КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

ІМЕНІ ТАРАСА ШЕВЧЕНКА

Факультет комп'ютерних наук та кібернетики

Кафедра теорії та технології програмування

Кваліфікаційна робота

на здобуття ступеня бакалавра

за спеціальністю 122 Комп'ютерні науки

на тему:

МОБІЛЬНИЙ ЗАСТОСУНОК ВІДСТЕЖЕННЯ МАРШРУТІВ ПРОГУЛЯНОК

Виконала студентка 4 курсу
Олена БОНДАРЄВА _____

Науковий керівник:
доцент, к.пед.н.
Наталія РУСІНА _____

Засвідчую, що в цій роботі немає запозичень з праць інших авторів без відповідних посилань.
Студентка_____

Роботу розглянуто й допущено до захисту на засіданні кафедри теорії та технології програмування
«05» червня 2023 р., протокол №18

Завідувач кафедри
Микола НІКІТЧЕНКО _____

Київ – 2023

РЕФЕРАТ

Обсяг роботи 50 сторінок, 13 ілюстраціїй, 23 джерел посилань.

МОБІЛЬНИЙ ЗАСТОСУНОК, МАРШРУТИ ПРОГУЛЯНОК, KOTLIN, АРХІТЕКТУРА MVVM, ОБ'ЄКТНО-РЕЛЯЦІЙНОГО ВІДОБРАЖЕННЯ, GOOGLE MAPS SDK.

Об'єктом розробки є процес аналізу маршрутів для прогулянок на свіжому повітрі.

Предметом роботи є мобільні застосунки орієнтовані на людей, які займаються активним відпочинком.

Метою роботи є розробка мобільного застосунку “StrollTrack” для піших прогулянок.

Інструменти розроблення: середовище програмування Android Studio; мова програмування Kotlin; бібліотека компонентів архітектури Android – Room, що забезпечує рівень абстракції над SQLite.

Результати роботи: проведено огляд ринку сучасних мобільних застосунків, проаналізовано потреби користувачів та розроблено мобільний застосунок для запису маршрутів пішохідних прогулянок.

ЗМІСТ

СКОРОЧЕННЯ ТА УМОВНІ ПОЗНАЧЕННЯ

API – Application Programming Interface, прикладний програмний інтерфейс

DAO – Decentralized autonomous organization, децентралізована автономна організація

GPS – Global Positioning System, система глобального позиціювання

JVM – Java Virtual Machine, віртуальна машина Java

MVVM – Model-View-ViewModel, модель-викляд-модель вигляду

ORM – Object-relational mapping, об'єктно-реляційна проєкція

SDK – Software Development Kit, комплект для розробки програмного забезпечення

SQL – Structured query language, мова структурованих запитів

XML – Extensible Markup Language, розширювана мова розмітки

ВСТУП

Оцінка сучасного стану об’єкта розробки. Мобільні застосунки для запису маршрутів прогулянок забезпечують важливу функцію в сучасному суспільстві, де активний спосіб життя та здоровий спосіб навколишнього середовища набуває все більшого значення. Ці програмні продукти надають користувачам можливість відстежувати та реєструвати свої маршрути прогулянок, а також отримувати різноманітні дані про них.

Дослідження щодо здоров'я та фізичної активності [1] підтверджують, що регулярна фізична активність, включаючи прогулянки, має безліч корисних впливів на здоров'я. Мобільні програми для запису маршрутів прогулянок стимулюють людей до активності, допомагаючи відстежити кількість кроків, відстань, тривалість прогулянки та інші параметри. Це сприяє підвищенню усвідомлення про особисту фізичну активність та може спонукати до більш активного способу життя.

Актуальність роботи та підстави для її виконання. Застосунок “StrollTrack” може бути корисним для широкого кола осіб у різних ситуаціях, наприклад для осіб, які займаються фітнесом чи активним відпочинком. Люди, які надають пріоритет спорту, зокрема фізичним вправам, можуть використовувати застосунок для відстеження дистанції, яку проходить та моніторингу свого прогресу. Він слугує цінним інструментом для підтримки мотивації, досягнення цілей у фітнесі та підтримання активного способу життя. Громадяни, які мають собак і полюбляють досліджувати разом з ними нові стежки, використовуватимуть цей застосунок для відстеження та збереження порібного шляху. Люди, які просто люблять активний відпочинок на свіжому повітра, такі як піші прогулянки, біг підтюпцем або вивчення природи, можуть скористатися функціями мапування та відстеження маршрутів у застосунку.

Мета й завдання роботи. Мета роботи полягає в розробці мобільного застосунку “StrollTrack” для піших прогулянок.

Основні завдання, що потрібно вирішити для досягнення мети кваліфікаційної роботи:

- провести аналіз існуючих аналогів на сучасному ринку застосунків;
- розробити мобільний застосунок для пішохідних прогулянок;
- протестувати застосунок;
- поглибити та закріпити навички роботи з мовою програмування Kotlin.

Об’єкт, методи й засоби розробки. Об’єктом розроблення є процес аналізу маршрутів для прогулянок на свіжому повітрі. Для розробки застосунку було використано інструментальні засоби: середовище програмування Android Studio [2]; мова програмування Kotlin [3]; бібліотека компонентів архітектури Android – Room [4], що забезпечує рівень абстракції над SQLite [5].

Можливі сфери застосування. Мобільний застосунок “StrollTrack” розрахований на користувачів, які полюбляють пішохідні прогулянки та хочуть досліджувати нові маршрути. Особливо цікавим цей застосунок буде для людей, які вигулюють домашніх тваринок. Ця програма може бути корисною як і для тих, хто хоче зберігати цікаві стежки, так і для тих, хто слідкує за своєю фізичною активністю, рахуючи пройдену відстань.

РОЗДІЛ 1. ОГЛЯД НАЯВНИХ НА РИНКУ ЗАСТОСУНКІВ

В останні роки швидко набирає популярність здоровий спосіб життя, активний відпочинок на свіжому повітрі, наприклад походи і прогулянки. Із широким використанням смартфонів мобільні застосунки стали невід'ємною частиною сучасного життя людей, тому вони є зручним інструментом для користувачів, які прагнуть відстежувати свої маршрути та зберігати для використання у майбутньому. Нижче наведено огляд декількох популярних програмних продуктів для мобільних пристроїв: «WalkTracker: Hiking Trails» [6], «My Walk Tracker» [7] і «My Track» [8].

Мобільний застосунок WalkTracer: Hiking Trails

WalkTracker: Hiking Trails – це спеціальний мобільний застосунок, що призначений для любителів активного відпочинку у вигляді піших прогулянок та походів. Він має такі функції: відстеження пройдених маршрутів, дослідження нових стежок, запис усіх даних після походів, прогулянок та ведення статистики (рисунок 1.1) [6].

Рисунок 1.1 – Фрагмент екрану WalkTracker з картою

Основні функції WalkTracker включають:

1. Відстеження маршруту: WalkTracker використовує технологію GPS для відстеження точного маршруту користувача в реальному часі. Ця функція дозволяє користувачам бачити своє поточне місцезнаходження, пройдену відстань і набір висоти під час прогулянок.

2. Імпорт пішохідних стежок: застосунок надає користувачеві можливість імпортувати нові пішохідні маршрути. Це дає можливість досліджувати більше стежок та урізноманітнює прогулянки.

3. Історія маршруту: WalkTracker зберігає історію маршруту користувача, дозволяє йому переглядати минулі прогулянки та відстежувати свій прогрес, оглянувши статистику, яка побудована на збережених даних. Ця функція дає можливість користувачам бачити покращення своєї продуктивності [6].

Зупинимось на перевагах WalkTracker.

Можливість імпорту маршрутів: WalkTracker дає можливість додавати нові пішохідні маршрути, створені іншими людьми. Імпорт нових стежок допомагає користувачеві спробувати більше цікавих шляхів.

Точне GPS-відстеження: функція GPS-відстеження WalkTracker забезпечує точне відстеження маршруту, надаючи користувачам усі дані про їх місцезнаходження в реальному часі, пройдену відстань, швидкість і набір висоти. Завдяки цій інформації користувач може контролювати свій прогрес.

Історія маршруту та відстеження прогресу: застосунок має можливість переглядати пройдені маршрути та відстежувати свій прогрес з часом за допомогою статистики.

Також доречно зазначити недоліки.

Обмеження в записах активностей: в першу чергу WalkTracker призначений для піших прогулянок і може не підійти користувачам, які займаються іншими видами активності, наприклад: біг або їзда на велосипеді.

Користувачам, які бажають більш універсальну програму для відстеження їх активності на свіжому повітрі, доведеться шукати інші застосунки.

Залежність від GPS: для точного відстеження застосунок WalkTracker використовує технологію GPS, тому на протязі усієї прогулянки йому потрібен стабільний сигнал GPS. У місцях, де поганий сигнал, або під час довготривалих походів у віддалені місця точність відстеження програми може бути погіршена.

Доступність: доступність WalkTracker може відрізнятися залежно від мобільної платформи та регіону. Для деяких пристроїв застосунок може бути недоступним або для доступу до деяких функцій треба змінити регіон. Це впливає на те, що не всі користувачі зможуть повноцінно користуватись програмо.

Платформа: застосунок має версію тільки для Android, що зменшує кількість потенційних користувачів

Застосунок My Walk Tracker

My Walk Tracker — це універсальний застосунок для мобільних пристроїв, який призначений для користувачів, які займаються різними видами діяльності, включаючи прогулянки і біг (рисунок 1.2) [7].

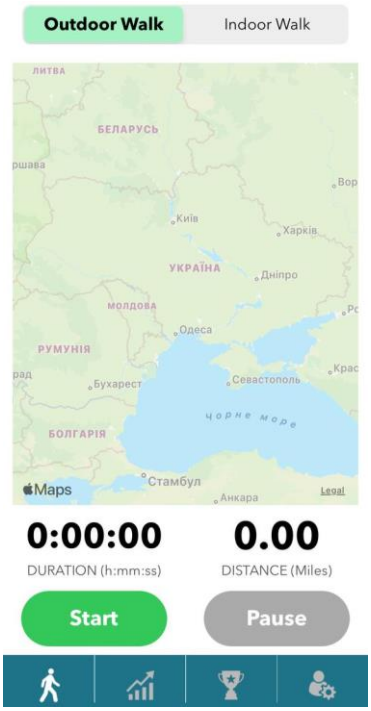

Рисунок 1.2 – Фрагмент екрану My Walk Tracker з картою

Застосунок My Walk Tracker пропонує такі функції:

1. Відстеження руху: My Walk Tracker використовує GPS і датчики руху для точного відстеження рухів користувачів під час прогулянок. Він записує важливі показники: відстань, тривалість, середня швидкість та спалені калорії.

2. Імпорт нових стежок: My Walk Tracker має можливість завантажувати нові пішохідні маршрути.

3. Історія шляхів та ведення статистики: My Walk Tracker зберігає пройдені маршрути користувача, дозволяє йому переглядати минулі прогулянки та відстежувати статистику. Ця функція дає можливість користувачам бачити прогрес.

4. Персоналізовані цілі: користувачі цього застосунка можуть встановлювати власні цілі у своєму профілі. My Walk Tracker дозволяє визначати цілі, які пов’язані з відстанню, часом або витратою калорій. Таким чином користувач має можливість відслідковувати свій прогрес.

5. Аудіопідказка: для того, щоб інформувати користувачів про їхній прогрес під час прогулянок, My Walk Tracker періодично надає аудіопідказки. Ця функція оголошує про зупинки, пройдену відстань, швидкість та іншу статистику. Це допомагає користувачам не відволікатись на перевірку цих даних на пристрої.

6. Інтеграція з застосунками для здоров'я: My Walk Tracker легко інтегрується з популярними застосунками для здоров'я та фітнесу, дозволяючи користувачам синхронізувати дані [7].

До переваги застосунку можна віднести:

- відстеження прогресу: застосунок My Walk Tracker може переглядати пройдені маршрути;
- імпорт нових маршрутів: My Walk Tracker дає можливість додавати нові пішохідні маршрути, створені іншими людьми. Імпорт нових стежок допомагає користувачеві спробувати більше цікавих шляхів;
- індивідуальне встановлення цілей: можливість встановлювати персональні цілі. Відстежуючи свій прогрес, користувачі залишаються мотивованими та зосередженими на досягненні бажаних результатів;
- звуковий зв'язок: функція звукового зворотнього зв'язку My Walk Tracker забезпечує передачу оновлених даних користувачу під час прогулянок, тим самим усуваючи необхідність перевірки програми чи пристрою;
- інтеграція з застосунками для здоров'я: інтеграція My Walk Tracker з іншими застосунками для здоров'я та фітнесу дозволяє користувачам переглядати загальний прогрес у фізичній формі;
- з'єднання з іншими пристроями: є можливість синхронізувати застосунок з Apple Watch.

Також варто зазначити недоліки:

- обмеження в записах активностей: My Walk Tracker зосереджено на відстеженні активності, яка пов’язана з ходьбою, та не має спеціальні функції для видів активності на свіжому повітрі, таких як їзда на велосипеді;
- залежність програми: функціональні можливості My Walk Tracker залежать від самої програми та часу роботи батареї пристрою. Користувачі повинні переконатися, що їх пристрій має достатній заряд акумулятора та наявність сигналу GPS, щоб застосунок працював правильно для точного відстеження;
- платформа: застосунок має версію тільки для iOS, що зменшує кількість потенційних користувачів.

1.1 Мобільний застосунок My Track

My Track — це комплексний мобільний застосунок, розроблений для відстеження різноманітних видів діяльності на свіжому повітрі, зокрема ходьби, бігу, їзди на велосипеді тощо (рисунок 1.3) [8].

Рисунок 1.3 – Фрагмент екрану My Track з картою

Застосунок пропонує такі функції:

1. Відстеження кількох дій: My Track дозволяє користувачам відстежувати різні типи дій за допомогою однієї програми. Він надає спеціальні режими для ходьби, бігу, катання на лижах чи човнах, їзди на велосипеді та інших занять на свіжому повітрі, забезпечуючи точне відстеження та індивідуальну статистику.

2. Відстеження декількох дій: My Track дозволяє користувачам відстежувати різні типи дій за допомогою однієї програми. Він надає спеціальні режими для ходьби, бігу, їзди на велосипеді та інших занять на свіжому повітрі, забезпечуючи точне відстеження та індивідуальну статистику.

3. Відстеження в реальному часі: застосунок дозволяє користувачам ділитися своїм місцем розташування та прогресом активності в реальному часі з іншими користувачами. Ця функція ідеально підійде для людей, які беруть участь у групових походах або просто хочуть ділитися своїм міцезнаходженням з друзями.

4. Історія: My Track зберігає маршрути користувача, які він пройшов разом з усіма його даними. Пізніше на цьому робиться аналіз. Користувач може переглядати свох минулі прогулянки та повторювати їх в майбутньому.

5. Аналіз продуктивності: My Track забезпечує детальний аналіз продуктивності на основі записаних даних. Користувачі можуть переглядати та порівнювати свою статистику щодо кількох видів діяльності, включаючи відстань, швидкість, висоту та, частоту серцевих скорочень, якщо пристрій підтримує.

6. Плани маршрутів: My Track пропонує заздалегідь розроблені плани для користувачів, які хочуть покращити свої успіхи в конкретній активності на свіжому повітрі або підготуватися до певних подій. Це допомагає швидше досягати поставлених цілей.

7. Імпорт та експорт маршрутів: застосунок дає можливість додавати нові шляхи для пішохідних активностей. Експорт стежок даже можливість передати маршрут іншому користувачеві.

8. Озвучка: функція озвучування My Track покращує концентрацію користувача під час прогулянох або походів, оскільки зменшує потребу в постійній перевірці пристрою. Застосунок забезпечує періодичне звукове оповіщення.

9. Групування: можливість утворювати групи та запрошувати туди друзів, щоб ділитись маршрутом. Це є дуже зручним для групових походів.

10. Сумісність з іншими пристроями: My Track легко інтегрується з різноманітними пристроями, такими як розумні годинники та фітнес-трекери [8].

Переваги:

Відстеження кількох видів діяльності: здатність My Track відстежувати різні види діяльності на свіжому повітрі. Користувачі можуть перемикатися між ходьбою, бігом, їздою на велосипеді та іншими видами діяльності без

використання декількох програм, що спрощує процес відстеження. Тому цей застосунок є досить універсальним

Відстеження в реальному часі та безпека: функція відстеження в реальному часі My Track підвищує безпеку під час активного відпочинку, оскільки вона дозволяє користувачам ділитися своїм місцезнаходженням у реальному часі з іншими. Ця функція особливо корисна для осіб, які беруть участь у групових походах, або для тих, хто хоче забезпечити свою безпеку під час самотніх прогулянок. Особливо корисно в нічний час.

Імпорт та експорт стежок: My Track має можливість імпортувати нові маршрути для прогулянок та експортувати власні пройдені стежки, щоб потім надіслати файлом своїм друзям.

Голосовий супровід: функція голосового супроводу в застосунку My Track допомагає користувачеві залишатись зосередженим на прогулянці на свіжому повітрі, але в той самий час періодично отримувати оновлення даних без використання пристрою.

Соціалізація: Користувач має можливість утворювати групи в застосунку My Track, щоб організовувати походи та ділитись маршрутом.

Аналіз продуктивності та планування: функція аналізу продуктивності My Track надає користувачам цінну інформацію у вигляді статистики про їхню діяльність, допомагаючи їм побачити їхній прогрес і визначити нові цілі. Планування маршрутів додатково підтримує користувачів у досягненні їхніх цілей.

Також варто розповісти про недоліки:

Сумісність пристрою: сумісність My Track із переносними пристроями та певними функціями може залежати від конкретної моделі пристрою та операційної системи користувача. Користувачі повинні переконатися, що їхні пристрої сумісні та відповідають необхідним вимогам для оптимального використання.

Потенційна складність: Широкий набір функцій у My Track може бути приголомшливим для користувачів, які віддають перевагу простішому та оптимізованому відстеженню. Користувачі, яким потрібна зрозумілий та мінімалістичний застоунок відстеження, можуть вважати безліч параметрів і налаштувань у My Track непотрібними або заплутаними.

Операційна система: Застосунок підтримується лише на системі Android [9].

Підсумовуючи огляд деяких застосунків, а саме: WalkTracker: Hiking Trails, My Walk Tracker та My Track, хотілось ба зазанчити, що це різні мобільні застосунки, що пропонують певні функції для відстеження маршрутів під час активного відпочинку. У той час як WalkTracker зосереджується на піших прогулянках, My Walk Tracker підтримує великий вибір піших прогулянок і робить акцент на персоналізованому встановленні цілей і звуковому відгуку. З іншого боку, My Track забезпечує відстеження кількох видів діяльності, аналіз ефективності, плани навчання, дослідження маршруту та сумісність з іншими пристроями. Кожен застосунок має свої переваги та недоліки.

РОЗДІЛ 2. ОГЛЯД ВИКОРИСТАНИХ ТЕХНОЛОГІЙ

У розділі представлено теорія огляду технологій, використаних для розробки мобільного застосунка “StrollTrack”.

2.1.Платформа Android

Застосунок “StrollTrack” розроблений для платформи Android [2], яка є однією з найбільш широко використовуваних мобільних операційних систем у світі.

Платформа Android - це операційна система з відкритим вихідним кодом, розроблена спеціально для мобільних пристроїв. Вона надає повний набір інструментів, бібліотек та API, які дозволяють розробникам створювати надійні та багатофункціональні застосунки для смартфонів, планшетів, носіїв та інших пристроїв. Нижче наведено огляд платформи Android та причин, чому вона підходить для розробки застосунку для відстежування маршрутів “StrollTrack”:

Відкритий вихідний код та широка база користувачів: Android - це платформа з відкритим вихідним кодом, що означає, що вихідний код знаходиться у вільному доступі для розробників. Ця відкритість призвела до створення великої та різноманітної спільноти розробників, що спричинило появу широкого спектру ресурсів, бібліотек та підтримки спільноти. Велика база користувачів пристроїв Android також представляє значну ринкову можливість для застосунку “StrollTrack”.

Інструменти розробки: Android Studio, офіційне інтегроване середовище розробки для Android, надає багатофункціональне середовище розробки. Воно пропонує інструменти для редагування коду, налагодження, тестування та профілювання, що полегшує розробникам створення, тестування та оптимізацію своїх застосунків.

Багатий користувацький інтерфейс: Android пропонує багатий набір компонентів інтерфейсу користувача та віджетів, які дозволяють розробникам

створювати візуально привабливі та інтерактивні користувацькі інтерфейси. Платформа підтримує різні макети, види, анімацію та стилі, що дозволяє розробникам створювати привабливі та інтуїтивно зрозумілі користувацькі інтерфейси.

Сумісність з різними пристроями та фрагментація: Android підтримує широкий спектр пристроїв з різними розмірами екранів, роздільною здатністю та апаратними можливостями. Розробники можуть оптимізувати макет і функціональність своїх застосунків, щоб адаптувати їх до різних пристроїв і забезпечити узгоджений користувацький досвід. Хоча фрагментація пристроїв може створювати проблеми, Android надає рекомендації, бібліотеки підтримки та функції адаптивного інтерфейсу для вирішення цих проблем.

Доступ до функцій пристрою: Android дозволяє розробникам отримати доступ до широкого спектра функцій і датчиків пристрою, таких як GPS, камера, акселерометр тощо. Це особливо актуально для застосунку “StrollTrack”, який вимагає доступу до сервісів визначення місцезнаходження та датчиків для запису маршрутів і відстеження переміщень користувача [9].

2.2. Мова програмування Kotlin

Kotlin — це мова програмування, вибрана для розробки застосунку “StrollTrack”. Kotlin - це сучасна мова програмування зі статичною типізацією, яка працює на віртуальній машині Java (JVM) і повністю сумісна з Java. Він був розроблений JetBrains з метою підвищення продуктивності праці розробників і зручності читання коду [3].

Рішення використати Kotlin у проекті було обумовлене кількома факторами. По-перше, Kotlin пропонує лаконічний і виразний синтаксис, що скорочує шаблонний код і робить кодову базу зручнішою у супроводі. Він також надає ряд мовних функцій, таких як нульова безпека та виведення типів, які підвищують надійність коду та знижують ймовірність помилок під час виконання.

Крім того, гарна сумісність Kotlin з існуючими кодовими базами Java дозволяє легко інтегруватися з платформами та бібліотеками Android. Він забезпечує розширену підтримку парадигм функціонального програмування та пропонує такі функції, як співпрограми для ефективного асинхронного програмування, що дозволяє розробникам писати більш ефективний та чуйний код [10].

2.3. Бібліотека Room

Room - це бібліотека компонентів архітектури Android, яка забезпечує рівень абстракції над SQLite [5], традиційним рішенням баз даних для Android. Room полегшує процес роботи з локальною базою даних, надаючи підхід об'єктно-реляційного відображення (ORM) [4].

Рішення використовувати Room у “StrollTrack” пов'язане з його численними перевагами. Room пропонує перевірку запитів під час компіляції, гарантуючи, що SQL запити перевіряються під час компіляції, а не під час виконання. Це допомагає виявляти помилки на ранньому етапі та запобігає збоям під час виконання через синтаксичні помилки SQL.

Room також надає можливості зіставлення об'єктів, дозволяючи розробникам зіставляти таблиці бази даних із класами даних Kotlin, зменшуючи обсяг стандартного коду, необхідного для операцій із базою даних. Він легко інтегрується з LiveData [12] та ViewModel [13], забезпечуючи ефективне спостереження за даними та їх оновлення у відповідь на зміни у базі даних.

Крім того, Room за замовчуванням обробляє операції з базою даних у фонових потоках, гарантуючи, що запити до бази даних і оновлення не блокують основний потік інтерфейсу користувача, тим самим підтримуючи плавний і чуйний інтерфейс користувача [11].

2.4. Платформа OpenStreetMap

OpenStreetMap - це картографічна платформа з відкритим вихідним кодом [14], яка надає безкоштовні та редаговані картографічні дані. Вона пропонує бібліотеку osmdroid для інтеграції функціональності OpenStreetMap у застосунки для Android. Опис бібліотеки та причини використання її в проекті.

Дані OpenStreetMap: OpenStreetMap надає повний і актуальний набір картографічних даних, що включає дороги, орієнтири, пам'ятки та інші географічні об'єкти. Використовуючи OpenStreetMap, проект може використовувати ці багаті дані для відображення точних і детальних карт для користувачів.

Підтримка автономних карт: Бібліотека osmdroid підтримує функціонал офлайн-карт, що дозволяє користувачам завантажувати картографічні плитки і використовувати їх без підключення до інтернету. Це особливо корисно для застосунків для запису маршрутів, оскільки користувачі можуть захотіти записувати маршрути в районах з обмеженим або відсутнім підключенням до Інтернету.

Налаштування та гнучкість: Бібліотека osmdroid пропонує високий ступінь кастомізації та гнучкості. Розробники можуть налаштовувати стиль мапи, додавати накладання, створювати власні маркери та візуалізації, щоб покращити користувацький досвід та пристосувати його до конкретних вимог програми.

Легкість та ефективність: Бібліотеку osmdroid розроблено легкою та ефективною, що забезпечує плавний рендеринг мап та оптимальну продуктивність навіть на пристроях з обмеженими ресурсами. Це має вирішальне значення для програми для запису маршрутів, оскільки вона повинна обробляти оновлення місцезнаходження в реальному часі і забезпечувати безперебійну роботу з картою.

Розробляється спільнотою та з відкритим вихідним кодом: OpenStreetMap та бібліотека osmdroid є проектами з відкритим вихідним кодом, що розвиваються спільнотою. Це означає, що розробники мають доступ до активної

спільноти учасників і можуть отримати вигоду від постійного розвитку, виправлення помилок і вдосконалення. Це також дає можливість розробникам зробити свій внесок у проект і розширити можливості бібліотеки.

Економічно ефективне рішення: У порівнянні з іншими картографічними рішеннями, які можуть вимагати дорогого ліцензування або передплати, OpenStreetMap і бібліотека osmdroid пропонують економічно ефективне рішення для інтеграції карт в застосунрок. Вільний і відкритий характер даних OpenStreetMap усуває необхідність додаткових витрат, пов'язаних з доступом до картографічних даних.

Таким чином, проект використовує бібліотеку OpenStreetMap (osmdroid) через її багаті картографічні дані, підтримку офлайн-карт, можливості кастомізації, легкість, розробку під керівництвом спільноти та економічну ефективність. Ці особливості роблять її підходящим вибором для реалізації картографічних функцій застосунку відстеження маршрутів [15].

2.5. Архітектура MVVM

Застосунок “StrollTrack” слідує за архітектурним шаблоном Model-View-ViewModel (MVVM). MVVM – це шаблон проектування, який відокремлює логіку уявлення від базових даних та забезпечує чіткий поділ завдань [16].

У MVVM модель представляє дані та бізнес-логіку, представлення представляє інтерфейс користувача, а модель представлення діє як посередник між поданням і моделлю, надаючи дані для відображення та обробки взаємодій з користувачем.

Архітектура MVVM була обрана за її переваги з точки зору організації коду, тестування та зручності обслуговування. Це дозволяє використовувати модульну та незв'язану кодову базу, що спрощує її розуміння та підтримку. Поділ обов'язків дозволяє розробникам незалежно працювати над різними частинами застосунка, не впливаючи на роботу один одного.

MVVM також спрощує тестування, оскільки ViewModel можна легко тестувати без використання інтерфейсу користувача. Він просуває використання платформ прив'язки даних і реактивного програмування, такого як LiveData, для забезпечення чуйного та реактивного інтерфейсу користувача [16].

2.6. Компоненти архітектури Android

Компоненти архітектури Android — це набір бібліотек, що надаються Google, які спрощують розробку надійних, модульних та зручних у супроводі програм для Android. До них відносяться LiveData, ViewModel, Room та Data Binding [17].

LiveData — це клас власника даних з урахуванням життєвого циклу, який дозволяє компонентам відстежувати зміни даних і автоматично оновлювати інтерфейс користувача при зміні даних. LiveData забезпечує реактивний підхід до програмування, гарантуючи, що інтерфейс користувача залишається в курсі останніх даних [12].

ViewModel - це клас, який зберігає і управляє даними, пов'язаними з інтерфейсом користувача, з урахуванням життєвого циклу. Він витримує зміни конфігурації, такі як повороти екрана, і забезпечує доступ до даних для декількох компонентів інтерфейсу користувача, таких як фрагменти або дії. ViewModel відокремлює дані, пов'язані з інтерфейсом користувача, від компонентів інтерфейсу користувача, просуваючи більш чисту і більш модульну архітектуру [13].

Room, як обговорювалося раніше, є бібліотекою ORM, яка спрощує операції з базою даних і забезпечує безшовну інтеграцію з іншими компонентами архітектури Android [4].

Data Binding — це бібліотека, яка забезпечує декларативну прив'язку компонентів інтерфейсу користувача до джерел даних. Це дозволяє розробникам писати виразні та короткі макети XML [18], які безпосередньо пов'язані з відповідними даними у ViewModel [17].

Використання компонентів архітектури Android у “StrollTrack” сприяє модульності, тестованості та ремонтопридатності. Він за безпечує міцну основу для створення надійних та добре структурованих застосунків для Android.

2.7. Шаблон проєктування Coroutines

Coroutines – це шаблон проектування паралелізму, представлений у Kotlin, який спрощує асинхронне програмування. Вони забезпечують простий та ефективний спосіб виконання тривалих завдань, таких як мережеві запити або операції з базою даних без блокування основного потоку [19].

У програмі “StrollTrack” співпрограми використовуються для асинхронних операцій із базою даних, таких як вставка треків у базу даних Room. Співпрограми покращують швидкість відгуку та продуктивність програми, виконуючи трудомісткі завдання поза основним потоком.

Використовуючи співпрограми, “StrollTrack” забезпечує плавну і швидку взаємодію з користувачем, ефективно виконуючи трудомісткі операції.

Також застосунок “StrollTrack” включає різні діалогові вікна і службові функції для поліпшення взаємодії з користувачем і оптимізації загальних завдань. Діалоги, реалізовані за допомогою AlertDialog, використовуються для запиту дій користувача або відображення важливих повідомлень [20].

Діалогові вікна дозволяють застосунку повідомляти користувачеві важливу інформацію, наприклад, включати служби позиціонування або підтверджувати дії користувача. Вони сприяють загальному зручності

використання та інтерактивності програми, забезпечуючи інформування та залучення користувачів.

Службові функції, такі як функція showToast та функції перевірки дозволів, надають повторно використовувані фрагменти коду для загальних операцій у програмі. Ці функції спрощують загальні завдання, такі як відображення спливаючих повідомлень або перевірка дозволів шляхом інкапсуляції необхідного коду у зручному для повторного використання та короткому форматі [21].

Включаючи діалогові вікна та службові функції, “StrollTrack” покращує взаємодію з користувачем, покращує повторне використання коду та скорочує зусилля з розробки.

У розділі представлено огляд технологій, використаних для розробки програми “StrollTrack”. Ми обговорили платформу Android та її широкі можливості для розробки мобільних застосунків. Ми вивчили мову програмування Kotlin та його переваги з погляду читабельності коду та сумісності.

Ми детально вивчили базу даних Room та її переваги для локального зберігання даних та управління ними. Ми вивчили SDK Google Maps та його функції картографування та місцезнаходження. Ми також розглянули архітектуру MVVM та її переваги з точки зору організації коду та можливості тестування.

Крім того, ми вивчили компоненти архітектури Android, включаючи LiveData, ViewModel, Room та Data Binding, а також їхній внесок у створення надійних та зручних у супроводі застосунків Android. Ми обговорили співпрограми та їх роль в асинхронному програмуванні, а також діалоги та службові функції для покращення взаємодії з користувачем та повторного використання коду.

РОЗДІЛ 3. ПРИЗНАЧЕННЯ ТА ВИМОГИ СТВОРЕННЯ МОБІЛЬНОГО ЗАСТОСУНКУ

3.1. Призначення застосунку та вимоги користувача

Застосунок призначений для людей, які не хочуть тримати в голові зайву інформацію. Наприклад: стежки для прогулянок з собакою, маршрут до нового місця в лісі для пікніка, дорога до улюбленого магазину чи нового ігрового майданчика для дитини. Приорітетом нашого застосунку буде простота інтерфейсу, для того, щоб діти та літні люди могли без проблем користуватисб цим застосунком.

Щоб розробити ефективний мобільний застосунок для зберігання маршрутів прогулянок, важливо розуміти основні потреби та вимоги цільової аудиторії. Нижче наведено деякі ключові моментти, які потрібно враховувати щодо основних потреб користувача.

Відстеження маршруту: користувачі потребують надійної та точної функції відстеження маршруту, яка використовує технологію GPS для запису своїх маршрутів. Ця функція має забезпечувати оновлення дистанції, тривалості, швидкості та інших відповідних показників у режимі реального часу.

Карти та навігація: користувачам потрібні функції картографування та навігації, що дозволятимуть їм переглядати своє поточне місцезнаходження та відстежувати пройдений маршрут на карті.

Візуалізація даних: користувачі очікують, що застоунок представить записані дані у легко зрозумілому та простому форматі.

Встановлення цілей і відстеження прогресу: користувачів, яким застосунок треба для фіксування досягнень, може зацікавити можливість встановлювати персональні цілі, наприклад: цілі дистанції, цілі спалювання калорій або цілі пов’язані часу. Тому у майбутньому можна додати таку функцію, щоб охопити більше користувачів.

3.2 Визначення функціональних вимог до програми

Виходячи з визначених потреб користувача, функціональні вимоги до мобільного застосунка “StrollTrack” можна визначити наступним чином:

Відстеження GPS: застоунок повинна використовувати технологію GPS для точного відстеження та запису маршрутів прогулянок користувачів. Він має надавати оновлення в режимі реального часу щодо відстані, тривалості, темпу та інших відповідних показників, щоб забезпечити точне відстеження їхніх сеансів ходьби.

Карти та навігація: “StrollTrack” має включати функцію картографування, яка дозволяє користувачам переглядати своє поточне місцезнаходження, відстежувати свій прогрес на карті. Також при покращенні програми можна буде додати навігаційну підсказку. Ця функція надаватиме чіткі візуальні підказки, наприклад виділені маршрути або шляхові точки, щоб допомогти користувачам слідувати бажаним шляхом.

Візуалізація даних: застоунок повинен представляти записані дані у легко зрозумілому форматі. Оскільки застосунок створений саме для зберігання маршрутів, то має бути вибір зберігати чи ні маршрут в базу даних. При зберігані маршрута мають зберігатись також усі показники.

Платформа: Проаналізувавши статистику щодо кількості користувачів платформами Android та iOS, бачимо, що в першу чергу краще реалізувати застосунок саме для операційної системи Android, щоб охопити більшу цільову аудиторію по всьому світу. Але в майбутньому можна зробити версію для iOS [22].

Враховуючи ці функціональні вимоги, розробка “StrollTrack” може відповідати конкретним потребам і очікуванням цільових користувачів, забезпечуючи зручну та ефективну програму для запису пішохідних маршрутів.

Для оцінки придатності існуючих аналогів у задоволенні вимог користувачів, виявлених для мобільного застосунка “StrollTrack”, необхідне порівняння їх функціональних можливостей. Наступний аналіз оцінює

відповідність аналогів (WalkTracker: Hiking Trails, My Walk Tracker та My Track) вивленим вимогам користувача.

3.3 Порівняння існуючих аналогів із вимогами користувача

Для оцінки придатності існуючих аналогів у задоволенні вимог користувачів, виявлених для мобільного застосунка “StrollTrack”, необхідне порівняння їх функціональних можливостей. Наступний аналіз оцінює відповідність аналогів (WalkTracker: Hiking Trails, My Walk Tracker та My Track) виявленим вимогам користувача.

Відстеження маршруту: всі три аналоги забезпечують можливості відстеження маршруту на основі GPS, що дозволяє користувачам записувати свої пішохідні маршрути. Вони пропонують оновлення в режимі реального часу про відстань, тривалість, а іноді і додаткові показники, такі як темп і набір висоти. Тому з погляду відстеження маршруту аналоги відповідають вимогам користувача.

Картування та навігація: WalkTracker та My Track пропонують функції картографування та навігації, відображаючи поточне місцезнаходження користувача, відстежуючи його просування на карті та надаючи допомогу у навігації. My Walk Tracker, з іншого боку, орієнтований переважно на ходьбу і може мати більш обмежені можливості картування та навігації. Щоб повністю задовольнити вимоги користувача, “StrollTrack” має забезпечувати надійні функції картографування та навігації.

Візуалізація даних: WalkTracker, My Walk Tracker та My Track тією чи іншою мірою надають функції візуалізації даних. Вони представляють записані дані, такі як пройдена відстань, спалені калорії та інші відповідні показники візуально привабливих форматах. “StrollTrack” повинен пропонувати комплексні та настроювані параметри візуалізації даних для задоволення вимог користувача.

Постановка цілей та відстеження прогресу: My Walk Tracker та My Track дозволяють користувачам встановлювати персональні цілі та відстежувати їх прогрес у досягненні цих цілей. WalkTracker більше фокусується на дослідженні стежок і може мати обмежені функції встановлення цілей. “StrollTrack” повинен включати надійні можливості постановки цілей і відстеження прогресу для ефективного задоволення вимог користувача.

Соціальна взаємодія та спільне використання: My Walk Tracker та My Track пропонують функції соціальної інтеграції, які дозволяють користувачам спілкуватися з іншими користувачами, ділитися своїми маршрутами та досвідом, а також брати участь у спільноті. WalkTracker може мати більш обмежені можливості соціальної взаємодії. Щоб задовольнити вимоги користувача, “StrollTrack” повинен містити соціальні функції, які сприяють взаємодії та обміну інформацією між користувачами.

Визначення переваг запропонованого застосування порівняно з аналогами

При розгляді переваг пропонованого застосунка “StrollTrack” в порівнянні з існуючими аналогами слід виділити наступні моменти:

Універсальність: “StrollTrack” призначений для широкого спектра прогулянок, включаючи звичайні прогулянки, біг підтюпцем та біг. Він розрахований на будь-який рух, оскільки його ціль саме запамятати маршрут та зберігти найголовніші дані про нього: відстань, час, швидкість мередня

Зручний інтерфейс: “StrollTrack” розрахований на те, що людина зберігатиме улюблені та цікаві маршрути, тому функціонал має бути простим і направленим саме на це. В аналогах дуже багато різних кнопок статистик і вони розраховані на аудиторію, яка хоче зберігати саме свої успіхи, тому там статистика доречна. Наш застосунок має іншу ціль, тому має віддавати перевагу зручному інтерфейсу, що забезпечує простоту використання та інтуїтивно зрозумілу навігацію. Добре продуманий та інтуїтивно зрозумілий інтерфейс може покращити взаємодію з користувачем та сприяти ефективній взаємодії з застосунком.

Підкреслюючи ці переваги, “StrollTrack” може позиціонувати себе як конкурентоспроможний та бажаний варіант для користувачів, яким потрібен комплексний та зручний мобільний застосунок саме для запису пішохідних маршрутів. Нижче можна буде переглянути (рисунок 3.1) порівняння нашого застосунка з існуючими аналогами.

	My Track	Hiking Trails	My Walk Tracker	StrollTrack
Відстеження шляху	+	+	+	+
Мапування шляху	+	+	+	+
Імпорт шляху	+	+	+	-
Експорт шляху	-	+	-	-
Andriod Platform	+	+	-	+
IOS Platform	-	-	+	-
Голосовий супровід	+	-	+	-
Ведення статистики	+	+	+	-/+
Персоналізовані цілі	-	-	+	-
Простота інтерфейсу	-	-	-	+
Універсальність додатку	+	-	-	+

Рисунок 3.1 – Порівняння застосунків

Приорітетністю застосунку “StrollTrack” є саме зручний та інтуїтивно зрозумілий інтерфейс.

РОЗДІЛ 4. РЕАЛІЗАЦІЯ МОБІЛЬНОГО ЗАСТОСУНКУ

4.1. Опис взаємодій файлів застосунку

Клас MainActivity.kt: є точкою входу до програми. Він розширює клас AppCompatActivity.

У методі onCreate макет роздувається за допомогою класу ActivityMainBinding.

Метод onBottomNavigClick налаштовує слухача для нижньої навігаційної панелі і на основі вибраного елемента відкриває відповідний фрагмент за допомогою функції openFragment.

Функція openFragment - це утиліта, визначена у файлі utils/Exten.kt, яка замінює поточний фрагмент у контейнері placeHolder на вказаний фрагмент.

MainActivity встановлює початковий фрагмент на MainFragment шляхом виклику openFragment(MainFragment.newInstance()) у методі onCreate.

Клас MainFragment - це фрагмент, який відображає основний вміст програми. Він містить елементи інтерфейсу користувача і обробляє взаємодію з користувачем, пов'язану з основною функціональністю програми.Фрагмент взаємодіє з MainViewModel для отримання та відображення даних.Він також може взаємодіяти з іншими утилітарними класами або сервісами для виконання певних завдань.

Клас SettingsFragment - це фрагмент, який дозволяє користувачам налаштовувати параметри програми. Він надає елементи інтерфейсу користувача і обробляє взаємодію користувача з налаштуваннями.Він може взаємодіяти з класом SharedPreferences або іншими утилітами для зберігання та отримання даних налаштувань.

Клас TracksFragment є фрагментом, який відображає список відстежуваних елементів (треків) з бази даних. Він отримує дані з MainViewModel і відображає їх у форматі списку.Він може обробляти дії користувача для вибору або

видалення треків, які потім передаються назад до MainViewModel для подальшої обробки.

Клас MainViewModel - це клас ViewModel, який діє як міст між компонентами інтерфейсу користувача (наприклад, фрагментами) та джерелами даних (наприклад, базою даних).Він надає дані компонентам інтерфейсу та обробляє дії користувача.MainViewModel взаємодіє з базою даних Maindb через пов'язаний з нею DAO (Data Access Object) для виконання операцій з базою даних.Вона відкриває об'єкти LiveData для спостереження за змінами в даних і відповідно оновлює інтерфейс користувача.MainViewModel може також містити бізнес-логіку або виконувати додаткові операції на основі дій користувача.

Клас LocationService - це клас сервісу, який відповідає за відстеження місцезнаходження користувача у фоновому режимі. Він використовує FusedLocationProviderClient для отримання оновлень місцезнаходження.

LocationService обчислює пройдену відстань і зберігає дані про місцезнаходження у geoPointsList.Він транслює оновлення місцезнаходження та пройдену відстань за допомогою LocalBroadcastManager.

Клас даних LocationModel представляє модель даних про місцезнаходження. Він містить таку інформацію, як швидкість, відстань та список географічних точок.

Точки GeoPoints представляють координати широти та довготи [23].

Об'єкт DialogManager надає утиліти для відображення діалогових вікон. Він має функції для показу діалогового вікна увімкнення локації та діалогового вікна збереження.Ці діалоги показуються користувачеві для певних дій або підказок.

Файл Exten.kt містить функції розширення для класів Fragment і AppCompatActivity. Ці функції розширення надають додаткову функціональність, наприклад, відкриття фрагментів або відображення повідомлень тостів.

Об'єкт TimeU надає утиліти для форматування часу і дати. Він має функції для ф орматування рядків часу і дати на основі заданого формату.

Клас Maindb представляє екземпляр бази даних Room для програми. Він визначає сутності бази даних, DAO (Data Access Object - об'єкт доступу до даних) та версію бази даних. Він надає одиночний екземпляр бази даних за допомогою функції getDatabase.

Клас MainApp є користувацьким класом застосунка, який слугує точкою входу для застосунка. Він надає доступ до ресурсів рівня застосунка, таких як екземпляр бази даних.

Взаємодію між цими файлами можна підсумувати наступним чином:

MainActivity слугує точкою входу для застосунка. Він ініціалізує інтерфейс користувача і налаштовує нижню панель навігації. Фрагменти MainFragment, SettingsFragment і TracksFragment відображаються на основі вибору користувача у нижній навігаційній панелі. Компоненти (фрагменти) інтерфейсу взаємодіють з MainViewModel для отримання даних, виконання дій або оновлення інтерфейсу. MainViewModel взаємодіє з базою даних Maindb через пов'язаний з нею DAO для виконання операцій з базою даних. MainViewModel також обробляє оновлення місцезнаходження від LocationService і транслює ці оновлення до інтерфейсу користувача. Інші утиліти, такі як DialogManager, Exten.kt та TimeU, надають додаткову функціональність та утиліти для відображення діалогів, керування фрагментами, форматування часу та дати, а також обробки дозволів. Клас MainApp надає доступ до ресурсів рівня програми, таких як екземпляр бази даних.

4.2. Опис реалізації коду програми

Нижче проведемо опис файлів програми.

Інтерфейс Dao з трьома методами для вставки, пошуку та видалення треків у базі даних Room. Анотації @Insert, @Query та @Delete містять інструкції для Room щодо того, як обробляти ці операції з базою даних (рисунок 4.1).

```
import ...
@Dao
interface Dao {
    @Insert
    suspend fun insertTrack(trackIt: TrackIt)
    @Query("SELECT * FROM TRACK")
    fun getAllTracks(): Flow<List<TrackIt>>
    @Delete
    suspend fun deleteTrack(trackIt: TrackIt)
}
```

Рисунок 4.1 – Файл Dao.kt

Клас Maindb надає метод створення та доступу до бази даних Room у застосунка Android. Він визначає абстрактний метод для отримання екземпляра DAO і включає companion object зі статичним методом для отримання одиночного екземпляра бази даних. Розглянемо складові цього файлу і опишемо, за що вони відповідають:

клас Maindb, який розширює базу даних RoomDatabase. Він слугує головною точкою доступу до бази даних Room.

абстрактний метод getDao(), який повертає екземпляр інтерфейсу Dao. Цей метод має бути реалізований підкласами Maindb для забезпечення доступу до методів DAO.

статичний метод в companion object, названий getDatabase(). Він отримує об'єкт Context як параметр і повертає екземпляр бази даних Maindb. Він перевіряє, чи змінна INSTANCE дорівнює нулю, і якщо так, то створює новий екземпляр за допомогою Room.databaseBuilder() і присвоює його INSTANCE. Метод гарантує, що буде створено і повернуто лише один екземпляр бази даних.

Клас TrackAdapter відповідає за управління списком елементів TrackIt і прив'язку їх до відповідних представлень у RecyclerView. Клас Holder обробляє

події кліку. Клас Comparator забезпечує ефективне порівняння елементів для оновлень (рисунок 4.2).

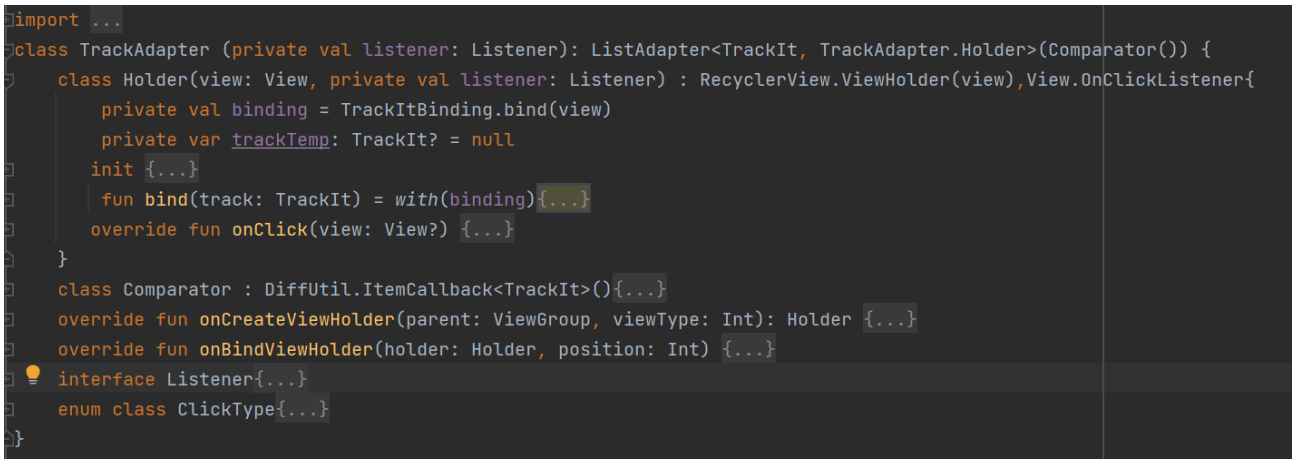

Рисунок 4.2 – Файл TrackAdapter.kt

Клас даних TrackIt представляє сутність маршрута у базі даних Room. Він має властивості, які відповідають стовпцям у таблиці "track", включаючи стовпець первинного ключа. Анотації ColumnInfo надають метадані про назви стовпців, а анотація Entity позначає клас як сутність у базі даних

@PrimaryKey(autoGenerate = true) val id: Int?: Ця анотація застосовується до властивості id, вказуючи, що вона є стовпцем первинного ключа в сутності.

@ColumnInfo(name = "time") val time: String: стовпчик з назвою "time" у базі даних та має тип String.

@ColumnInfo(name = "date") val data: String: стовпчик з назвою "date" у базі даних та має тип String.

@ColumnInfo(name = "distance") val distance: String: стовпчик з назвою "distance" у базі даних та має тип String.

@ColumnInfo(name = "velocity") val speed: String: стовпчик з назвою "velocity" у базі даних та має тип String.

@ColumnInfo(name = "geo_points") val geoPoints: String: стовпчик з назвою " geo_points " у базі даних та має тип String.

Тепер опишемо складові наших фрагментів: MainFragment, SettingsFragment, TracksFragment, ViewTrackFragment.

Перший складається з:

У класі MainFragment оголошено різні властивості, зокрема pLauncher, binding, timer, startTime, pl, locationModel, mLocOverlay, isServiceRunning, firstStart і model. Ці властивості використовуються для зберігання посилань на різні елементи інтерфейсу, моделі даних та інформацію про стан.

Метод onCreateView() відповідає за роздування макета фрагмента і повернення кореневого подання. У цьому методі використовується клас FragmentMainBinding для розгортання макета і повертається кореневе подання.

Перевизначений метод onViewCreated() з класу Fragment, який викликається після onCreateView() і використовується для ініціалізації представлень фрагмента та налаштування слухачів подій. У цьому методі виконуються такі операції: реєстрація прав доступу, налаштування слухачів кліків, перевірка стану сервісу, часу оновлення та реєстрація приймача місцезнаходження.

Метод startLocService() викликається для запуску служби визначення місцезнаходження. Він запускає LocationService, використовуючи або startForegroundService(), або startService(), залежно від версії Android. Він також оновлює елементи інтерфейсу і запускає таймер.

Метод setOnClick() використовується для встановлення слухачів кліків для кнопок пуск/стоп та центрування.

Метод centerLocation() центрує карту за поточним розташуванням за допомогою mLocOverlay та методу map.controller.animateTo().

Метод locationUpdates() оновлює елементи інтерфейсу.

Метод updateTime() оновлює елемент інтерфейсу tvTime.

Метод startTimer() запускає таймер.

Метод getAverSpeed() обчислює середню швидкість на основі пройденої відстані та часу, що минув.

Метод getCurTime() обчислює поточний час на основі часу старту.

Метод geoPointsToString() перетворює список об'єктів GeoPoint.

Метод starStopService(): викликається при натисканні кнопки запуску/зупинки. Він запускає або зупиняє сервіс, оновлює елементи інтерфейсу, скасовує таймер і показує діалогове вікно для збереження треку.

Метод getTrackItem(): створює об'єкт TrackIt на основі поточних даних треку.

Метод checkServState(): перевіряє стан сервісу і відповідно оновлює елементи інтерфейсу.

Метод settingsOsm(): відповідає за налаштування параметрів та конфігурації OpenStreetMap.

Метод initOSM(): ініціалізує OpenStreetMap, налаштовуючи мапу, провайдера локацій та інші параметри.

Метод checkPermissionAfter10() перевіряє дозволи на точне та фонове розташування для версій Android після Android 10. Якщо дозволи надано, він ініціалізує OpenStreetMap і перевіряє налаштування місцезнаходження; в іншому випадку, він запускає запит на отримання дозволу.

Метод checkPermissionBefore10() перевіряє дозвіл на точне визначення місцезнаходження для версій Android до Android 10. Якщо дозвіл надано, він ініціалізує OpenStreetMap і перевіряє налаштування місцезнаходження; в іншому випадку, він запускає запит на дозвіл.

Метод checkLocationEnabled() перевіряє, чи увімкнено локацію на пристрої, і показує діалогове вікно, щоб увімкнути її, якщо це необхідно.

Метод addPoint() додає точку до полілінії на карті.

Метод updatePolyline() оновлює полілінію на мапі, додаючи нові точки.

Клас "SettingsFragment" є фрагментом, що відповідає за відображення та керування налаштуваннями програми. Давайте пройдемося по коду і зрозуміємо його функціональність:

Клас розширює клас "PreferenceFragmentCompat", який надає фреймворк для створення екрану налаштувань з використанням налаштувань.

Усередині методу onCreatePreferences() XML-файл налаштувань (main_preferences.xml) роздувається і встановлюється в якості налаштувань для фрагмента за допомогою методу "setPreferencesFromResource".

Метод init() ініціалізує налаштування та встановлює приймач змін для кожного налаштування.

Метод onChangeListener() повертає екземпляр "Preference.OnPreferenceChangeListener", який обробляє зміни налаштувань. Він перевіряє ключ зміненого параметра і виконує відповідні дії на основі цього.

У методі onTimeChange() значення часового параметра перетворюється у формат, придатний для відображення, і відповідно оновлює заголовок параметра.

Метод initPrefs() ініціалізує налаштування початковими значеннями. Він отримує спільні налаштування, зчитує збережені значення для параметрів часу та кольору, а також оновлює назву та піктограму відповідного параметра.

Загалом, клас SettingsFragment відповідає за створення та керування екраном налаштувань, обробку змін налаштувань та оновлення інтерфейсу на основі вибраних налаштувань.

Клас TracksFragment відповідає за відображення списку доріжок у RecyclerView та обробку подій кліку на елементах доріжок. Він взаємодіє з MainViewModel для отримання треків і виконання таких дій, як видалення треку або відкриття треку для перегляду. Цей фрагмент має такі складові:

клас TracksFragment розширює клас Fragment і реалізує інтерфейс TrackAdapter.Listener. Інтерфейс TrackAdapter.Listener використовується для обробки подій кліку у TrackAdapter.

Метод onCreateView() викликається для створення ієрархії представлень, пов'язаних з фрагментом.

Метод onViewCreated() викликається після створення ієрархії представлень. У цьому методі ініціалізується RecyclerView та отримуються треки з MainViewModel.

companion object містить функцію newInstance(), яка є звичайною практикою для створення нового екземпляра фрагмента. Вона повертає новий екземпляр TracksFragment.

Метод onClick() викликається, коли натискається елемент доріжки у RecyclerView. Він отримує натиснутий елемент треку та тип події кліку. На основі типу кліку він виконує різні дії. Якщо тип кліку - ВИДАЛИТИ, він викликає метод, щоб видалити доріжку. Якщо тип кліку OPEN, він встановлює поточну доріжку і відкриває її іншим методом.

Клас ViewTrackFragment відповідає за відображення детальної інформації про конкретну доріжку. Він використовує MainViewModel для отримання поточної доріжки і відповідно оновлює інтерфейс користувача. Він використовує бібліотеку OpenStreetMap для відображення треку на карті, включаючи полілінію, що представляє маршрут треку, та маркери стартової і фінішної позицій (рисунок 4.3).

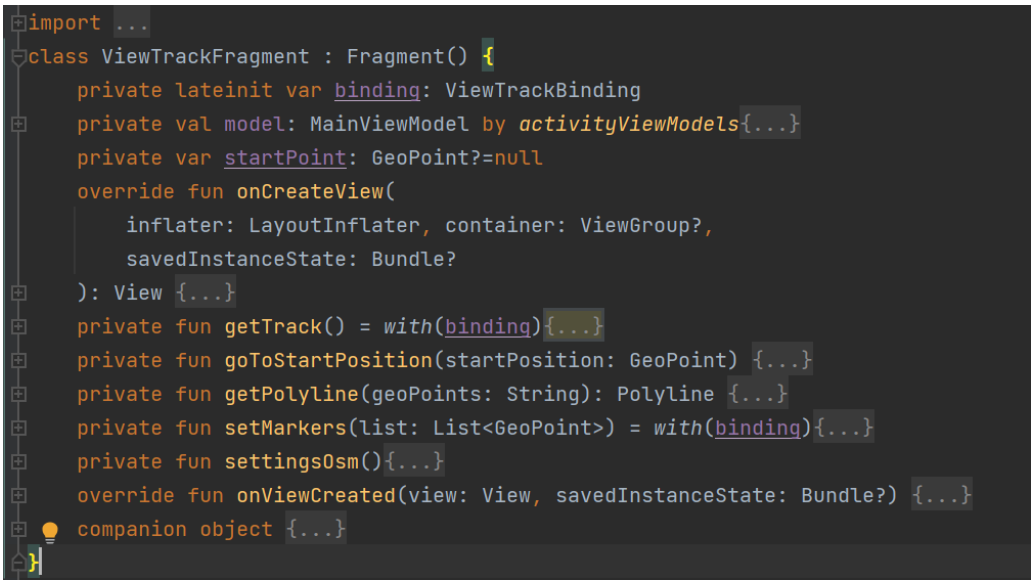

Рисунок 4.3 – Файл ViewTrackFragment.kt

Клас LocationModel реалізує інтерфейс Serializable, що дозволяє серіалізувати та десеріалізувати екземпляри цього класу для збереження даних або комунікаційних цілей.

Клас даних LocationModel має три властивості:

1. velocity представляє швидкість, як значення типу Float
2. distance відображає відстань, пройдену від початкової точки до поточного місцезнаходження, як значення типу Float
3. geoPointsList представляє список об'єктів GeoPoint, які визначають географічні координати місцезнаходження. Являє собою ArrayList<GeoPoint>

Клас LocationService розширює клас Service і виконує наступні завдання:

1. Відстежує місцезнаходження користувача, запитуючи оновлення місцезнаходження за допомогою FusedLocationProviderClient з сервісів Google Play.

2. Обчислює пройдену відстань на основі оновлень місцезнаходження користувача.

3. Підтримує список об'єктів GeoPoint, що представляють координати місцезнаходження користувача в часі.

4. Надсилає оновлення місцезнаходження за допомогою локальної трансляції.

5. Створює сповіщення служби на передньому плані, щоб показати, що відстеження місцезнаходження запущено.

6. Ініціалізує налаштування місцезнаходження і запускає оновлення місцезнаходження на основі бажаного інтервалу (рисунок 4.4).

```
class LocationService: Service() {
    private var distance=0.0f
    private var lastLocation :Location?=null
    private lateinit var locProvider: FusedLocationProviderClient
    private lateinit var locRequest:LocationRequest
    private lateinit var geoPointsList: ArrayList<GeoPoint>
    private var isDebug = false
    override fun onBind(intent: Intent?): IBinder? {...}
    override fun onStartCommand(intent: Intent?, flags: Int, startId: Int): Int {...}
    override fun onCreate() {...}
    override fun onDestroy() {...}
    @SuppressLint("SuspiciousIndentation")
    private fun startNotification() {...}
    private val locCallBack = object : LocationCallback() {...}
    private fun sendLocData(locModel: LocationModel) {...}
    private fun initLocation(){...}
    private fun startLocationUpdates() {...}
    companion object {...}

}
```

Рисунок 4.4 – Файл LocationService

У файлі DialogManager об’єкт DialogManager має два методи:

Метод showLocEnableDialog відображає діалогове вікно із запитом до користувача увімкнути служби визначення місцезнаходження. Він отримує Context та Listener як параметри. Діалогове вікно відображає заголовок, повідомлення та дві кнопки: "Так" і "Ні". Коли натискається кнопка "Так", викликається метод onClick .

Метод showSaveDialog відображає діалогове вікно із запитом на збереження елемента доріжки. Він отримує параметри Context, необов'язковий

елемент TrackIt та Listener. У діалоговому вікні відображається інформація про елемент треку, зокрема час, швидкість та відстань. Воно також містить кнопки "Зберегти" і "Скасувати". Коли натискається кнопка "Зберегти", викликається метод onClick.

Файл Exten надає наступні функції розширення:

Функція Fragment.openFragment дозволяє замінити фрагмент у контейнері, визначеному ідентифікатором ресурсу R.id.placeHolder. Вона використовує supportFragmentManager асоційованого AppCompatActivity для виконання транзакції фрагмента. При заміні фрагментів застосовується анімація переходу від одного фрагмента до іншого.

Функція AppCompatActivity.openFragment подібна до попередньої функції openFragment, але може бути викликана безпосередньо на екземплярі AppCompatActivity. Вона перевіряє, чи поточний фрагмент збігається з тим, що відкривається, щоб уникнути надлишкових транзакцій фрагментів.

Функція showToast показує коротке повідомлення за допомогою методу Toast.makeText.

Функція showToast (AppCompatActivity) подібна до попередньої функції showToast, але може бути викликана безпосередньо на екземплярі AppCompatActivity.

Функція checkPermission перевіряє, чи надано дозвіл, вказаний у рядку дозволів чи ні.

У файлі Time U є об’єкт TimeU, який надає наступні функції:

Функція getTime отримує параметр timeInMillis типу Long і повертає відформатований рядок часу. Для форматування часу використовується клас SimpleDateFormat з шаблоном "HH:mm:ss". Перед форматуванням значення часу встановлюється часовий пояс UTC.

Функція getDate повертає відформатований рядок дати і часу, що представляє поточну дату і час. Він отримує поточну дату і час з методу Calendar.getInstance().

У класі MainActivity є такі методи:

метод onCreate,у якому макет роздувається за допомогою класу ActivityMainBinding, який автоматично генерується на основі XML-файлу макета. Зв'язка binding.root представляє кореневе подання макета, і воно встановлюється як подання вмісту активності за допомогою setContentView(binding.root). У цьому методі спочатку відкривається MainFragment.

Метод onBottomNavigClick відповідає за обробку подій вибору елементів нижньої панелі навігації. Усередині цього методу використовується оператор when для визначення вибраного елемента на основі його ID. Залежно від обраного елемента відкривається відповідний фрагмент за допомогою функції розширення openFragment.

openFragment є користувацькою функцією розширення, визначеною в пакеті utils. Вона використовується для заміни поточного фрагмента новим фрагментом у контейнері фрагментів активності.

Файл MainViewModel керує та надає необхідні дані для MainActivity, а також надає функції для взаємодії з базовою базою даних. В ньому є клас MainViewModel, який розширює клас ViewModel з компонентів архітектури Android. Він зберігає дані і надає їх компонентам інтерфейсу користувача (MainActivity), витримуючи зміни конфігурації. В цьому класі є:

1. Функція insertTrack - це функція підпрограми, яка вставляє об'єкт TrackIt в базу даних за допомогою функції DAO insertTrack. Вона запускається у viewModelScope для асинхронного виконання операції з базою даних.

2. Функція deleteTrack - це підпрограмна функція, яка видаляє об'єкт TrackIt з бази даних за допомогою функції deleteTrack DAO. Вона також запускається у viewModelScope.

3. Клас ViewModeFactory є фабричним класом, який надає екземпляри MainViewModel. Він реалізує інтерфейс ViewModelProvider.Factory і перевизначає функцію create.

Використовуючи клас MainApp, ви можете отримати доступ до бази даних програми (властивості бази даних) з різних компонентів програми, таких як activities, fragments та view models. Це забезпечує централізований і зручний спосіб керування та доступу до ресурсів рівня застосунка

клас MainApp розширює клас Application, що надається фреймворком Android. Він представляє сам застосунок і створюється перед будь-яким іншим компонентом програми.

.

РОЗДІЛ 5. ІНСТРУКЦІЯ КОРИСТУВАЧА

В застосунку реалізовано 3 екрани: “Tracks”, “Home”, “Settings”.

Початковий екран, на який потрапляє користувач при запуску застосунка, це екран “Home”

“Home”: головний фрагмент з картою (рисунок 5.1), на якому записується маршрут. “Time”-таймер, який вмикається після запуску маршрута, та показує час ходьби. “Average velocity”- середня шивдкість впродовж всього маршруту. “Velocity”- швидкість в поточний момент руху. “Distance”- відстань, яку пройшов користувач. Справа зверху кнопка для того, щоб побачити своє поточне місцезнаходження. Знизу кнопка для запуску (рисунок 5.1) та зупинки маршруту (рисунок 5.2).

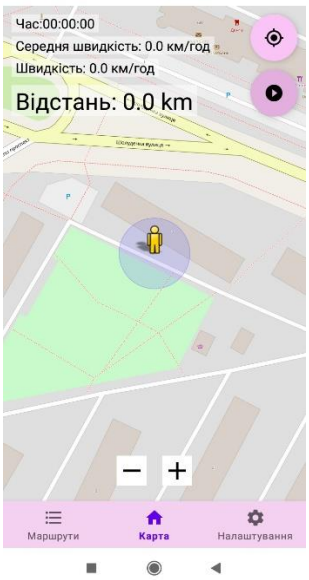

Рисунок 5.1 – Фрагмент карти до старту маршрута

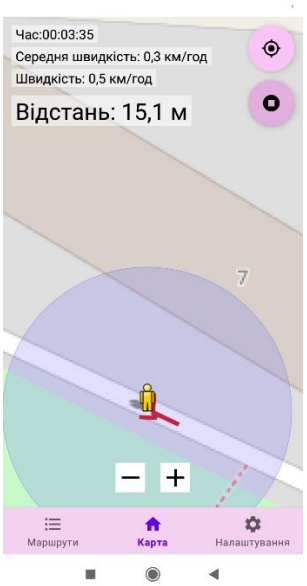

Рисунок 5.2 – Фрагмент карти після старту маршрута

Завдяки навігаційній панелі користувач може відкрити ще два фрагменти: “Tracks” та “Settings”

“Tracks”: фрагмент (рисунок 5.3) показує список збережених маршрутів. Натиснувши на маршрут, відкриється фрагмент “ViewTrackFragment”, де зображена карта з цим маршрутом (рисунок 5.4). Також є кнопка для видалення маршруту.

Рисунок 5.3 – Фрагмент зі списком маршрутів

Рисунок 5.4 –Фрагмент з відкритим збереженим маршрутом

“Settings”: фрагмент з налаштуваннями застосунка. “Update Time”-кнопка для зміни частоти оновлення застосунка. “Track color”-кнопка для зміни кольора лінії, яка малює маршрут (рисунок 5.5).

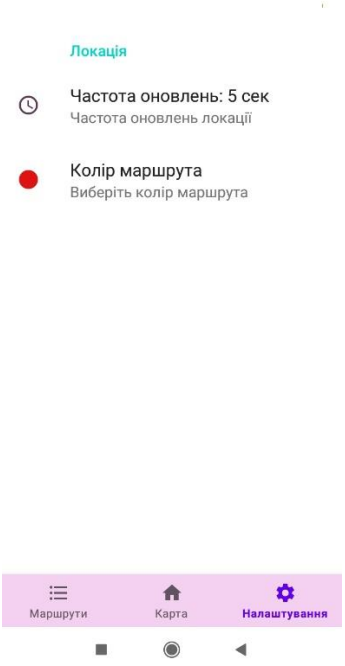

Рисунок 5.5 – Фрагмент з налаштуванням застосунка

Інтерфейс застосунку “StrollTrack” вийшов інтуєтивно зрозумілим та простим, тому користувач з легкістю зможе ним користуватись.

ВИСНОВКИ

В процесі виконання кваліфікаційної роботи, відповідно до поставленої мети були виконані завдання:

- проведений аналіз існуючих аналогів на сучасному ринку застосунків;
- розроблено мобільний застосунок для пішохідних прогулянок;
- протестувано застосунок;
- поглиблено та закріплено навички роботи з мовою програмування Kotlin.

Результатом роботи є розроблений застосунок “StrollTrack”, який може відстежувати та зберігати маршрути користувачів, що повністю відповідає меті роботи.

У зв'язку із зростанням популярності активного способу життя застосунок залишатиметься актуальним в майбутньому. Особливо він буде корисним для людей, які мають собак та хочуть зберігати стежки для прогулянок з ними. Для такого кола користувачів він завжди буде потрібен. Для дітей та людей похилого віку цей застосунок буде затребуваним незалежно від часу чи трендів, оскільки для них головне його завдання – відстежити та зберігти потрібний шлях, а це залишатиметься актуальним завжди.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Physical activity [Електронний ресурс] – Режим доступу до ресурсу: https://www.who.int/news-room/fact-sheets/detail/physical-activity.

2. Burnette E. Hello, Android: Introducing Google's Mobile Development Platform 4th Edition / E. Burnette., 2015. – 252 с.

3. Leiva A. Kotlin for Android Developers: Learn Kotlin the easy way while developing an Android App / Antonio Leiva., 2016. – 240 с. – (reateSpace Independent Publishing Platform).

4. Murphy M. Elements of Android Room / Mark Murphy., 2022. – 266 с. – (CommonsWare, LLC).

5. Save data using SQLite [Електронний ресурс] – Режим доступу до ресурсу: https://developer.android.com/training/data-storage/sqlite.

6. WalkTracker: Hiking Trails [Електронний ресурс] – Режим доступу до ресурсу: https://play.google.com/store/apps/details?id=net.videgro.walktracker&hl=en_US.

7. My Walk Tracker [Електронний ресурс] – Режим доступу до ресурсу: https://apps.apple.com/us/app/my-walk-tracker/id1515421087.

8. My Track [Електронний ресурс] – Режим доступу до ресурсу: https://play.google.com/store/apps/details?id=com.zihua.android.mytracks&hl=en_US.

9. Schiefer L. Inside the Android OS: Building, Customizing, Managing and Operating Android System Services / L. Schiefer, G. Meike., 2021. – 272 с. – (Addison-Wesley Professional).

10. Subramaniam V. Programming Kotlin / Venkat Subramaniam., 2019. – 460 с. – (Pragmatic Bookshelf).

11. Save data in a local database using Room [Електронний ресурс] – Режим доступу до ресурсу: https://developer.android.com/training/data-storage/room.

12. LiveData overview [Електронний ресурс] – Режим доступу до ресурсу: https://developer.android.com/topic/libraries/architecture/livedata.

13. ViewModel overview [Електронний ресурс] – Режим доступу до ресурсу: https://developer.android.com/topic/libraries/architecture/viewmodel.

14. About OpenStreetMap [Електронний ресурс] – Режим доступу до ресурсу: https://openstreetmap.or.id/en/tentang-openstreetmap/.

15. "Hello osmdroid World" [Електронний ресурс] – Режим доступу до ресурсу: https://osmdroid.github.io/osmdroid/How-to-use-the-osmdroid-library.html.

16. Elias C. Understanding MVVM pattern for Android in 2021 [Електронний ресурс] / Christopher Elias. – 2021. – Режим доступу до ресурсу: https://proandroiddev.com/understanding-mvvm-pattern-for-android-in-2021-98b155b37b54.

17. Data Binding Library [Електронний ресурс] – Режим доступу до ресурсу: https://developer.android.com/topic/libraries/data-binding.

18. XML in Android: Basics And Different XML Files Used In Android [Електронний ресурс] – Режим доступу до ресурсу: https://abhiandroid.com/ui/xml.

19. Kotlin coroutines on Android [Електронний ресурс] – Режим доступу до ресурсу: https://developer.android.com/kotlin/coroutines.

20. AlertDialog [Електронний ресурс] – Режим доступу до ресурсу: https://developer.android.com/reference/android/app/AlertDialog.

21. Toasts overview [Електронний ресурс] – Режим доступу до ресурсу: https://developer.android.com/guide/topics/ui/notifiers/toasts.

22. Android vs. Apple Market Share: Leading Mobile Operating Systems (OS) (May 2023) [Електронний ресурс] – Режим доступу до ресурсу: https://www.bankmycell.com/blog/android-vs-apple-market-share/.

23. Location [Електронний ресурс] – Режим доступу до ресурсу: https://developer.android.com/reference/kotlin/android/location/Location.