукати потенційного інвестора профінансувати впровадження даної розробки та виведення її на ринок.

5.4 Висновки

Згідно проведених досліджень рівень комерційного потенціалу розробки за темою «Розробка методу та програмного засобу для навігації із доповненою реальністю» становить 40,3 бали, що, свідчить про комерційну важливість проведення даних досліджень (рівень комерційного потенціалу вище середнього).

Також термін окупності становить 0,54 р., що менше 3-х років, що свідчить про комерційну привабливість науково-технічної розробки і може спонукати потенційного інвестора профінансувати впровадження даної розробки та виведення її на ринок.

Отже можна зробити висновок про доцільність проведення науково-дослідної роботи за темою «Розробка методу та програмного засобу для навігації із доповненою реальністю».

ВИСНОВОК

Додатки з доповненою реальністю дозволяють нам бачити світ по-новому. Вони додають віртуальні об'єкти до реального світу, що може бути використано для різних цілей. У навчанні AR-додатки можуть використовуватися для створення інтерактивних навчальних матеріалів, які допомагають студентам краще засвоювати інформацію. У сфері розваг AR-додатки можуть використовуватися для створення захоплюючих ігрових experiences, а також для надання додаткової інформації про реальні об'єкти та місця. У медицині AR-додатки можуть використовуватися для навчання студентів-медиків, а також для надання допомоги лікарям у діагностиці та лікуванні пацієнтів. Зростаюча популярність AR-додатків свідчить про те, що ця технологія має великий потенціал для зміни нашого життя. Розширена реальність може зробити світ більш інформативним, захоплюючим і корисним.

У першому розділі МКР було проаналізовано сучасний стан предметної області, визначено актуальність та важливість дослідження технології доповненої реальності. Проведено порівняльний аналіз схожих додатків для навігації, виділено їх переваги та недоліки. Виконано постановку завдань для дослідження та розробки методу для навігації в доповненій реальності.

У другому розділі було виконано опис методів та алгоритмів для навігації в доповненій реальності, удосконалено метод навігації в доповненій реальності шляхом використання фільтрів Калмана та Маджвіка, а також інтеграцією системи VPS що підвищує точність позиціонування та загальну чіткість навігації. Розглянуто підходи до розробки інтерфейсів користувача. Розроблено структурну схему додатку.

В третьому розділі було виконано варіантний аналіз та обґрунтування вибору методів реалізації програмного засобу. Було вирішено використати технології нативної розробки мобільних додатків з використанням мов

програмування Java та Kotlin, інтегроване середовище розробки Android Studio та бібліотеку для роботи з доповненою реальністю BeyondAR.

В четвертому розділі МКР було розглянуто засоби та методи тестування програмного забезпечення. Було проведено тестування мобільного додатку. Розроблено інструкцію користувача.

В п’ятому розділі було обгрунтовано економічну доцільність розробки програмного засобу, обраховано потенційні витрати та можливий прибуток програмного засобу.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Augmented Reality (AR) Defined, With Examples and Uses [Електронний ресурс]. – Режим доступу: https://www.investopedia.com/terms/a/augmented-reality.asp

2. GPS [Електронний ресурс]. – Режим доступу: https://www.faa.gov/about/office_org/headquarters_offices/ato/service_units/techops/navservices/gnss/gps/howitworks

3. Google Maps [Електронний ресурс]. – Режим доступу: https://www.google.com.ua/maps/

4. Apple Maps [Електронний ресурс]. – Режим доступу: https://apps.apple.com/us/app/apple-maps/id915056765

5. ARCity [Електронний ресурс]. – Режим доступу: https://www.blippar.com/blog/2017/11/06/welcome-ar-city-future-maps-and-navigation

6. Why Augmented Reality Is One Of The Most Promising Experimental Technologies Of This Decade https://www.forbes.com/sites/forbestechcouncil/2023/02/06/why-augmented-reality-is-one-of-the-most-pro

7. Augmented Reality Development Platforms [Електронний ресурс] – Режим доступу до ресурсу: https://www.trustradius.com/augmented-reality-development.

8. Raul Mur-Artal, J. M. M. Montiel ´ , Member, IEEE, and Juan D. Tardos´ , Member, IEEE. ORB-SLAM: A Versatile and Accurate Monocular SLAM System – 2015 – 1163 с.

9. Marker-based Augmented Reality using OpenCV [Електронний ресурс] – Режим доступу до ресурсу: https://medium.com/p/b851b82be4dc

10. Dikshant Jopat , Krutarth Makwana , Hardik Dhanmeher , Joyce Lemos. Indoor Navigation System Using Augmented Reality – 2020 – 18 с.

11. The Complete Guide on Visual Positioning System [Електронний ресурс] – Режим доступу до ресурсу: https://www.techfunnel.com/information-technology/visual-positioning-system/

12. Madgwick Orientation Filter [Електронний ресурс] – Режим доступу до ресурсу: https://ahrs.readthedocs.io/en/latest/filters/madgwick.html

13. Felix Govaers. Introduction and Implementations of the Kalman Filter – 2019 – 128 с.

14. Grady Booch. The Unified Modeling Language User Guide – 2005. – 475 с.

15. ARCore and Arkit, What is under the hood: SLAM (Part 2) [Електронний ресурс] – Режим доступу до ресурсу: https://medium.com/@rabimba/arcore-and-arkit-what-is-under-the-hood-slam-part-2-5a5271d30449

16. Swift: Revolutionizing iOS App Development [Електронний ресурс] – Режим доступу до ресурсу: https://levelup.gitconnected.com/swift-revolutionizing-ios-app-development-a837213297e8

17. Android Studio Platforms [Електронний ресурс] – Режим доступу до ресурсу: https://developer.android.com/studio

18. A Detailed Comparison of Java vs Kotlin for Android Development, Performance Enhancement, and Linguistic Features [Електронний ресурс] – Режим доступу до ресурсу: https://medium.com/stackademic/an-detailed-comparison-of-java-vs-kotlin-for-android-development-performance-enhancement-and-73f2a74188a8

19. How to Build an Augmented Reality GPS App [Електронний ресурс] – Режим доступу до ресурсу: https://www.biztechcs.com/augmented-reality-gps-app/

20. Методичні вказівки до виконання економічної частини магістерських кваліфікаційних робіт / Уклад. : В. О. Козловський, О. Й. Лесько, В. В. Кавецький. – Вінниця : ВНТУ, 2021. – 42 с.

21. Кавецький В. В. Економічне обґрунтування інноваційних рішень: практикум / В. В. Кавецький, В. О. Козловський, І. В. Причепа – Вінниця : ВНТУ, 2016. – 113 с.

ДОДАТКИ

Додаток А. Технічне завдання

Міністерство освіти і науки України
Вінницький національний технічний університет
Факультет інформаційних технологій та комп'ютерної інженерії

ЗАТВЕРДЖУЮ
д.т.н., проф. О. Н. Романюк
" 19 " вересня 2023 р.

Технічне завдання

на магістерську кваліфікаційну роботу

«Розробка методу та програмного засобу для навігації із доповненою реальністю»

за спеціальністю 121 – Інженерія програмного забезпечення

Керівник магістерської кваліфікаційної роботи:
к.т.н., доц. каф. Бабюк Н.П.
" 20 " вересня 2023 р.

Виконав: студент гр. ЗПІ-22м Кагальняк Р.Ю.
" 20 " вересня 2023 р.

Вінниця – 2023 рік

1. Найменування та галузь застосування

Магістерська кваліфікаційна робота: «Розробка методу та програмного засобу для навігації із доповненою реальністю».

Галузь застосування – користувацький мобільний додаток, персональна навігаційна система.

2. Підстава для розробки.

Підставою для виконання магістерської кваліфікаційної роботи (МКР) є індивідуальне завдання на МКР та наказ № 247 від «18» вересня 2023 року ректора по ВНТУ про закріплення тем МКР.

3. Мета та призначення розробки.

Метою магістерської роботи є спрощення процесу навігації, завдяки розробці навігатора з доповненою реальністю.

Призначення роботи – надання послуг навігації та побудови маршрутів.

4. Вихідні дані для проведення НДР

1. Why Every Organization Needs an Augmented Reality Strategy [Електронний ресурс] – Режим доступу до ресурсу: https://hbr.org/2017/11/why-every-organization-needs-an-augmented-reality-strategydas
2. Augmented Reality – The Past, The Present and The Future [Електронний ресурс] – Режим доступу до ресурсу: https://www.interaction-design.org/literature/article/augmented-reality-the-past-the-present-and-the-future
3. Why Augmented Reality Is One Of The Most Promising Experimental Technologies Of This Decade https://www.forbes.com/sites/forbestechcouncil/2023/02/06/why-augmented-reality-is-one-of-the-most-promising-experimental-technologies-of-this-decade/?sh=4366e4fd3c85

5. Технічні вимоги

МКР повинна задовольняти такі вимоги:

– проведений аналіз сучасного стану досліджуваної предметної області;

– проведення порівняльного аналізу аналогів;

– удосконалення методу об’єктної доповненої реальності, розробка алгоритмів роботи програмного засобу;

– реалізація програмного засобу для об’єктної доповненої реальності;

– тестування створеного програмного засобу;

– економічне обґрунтування доцільності наукового дослідження.

6. Конструктивні вимоги.

Конструкція пристрою повинна відповідати естетичним та ергономічним вимогам, повинна бути зручною в обслуговуванні та керуванні.

Графічна та текстова документація повинна відповідати діючим стандартам України.

7. Перелік технічної документації, що пред'являється по закінченню робіт:

- пояснювальна записка до МКР;
- технічне завдання;
- лістинги програми.

8. Вимоги до рівня уніфікації та стандартизації

При розробці програмних засобів слід дотримуватися уніфікації і ДСТУ.

9. Стадії та етапи розробки.

№	Назва етапів виконання магістерської роботи	Строк виконання етапів роботи	Примітка
1	Постановка задачі роботи	20.09.23 25.09.23	вик
2	Огляд існуючих методів та програмних засобів	25.09.23 01.10.23	вик
3	Аналіз існуючих методів навігації з доповненою рельністю	01.10.23 12.10.23	вик
4	Розробка методу навігації з доповненою рельністю	13.10.23 22.10.23	вик
5	Розробка інтерфейсу та тестування програми	23.10.23 05.11.23	вик
6	Підготовка матеріалів та опис розробки програмного засобу	05.11.23 10.11.23	вик
7	Розрахунок економічної частини роботи	20.11.23 01.12.23	вик
8	Оформлення пояснювальної записки та ілюстративного матеріалу	01.12.23 04.12.23	вик

10. Порядок контролю та прийняття.

Виконання етапів магістерської кваліфікаційної роботи контролюється керівником згідно з графіком виконання роботи. Прийняття магістерської кваліфікаційної роботи здійснюється ДЕК, затвердженою зав. Кафедрою згідно з графіком.

Додаток Б. Протокол перевірки роботи

ПРОТОКОЛ ПЕРЕВІРКИ НАВЧАЛЬНОЇ (КВАЛІФІКАЦІЙНОЇ) РОБОТИ

Назва роботи: Розробка методу та програмного засобу для навігації із доповненою реальністю

Тип роботи: магістерська кваліфікаційна робота

Підрозділ : кафедра програмного забезпечення, ФІТКІ, ЗПІ – 22м

Unicheck	
Оригінальність	**94,5%**
Схожість	5,5%

Науковий керівник: к.т.н., доцент кафедри ПЗ Бабюк Н. П.

Аналіз звіту подібності

■ **Запозичення, виявлені у роботі, оформлені коректно і не містять ознак плагіату.**

□ Виявлені у роботі запозичення не мають ознак плагіату, але їх надмірна кількість викликає сумніви щодо цінності роботи і відсутності самостійності її автора. Роботу направити на доопрацювання.

□ Виявлені у роботі запозичення є недобросовісними і мають ознаки плагіату та/або в ній містяться навмисні спотворення тексту, що вказують на спроби приховування недобросовісних запозичень.

Особа, відповідальна за перевірку _____ Черноволик Г. О.

Опис прийнятого рішення: **допустити до захисту**

Ознайомлені з повним звітом подібності, який був згенерований системою Unicheck

Автор роботи _____ Кагальняк Р. Ю.

Керівник роботи _____ Бабюк Н. П.

Додаток В. Лістинг програми

```
package com.kahalniak.bakalavr.arnavigation.ar;


import java.util.ArrayList;
import java.util.List;
import java.util.concurrent.BlockingQueue;
import java.util.concurrent.LinkedBlockingQueue;
import java.util.concurrent.ThreadPoolExecutor;
import java.util.concurrent.TimeUnit;

import android.app.Activity;
import android.content.Context;
import android.content.pm.PackageManager;
import android.hardware.Camera;
import android.hardware.SensorManager;
import android.os.Bundle;
import androidx.fragment.app.Fragment;
import android.util.Log;
import android.view.LayoutInflater;
import android.view.MotionEvent;
import android.view.View;
import android.view.ViewGroup;
import android.widget.FrameLayout.LayoutParams;
import android.widget.RelativeLayout;
import android.widget.TextView;
import android.view.View.OnClickListener;
```

```
import android.view.View.OnTouchListener;

import com.beyondar.android.opengl.renderer.ARRenderer.FpsUpdatable;
import com.beyondar.android.sensor.BeyondarSensorManager;
import com.beyondar.android.util.math.geom.Ray;
import com.beyondar.android.view.OnClickBeyondarObjectListener;
import com.beyondar.android.world.BeyondarObject;
import com.beyondar.android.world.World;

public class ArFragmentSupport extends Fragment implements
FpsUpdatable,OnClickListener,
        OnTouchListener{

    private static final int CORE_POOL_SIZE = 1;
    private static final int MAXIMUM_POOL_SIZE = 1;
    private static final long KEEP_ALIVE_TIME = 1000; // 1000 ms

    private ArSurfaceView mBeyondarCameraView;
    private ArBeyondarGLSurfaceView mBeyondarGLSurface;
    private TextView mFpsTextView;
    private RelativeLayout mMainLayout;

    private Camera mCamera;
    private Camera.Parameters param;

    private World mWorld;
```

```
    private OnTouchBeyondarViewListenerMod mTouchListener;
    //private OnTouchBeyondarViewListener mTouchListener;
    private OnClickBeyondarObjectListener mClickListener;

    private float mLastScreenTouchX, mLastScreenTouchY;

    private ThreadPoolExecutor mThreadPool;
    private BlockingQueue<Runnable> mBlockingQueue;

    private SensorManager mSensorManager;

    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        mBlockingQueue = new LinkedBlockingQueue<Runnable>();
        mThreadPool = new ThreadPoolExecutor(CORE_POOL_SIZE,
MAXIMUM_POOL_SIZE, KEEP_ALIVE_TIME,
                TimeUnit.MILLISECONDS, mBlockingQueue);
    }

    @Override
    public void onAttach(Activity activity) {
        super.onAttach(activity);
        mSensorManager = (SensorManager)
activity.getSystemService(Context.SENSOR_SERVICE);
    }

    private void init() {
```

```
        android.view.ViewGroup.LayoutParams params = new
LayoutParams(ViewGroup.LayoutParams.MATCH_PARENT,
                ViewGroup.LayoutParams.MATCH_PARENT);

        mMainLayout = new RelativeLayout(getActivity());
        mBeyondarGLSurface = createBeyondarGLSurfaceView();
        mBeyondarGLSurface.setOnTouchListener(this);

        mBeyondarCameraView = createCameraView();

        mMainLayout.addView(mBeyondarCameraView, params);
        mMainLayout.addView(mBeyondarGLSurface, params);

        mBeyondarGLSurface.setMaxDistanceToRender(1000f);
        Log.d("ARFRAGG", "init:
MaxDistRender"+mBeyondarGLSurface.getMaxDistanceToRender());
    }

    private void checkIfSensorsAvailable() {
        PackageManager pm = getActivity().getPackageManager();
        boolean compass =
pm.hasSystemFeature(PackageManager.FEATURE_SENSOR_COMPASS);
        boolean accelerometer =
pm.hasSystemFeature(PackageManager.FEATURE_SENSOR_ACCELEROMETER);
        if (!compass && !accelerometer) {
            throw new IllegalStateException(getClass().getName()
                    + " can not run without the compass and the acelerometer sensors.");
        } else if (!compass) {
```

```
            throw new IllegalStateException(getClass().getName() + " can not run without
the compass sensor.");
        } else if (!accelerometer) {
            throw new IllegalStateException(getClass().getName()
                    + " can not run without the acelerometer sensor.");
        }
    }

    protected ArBeyondarGLSurfaceView createBeyondarGLSurfaceView() {
        return new ArBeyondarGLSurfaceView(getActivity());
    }

    protected ArSurfaceView createCameraView() {
        mCamera=getCameraInstance();
        param=mCamera.getParameters();


if(param.getSupportedFocusModes().contains(Camera.Parameters.FOCUS_MODE_CO
NTINUOUS_PICTURE))

param.setFocusMode(Camera.Parameters.FOCUS_MODE_CONTINUOUS_PICTURE
);
        mCamera.setParameters(param);


        return new ArSurfaceView(getActivity(),mCamera);
    }

    public static Camera getCameraInstance(){
```

```
        Camera c = null;
        try {
            c = Camera.open(); // attempt to get a Camera instance
        }
        catch (Exception e){
            Log.d("Main Activity", "getCameraInstance: ERROR"+e.getMessage());
            // Camera is not available (in use or does not exist)
        }
        return c; // returns null if camera is unavailable
    }

    public ArSurfaceView getCameraView() {
        return mBeyondarCameraView;
    }

    public com.kahalniak.bakalavr.arnavigation.ar.ArBeyondarGLSurfaceView
getGLSurfaceView() {
        return mBeyondarGLSurface;
    }

    @Override
    public View onCreateView(LayoutInflater inflater, ViewGroup container, Bundle
savedInstanceState) {
        init();
        startRenderingAR();
        return mMainLayout;
    }
```

```
    @Override
    public void onResume() {
        super.onResume();
        mBeyondarCameraView.startPreviewCamera();
        mBeyondarGLSurface.onResume();
        BeyondarSensorManager.resume(mSensorManager);
        if (mWorld != null) {
            mWorld.onResume();
        }
    }

    @Override
    public void onPause() {
        super.onPause();
        mBeyondarCameraView.releaseCamera();
        mBeyondarGLSurface.onPause();
        BeyondarSensorManager.pause(mSensorManager);
        if (mWorld != null) {
            mWorld.onPause();
        }
    }

    public void setOnTouchBeyondarViewListener(OnTouchBeyondarViewListenerMod
listener) {
        mTouchListener = listener;
    }
```

```
    public void setOnClickBeyondarObjectListener(OnClickBeyondarObjectListener
listener) {
        mClickListener = listener;
        mMainLayout.setClickable(listener != null);
        mMainLayout.setOnClickListener(this);
    }

    @Override
    public boolean onTouch(View v, final MotionEvent event) {
        mLastScreenTouchX = event.getX();
        mLastScreenTouchY = event.getY();

        if (mWorld == null || mTouchListener == null || event == null) {
            return false;
        }
        mTouchListener.onTouchBeyondarView(event, mBeyondarGLSurface);
        return false;
    }

    @Override
    public void onClick(View v) {
        if (v == mMainLayout) {
            if (mClickListener == null) {
                return;
            }
            final float lastX = mLastScreenTouchX;
            final float lastY = mLastScreenTouchY;
```

```
        mThreadPool.execute(new Runnable() {
            @Override
            public void run() {
                final ArrayList<BeyondarObject> beyondarObjects = new
ArrayList<BeyondarObject>();
                mBeyondarGLSurface.getBeyondarObjectsOnScreenCoordinates(lastX,
lastY, beyondarObjects);
                if (beyondarObjects.size() == 0)
                    return;
                mBeyondarGLSurface.post(new Runnable() {
                    @Override
                    public void run() {
                        OnClickBeyondarObjectListener listener = mClickListener;
                        if (listener != null) {
                            Log.d("ArFragment", "run: ListenerSet");
                            listener.onClickBeyondarObject(beyondarObjects);
                        }
                    }
                });
            }
        });
    }
}

public World getWorld() {
    return mWorld;
}
```

```
public void setWorld(World world) {
    try {
        checkIfSensorsAvailable();
    } catch (IllegalStateException e) {
        throw e;
    }
    mWorld = world;
    mBeyondarGLSurface.setWorld(world);
}

public void setSensorDelay(int delay) {
    mBeyondarGLSurface.setSensorDelay(delay);
}

public int getSensorDelay() {
    return mBeyondarGLSurface.getSensorDelay();
}

public void setFpsUpdatable(FpsUpdatable fpsUpdatable) {
    mBeyondarGLSurface.setFpsUpdatable(fpsUpdatable);
}

public void stopRenderingAR() {
    mBeyondarGLSurface.setVisibility(View.INVISIBLE);
}
```

```
    public void startRenderingAR() {
        mBeyondarGLSurface.setVisibility(View.VISIBLE);
    }

    public List<BeyondarObject> getBeyondarObjectsOnScreenCoordinates(float x, float
y) {
        ArrayList<BeyondarObject> beyondarObjects = new
ArrayList<BeyondarObject>();
        mBeyondarGLSurface.getBeyondarObjectsOnScreenCoordinates(x, y,
beyondarObjects);
        return beyondarObjects;
    }

    public void getBeyondarObjectsOnScreenCoordinates(float x, float y,
                                        ArrayList<BeyondarObject> beyondarObjects) {
        mBeyondarGLSurface.getBeyondarObjectsOnScreenCoordinates(x, y,
beyondarObjects);
    }

    public void getBeyondarObjectsOnScreenCoordinates(float x, float y,
                                        ArrayList<BeyondarObject> beyondarObjects, Ray
ray) {
        mBeyondarGLSurface.getBeyondarObjectsOnScreenCoordinates(x, y,
beyondarObjects, ray);

    }

    public void setPullCloserDistance(float maxDistanceSize) {
```

```
    mBeyondarGLSurface.setPullCloserDistance(maxDistanceSize);
}

public float getPullCloserDistance() {
    return mBeyondarGLSurface.getPullCloserDistance();
}

public void setPushAwayDistance(float minDistanceSize) {
    mBeyondarGLSurface.setPushAwayDistance(minDistanceSize);
}

public float getPushAwayDistance() {
    return mBeyondarGLSurface.getPushAwayDistance();
}

public void setMaxDistanceToRender(float meters) {
    mBeyondarGLSurface.setMaxDistanceToRender(meters);
}

public float getMaxDistanceToRender() {
    return mBeyondarGLSurface.getMaxDistanceToRender();
}

public void setDistanceFactor(float meters)
{
    mBeyondarGLSurface.setDistanceFactor(meters);
}
```

```
    public float getDistanceFactor(){
        return mBeyondarGLSurface.getDistanceFactor();
    }


    public void showFPS(boolean show) {
        if (show) {
            if (mFpsTextView == null) {
                mFpsTextView = new TextView(getActivity());
                mFpsTextView.setBackgroundResource(android.R.color.black);

mFpsTextView.setTextColor(getResources().getColor(android.R.color.white));
                LayoutParams params = new
LayoutParams(ViewGroup.LayoutParams.WRAP_CONTENT,
                        ViewGroup.LayoutParams.WRAP_CONTENT);
                mMainLayout.addView(mFpsTextView, params);
            }
            mFpsTextView.setVisibility(View.VISIBLE);
            setFpsUpdatable(this);
        } else if (mFpsTextView != null) {
            mFpsTextView.setVisibility(View.GONE);
            setFpsUpdatable(null);
        }
    }

    @Override
    public void onFpsUpdate(final float fps) {
```

```
    if (mFpsTextView != null) {
        mFpsTextView.post(new Runnable() {
            @Override
            public void run() {
                mFpsTextView.setText("fps: " + fps);
            }
        });
    }
}

public void setBeyondarViewAdapter(BeyondarViewAdapter adapter) {
    mBeyondarGLSurface.setBeyondarViewAdapter(adapter, mMainLayout);
}


public void forceFillBeyondarObjectPositionsOnRendering(boolean fill) {
    mBeyondarGLSurface.forceFillBeyondarObjectPositionsOnRendering(fill);
}

public void fillBeyondarObjectPositions(BeyondarObject beyondarObject) {
    mBeyondarGLSurface.fillBeyondarObjectPositions(beyondarObject);
}

@Deprecated
public void setMaxFarDistance(float maxDistanceSize) {
    setPullCloserDistance(maxDistanceSize);
}
```

```
    @Deprecated
    public float getMaxDistanceSize() {
        return getPullCloserDistance();
    }

    @Deprecated
    public void setMinFarDistanceSize(float minDistanceSize) {
        setPushAwayDistance(minDistanceSize);
    }

    @Deprecated
    public float getMinDistanceSize() {
        return getPushAwayDistance();
    }
}
```

```
package com.kahalniak.bakalavr.arnavigation.ar;

import android.location.Location;

import com.beyondar.android.plugin.BeyondarObjectPlugin;
import com.beyondar.android.plugin.GeoObjectPlugin;
import com.beyondar.android.util.math.Distance;
import com.beyondar.android.world.BeyondarObject;
import com.beyondar.android.world.GeoObject;
```

```
import java.util.Iterator;

public class ArObject extends BeyondarObject {

    private double mLongitude;
    private double mLatitude;
    private double mAltitude;
    private String placeId;

    public ArObject(long id) {
        super(id);
        this.setVisible(true);
    }

    public ArObject() {
        this.setVisible(true);
    }

    public void setPlaceId(String placeId){
        this.placeId=placeId;
    }

    public String getPlaceId(){
            return placeId;
    }
```

```
    public void setGeoPosition(double latitude, double longitude) {
        this.setGeoPosition(latitude, longitude, this.mAltitude);
    }

    public void setGeoPosition(double latitude, double longitude, double altitude) {
        this.mLatitude = latitude;
        this.mLongitude = longitude;
        this.mAltitude = altitude;
        Object var7 = this.lockPlugins;
        synchronized(this.lockPlugins) {
            Iterator var9 = this.plugins.iterator();

            while(var9.hasNext()) {
                BeyondarObjectPlugin plugin = (BeyondarObjectPlugin)var9.next();
                if(plugin instanceof GeoObjectPlugin) {
                    ((GeoObjectPlugin)plugin).onGeoPositionChanged(latitude, longitude,
altitude);
                }
            }

        }
    }

    public double getLongitude() {
        return this.mLongitude;
    }
```

```
    public double getAltitude() {
        return this.mAltitude;
    }

    public double getLatitude() {
        return this.mLatitude;
    }

    public void setLocation(Location location) {
        if(location != null) {
            this.setGeoPosition(location.getLatitude(), location.getLongitude());
        }
    }

    public double calculateDistanceMeters(GeoObject geo) {
        return this.calculateDistanceMeters(geo.getLongitude(), geo.getLatitude());
    }

    public double calculateDistanceMeters(double longitude, double latitude) {
        return Distance.calculateDistanceMeters(this.getLongitude(), this.getLatitude(),
longitude, latitude);
    }

}
```

Додаток Г. Графічна частина

ГРАФІЧНА ЧАСТИНА

РОЗРОБКА МЕТОДУ ТА ПРОГРАМНОГО ЗАСОБУ ДЛЯ НАВІГАЦІЇ ІЗ ДОПОВНЕНОЮ РЕАЛЬНІСТЮ

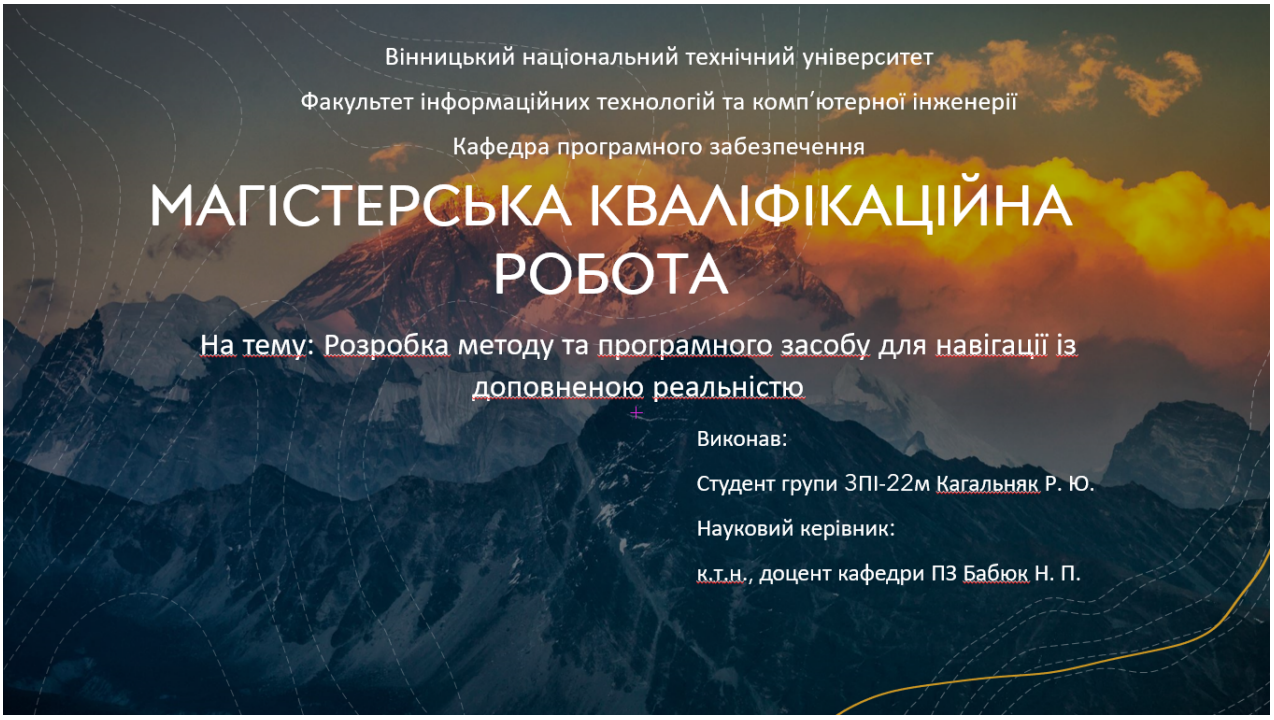

Рисунок Г.1 – Титульний слайд презентації

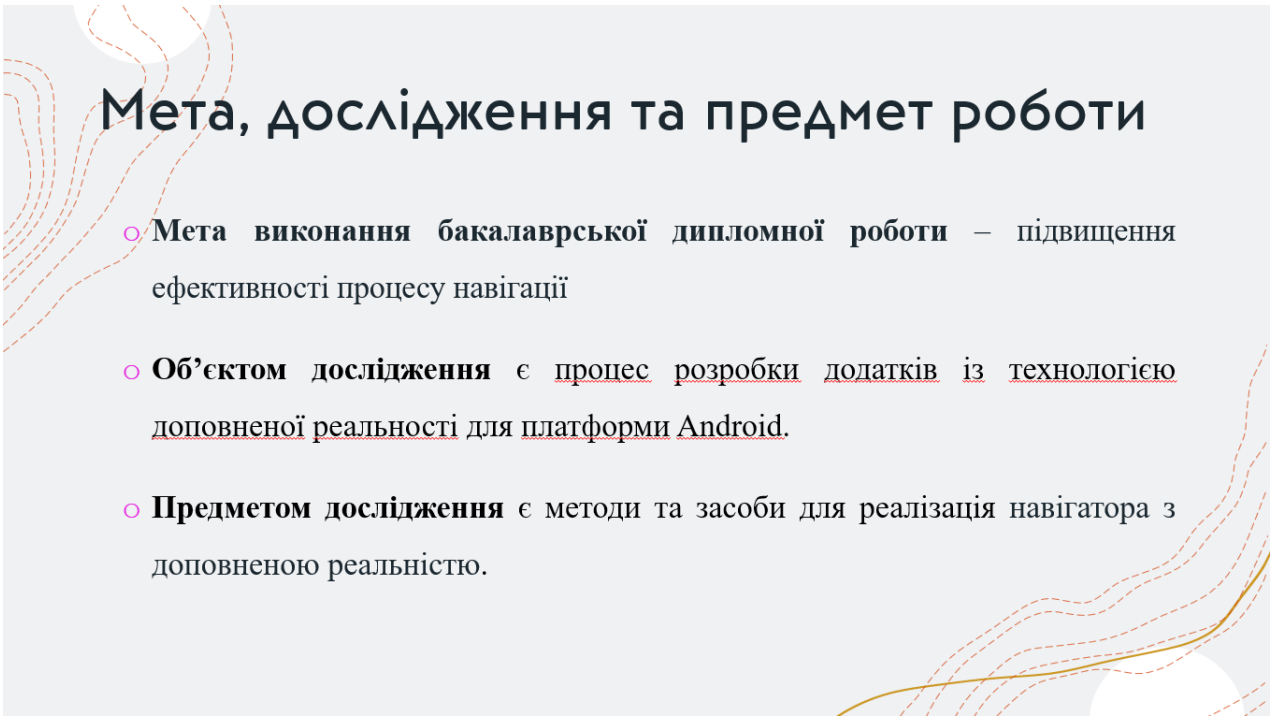

Рисунок Г.2 – Мета, дослідження та предмет роботи

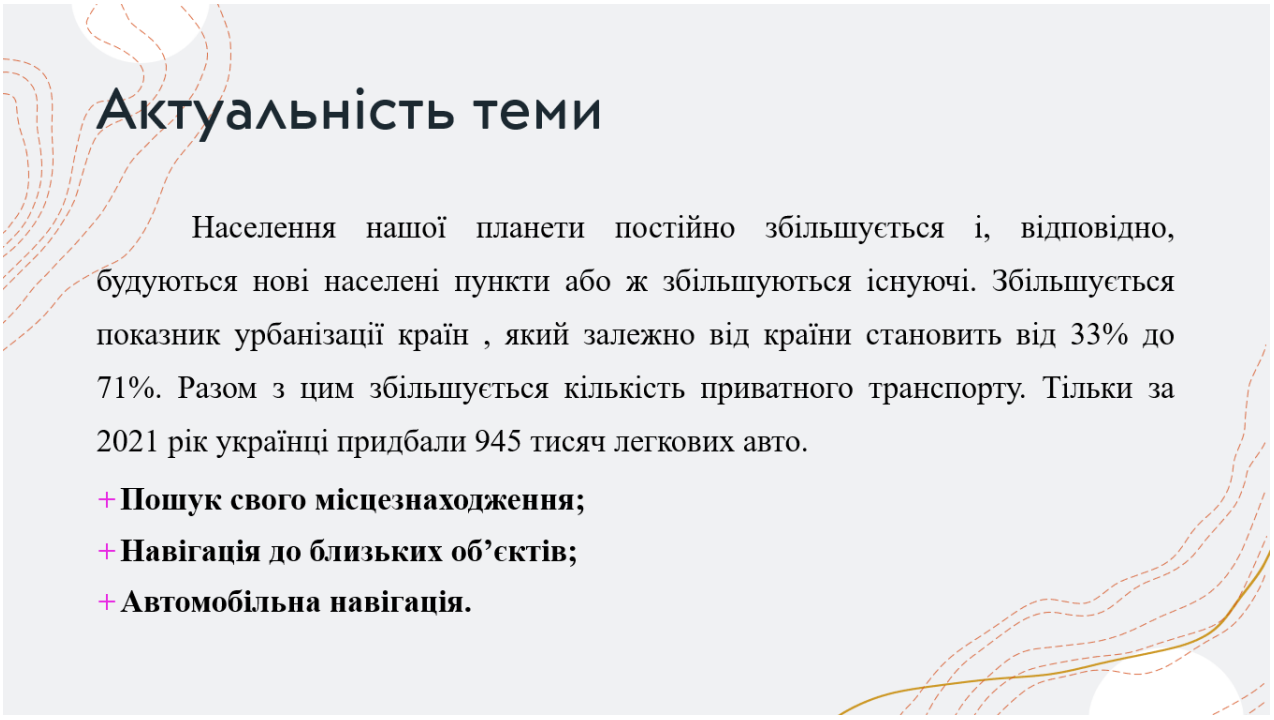

Рисунок Г.3 – Актуальність роботи

Рисунок Г.4 – Новизна та практична цінність роботи

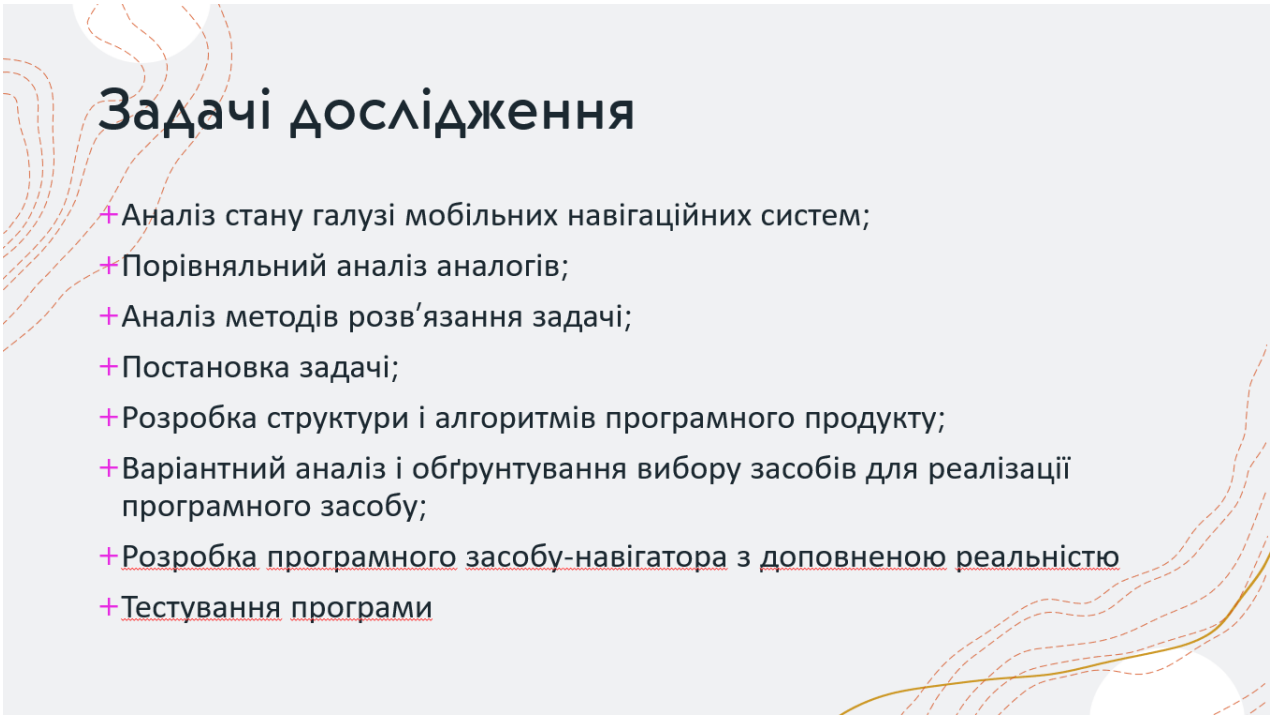

Рисунок Г.5 – Задачі дослідження

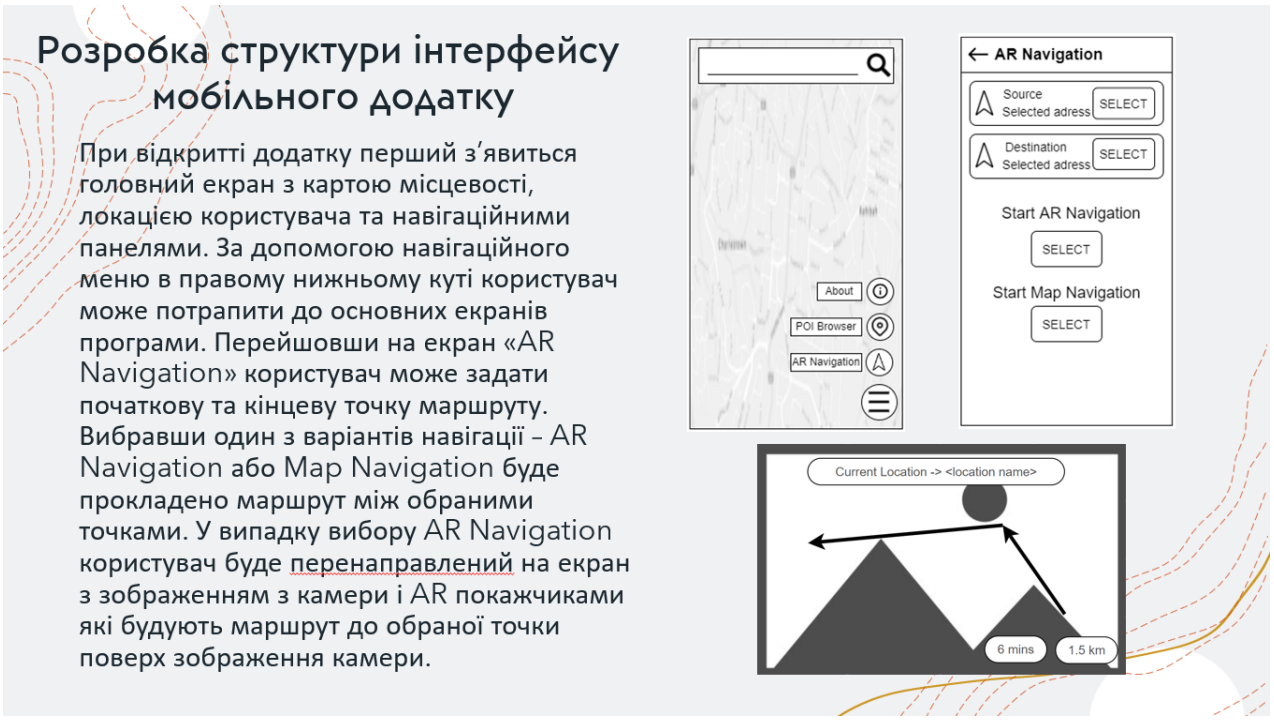

Рисунок Г.6 – Розробка структури інтерфейсу мобільного додатку

Рисунок Г.7 – Блок схема алгоритму створення і нанесення навігаційних вказівників

Рисунок Г.8 – Опис фільтру Калмана

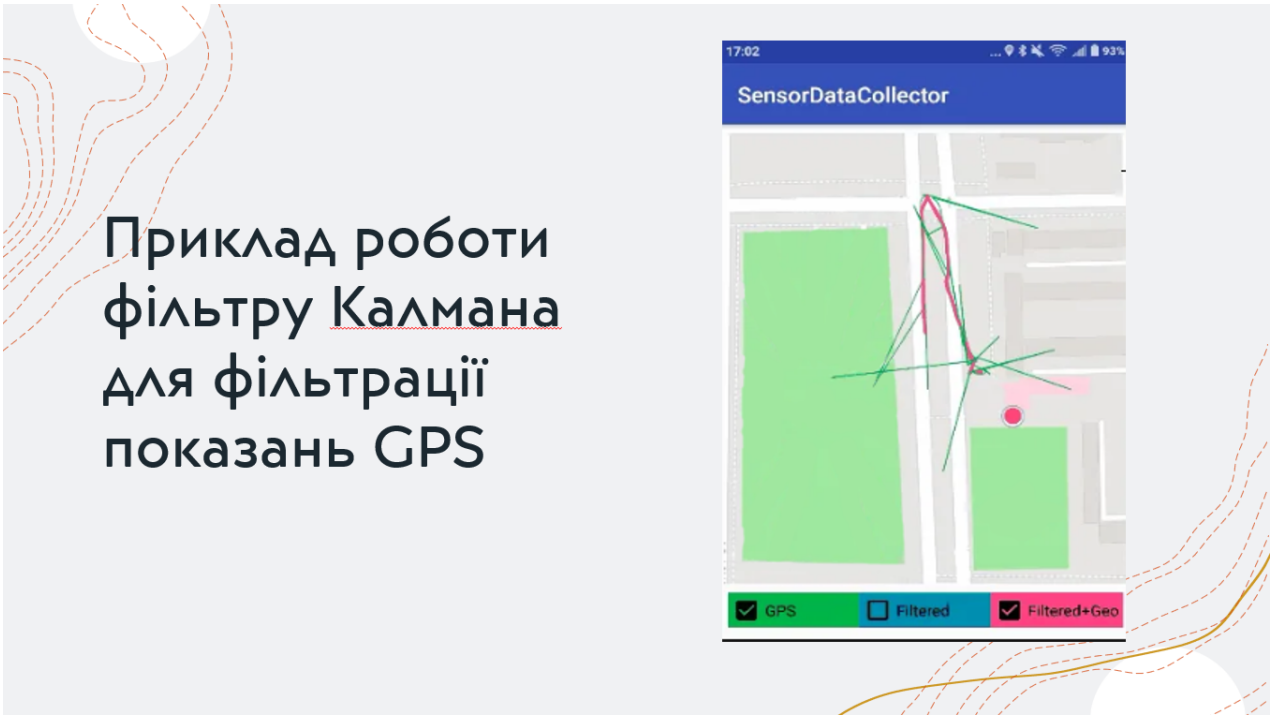

Рисунок Г.9 – Приклад роботи фільтру Калмана

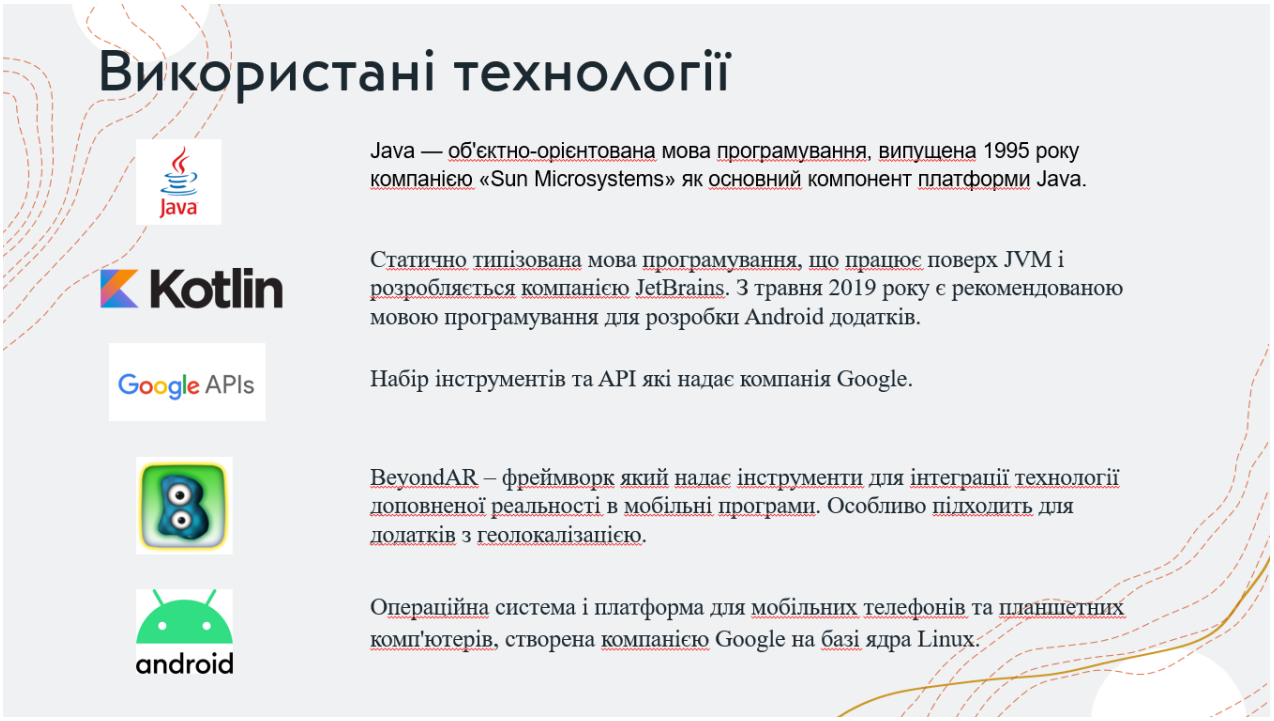

Рисунок Г.10 – Використані технології

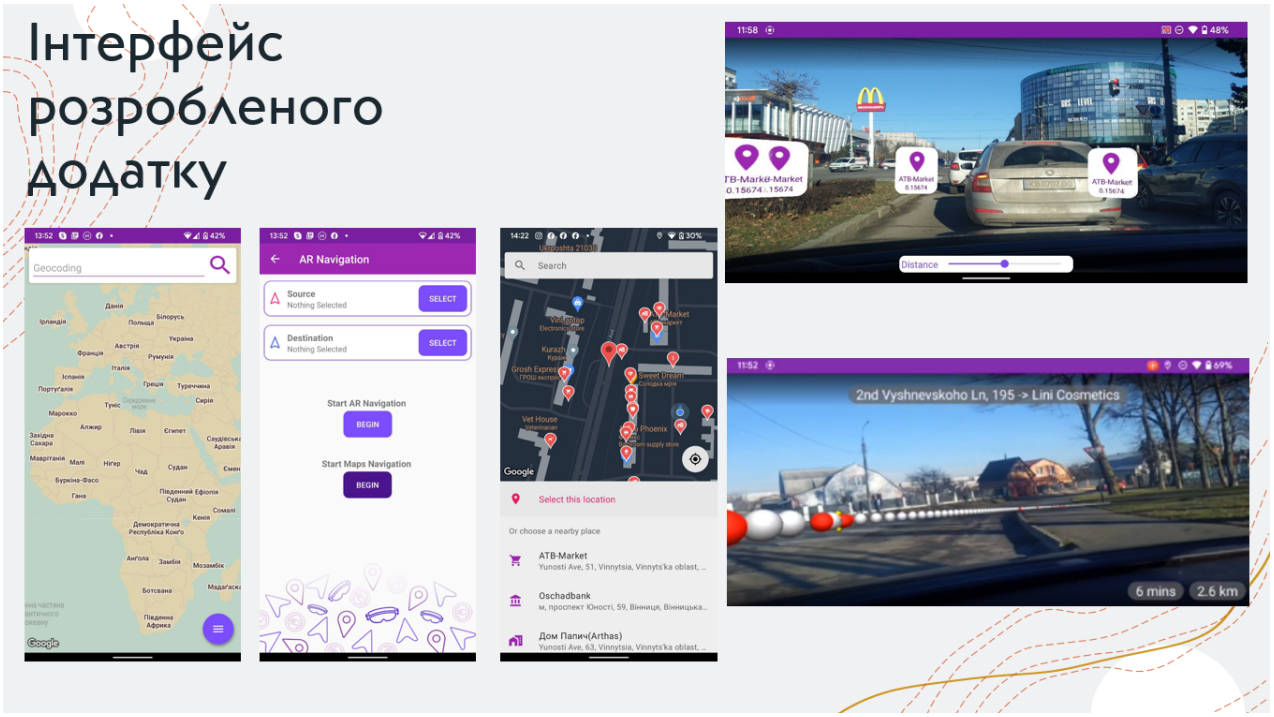

Рисунок Г.11 – Інтерфейс додатку

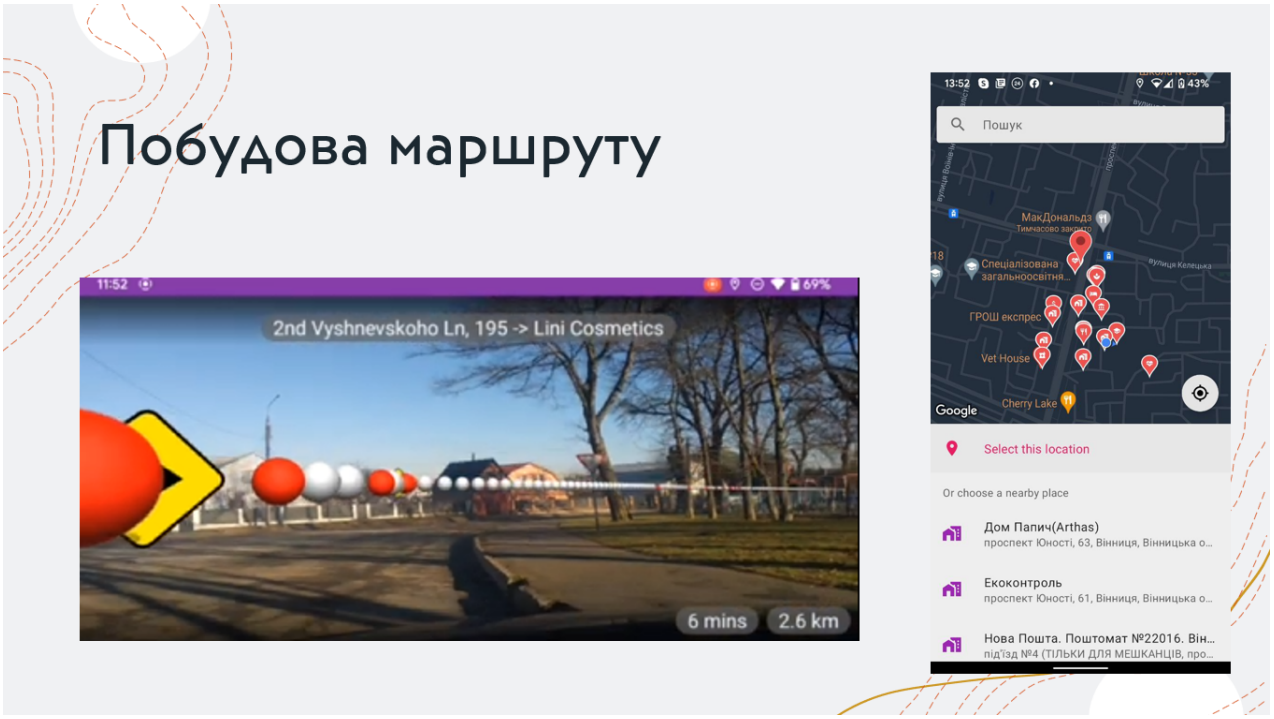

Рисунок Г.12 – Побудова маршруту

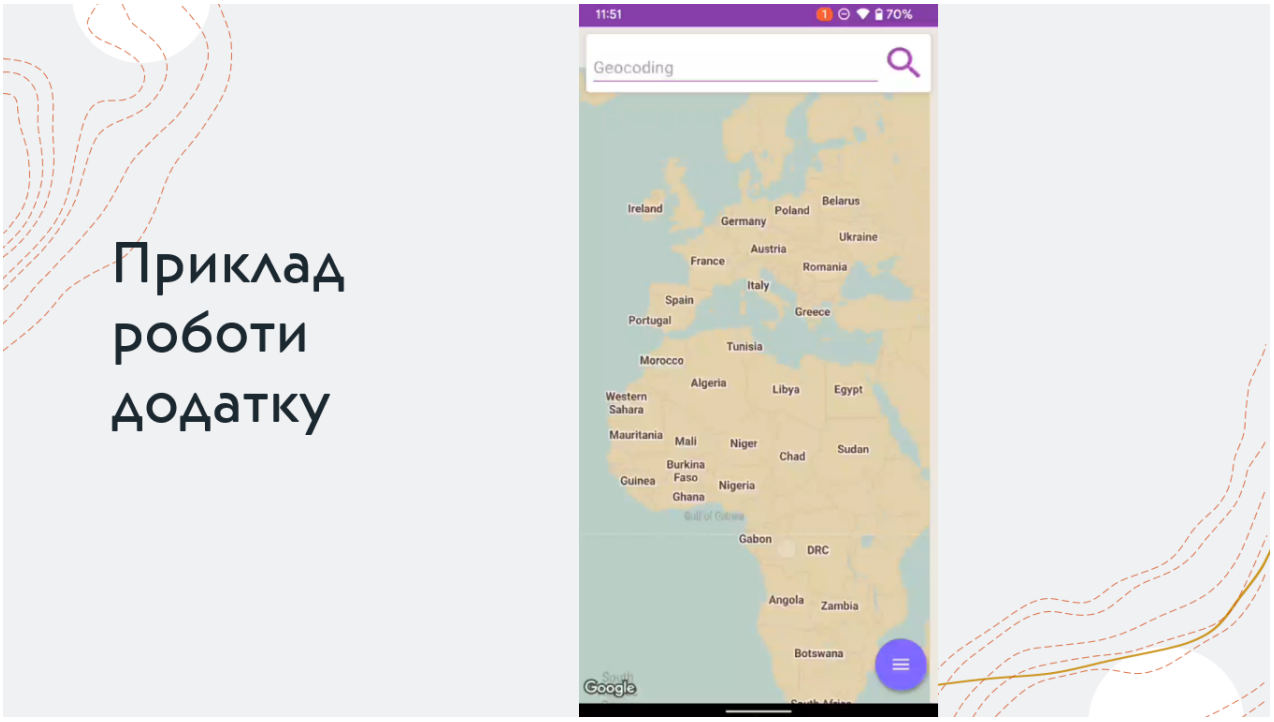

Рисунок Г.13 – Приклад роботи додатку

Рисунок Г.14 – Апробація та публікації роботи

Рисунок Г.15 – Фінальний слайд презентації