Вінницький національний технічний університет
(повне найменування вищого навчального закладу)

Факультет інтелектуальних інформаційних технологій та автоматизації
(повне найменування інституту, назва факультету (відділення))

Кафедра комп'ютерних наук
(повна назва кафедри (предметної, циклової комісії))

МАГІСТЕРСЬКА КВАЛІФІКАЦІЙНА РОБОТА

на тему:

«Інформаційна технологія надання рекомендацій щодо спортивних активностей»

Виконав: студент 2-го курсу, групи 3КН-23м
спеціальності 122 «Комп'ютерні науки»
(шифр і назва напряму підготовки, спеціальності)
_____ Лесков С. Д.
(прізвище та ініціали)

Керівник: к. т. н., доцент каф. КН
_____ Арсенюк І. Р.
(прізвище та ініціали)
«05» 12 2024 р.

Опонент: к. т. н., доцент каф. АПТ
_____ Кабачій В. В.
(прізвище та ініціали)
«05» 12 2024 р.

Допущено до захисту
Завідувач кафедри КН
д. т. н., проф. Яровий А. А.
(прізвище та ініціали)
«06» 12 2024 р.

Вінниця ВНТУ – 2024 рік

Вінницький національний технічний університет
Факультет інтелектуальних інформаційних технологій та автоматизації
Кафедра комп'ютерних наук
Рівень вищої освіти ІІ-й (магістерський)
Галузь знань – 12 «Інформаційні технології»
Спеціальність – 122 «Комп'ютерні науки»
Освітньо-професійна програма – «Системи штучного інтелекту»

ЗАТВЕРДЖУЮ
Завідувач кафедри КН
Д. т. н., проф. Яровий А. А.
11.10 2024 року

З А В Д А Н Н Я

НА МАГІСТЕРСЬКУ КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТУ

Лесков Сергій Дмитрович
(прізвище, ім'я, по батькові)

1. Тема роботи: «Інформаційна технологія надання рекомендацій щодо спортивних активностей»
Керівник роботи: к. т. н., доцент кафедри КН Арсенюк І. Р.
Затверджені наказом вищого навчального закладу від "17. 09." 2024 року № 310
2. Строк подання студентом роботи 11.11 2024 року
3. Вихідні дані до роботи: кількість параметрів характеристики користувача – не менше 9; вік користувачів додатку – не менше 6 років; частота збору даних користувача під час спортивних активностей – не рідше ніж кожні 5 хвилин; кількість користувачів для тестування – не менше 100 осіб; операційна система – мобільна (Android); мова програмування – об'єктно-орієнтована.
4. Зміст текстової частини:
вступ, аналіз сучасного стану надання рекомендацій щодо спортивних активностей, огляд існуючих систем-аналогів, програмна реалізація системи рекомендації, економічна частина, висновки, перелік використаних джерел, додатки.
5. Перелік ілюстративного матеріалу (з точним зазначенням обов'язкових креслень): алгоритм роботи програми надання рекомендацій щодо спортивних активностей, структурна схема функціонування інформаційної технології; UML діаграма класів, робочі вікна програми надання рекомендацій щодо спортивних активностей, результати тестування програми надання рекомендацій щодо спортивних активностей.

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийня
1-3	Арсенюк І. Р., к. т. н., доц. каф. КН		
4	Адлер О. О., к. т. н., доц. каф. ЕПВМ		

7. Дата видачі завдання 02.09. 2024 року

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів магістерської кваліфікаційної роботи	Строк виконання етапів роботи	Примітка
1	Аналіз предметної області рекомендацій щодо спортивних активностей	02.09.24 – 14.09.24	Розділ 1
2	Розробка інформаційної технології надання рекомендацій щодо спортивних активностей	15.09.24 – 25.09.24	Розділ 2
3	Програмна реалізація інформаційної технології надання рекомендацій	26.09.24 – 02.10.24	Розділ 3
4	Підготовка економічної частини	23.10.24 – 01.11.24	Розділ 4
5	Апробація результатів дослідження	02.11.24 – 04.11.24	Матеріали конференц авторське право на програму
6	Оформлення пояснювальної записки, графічного матеріалу та презентації	05.11.24 – 11.11.24	Пояснювальн записка, графічний матеріал, презентаці

Студент _____ (підпис) Лесков С.Д.

Керівник роботи _____ (підпис) Арсенюк І. Р.

АНОТАЦІЯ

УДК 004.42

Лесков С. Д. Інформаційна технологія надання рекомендацій щодо спортивних активностей. Магістерська кваліфікаційна робота зі спеціальності 122 Комп'ютерні науки, освітня програма – Системи штучного інтелекту. Вінниця: ВНТУ, 2024. 113 с.

Укр. мовою. Бібліогр.: 25 назв; рис.: 17; табл.: 15.

У магістерській кваліфікаційній роботі виконано аналіз існуючих програмних рішень у галузі спортивних активностей. Досліджено основні підходи щодо реалізації інформаційної технології. Здійснено аналіз аналогів, розглянуто методи та технології, які використовуються для вирішення подібних задач та обгрунтована доцільність розробки. Розроблено структурну модель інформаційної технології, що сприяє розширенню функціональних можливостей інформаційної технології.

Розрозблений додаток дає можливість користувачу отримати рекомендації щодо спортивних активностей, на основі вказаних користувачем даних, зі зручним та інтуїтивно зрозумілим інтерфейсом.

У роботі наведено результати досліджень та аналіз роботи розробленої інформаційної технології надання рекомендацій щодо спортивних активностей. Доведена економічна доцільність розробки.

Ключові слова: рекомендації щодо спортивних активностей, спортивні активності, машине навчання, інформаційна технологія, операційна система Android.

ABSTRACT

Leskov S. D. Information technology for providing recommendations for sports activities. Master's thesis on the specialty 122 Computer science, educational program – Artificial intelligence systems. Vinnytsia: VNTU, 2024. 113 p.

Ukraine language Bibliography: 25 titles; Fig.: 17; tab.: 15.

In the master's qualification work, an analysis of existing software solutions in the field of sports activities was performed. The main approaches to the implementation of information technology have been studied. An analysis of analogues was carried out, the methods and technologies used to solve similar problems were considered, and the reasonableness of the development was substantiated. A structural model of information technology has been developed, which contributes to the expansion of the functional capabilities of information technology.

The detailed application allows the user to receive recommendations for sports activities, based on the data specified by the user, with a convenient and intuitive interface.

The work presents the results of research and analysis of the work of the developed information technology for providing recommendations for sports activities. The economic feasibility of the development has been proven.

The illustrative part consists of 7 posters with simulation results.

Keywords: recommendations for sports activities, sports activities, machine learning, information technology, Android operating system

ЗМІСТ

ВСТУП

Актуальність теми. У сучасному світі фізична активність є важливим фактором підтримання здорового способу життя, але значна частина населення стикається з проблемою недостатньої мотивації та відсутності персоналізованих тренувальних програм, які б враховували індивідуальні особливості організму. Стандартні фітнес-додатки часто пропонують лише загальні рекомендації, що не враховують індивідуальних фізичних параметрів, таких як вага, зріст, рівень фізичної підготовки та медичні обмеження. Це може призводити до неефективності тренувань та підвищеного ризику травм.

Зростаюча популярність мобільних пристроїв на базі операційної системи Android створює нові можливості для розробки програмного забезпечення, що здатне запропонувати персоналізовані рішення для користувачів. ОС Android є однією з найпоширеніших мобільних платформ, що робить її ідеальною для створення додатків, які мають широке охоплення та можуть допомогти значній кількості користувачів підтримувати здоровий спосіб життя.

Одним з ключових аспектів актуальності є вибір мови програмування Kotlin, яка є офіційною мовою для розробки додатків під ОС Android. Kotlin дозволяє використовувати сучасні підходи до програмування, такі як асинхронність через корутини, що сприяє підвищенню продуктивності та зручності у розробці масштабованих додатків. Використання Kotlin у поєднанні з бібліотеками для машинного навчання, такими як TensorFlow Lite або ML Kit, дозволяє реалізувати алгоритми для адаптивної побудови тренувальних програм, враховуючи фізичні показники користувачів в реальному часі.

Актуальність дослідження також підкріплюється тенденцією до збільшення кількості користувачів, які прагнуть контролювати свою фізичну активність за допомогою мобільних додатків. Водночас існуючі рішення мають низку обмежень, таких як відсутність глибокої персоналізації та інтеграції з медичними даними користувача. Використання машинного навчання для створення персоналізованих рекомендацій може значно підвищити точність

тренувальних програм і сприяти досягненню користувачами кращих результатів, мінімізуючи ризики, пов'язані з фізичним перенавантаженням.

Таким чином, тема розробки інформаційної технології рекомендацій для фізичних активностей є актуальною з огляду на необхідність удосконалення сучасних тренувальних додатків. Використання технологій машинного навчання в середовищі Android з мовою програмування Kotlin дозволить створити ефективний інструмент для покращення мотивації користувачів і підвищення якості їх фізичної активності.

Зв'язок роботи з науковими програмами, планами, темами. Магістерська кваліфікаційна робота виконана відповідно до напряму наукових досліджень кафедри комп’ютерних наук Вінницького національного технічного університету 22 К1. "Розробка прикладних інтелектуальних інформаційних технологій та систем".

Мета та завдання досліджень. Мета магістерської кваліфікаційної роботи – розширення функціональних можливостей програмного додатку надання рекомендацій щодо спортивних активностей.

Для досягнення мети роботи потрібно виконати такі задачі:

– Здійснити аналіз існуючих систем надання рекомендацій щодо спортивних активностей, виявити їхні переваги та недоліки.

– Обґрунтувати вибір технологій і методів, що будуть використані для реалізації програмного додатку (вибір алгоритмів машинного навчання, середовища розробки та інструментів).

– Розробити структуру та архітектуру інформаційної системи, яка забезпечить адаптивне формування рекомендацій для користувачів на основі їхніх фізичних параметрів.

– Реалізувати інформаційну технологію рекомендацій у програмному додатку для ОС Android, використовуючи об’єктно орієнтовану мову програмування та інтегруючи бібліотеки для машинного навчання.

– Виконати тестування розробленого програмного продукту та здійснити його аналіз.

– Здійснити огляд існуючих програмних рішень.

Об’єкт дослідження – процес формування рекомендацій щодо спортивних активностей.

Предмет дослідження – програмне забезпечення та інформаційна технологія надання рекомендацій щодо спортивних активностей.

Методи дослідження. У роботі використані такі методи наукових досліджень: методи системного аналізу для дослідження існуючих систем рекомендацій щодо спортивних активностей; методи математичної статистики для аналізу та обробки фізичних даних користувачів, а також для оцінки ефективності роботи алгоритмів; методи об'єктно-орієнтованого програмування для розробки програмного додатку під ОС Android з використанням об’єктно орієнтованої мови.

Наукова новизна одержаних результатів.

Набула подальшого розвитку інформаційна технологія надання рекомендацій щодо спортивних активностей, яка на відміну від існуючих систем відрізняється розширеними функціональними можливостями та використанням алгоритмів машинного навчання, що дозволяє підвищити персоналізацію та точність тренувальних програм у мобільних додатках.

Практичне значення одержаних результатів полягає в тому, що на основі здійснених досліджень розроблено вдосконалено програмне забезпечення для надання персоналізованих рекомендацій щодо спортивних активностей.

Запропонована інформаційна технологія сприяє підвищенню точності та адаптивності тренувальних програм, зокрема:

– розроблено алгоритм адаптивного формування тренувальних програм на основі фізичних показників користувачів з використанням машинного навчання;

– створено програмний додаток під ОС Android, що використовує алгоритми машинного навчання для персоналізації рекомендацій щодо фізичних активностей.

Достовірність теоретичних положень магістерської кваліфікаційної роботи підтверджується коректністю постановки завдання, коректністю використання математичного апарату методів дослідження, експериментальними дослідженнями тестування програмної реалізації інформаційної технології надання рекомендацій щодо спортивних активностей. Адекватність розроблених математичних моделей підтверджується результатами експериментальних досліджень.

Особистий внесок здобувача. Усі результати, що наведені у магістерській кваліфікаційній роботі, отримані самостійно. У працях, які написано у співавторстві, здобувачу належать: аналіз процесу розв'язання задачі надання рекомендацій щодо спортивних активностей [1].

Апробація результатів роботи. Результати досліджень було апробовано на Міжнародній мультидисциплінарній науковій інтернет-конференції "Світ наукових досліджень" (м. Тернопіль, Україна, м. Ополе, Польща, 22-23 жовтня 2024 р.).

Публікації. За результатами магістерської кваліфікаційної роботи опубліковано матеріали конференції на науково-технічних конференції [1] та подано заявку на реєстрацію авторського права на твір у формі комп’ютерної програми – «Програмний додаток рекомендацій щодо спортивних активностей».

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ РЕКОМЕНДАЦІЙ ЩОДО СПОРТИВНИХ АКТИВНОСТЕЙ

1.1 Аналіз сучасного стану надання рекомендацій щодо спортивних активностей

В умовах сучасного життя багато людей працюють у сидячому режимі, що збільшує ризики для здоров'я, пов'язані з гіподинамією. Тому важливо створювати індивідуальні тренувальні програми, які будуть враховувати фізичні можливості користувачів, їхні цілі та поточний стан здоров'я.

Високопродуктивні обчислювальні системи, алгоритми машинного навчання та мобільні платформи створюють умови для впровадження адаптивних тренувальних програм, які враховують індивідуальні фізичні параметри, стан здоров’я та цілі користувачів. Інформаційна технологія надання рекомендацій дозволяє інтегрувати фізичні показники зібрані за допомогою фітнес-трекерів, сенсорів та мобільних додатків, що створює можливості для динамічного моніторингу і корекції тренувань.

Системи рекомендацій для спортивних активностей на основі машинного навчання дозволяють аналізувати фізичні дані користувачів, такі як вага, зріст, рівень активності, і надавати їм тренувальні програми, адаптовані до їхніх індивідуальних потреб. Вони допомагають людям досягати своїх спортивних цілей, підтримувати мотивацію та уникати перевантажень. Окрім того, такі системи можуть бути інтегровані з фітнес-трекерами, що дозволяє отримувати актуальні дані в режимі реального часу і коригувати тренувальні плани на їх основі [2].

Системи розробки рекомендацій щодо спортивних активностей мають широкий спектр застосування, серед яких можна виділити:

- Фітнес-індустрію, де додатки для тренувань використовуються як професійними тренерами, так і любителями спорту. Вони допомагають створювати індивідуальні плани тренувань, відстежувати прогрес і

досягати конкретних фізичних цілей (схуднення, нарощування м'язової маси тощо).

− Медичні заклади для реабілітаційних програм пацієнтів. На основі медичних показників та фізичної активності пацієнтів можна створювати індивідуальні програми, спрямовані на відновлення після операцій або хвороб.

− Для підтримки фізичної активності своїх співробітників з метою зменшення стресу, підвищення продуктивності та поліпшення загального самопочуття. Системи рекомендацій щодо активності можуть допомогти в створенні таких програм.

− Для професійних спортсменів та команд системи рекомендацій допомагають створювати програми підготовки до змагань, враховуючи індивідуальні фізичні дані та мету тренувань.

− Персональними тренерами для розробки більш точних та індивідуалізованих програм тренувань для клієнтів, дозволяючи відстежувати їхній прогрес та коригувати програми на основі отриманих даних.

1.2 Огляд існуючих систем-аналогів

Системи рекомендацій є важливим інструментом для підтримки фізичної активності та здорового способу життя. Вони надають користувачам інформацію про те, як оптимізувати тренувальні програми, зважаючи на фізичний стан, індивідуальні потреби та цілі. Сучасні системи рекомендацій орієнтовані на аналіз фізичної активності користувачів за допомогою різних датчиків і сенсорів. Вони збирають дані про кроки, частоту серцевих скорочень, витрачені калорії, а також фіксують інші показники під час тренувань.

Існує велика кількість популярних систем, які надають статичні рекомендації щодо спортивних активностей. Ці системи використовують дані про фізичну активність користувача для генерації тренувальних програм

орієнтовані на конкретний вид спорту, надаючи індивідуальні плани, виходячи з фізичних параметрів або цілей. У цьому підрозділі розглянемо найбільш популярні системи-аналоги, а також їхні переваги та недоліки.

Nike Training Club (NTC) – це популярний фітнес-додаток, розроблений для забезпечення користувачів тренувальними програмами, адаптованими до їхнього рівня фізичної підготовки. Додаток пропонує широкий спектр вправ, які можна виконувати як вдома, так і в спортивному залі, що робить його зручним інструментом для занять спортом незалежно від умов. Однією з ключових переваг NTC є різноманітність тренувальних планів, які враховують різні рівні фізичної підготовки, що дозволяє задовольнити потреби як новачків, так і досвідчених спортсменів.

Особливо слід відзначити високий рівень професіоналізму інструкцій, адже всі тренування розроблені тренерами Nike, що забезпечує якісний підхід до виконання вправ. Ще однією важливою особливістю є те, що додаток є безкоштовним і доступним для більшості користувачів, що сприяє його популярності серед широкого кола людей.

Проте NTC має й деякі недоліки, які обмежують його функціональність. Одним із найбільш помітних є відсутність динамічної адаптації тренувань у реальному часі. Це означає, що додаток не враховує фізичні дані користувача, такі як пульс або витрачені калорії, для коригування тренувань у процесі їх виконання. Крім того, інтеграція з трекерами активності є обмеженою, що може створити незручності для тих, хто звик використовувати носимі пристрої для моніторингу своїх досягнень.

Ще одним аспектом, який потребує вдосконалення, є персоналізація. Хоча програми тренувань враховують рівень підготовки користувача, можливості підлаштування під індивідуальні цілі чи фізичні обмеження залишаються недостатньо розвиненими. Це може стати перешкодою для користувачів з особливими потребами або тих, хто прагне отримати більш точні й адаптовані рекомендації.

Таким чином, Nike Training Club є корисним додатком для загального використання, особливо для початківців і тих, хто шукає доступний інструмент для занять спортом. Водночас його обмежена адаптивність і інтеграція залишають простір для покращень, які могли б зробити додаток ще більш універсальним і ефективним.

Загальний вигляд додатку Nike Training (рис. 1.1).

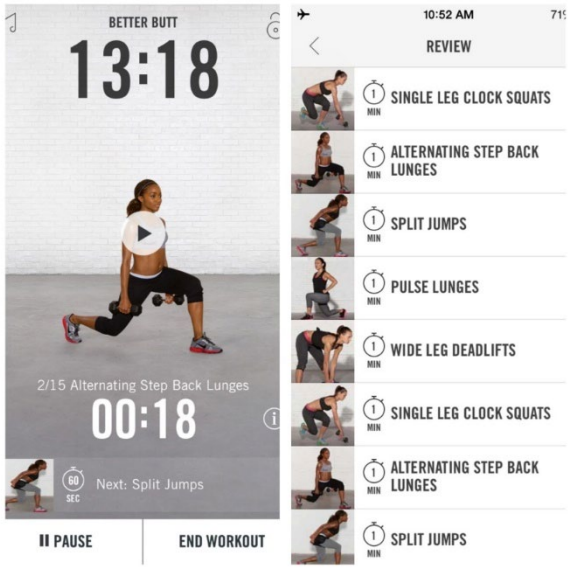

Рисунок 1.1 – Програма Nike Training Club

MyFitnessPal – це інструмент для моніторингу харчування та фізичної активності, який став популярним серед користувачів, що прагнуть підтримувати здоровий спосіб життя. Основною перевагою додатку є його велика база продуктів, яка дозволяє точно підраховувати калорії, макро- та мікронутрієнти, спрощуючи контроль за харчуванням. Завдяки цій функції MyFitnessPal є зручним рішенням для тих, хто прагне покращити свій раціон, досягти певних цілей у вазі або просто підтримувати баланс у харчуванні.

Ще однією важливою особливістю додатку є інтеграція з різноманітними фітнес-трекерами, що забезпечує автоматичний збір даних про фізичну активність, такі як кількість кроків, витрачені калорії чи тривалість тренувань. Ця інтеграція робить додаток корисним не лише для контролю харчування, але й для загального моніторингу здоров'я. Окрім того, MyFitnessPal пропонує

інструменти для планування харчування та тренувань, що дозволяють користувачам досягати встановлених цілей через системний підхід.

Попри всі переваги, додаток має і деякі обмеження. Його основний фокус залишається на харчуванні, що обмежує його можливості для користувачів, які шукають рішення для комплексного управління тренуваннями. MyFitnessPal не надає індивідуальних тренувальних планів, адаптованих до фізичного стану користувача в реальному часі, що робить його менш корисним для тих, хто очікує глибокої персоналізації у сфері фізичних активностей.

Крім того, інтеграція з популярними фітнес-трекерами є не завжди повною, що може створювати труднощі для користувачів, які вже мають досвід роботи з іншими екосистемами. Відсутність глибокої синхронізації може призвести до часткової втрати даних або потреби у ручному введенні інформації. Таким чином, MyFitnessPal є ефективним інструментом для тих, хто зосереджений на харчуванні та базовому моніторингу активності. Водночас його обмеження у сфері персоналізованих тренувальних планів та інтеграції з деякими пристроями свідчать про наявність потенціалу для вдосконалення, що могло б зробити додаток більш універсальним.

Загальний вигляд додатку MyFitnessPal (рис. 1.2).

Рисунок 1.2 – Приклад роботи MyFitnessPal

Google Fit – це зручний і безкоштовний додаток, розроблений для відстеження фізичної активності користувачів. Його ключовою перевагою є широка інтеграція з різними фітнес-трекерами та пристроями Android, що робить його універсальним рішенням для збору даних про фізичну активність. Google Fit дозволяє відстежувати такі важливі показники, як кількість кроків, витрачені калорії, пульс та інші, що забезпечує повний огляд фізичного стану користувача.

Додаток є повністю безкоштовним для власників пристроїв на базі Android, що робить його доступним для широкого кола користувачів. Завдяки простоті використання Google Fit стає ефективним інструментом для базового моніторингу активності та здоров'я, особливо для тих, хто тільки починає стежити за своїм фізичним станом.

Однак, незважаючи на свої переваги, Google Fit має певні недоліки, які обмежують його функціональність. Додаток не надає індивідуальних тренувальних програм, побудованих на основі фізичних даних користувача, що робить його менш корисним для користувачів, які шукають персоналізовані рекомендації. Основний акцент додатку зроблено на збиранні та відображенні даних, але він не пропонує інтерактивних тренувальних планів або рекомендацій, що б адаптувалися до змін у фізичному стані користувача.

Обмежені можливості динамічного коригування тренувань також є слабким місцем Google Fit. Додаток не здатний оперативно аналізувати поточний стан користувача та відповідно змінювати тренувальні рекомендації, що знижує його ефективність для активних спортсменів або користувачів із спеціальними потребами.

Таким чином, Google Fit є гарним вибором для базового моніторингу фізичної активності завдяки інтеграції з різними пристроями та простоті використання. Проте його можливості залишаються обмеженими, особливо у контексті персоналізації та адаптивності тренувального процесу, що створює простір для подальшого вдосконалення.

Загальний вигляд додатку Google Fit (рис. 1.3).

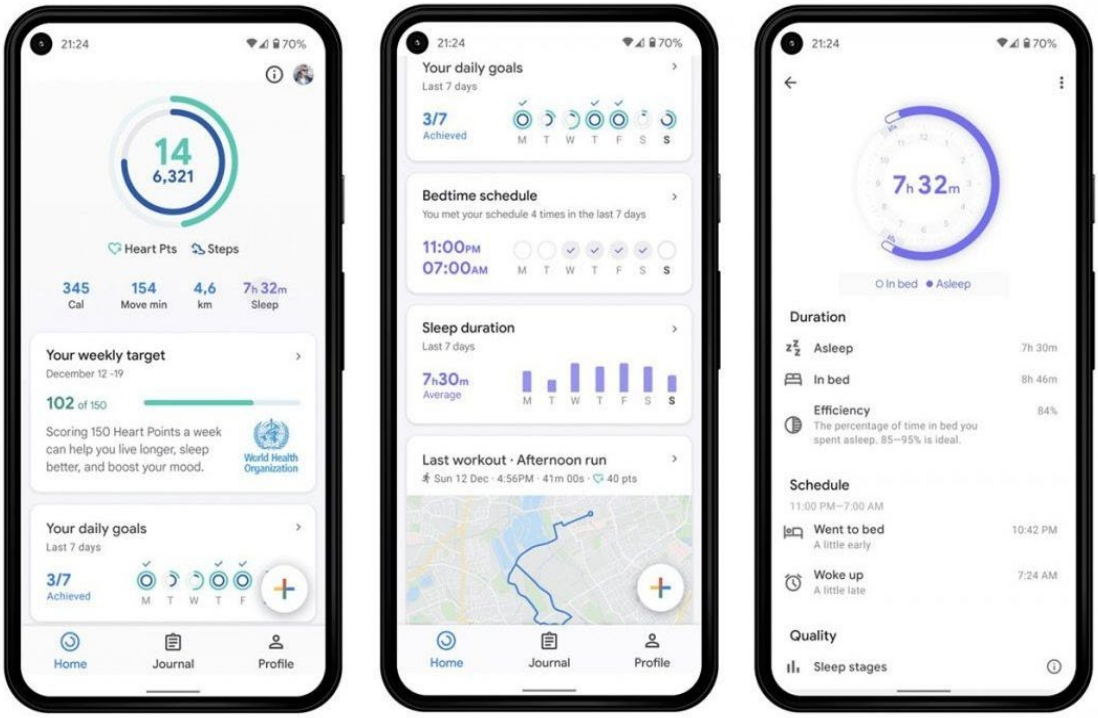

Рисунок 1.3 – Приклад роботи Google Fit

На таблиці 1.1 наведено переваги та недоліки систем-аналогів.

Таблиця 1.1 – Порівняльна таблиця існуючих систем-аналогів

Додаток	Характеристики
1	2
Nike Training Club	+ Широкий вибір тренувальних програм. + Професійні інструкції. - Відсутність адаптації в реальному часі. - Обмежена інтеграція з трекерами. - Недостатня персоналізація.

Продовження таблиці 1.1

1	2
MyFitnessPal	+ Велика база продуктів для підрахунку калорій. + Інтеграція з фітнес-трекерами. + Планування харчування та тренувань. - Основний фокус на харчуванні, а не тренуваннях. - Відсутність адаптивних тренувальних програм.
Google Fit	+ Підтримка багатьох пристроїв. + Моніторинг різних показників. + Безкоштовний доступ для користувачів Android. - Відсутність індивідуальних тренувальних програм. - Основний акцент на моніторингу, а не рекомендаціях. - Немає функцій динамічного коригування тренувань.

Слід зазначити, що наведені системи орієнтовані на більш вузький спектр можливостей, також враховуючи статичні рекомендації які надаються суто по тематиці поточного додатку.

1.3 Застосування сучасних підходів надання рекомендацій щодо спортивних активностей

Система надання рекомендацій щодо спортивних активностей у мобільному додатку на Android потребує вибору й налаштування відповідної математичної моделі машинного навчання, яка забезпечить високу точність персоналізованих рекомендацій. Завдання включає обробку вхідних даних користувача, таких як вік, вага, стан здоров'я, рівень фізичної підготовки тощо, для прогнозування відповідної спортивної активності та її інтенсивності.

Для порівняння було обрано кілька популярних додатків, які забезпечують користувачів рекомендаціями щодо фізичних активностей на основі персональних даних. Основними критеріями для вибору аналогу були:

— Наявність функціоналу для збору фізичних параметрів користувача (вага, зріст, вік, рівень фізичної підготовки тощо).

— Застосування алгоритмів машинного навчання або інших технологій для формування індивідуальних тренувальних програм.

— Можливість відстеження прогресу користувача та адаптації рекомендацій залежно від результатів тренувань.

— Інтеграція з фітнес-трекерами та іншими носимими пристроями для автоматичного збору даних про активність користувача.

У контексті розробки мобільного додатку під Android для надання рекомендацій щодо спортивних активностей основною задачею є вибір алгоритму машинного навчання, який здатен працювати з послідовними даними користувача, такими як фізичні параметри, історія тренувань та дані з фітнес-трекерів.

Для цього необхідно враховувати не лише складність задачі, але й обмежені обчислювальні ресурси мобільних пристроїв:

− Обробка послідовних даних з фітнес-трекерів зазвичай мають часову залежність, тому використовувати класичні методи машинного навчання, як-от лінійні моделі, не є оптимальним вибором. Для цього потрібні рекурентні нейронні мережі (RNN), оскільки вони здатні працювати з послідовними даними. Водночас застосування моделі LSTM (Long Short-Term Memory) може покращити результати, оскільки вона здатна зберігати довготривалу інформацію [5].

− Розширення можливостей RNN через TensorFlow Lite: TensorFlow Lite дозволяє використовувати моделі на основі RNN/LSTM під Android без значного споживання ресурсів. Ця бібліотека оптимізована для мобільних пристроїв, що дає змогу використовувати складні моделі машинного навчання, такі як LSTM, прямо на мобільному пристрої,

дозволяючи системі динамічно адаптувати рекомендації в режимі реального часу [6].

Схема роботи TensorFlow Lite зображена на рисунку 1.4.

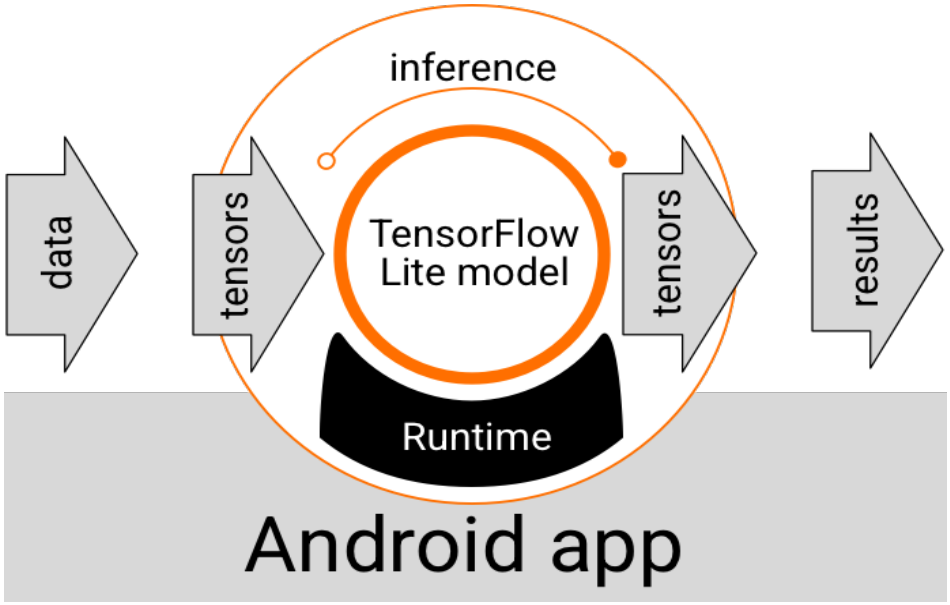

Рисунок 1.4 – Схема роботи TensorFlow Lite

– Класифікаційні алгоритми: логістична регресія або дерева рішень можуть бути застосовані для первинної класифікації користувачів за рівнем фізичної активності. Наприклад, для нових користувачів, які не мають історичних даних, такі алгоритми можуть бути використані для класифікації за введеними параметрами (вага, зріст, вік) [7].

– Підкріплювальне навчання: для покращення результатів системи у випадках, коли потрібно врахувати зворотний зв'язок від користувача, можна застосовувати алгоритми підкріплювального навчання. Це дозволить системі оптимізувати рекомендації на основі прогресу користувача під час тренувань. Такий підхід особливо корисний для мобільних додатків, які мають функцію постійного відстеження результатів тренувань [8].

На основі аналізу особливостей задачі, найбільш підходящими є рекурентні нейронні мережі (RNN) та LSTM, оскільки ці моделі здатні обробляти послідовні

дані і зберігати важливі характеристики з минулих тренувань. TensorFlow Lite забезпечує оптимізацію використання цих моделей на мобільних пристроях з мінімальними ресурсами [9].

Хоча обидві технології належать до класу рекурентних мереж, їхня архітектура, можливості та ефективність у вирішенні складних завдань мають суттєві відмінності. Наведена таблиця демонструє порівняльні характеристики RNN та LSTM, підкреслюючи їхні переваги та недоліки.

Таблиця 1.2 – порівняльна характеристика моделей нейроних мереж

Назва	Характеристики
Рекурентні нейронні мережі (RNN)	− зручність для обробки послідовних даних; − ефективність для задач із короткотривалими залежностями схильність до затухання або вибухання градієнта; − обмежена здатність зберігати контекст; − проблеми з обробкою довготривалих залежностей.
LSTM (Long Short-Term Memory)	− Здатність працювати з короткотривалими й довготривалими залежностями − використання комірок пам’яті для збереження контексту − усунення проблеми затухання градієнта − складність реалізації − високі вимоги до обчислювальних ресурсів – тривале налаштування та тренування.

Проаналізовані характеристики показують, що RNN є більш простими і підходять для задач із короткими залежностями у даних, однак вони мають обмеження у роботі з довготривалими залежностями через затухання градієнта. LSTM, завдяки використанню комірок пам’яті та шлюзів, ефективно справляються з цими проблемами, але вимагають більших обчислювальних ресурсів.

Таким чином, вибір між RNN і LSTM залежить від конкретної задачі: для простих послідовностей RNN можуть бути більш доцільними, тоді як для задач із довготривалими залежностями LSTM є оптимальним вибором.

1.4 Постановка задачі дослідження

Забезпечити врахування індивідуальних особливостей стану здоров’я (нещодавні хвороби, вади з дитинста тощо), в результаті чого буде надано список рекомендацій щодо спортивних активностей, серед яких користувач має можливість обрати ту рекомендацію, яка йому підійде.

Задача формування рекомендацій спрямована на вирішення проблем недостатньої персоналізації в існуючих системах, підвищення ефективності тренувального процесу та мінімізацію ризиків для здоров'я користувачів. Реалізація цієї задачі дозволить користувачам отримувати персоналізовані рекомендації, які будуть адаптовані до їхніх індивідуальних особливостей і допоможуть досягти кращих результатів у тренуваннях.

Сучасний підхід до підтримання здорового способу життя потребує використання індивідуальних програм тренувань, які базуються на фізичних характеристиках і рівні активності користувача. Проте більшість існуючих систем для надання рекомендацій щодо спортивних активностей пропонують універсальні рішення, що не враховують індивідуальних особливостей користувачів, таких як вага, зріст, вік, медичні обмеження, попередній рівень підготовки тощо. Це значно знижує ефективність таких рекомендацій і може призводити до травм або недостатньої результативності.

Отже, необхідно розробити інформаційну технологію, яка б забезпечувала створення індивідуальних тренувальних програм для кожного користувача, виходячи з його індивідуальних параметрів.

Основними складовими цієї задачі є:

– Збір даних користувача для надання точних рекомендацій необхідно автоматично отримувати дані про фізичні показники користувачів, такі як

вага, зріст, індекс маси тіла, рівень фізичної активності та історія тренувань. Ці дані можуть бути отримані з мобільних пристроїв (смартфонів, фітнес-трекерів) та сенсорів, що відстежують фізичну активність користувача.

– Після отримання даних, система повинна аналізувати їх за допомогою алгоритмів машинного навчання. Це дозволить оцінити поточний стан користувача і визначити оптимальні навантаження, які будуть не лише ефективними, але й безпечними. Важливо враховувати такі параметри, як рівень фізичної підготовки, стан здоров'я, попередній тренувальний досвід та інші індивідуальні фактори.

– Формування персоналізованих рекомендацій для створення персоналізованих програм тренувань. Вони повинні адаптуватися в режимі реального часу до змін у фізичному стані користувача, рівня його активності та прогресу у виконанні програм. Це дозволяє формувати рекомендації, які відповідають поточним потребам та можливостям користувача, забезпечуючи ефективність і мотивацію до регулярних занять спортом.

– Інтеграція з мобільними додатками передбачає розробку програмного додатку для ОС Android з використанням мови програмування Kotlin. Цей додаток повинен забезпечувати зручний інтерфейс для користувачів, інтеграцію з фітнес-трекерами та іншими носимими пристроями, а також можливість автоматичного оновлення рекомендацій на основі отриманих даних.

1.5 Висновок до розділу 1

У першому розділі було здійснено аналіз предметної області рекомендацій щодо спортивних активностей. Розглянуто сучасний стан систем, які пропонують індивідуальні тренувальні програми, а також проведено огляд основних технологій, що використовуються в цій сфері. Особливу увагу

приділено алгоритмам машинного навчання, які дозволяють досягати високої точності та персоналізації рекомендацій.

Аналіз існуючих систем-аналогів, таких як Nike Training Club, MyFitnessPal та Google Fit, дозволив виявити їхні переваги та недоліки. Серед ключових переваг були відзначені доступність, широкий функціонал і інтеграція з різними трекерами активності. Водночас виявлено обмеження, зокрема недостатню адаптивність рекомендацій у реальному часі, обмежену персоналізацію та обмеження в динамічному моніторингу.

Було сформульовано основну проблему – відсутність у сучасних системах можливості створювати глибоко персоналізовані тренувальні програми, що враховують унікальні показники користувачів. Це підкреслює необхідність створення інформаційної технології, яка б забезпечувала не лише статичні рекомендації, але й динамічне коригування тренувальних планів.

Постановка задачі дослідження передбачає розробку мобільного додатку з використанням алгоритмів машинного навчання, що дозволить інтегрувати сучасні методи обробки даних у тренувальний процес. У наступних розділах буде запропоновано архітектуру такої системи, а також представлено її програмну реалізацію.

2 РОЗРОБКА ІНФОРМАЦІЙНОЇ ТЕХНОЛОГІЇ РОЗРОБКИ РЕКОМЕНДАЦІЙ ЩОДО СПОРТИВНИХ АКТИВНОСТЕЙ

2.1 Проектування архітектури мобільного додатку для надання рекомендацій щодо спортивних активностей

Для проектування архітектури мобільного додатку для надання рекомендацій щодо спортивних активностей є важливим етапом у розробці. Основною метою проектування архітектури є створення структури додатку, яка буде забезпечувати його ефективність, розширюваність, легкість у тестуванні та підтримці.

Архітектури проєктування Android-додатків, такі як MVC (Model-View-Controller) [12], MVP (Model-View-Presenter) [13], MVVM та MVI (Model-View-Intent) [14], є ключовими підходами до забезпечення чіткого поділу обов'язків між компонентами застосунку. MVC є класичним підходом, що розділяє програму на модель (дані та логіку), представлення (інтерфейс користувача) та контролер (обробка подій). Проте у контексті Android MVC демонструє суттєві недоліки, зокрема сильну зв'язаність компонентів і ускладненість тестування. MVP удосконалює цей підхід, делегуючи всю логіку роботи з інтерфейсом презентеру, що суттєво знижує залежність між View і Model, роблячи проєктування більш модульним. Однак, Presenter може бути перевантаженим логікою у складних застосунках.

MVVM є сучасним підходом, що забезпечує двостороннє зв'язування даних між інтерфейсом і моделлю за допомогою компонента ViewModel. Такий підхід сприяє слабкій зв'язаності компонентів, що спрощує підтримку та масштабованість додатків. Крім того, MVVM добре інтегрується з бібліотеками Jetpack, такими як LiveData та Data Binding, завдяки чому є ефективним у розробці сучасних Android-застосунків. MVI фокусується на односпрямованому потоці даних, де стан програми контролюється через уніфікований потік подій, що робить архітектуру прозорою, але більш складною для реалізації у простих

проєктах. Завдяки централізованому управлінню станами, чіткому розділенню відповідальностей і реактивному підходу, MVI сприяє створенню стабільного й передбачуваного програмного забезпечення. Проте його використання доцільне переважно для середніх і великих проектів через складність реалізації та вищі вимоги до розробників.

У таблиці 2.1 наведено характеристики наявних архітектурних рішень у розробці мобільних додатків під Android.

Таблиця 2.1 – Порівняльна характеристика існуючих архітектурних рішень

Архітектура	Характеристики
1	2
MVC (Model-View-Controller)	- Поділ відповідальності між Model (дані та логіка), View (інтерфейс користувача) та Controller (обробка подій). - Взаємодія: View напряму взаємодіє з Controller і Model. - Використання: класична архітектура, але в Android може спричиняти сильну зв'язність між компонентами. - Недоліки: важкість тестування, можливий перерозподіл логіки між View і Controller.
MVP (Model-View-Presenter)	- Поділ на Model (дані та логіка), View (інтерфейс) та Presenter (логіка представлення). - Взаємодія: View взаємодіє з Presenter, а Presenter - з Model. - Використання: значно зменшує зв'язність, дозволяє легше реалізувати юніт-тести.

Продовження таблиці 2.1

1	2
MVVM (Model-View-ViewModel)	- Поділ на Model (дані та логіка), View (інтерфейс) і ViewModel (логіка зв'язку між View і Model). - Підходить для проектів з використанням Jetpack, LiveData, або RxJava. - ViewModel забезпечує двостороннє зв'язування даних між View і Model. - Легко тестується, завдяки відсутності прив'язки до View. - Може ускладнитися через використання складних патернів.
MVI (Model-View-Intent)	- Орієнтована на уніфікований потік даних. - Intent передає події користувача у вигляді односпрямованого потоку, Model генерує стан, який рендериться View. - Популярна у проектах із складним станом або реактивним підходом. - Прозорий потік даних, легкість відстеження змін. - Високий рівень складності для простих додатків.

Слід зазначити, що з наведених архітектурних рішень у розробці мобільного додатку доцільне використання архітекурного патерну MVVM, оскільки сучасній розробці мобільних додатків MVVM є найбільш актуальним підходом завдяки своїй інтеграції з екосистемою Jetpack, що забезпечує ефективне управління станом і даними в динамічних інтерфейсах. Завдяки слабкій зв’язаності компонентів та легкості тестування ViewModel, MVVM дозволяє створювати масштабовані, модульні та легко підтримувані програми. [9] Ця архітектура також сприяє швидшій розробці завдяки автоматизації завдань, таких як зв’язування даних, що критично для великих і складних

проєктів. Таким чином, MVVM поєднує в собі зручність, сучасні технології та адаптивність, що робить її незамінною для сучасної Android-розробки.

Архітектурний патерн MVVM має наступні переваги:

1) Дозволяє здійснювати розподіл відповідальності між компонентами архітектури. Модель (Model) відповідає за бізнес-логіку та доступ до даних. Представлення (View) відповідає за відображення даних та взаємодію з користувачем. ViewModel забезпечує зв'язок між моделлю та представленням, обробляє події та оновлює дані.

2) Дозволяє здійснити ефективне тестування компонентів окремо. Модель можна тестувати незалежно від інтерфейсу користувача. ViewModel можна тестувати за допомогою модульних тестів, оскільки вона не залежить від платформи. Представлення можна тестувати за допомогою інструментів для тестування інтерфейсу користувача.

3) Забезпечує гнучкість та розширюваність додатків завдяки розділенню відповідальностей, можна змінювати та розширювати компоненти архітектури окремо, не впливаючи на решту. Наприклад, можна змінювати логіку моделі, не змінюючи код представлення, або замінювати представлення без впливу на модель.

4) Підтримує декларативний підхід до розробки завдяки зв'язуванню даних (Data Binding), можна встановлювати зв'язок між елементами інтерфейсу та даними моделі, що дозволяє автоматично оновлювати відображення даних при їх зміні. Це спрощує розробку та підтримку інтерфейсу користувача [6].

5) Є одним з найпопулярніших архітектурних патернів у мобільній розробці, особливо в контексті розробки на платформі Android. Він інтегрується з багатьма фреймворками та бібліотеками, такими як Android Architecture Components і Data Binding, що спрощує розробку додатків та забезпечує кращу підтримку.

6) Забеспечує ефективну взаємодію з даними, зокрема з використанням асинхронних операцій та обробки помилок. ViewModel може керувати

завантаженням даних з мережі або збереженням в базі даних, а також реагувати на зміни даних та оновлювати інтерфейс користувача.

7) Дозволяє чітко розділити логіку відображення та бізнес-логіку. Представлення відповідає лише за відображення даних та реагування на дії користувача, тоді як бізнес-логіка зосереджена в ViewModel і моделі. Це спрощує розробку та розуміння коду, а також дозволяє змінювати інтерфейс користувача без зміни бізнес-логіки.

Використання архітектурного патерну MVVM (рис. 2.1) має багато переваг, які полегшують розробку, тестування та підтримку мобільних додатків. Він дозволяє розділити відповідальності між компонентами архітектури, забезпечує гнучкість та розширюваність додатків, підтримує декларативне програмування та ефективну взаємодію з даними.

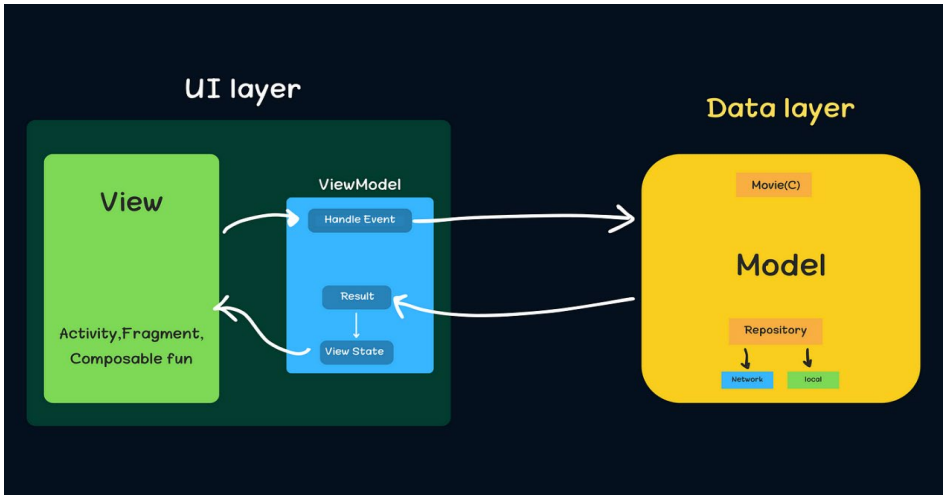

Рисунок 2.1 – Архітектура мобільного додатку Model-View-ViewModel

Усі складові частини архітектури (див. рис. 2.1) MVVM взаємодіють між собою згідно з такою послідовністю:

1) Користувач взаємодіє з View, виконуючи дії на інтерфейсі.

2) View передає ці дії до ViewModel, використовуючи прив'язку даних або команди (bindings/commands).

3) ViewModel отримує ці дії і виконує відповідну бізнес-логіку, яка може включати отримання або оновлення даних з Model.

4) ViewModel оновлює дані в своїх змінних, які зв'язані з View за допомогою прив'язки даних.

5) Зміна даних у ViewModel автоматично оновлює відображення відповідних елементів у View.

6) При необхідності, ViewModel може виконати додаткові дії, такі як збереження даних до бази даних або виклик зовнішніх сервісів.

Ця послідовність дозволяє забезпечити розділення відповідальностей та забезпечити зручну взаємодію між користувачем, View і ViewModel. ViewModel виступає в якості посередника, що забезпечує обмін даними та комунікацію між View та Model. Прив'язка даних дозволяє автоматично оновлювати відображення у View при зміні даних у ViewModel, що спрощує синхронізацію та забезпечує актуальність відображення.

Цей принциповий зв'язок, де ViewModel відповідає за управління даними та комунікацію між View та Model, є основою роботи архітектурного патерну MVVM. Він дозволяє забезпечити відокремлення бізнес-логіки від інтерфейсу користувача та зручне керування станом даних у мобільних додатках.

Актуальність архітектурного патерну MVVM (Model-View-ViewModel) полягає в його відповідності сучасним вимогам мобільної розробки та його внутрішній потужності. Враховуючи зростання складності та функціональності мобільних додатків, MVVM стає незамінним інструментом для забезпечення ефективності та підтримки додатків.

Завдяки розділенню відповідальностей між компонентами архітектури (модель, представлення та ViewModel), MVVM дозволяє покращити розширюваність та підтримку додатків. Кожен компонент виконує свою функцію і може бути модифікований або замінений незалежно від інших компонентів, що спрощує розробку та утримання коду.

Архітектурний патерн MVVM (Model-View-ViewModel) вирішує проблему ефективного управління даними та їх відображенням у мобільних додатках. Він забезпечує розділення відповідальностей між компонентами

архітектури, що дозволяє керувати станом даних, реагувати на їх зміни та оновлювати інтерфейс користувача шляхом зв'язування даних та відображення. Model представляє собою джерело даних та логіку їх обробки. ViewModel виступає посередником між Model та View, виконуючи функції управління даними та бізнес-логікою. View відображає дані та реагує на дії користувача.

MVVM дозволяє зручно керувати взаємодією з даними, оскільки ViewModel забезпечує збереження та оновлення стану даних, а також реагує на їх зміни безпосередньо з Model. Це усуває проблему ручного оновлення інтерфейсу користувача та спрощує роботу зі змінними даних.

Крім того, MVVM дозволяє забезпечити відокремлення бізнес-логіки від представлення даних, що полегшує тестування та розширення додатків. ViewModel може бути протестована незалежно від View, що спрощує процес тестування та забезпечує стабільність роботи додатку.

Архітектурний патерн MVVM вирішує проблему ефективного управління даними та їх відображенням у мобільних додатках шляхом розділення відповідальностей та забезпечення зручного керування станом даних та інтерфейсом користувача [9].

Під час розробки мобільного додатку з використанням архітектурного патерну MVVM, доцільно використати такі бібліотеки:

– Retrofit застосовується для здійснення мережевих викликів і отримання даних з віддаленого сервера. Можна використовувати анотації Retrofit для опису API-інтерфейсів та конфігурувати серіалізацію/десеріалізацію даних [10].

– Фреймворк Dagger-Hilt використовується для спрощення управління залежностями в додатку. Він забезпечує автоматичне генерування фабрик та контейнерів для впровадження залежностей [11].

– OkHttp призначенийй у використанні як HTTP-клієнт для взаємодії з сервером. Забезпечує можливість налаштування запитів, обробку статусу відповідей, перехоплення та обробку мережевих помилок [12].

– Kotlin-Coroutines дозволяє використовувати асинхронне

програмування у стилі сопрограм для виконання операцій введення/виведення та роботи з потоками без блокування основного потоку. Це допомагає уникнути проблем з блокуванням та забезпечити плавну роботу додатку [13].

– ViewModel є частиною архітектури MVVM і використовується для збереження та управління даними, пов'язаними з UI. Вона забезпечує стабільність даних під час поворотів екрану та управляє комунікацією між моделлю та відображенням [14].

– LiveData є реактивним компонентом, що надає спостерігачам (наприклад, UI-компонентам) можливість слідкувати за змінами даних, що зберігаються в ViewModel. Вона автоматично оновлює UI, коли данізмінюються [15].

– Gson використовується для серіалізації/десеріалізації об'єктів Java у формат JSON. Вона дозволяє легко перетворювати об'єкти в JSON-рядки та навпаки, спрощуючи обробку даних з віддалених джерел [16].

Ці бібліотеки можна поєднувати та використовувати разом для створення потужних мобільних додатків, використовуючи підхід MVVM для забезпечення модульності, легкості тестування та ефективності розробки.

У наведеній нижче діаграмі компонентів (рис. 2.2) ми можемо розглянути взаємодію компонентів, які реалізують коректну роботу мобільного додатку на всіх рівнях з’єднання.

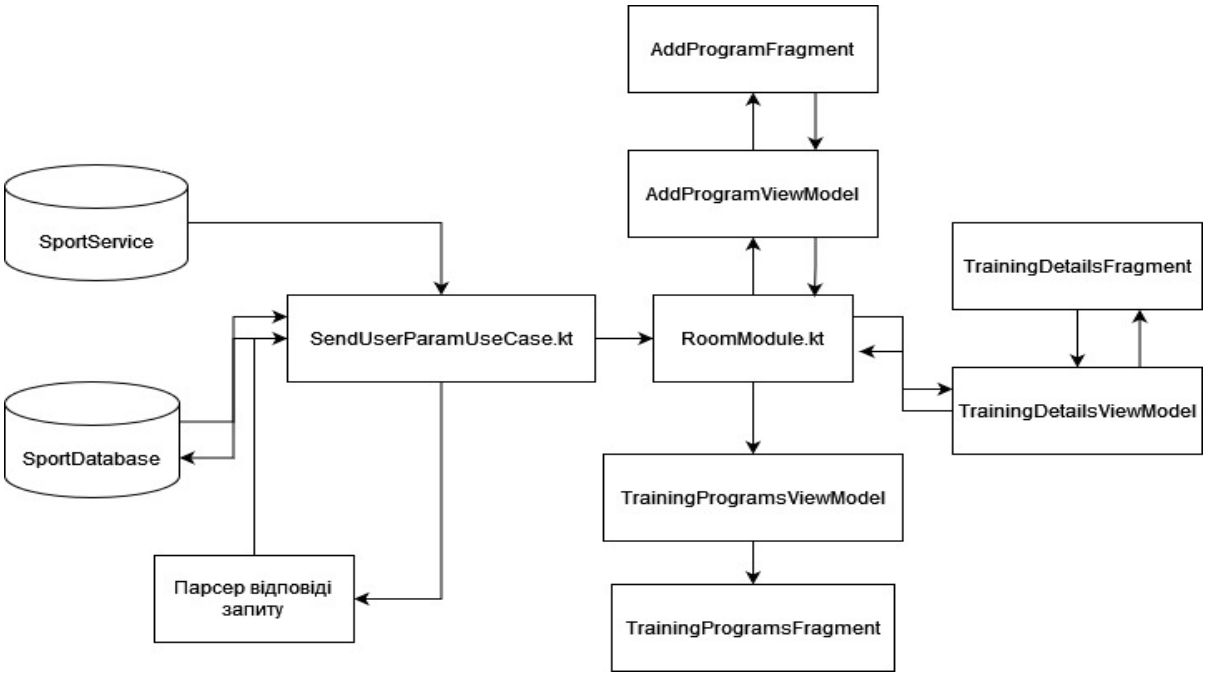

Рисунок 2.2 – Діаграма компонентів програмного додатку для надання рекомендацій щодо спортивних активностей

На рисунку 2.2 зображено діаграму, яка відображає як компоненти проекту взаємодіють між собою для коректної роботи мобільного додатку для користувацького читання, та як вони взаємодіють і обмінюються інформацією між собою.

2.2 Розробка структурної схеми інформаційної технології надання рекомендацій щодо спортивних активностей

Було розроблено структурну схему інформаційної технології надання рекомендацій (рис. 2.3).

Рисунок 2.3 – Структурна схема інформаційної технології рекомендацій щодо спортивних активностей

Цей зв'язок і послідовність дій забезпечують роботу інформаційної технології, що дозволяє забезпечити користувачам персоналізовані та цікаві рекомендації на основі їх вподобань та характеристик.

Розробка структурної схеми інформаційної технології оптичного розпізнавання тексту включає в себе наступні етапи:

1. На першому етапі користувач здійснює активну взаємодію з інтерфейсом додатку. Це може включати вибір налаштувань, запит на перегляд рекомендованих активностей або взаємодію з уже запропонованими рекомендаціями. Користувацький досвід на цьому етапі є ключовим, адже зручність та інтуїтивність інтерфейсу безпосередньо впливають на ефективність використання додатку.
2. Інтерфейс додатку обробляє дії користувача та формує запит, який надсилається до модуля рекомендацій. Цей запит може містити параметри, що визначають інтереси користувача, наприклад, вибраний тип активностей, фільтри за категоріями або інші персоналізовані критерії.
3. Використання вхідних даних модулем рекомендацій.
4. Модуль рекомендацій отримує запит та використовує наявні дані для його обробки. До таких даних належать:
   - Історія взаємодії користувача: дії, виконані в минулому, такі як обрані або переглянуті активності, історія пошуку тощо.
   - Вподобання користувача: задані користувачем параметри, наприклад, рівень тренування, рівень підготовки, фізичні обмеження, вади зі здоров’ям.
   - Характеристики видань (контенту): інформація про доступні спортивні активності, включаючи їх категорію, рівень складності.
5. Обробка даних за допомогою алгоритмів рекомендацій: отримані дані аналізуються за допомогою алгоритмів рекомендацій.
6. Генерація списку рекомендацій: на основі аналізу модуль рекомендацій формує список спортивних активностей, які максимально відповідають інтересам користувача. Цей список включає ключові параметри кожної активності, такі як її назва, опис, категорія, а також рекомендації щодо її виконання.
7. Передача рекомендацій до інтерфейсу додатку: інтерфейс відповідає за відображення цих даних у зручному форматі, що дозволяє користувачу швидко переглянути та оцінити запропоновані активності.

8. Взаємодія користувача з рекомендаціями: користувач отримує можливість вибору серед запропонованих рекомендацій.

Користувач може:

- Переглянути додаткову інформацію про активність, наприклад, опис, відгуки інших користувачів або поради.
- Додати активність до свого списку вподобань чи розкладу.
- Почати виконання активності, якщо це передбачено функціоналом додатку.

До даної інформаційної технології розробки рекомендації щодо спортивних активностей була розроблена UML-діаграма класів (рис.3.2).

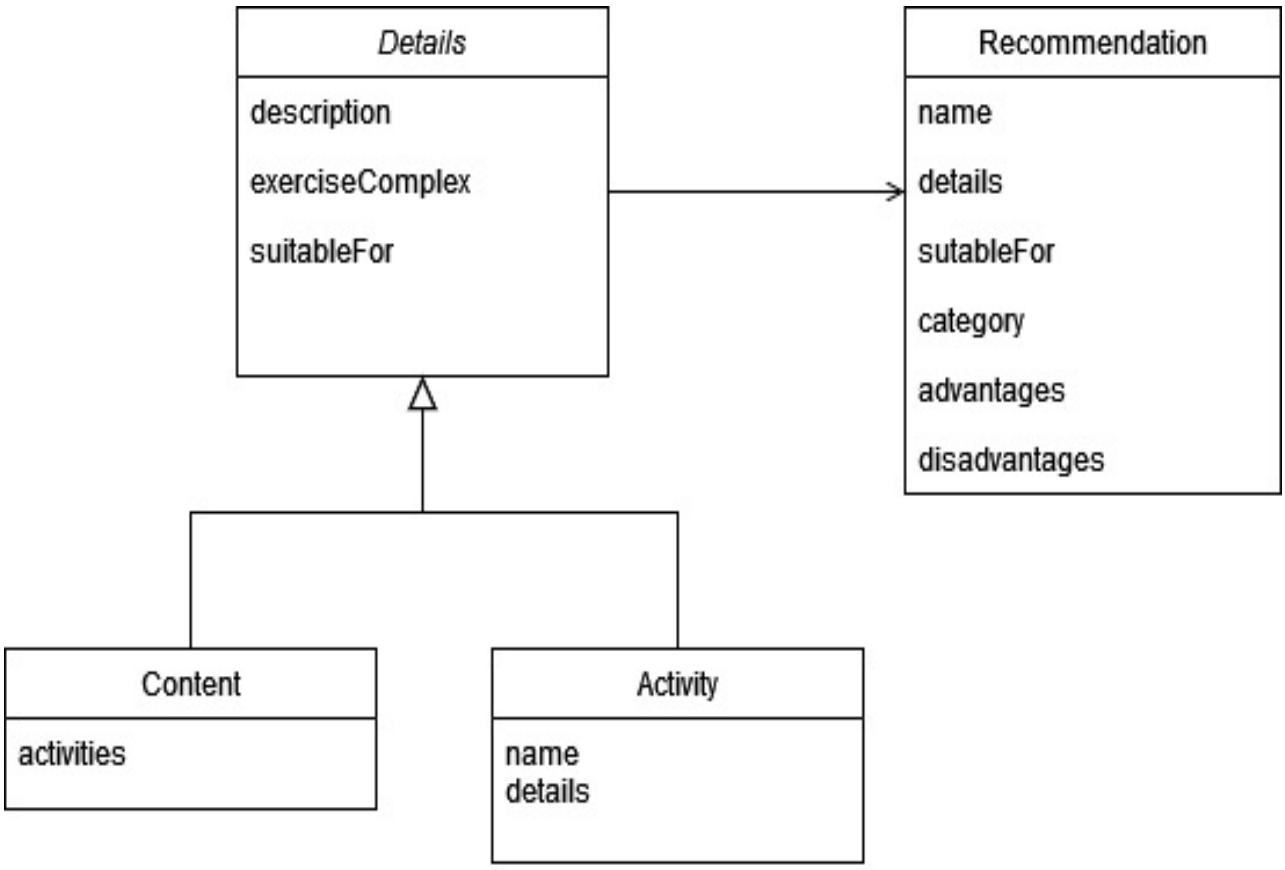

Рисунок 3.2 – UML-діаграма класів інформаційної технології розробки рекомендацій щодо спортивних активностей

2.3 Розробка алгоритму функціонування програмного додатку надання рекомендацій

Алгоритм функціонування програмного додатку для надання рекомендацій щодо спортивних активностей є ключовим компонентом системи, що визначає логіку її роботи. Основна мета алгоритму — забезпечення точного, персоналізованого та зручного формування рекомендацій на основі індивідуальних характеристик користувача.

Відповідно до дослідженої інформації на тему розробки інформаційної технології було розроблено схему алгоритму роботи програми (рис. 2.4)

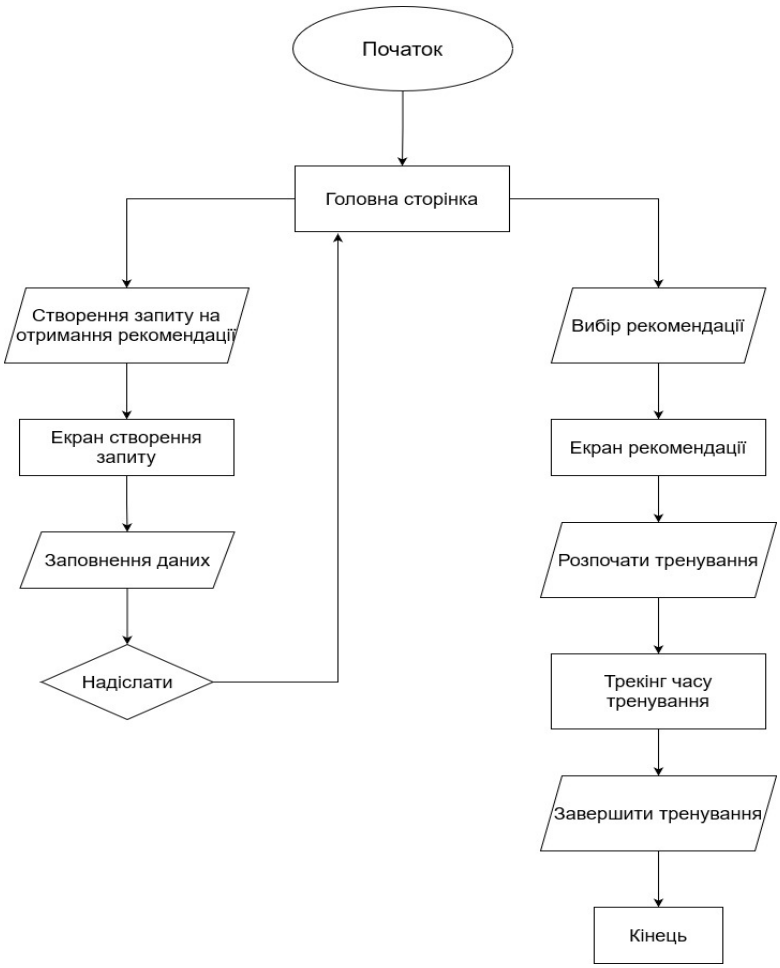

Рисунок 2.4 – Схема алгоритму функціонування мобільного додатку

На схемі алгоритму функціонування мобільного додатку (див рис. 2.4) видно наступний зв'язок і принцип роботи:

1) Користувач взаємодіє з інтерфейсом мобільного додатку, виконуючи

дії, такі як перегляд, пошук, додавання або редагування даних.

2) Інтерфейс передає дії користувача до відповідних модулів додатку.

3) Інформаційна технологія обробляє отримані дії, виконуючи відповідні операції, такі як отримання даних, виконання розрахунків або взаємодія з сервером.

4) Інформаційна техногологія взаємодіє з базою даних, зовнішніми сервісами або іншими модулями, щоб отримати необхідні дані або виконати потрібні операції.

5) Модулі обробляють дані, здійснюють необхідні обчислення та генерують результати.

6) Результати передаються назад до інтерфейсу, який відображає їх користувачеві.

7) Користувач може продовжувати взаємодіяти з додатком, виконуючи нові дії, що сприяє зміні стану додатку і активації відповідних модулів.

Цей зв'язок (рис. 2.4) та послідовність дій забезпечують функціонування мобільного додатку, дозволяють користувачеві взаємодіяти з додатком, отримувати необхідні результати та забезпечують правильну обробку даних та операцій.

Алгоритм функціонування програмного додатку побудований на основі інтеграції персональних даних користувача, машинного навчання та зручного інтерфейсу. Ця модель забезпечує персоналізований підхід до формування рекомендацій, підвищує мотивацію користувачів до регулярних занять спортом і сприяє досягненню поставлених цілей.

2.4 Архітектура математична модель та функціонування машинного навчання

У цьому підрозділі буде представлено загальний опис архітектури математичної моделі та функціонування систем машинного навчання, які використовуються для побудови рекомендаційних систем. Розглядатимуться

основні підходи до створення математичних моделей, їх інтеграція в програмні середовища та роль у забезпеченні персоналізованих рекомендацій. Особливу увагу буде приділено використанню рекурентних нейронних мереж (RNN) та їх модифікації — Long Short-Term Memory (LSTM), які забезпечують високу ефективність обробки послідовних даних.

Архітектура системи для надання спортивних рекомендацій під Android повинна включати кілька основних компонентів: збір даних, обробку та прогнозування результатів тренувань. Використання існуючих бібліотек, таких як TensorFlow Lite та ML Kit, дозволяє ефективно реалізувати цю архітектуру з урахуванням обмежень мобільних пристроїв.

Модуль збору даних: дані про фізичну активність користувача збираються через Google Fit API, який забезпечує доступ до параметрів, таких як кількість кроків, пульс, калорії тощо. Для реалізації цього модуля використовується інтеграція з API, яке дозволяє отримувати дані в реальному часі [10]. Інформація збирається і зберігається для подальшої обробки в мобільному додатку.

Модуль обробки даних: використовуючи бібліотеку TensorFlow Lite, зібрані дані передаються в модель машинного навчання для прогнозування. TensorFlow Lite дозволяє запускати моделі прямо на пристрої без потреби у постійному підключенні до серверів [11].

Модуль навчання та передбачення: на основі історичних даних користувача, модель LSTM прогнозує наступні кроки в тренувальній програмі. Наприклад, якщо користувач починає виконувати нову програму, модель адаптується до його поточного фізичного стану, використовуючи попередні дані для визначення оптимальних тренувальних навантажень [12].

Математично основою є рекурентні нейронні мережі (RNN), зокрема LSTM, які здатні моделювати часову залежність даних і зберігати інформацію про попередні стани системи. Рівняння для LSTM включають три основні компоненти: вхідні, вихідні та забувальні шари, які дозволяють мережі контролювати, які дані потрібно зберегти або забути [13]. Це дозволяє моделі

динамічно адаптуватися до змін у даних користувача, таких як покращення фізичної форми [14].

Система надання рекомендацій щодо спортивних активностей у мобільному додатку на Android потребує вибору й налаштування відповідної математичної моделі машинного навчання, яка забезпечить високу точність персоналізованих рекомендацій. Завдання включає обробку вхідних даних користувача, таких як вік, вага, стан здоров'я, рівень фізичної підготовки тощо, для прогнозування відповідної спортивної активності та її інтенсивності.

Для реалізації даної системи можна розглядати кілька основних математичних моделей. У таблиці 2.2 наведено основні моделі, що підходять для цієї задачі, з відповідними характеристиками.

Таблиця 2.2 – порівняльна характеристика мат. моделей

Модель	Характеристика	Переваги	Недоліки	Формули
1	2	3	4	5
Рекурентні нейронні мережі (RNN)	- Використання для послідовних даних - Проблеми з обробкою довготривалих залежностей - Затухання/вибухання градієнта	Обробка послідовностей, аналіз динамічних залежностей	Високі обчислювальні вимоги, потреба у великій вибірці даних	$h_t = \sigma(W_h h_{t-1} + W_x x_t + b)$

Продовження таблиці 2.2

1	2	3	4	5
Long Short-Term Memory (LSTM)	- Висока точність - Можливість роботи з довготривалими залежностями	Можливість роботи з довготривалими залежностями	- Складність реалізації - Великі вимоги до обчислювальних ресурсів	$f_t = \sigma(W_f[h_{t-1}, x_t] + b_f)$

Рекурентні нейронні мережі забезпечують ефективну обробку послідовних даних, однак їх продуктивність значно обмежується у випадках довготривалих залежностей. Модифікація LSTM усуває ці проблеми за рахунок спеціальних механізмів управління пам'яттю, що робить її ідеальним вибором для створення персоналізованих рекомендацій. Попри вищі вимоги до ресурсів, використання LSTM виправдано у складних завданнях, які потребують глибокого аналізу даних.

У формулі для RNN h_t — стан мережі у момент часу $t_t, W_t, W_x,$ — ваги для попереднього стану та поточного входу відповідно, x_t — вхід, b — зсув, σ — активаційна функція.

Щодо формули для LSTM, f_t, - шлюзи забуття, введення та виходу відповідно, W_f – ваги для відповідних шлюзів, $x_t, –$ вхідні дані, $b_f –$ зсуви, h_{t-1}- попередній та поточних приховані стани.

Для розробки інформаційної технології надання рекомендацій було обрано LSTM та RNN через їх здатність ефективно працювати з послідовними даними, такими як фізичні параметри користувачів та історія тренувань. LSTM, на відміну від класичних RNN, дозволяє зберігати інформацію про довготривалі залежності, що критично для моделювання складних зв’язків у даних. Завдяки цьому забезпечується висока точність рекомендацій, що є необхідною умовою для розробки персоналізованих рішень.

TensorFlow Lite дозволяє оптимізувати використання LSTM на мобільних пристроях, що зменшує споживання ресурсів і підвищує продуктивність додатку. Таким чином, вибір цих технологій є обґрунтованим із точки зору функціональності, ефективності та зручності інтеграції у програмну архітектуру.

2.5 Висновок до розділу 2

У розділі було детально розглянуто архітектурні, математичні аспекти розробки інформаційної системи для надання спортивних рекомендацій під Android. Використання бібліотеки TensorFlow Lite забезпечує можливість ефективної обробки даних та запуску моделей машинного навчання прямо на мобільних пристроях. Застосування Google Fit API надає доступ до фізичних показників користувача, дозволяючи автоматизувати процес збору даних. А динамічна адаптація рекомендацій на основі алгоритмів LSTM гарантує, що кожен користувач отримує персоналізовані тренувальні плани, які оновлюються залежно від його прогресу.

Система розроблена з урахуванням обмежень мобільних пристроїв, таких як обмеженість ресурсів та енергоспоживання, що робить її ефективною для використання на широкому спектрі Android-пристроїв.

3 ПРОГРАМНА РЕАЛІЗАЦІЯ ІНФОРМАЦІЙНОЇ ТЕХНОЛОГІЇ РОЗРОБКИ РЕКОМЕНДАЦІЙ ЩОДО СПОРТИВНИХ АКТИВНОСТЕЙ

3.1 Обґрунтування вибору мови та середовища програмування

У процесі розробки інформаційної технології для надання рекомендацій щодо спортивних активностей важливим етапом є вибір мови програмування та середовища розробки. У цьому розділі детально проаналізуємо особливості основних середовищ розробки для платформ Android та iOS, а також порівняємо мови програмування Kotlin і Swift, які є основними для розробки під відповідні платформи. Для цього буде наведено таблиці з характеристиками, що відображають переваги та недоліки кожного середовища та мови програмування.

Середовища розробки для Android і iOS мають свої унікальні функціональні можливості, які визначають їх вибір залежно від цілей і задач проекту. У таблиці наведено ключові аспекти, що дозволяють оцінити сильні та слабкі сторони Android Studio та Xcode.

Таблиця 3.1 – Характеристики середовищ розробки мобільних додатків

Назва	Характеристики
1	2
Android Studio	Підтримка Kotlin, Java та C++ для розробки додатків. Інтеграція з усім набором інструментів Android SDK. Розширення можливостей за допомогою плагінів. Високі вимоги до апаратних ресурсів. Довший час компіляції порівняно з іншими середовищами. Необхідність додаткової адаптації для різних пристроїв Android.

Продовження таблиці 3.1

1	2
Xcode	Повна інтеграція з iOS SDK і платформою Apple. Зручні інструменти для створення інтерфейсів (Storyboard). Просте тестування на реальних пристроях. Обмежена підтримка платформ, окрім екосистеми Apple. Доступне лише для macOS. Висока вартість обладнання для розробки.

Обидва середовища розробки мають свої сильні сторони, що залежать від цільової платформи. Android Studio забезпечує більше можливостей для кастомізації, тоді як Xcode є оптимальним вибором для інтеграції з продуктами Apple. Для розробки мобільного під операційну систему Android було обрано Android Studio через його гнучкість і широку аудиторію пристроїв, які підтримують Android.

У реалізації мобільних додатків також МДля реалізації додатку під операційну систему Android було обрано мову Kotlin та середовище Android Studio. Рішення приймалося на основі кількох ключових чинників, які обумовлюють ефективність та продуктивність розробки для мобільних пристроїв.

Kotlin – це сучасна мова програмування, яка є офіційно підтримуваною для розробки Android-додатків. Вона була створена компанією JetBrains як альтернатива Java, що дозволяє використовувати переваги сучасних підходів до розробки додатків.

Основні аргументи на користь вибору Kotlin включають:

– Повну сумісність з Java, що дозволяє використовувати наявні Java-бібліотеки та легко інтегрувати існуючі проєкти на Java в Kotlin. Це особливо корисно при використанні сторонніх бібліотек, таких як TensorFlow Lite та Google Fit API, які написані на Java.

– Синтаксична лаконічність що дозволяє значно скоротити кількість коду, необхідного для реалізації типових завдань, що сприяє зменшенню кількості помилок і підвищує читабельність коду. Лаконічний синтаксис мови спрощує процес розробки та робить код більш зрозумілим і підтримуваним. Наприклад, операції, які вимагають багато рядків коду в Java, можна виконати значно простіше в Kotlin.

– Безпека від null-посилань яка надає вбудовані механізми для запобігання поширеним помилкам, пов'язаним з null-посиланнями, які є однією з головних причин помилок у Java-програмах. Система обробки null-посилань у Kotlin зменшує ризик виникнення помилок під час виконання програм [24].

– Швидкий розвиток і підтримка: Kotlin має активну підтримку з боку Google як основна мова для Android. Це означає, що будь-які нові фічі Android будуть швидко інтегровані в Kotlin, що дозволяє розробникам отримувати переваги від новітніх технологій та інструментів.

– Підтримка функцій високого рівня: Kotlin пропонує ряд сучасних функцій, таких як розширення для колекцій, функції вищого порядку, корутини для асинхронної програмування, що дозволяє більш ефективно розробляти додатки, зокрема ті, які працюють з великою кількістю даних у реальному часі, як у випадку з додатками для збору даних з фітнес-трекерів і надання спортивних рекомендацій.

Android Studio – це офіційне інтегроване середовище розробки (IDE) для Android, створене та підтримуване компанією Google. Основними причинами вибору Android Studio як середовища розробки є:

– Офіційна підтримка та інтеграція з Android SDK є офіційним IDE для розробки під Android і має вбудовані інструменти для компіляції, тестування та налагодження додатків. Це значно спрощує процес розробки завдяки прямій інтеграції з Android SDK, а також доступу до найновіших версій бібліотек і інструментів для Android.

– Android Studio надає повну підтримку мови Kotlin з моменту офіційного оголошення Google про Kotlin як мову для розробки під Android. IDE пропонує спеціальні інструменти для швидкої конвертації коду з Java у Kotlin, а також підтримує всі сучасні функції цієї мови.

– Ефективні інструменти для тестування надає широкий спектр інструментів для тестування додатків, таких як Android Emulator та інтеграція з Firebase Test Lab. Це дозволяє тестувати додатки на різних версіях Android та на різних типах пристроїв, що є важливим для забезпечення стабільної роботи програми на різних платформах.

– Інтеграція з інструментами продуктивності надає інструменти для моніторингу продуктивності додатків, такі як Android Profiler. Це дозволяє розробникам відстежувати споживання пам'яті та продуктивність додатка, що є важливим для мобільних додатків, особливо тих, які використовують машинне навчання та працюють з великою кількістю даних.

3.2 Вибір спеціалізованої бібліотеки для програмної реалізації машинного навчання

Для розробки інформаційної технології надання рекомендацій щодо спортивних активностей на базі Android додатків важливим етапом є вибір спеціалізованої бібліотеки для машинного навчання, яка забезпечить ефективне виконання алгоритмів на мобільних пристроях. Для досягнення цієї мети було розглянуто кілька варіантів бібліотек, що підтримують мобільне машинне навчання, зокрема TensorFlow Lite та ML Kit. Обидві бібліотеки надають широкий спектр інструментів для інтеграції моделей машинного навчання у мобільні додатки.

TensorFlow Lite – це оптимізована версія TensorFlow для мобільних пристроїв і вбудованих систем. TensorFlow Lite дозволяє запускати моделі машинного навчання безпосередньо на мобільному пристрої, забезпечуючи

ефективну обробку даних навіть на пристроях із обмеженими обчислювальними ресурсами.

Основні особливості TensorFlow Lite:

– Оптимізація для мобільних пристроїв. TensorFlow Lite розроблено з урахуванням специфіки мобільних пристроїв, таких як низьке споживання ресурсів та енергії. Це дозволяє запускати моделі машинного навчання навіть на пристроях із середніми технічними характеристиками.

– Підтримка різних типів моделей. TensorFlow Lite підтримує різноманітні типи моделей машинного навчання, включаючи глибокі нейронні мережі (DNN), рекурентні нейронні мережі (RNN) та LSTM, що є особливо корисним для роботи з послідовними даними, такими як історія фізичної активності користувача.

– Підтримка апаратного прискорення. TensorFlow Lite використовує апаратне прискорення (наприклад, за допомогою GPU або спеціалізованих чипів, таких як DSP або NPU) для підвищення продуктивності, що робить можливим запуск складних моделей на мобільних пристроях із високою швидкістю.

– Використання попередньо навчених моделей. TensorFlow Lite дозволяє використовувати попередньо навчені моделі машинного навчання, що значно спрощує процес розробки. Моделі можна оптимізувати для мобільних додатків за допомогою методів, таких як квантоване навчання, що знижує обсяг моделі та підвищує продуктивність на мобільних пристроях.

Переваги використання TensorFlow Lite для проєкту:

– Низьке споживання ресурсів. TensorFlow Lite дозволяє запускати моделі машинного навчання з мінімальним навантаженням на процесор, що забезпечує тривалу роботу додатка без суттєвого впливу на автономність пристрою.

– Гнучкість у використанні різних моделей. Підтримка рекурентних нейронних мереж дозволяє використовувати TensorFlow Lite для аналізу

послідовних даних користувача, що є важливим для побудови рекомендацій щодо спортивних активностей.

– Підтримка різних платформ. TensorFlow Lite підтримує різні платформи, включаючи Android та iOS, що дозволяє легко переносити додаток між різними операційними системами.

ML Kit – це інструментарій машинного навчання, розроблений Google, який інтегрується з Firebase і дозволяє додавати можливості машинного навчання до мобільних додатків без необхідності створювати та навчати власні моделі. ML Kit включає ряд готових моделей для виконання різних завдань, таких як розпізнавання тексту, розпізнавання облич, класифікація зображень та інші.

Основні особливості ML Kit:

– Готові моделі. ML Kit надає попередньо навчені моделі для виконання типових завдань машинного навчання, що значно спрощує розробку додатків для мобільних пристроїв. Це дозволяє розробникам швидко інтегрувати функціональність машинного навчання у додаток.

– Хмарне та локальне виконання. ML Kit підтримує як виконання на пристрої, так і хмарне виконання, що забезпечує гнучкість у використанні складних моделей, які потребують більших обчислювальних ресурсів. Для завдань, які не вимагають великої обчислювальної потужності, модель можна виконувати безпосередньо на пристрої користувача.

Переваги використання ML Kit для проєкту:

– Легка інтеграція. ML Kit пропонує простий API для інтеграції готових моделей, що дозволяє швидко додати можливості машинного навчання до додатку без необхідності самостійно навчати моделі.

– Хмарні та локальні можливості. Підтримка виконання як на пристрої, так і в хмарі забезпечує високу гнучкість для виконання різних завдань.

– Спрощене навчання та налаштування. Розробники можуть легко використовувати готові моделі та налаштовувати їх відповідно до потреб проєкту, що значно скорочує час розробки.

Хоча обидві бібліотеки – TensorFlow Lite і ML Kit – є потужними інструментами для мобільного машинного навчання, для даного проєкту було обрано TensorFlow Lite. Головними причинами цього вибору є необхідність використання спеціалізованих моделей машинного навчання (RNN або LSTM) для обробки послідовних даних (історії активностей користувача), що є критичним для побудови рекомендацій щодо спортивних активностей. TensorFlow Lite забезпечує більш гнучку інтеграцію таких моделей і дозволяє оптимізувати їх для мобільних пристроїв.

Крім того, TensorFlow Lite надає можливість запуску складних моделей на пристроях із використанням апаратного прискорення, що забезпечує швидку обробку даних з мінімальним навантаженням на процесор і пам'ять пристрою. Підтримка TensorFlow Lite для різних типів платформ також робить його ідеальним вибором для кросплатформної розробки додатків.

3.3 Програмна реалізація інформаційної технології розробки рекомендацій щодо спортивних активностей

Під час розробки програмного додатку надання рекомендацій щодо спортивних активностей було реалізовано значну кількість рішень які було описано в другому розділі. Далі буде наведено приклади реалізації методів та класів, які відповідають за основний функціонал програмного додатку.

Основним завданням інформаційної технології для надання рекомендацій щодо спортивних активностей є динамічна генерація рекомендацій на основі фізичних параметрів користувача та рівнів складності тренування, активності та тренованості.

```
class FragmentViewBindingDelegate<T : ViewBinding>(
    val fragment: Fragment,
```

```
    val viewBindingFactory: (LayoutInflater) -> T
) : ReadOnlyProperty<Fragment, T> {

    private var binding: T? = null

    init {
        fragment.lifecycle.addObserver(object : DefaultLifecycleObserver {
            val viewLifecycleOwnerLiveDataObserver =
                Observer<LifecycleOwner?> {
                    val viewLifecycleOwner = it ?: return@Observer

                    viewLifecycleOwner.lifecycle.addObserver(object :
DefaultLifecycleObserver {
                        override fun onDestroy(owner: LifecycleOwner) {
                            binding = null
                        }
                    })
                }

            override fun onCreate(owner: LifecycleOwner) {

fragment.viewLifecycleOwnerLiveData.observeForever(viewLifecycleOwnerLiveDataObs
erver)
            }

            override fun onDestroy(owner: LifecycleOwner) {

fragment.viewLifecycleOwnerLiveData.removeObserver(viewLifecycleOwnerLiveDataObs
erver)
            }
        })
    }

    override fun getValue(thisRef: Fragment, property: KProperty<*>): T {
        val binding = binding
        if (binding != null) {
            return binding
        }

        val lifecycle = fragment.viewLifecycleOwner.lifecycle
        if (!lifecycle.currentState.isAtLeast(Lifecycle.State.INITIALIZED)) {
            throw IllegalStateException("Should not attempt to get bindings when
Fragment views are destroyed.")
        }

        return viewBindingFactory(thisRef.layoutInflater).also { this.binding =
it }
    }
}

fun <T : ViewBinding> Fragment.viewBinding(viewBindingFactory: (LayoutInflater)
-> T) =
    FragmentViewBindingDelegate(this, viewBindingFactory)

inline fun <T : ViewBinding> AppCompatActivity.viewBinding(
    crossinline bindingInflater: (LayoutInflater) -> T) =
    lazy(LazyThreadSafetyMode.NONE) {
        bindingInflater.invoke(layoutInflater)
    }
```

Клас FragmentViewBindingDelegate та функції viewBinding для Fragment та AppCompatActivity надають зручний та безпечний спосіб роботи з ViewBinding

без необхідності вручну зберігати та очищати посилання на об’єкт binding. Вони спрощують взаємодію з UI-компонентами та запобігають витокам пам'яті, що можуть виникати через неправильне управління життєвим циклом фрагментів та активностей.

FragmentViewBindingDelegate служить для спрощення роботи з ViewBinding у фрагментах. Він забезпечує зручний спосіб автоматично ініціалізувати ViewBinding, а також очищати його при знищенні вигляду фрагмента, щоб уникнути витоків пам'яті. Це особливо важливо, оскільки ViewBinding може утримувати посилання на вигляд, навіть коли фрагмент більше не використовується, що може призвести до витрат ресурсів.

Характеристиками даної функції відзначаються:

– Безпека щодо життєвого циклу автоматично очищає binding при завершенні життєвого циклу viewLifecycleOwner у фрагментах.

– Спрощення коду що зменшує код, необхідний для ініціалізації та очищення ViewBinding.

– Гнучкість яка сприяє подальше використання як у фрагментах, так і в активностях.

– Завдяки цьому підходу код стає більш надійним і зручним, відповідаючи особливостям життєвого циклу компонентів у Android.

```
interface SportService {
    suspend fun getRecommendations(request: SportRecommendationRequest): 
SportRecommendationResponse

}
```

Інтерфейс SportService був створений для побудови запиту на отримання рекомендацій щодо спортивних активностей. В даному інтерфейсі використовується Kotlin Coroutines, для побудови асинхроних методів, що дозволяє працювати в різних потоках системи.

Також для десириалізації відповідей з запитів використовується бібліотека Gson, для подальшого форматування відповідей з запитів в формат JSON.

```
data class SportRecommendationRequest(
    val messages: List<Message>
)

data class Message(
    val content: String
)

data class SportRecommendationResponse(
    val choices: List<Choice>
)

data class Choice(
    val message: Message
)
```

Клас SportRecommendationRequest призначений для формування запиту на надання рекомендацій щодо спортивних активностей. Клас SportRecommendationResponse призначений для отримання результатів запиту на отримання рекомендацій.

```
@Module
@InstallIn(SingletonComponent::class)
object RoomModule {

    @Provides
    @Singleton
    fun provideSportDatabase(@ApplicationContext context: Context):
SportDatabase =
        Room
            .databaseBuilder(context, SportDatabase::class.java,
"sport_activities_db")
            .fallbackToDestructiveMigration()
            .build()

    @Provides
    @Singleton
    fun provideSportDao(sportDatabase: SportDatabase): SportDao =
sportDatabase.sportDao()

@Provides
@Singleton
fun provideSportService(retrofit: Retrofit): SportApi =
    retrofit.create(SportService::class.java)
```

```
@Provides
@Singleton
fun provideGson(): Gson =
    Gson()
        .newBuilder()
        .setFieldNamingPolicy(FieldNamingPolicy.IDENTITY)
        .create()
}
@Module
@InstallIn(SingletonComponent::class)
object UseCasesModule {

    @Provides
    @Singleton
    fun provideSendUserParamsUseCase(sportService: SportService, sportDao:
SportDao) =
        SendUserParamsUseCase(sportService, sportDao)
}
```

У наведеному коді використовується бібліотека Dagger-Hilt для побудови графу залежностей у мобільному додатку. Давайте розглянемо кожен елемент коду та його роль в процесі побудови графу залежностей:

Анотація `@Module`. Ця анотація вказує Dagger-Hilt, що цей клас є модулем, який надає залежності для внедрення залежностей у додаток.

Анотація `@InstallIn(SingletonComponent::class)`. Ця анотація вказує, що модуль `AppModule` має бути встановлений у `SingletonComponent`. `SingletonComponent` визначає область життєвого циклу, в якій ці залежності будуть існувати.

```
@AndroidEntryPoint
class TrainingProgramsFragment: Fragment() {

    private val binding by viewBinding(FragmentTrainingProgramsBinding::inflate)

    private val viewModel: TrainingProgramsViewModel by viewModels()

    override fun onCreateView(
        inflater: LayoutInflater,
```

```
        container: ViewGroup?,
        savedInstanceState: Bundle?
    ): View = binding.root

    override fun onViewCreated(view: View, savedInstanceState: Bundle?) {
        super.onViewCreated(view, savedInstanceState)

        val adapter = TrainingProgramsAdapter()

        if (requireActivity() is ReplaceAbleToolBarView) {
            (requireActivity() as ReplaceAbleToolBarView).replaceTitle("Спорт")
        }

        binding.fbAddProgram.setOnClickListener {
            FragmentUtil.setFragmentIfAbsent(AddProgramFragment.newInstance(),
requireActivity().supportFragmentManager, R.id.fragment_container)
        }

        binding.rvTrainingPrograms.adapter = adapter

        viewLifecycleOwner.lifecycleScope.launch(Dispatchers.IO) {
            viewModel.sportActivities.collect {
                adapter.submitList(it)
            }
        }

        adapter.onTrainingClickListener = {
FragmentUtil.setFragmentIfAbsent(DetailsTrainingFragment.newInstance(it.id),
requireActivity().supportFragmentManager, R.id.fragment_container)
        }

    }
    override fun onResume() {
        super.onResume()
        viewModel.fetchSportRecommendations()
    }

    companion object {
        @JvmStatic
        fun newInstance(): TrainingProgramsFragment {
            return TrainingProgramsFragment()
        }
```

```
    }
}
```

У наведеному коді представлено клас TrainingProgramsFragment. Сторінка, на якій відображається список рекомендацій щодо спортивних активностей. Цей фрагмент коду представляє Fragment у Android, який відповідає за відображення списку тренувальних програм та взаємодію з ним.

Також для отримання рекомендацій щодо спортивних активностей слугує зміна viewModel, яка надає доступ до представлення списку.

```
@HiltViewModel
class TrainingProgramsViewModel @Inject constructor(
    private val sportDatabase: SportDatabase
) : BaseViewModel() {

    private val _sportActivities: MutableStateFlow<List<RecommendationEntity>> =
MutableStateFlow(listOf())
    val sportActivities: StateFlow<List<RecommendationEntity>> =
_sportActivities

    init {
        fetchSportRecommendations()
    }

    fun fetchSportRecommendations() {
        kotlin.runCatching {
            viewModelScope.launch(Dispatchers.IO) {

_sportActivities.emit(sportDatabase.sportDao().fetchSportActivities())
            }
        }.onFailure {
            Timber.e(it)
        }
    }
}
```

У наведеному коді зображен клас TrainingProgramsViewModel, який надає доступ до списку рекомендацій щодо спортивних активностей. Для передачі даних на екран списку рекомендацій використовується Kotlin Coroutines. Дана технологія орієнтована на реалізацію асинхроного програмування, тобто, роботи у різних потоках. Dispatchers.IO – параметр, який вказує, в якому саме потоці будуть надсилатись дані для відображення на користувацькому інтерфейсі, а саме – у фоні. Функція fetchSportRecommendations ініціюється у випадку створення екземпляру класа TrainingProgramsViewModel, у подальшому використані та роботі з даними.

```
class SendUserParamsUseCase @Inject constructor(
    private val sportApi: SportService,
    private val sportDao: SportDao
): Action<SportRecommendationRequest, List<Activity>> {
    override suspend fun execute(arg: SportRecommendationRequest):
Resource<List<Activity>> {
        return try {
            val response = sportApi.getRecommendations(arg)

            val choices = response.choices.first()
            val content = choices.message.content
            val typeTokenContent = object : TypeToken<Content>() {}.type
            val parsedContent = Gson().fromJson(content, typeTokenContent) as
Content

            val recommendationEntities = parsedContent.activities
                .map { RecommendationEntity.mapFrom(
                    Recommendation(it.name, it.details, it.details.suitableFor,
it.details.category, it.details.advantages, it.details.disadvantages))
                }
            sportDao.insert(recommendationEntities)
            Resource.Success(parsedContent.activities)
        } catch (t: Throwable) {
            Timber.e(t)
            Resource.Error(uiText =
UIText.DynamicString(t.localizedMessage.toString()))
        }
    }
}
```

У наведеному коді представлений клас SendUserParamsUseCase, який відповідає за формування запиту. Даний клас реалізовує інтерфейс Action, який приймає на вхід два типи даних, перший – для аргумента suspend функції execute(arg: T), другий – Resource<R> який спрямований на визначення типу даних що буде повертати функція execute.

Клас SendUserParamsUseCase потрібен для взаємодії з інтерфейсом який надає доступ до формування запиту та отримання результату надання рекомендацій та локальною базою даних. Він надсилає параметри користувача для отримання спортивних рекомендацій, зберігає ці дані у локальній базі та повертає результат для відображення в інтерфейсі. Така структура дозволяє ізолювати логіку запитів та роботу з базою в одному місці, що полегшує підтримку коду.

```
sealed class Resource<T>(val data: T? = null, val uiText: UIText? = null) {
    class Success<T>(data: T?) : Resource<T>(data)
    class Error<T>(uiText: UIText? = null, data: T? = null) : Resource<T>(data,
uiText)
}
```

Клас Resource створений для представлення різних станів операцій, які можуть мати успіх або зазнати помилки. Він часто використовується для обробки результатів запитів до API або операцій, які можуть завершитися як успішно, так і з помилкою. Всередині класу Resource представлені два класа, для обробки даних, у випадку успішного сценарію в клас конструктор класа Success встановлюється результат запиту.

Переваги використання Resource:

- Стандартизує обробку результатів операцій: Однаковий підхід до обробки успіхів і помилок у будь-яких запитах або операціях.
- Забезпечує зрозумілий API для роботи з результатами: Полегшує код, роблячи його більш зрозумілим і компактним.
- Гнучкість обробки помилок: Можна показувати користувачеві повідомлення про помилки через uiText, що дозволяє налаштовувати текст залежно від конкретної ситуації.

Отже, в даному підрозділі було розглянуто програмну реалізацію інформаційної технології.

3.4 Розробка інтерфейсу інформаційної технології надання рекомендацій

Інтерфейс створеної інформаційної технології оптичного розпізнавання тексту представлений у вигляді XML-сторінок [31]. Для роботи з окремими компонентами сторінки екрану була використана бібліотека ViewBinding, що покращує ефективність та працездатність додатку. Далі продемонстровано рисунок з наведеними характеристиками інших методів роботи з компонентами додатку (рис. 3.3).

Method	Elegance	Compile Type Safety	Build Speed
findViewById	✗	✗	✓
Data Binding	✓	✓	✗
Butterknife	✓	✗	✗
Kotlin Synthethic	✓	✗	✓
View Binding	✓	✓	✓

Рисунок 3.3 – Порівняльна характеристика ефективності методів роботи з XML-компонентами

Даний рисунок представляє порівняльний аналіз методів взаємодії з елементами користувацького інтерфейсу в Android. Аналіз базується на трьох критеріях:

- елегантність оцінює зручність, простоту та чіткість коду, створеного за допомогою відповідного методу.
- типо-безпека на етапі компіляції вказує, чи дозволяє метод виявляти помилки, пов’язані з доступом до елементів інтерфейсу, ще на етапі компіляції.
- швидкість збірки оцінює вплив методу на швидкість компіляції проєкту.

Методи, представлені в таблиці, порівнюються наступним чином:

- findViewById характеризується низькою елегантністю через громіздкість коду, відсутністю типобезпеки, але забезпечує високу швидкість збірки.
- Data Binding забезпечує високу елегантність і типобезпеку, однак значно уповільнює процес збірки через складність генерації відповідного коду.
- Butterknife відрізняється високою елегантністю, але позбавлений типобезпеки, а також негативно впливає на швидкість збірки.
- Kotlin Synthetic пропонує зручний і компактний підхід (елегантний), однак не підтримує типобезпеку. На швидкість збірки він не впливає.

– View Binding демонструє високу елегантність, типобезпеку і має мінімальний вплив на швидкість збірки, що робить його одним із найсучасніших і збалансованих підходів.

3.5 Тестування розробленої інформаційної технології надання рекомендацій

Для демонстрації працездатності виконаємо тест даної інформаційної технології. Для цього необхідно відкрити додаток розташований у меню мобільного девайсу, після чого ми опинимось на його головній сторінці (рис. 3.4).

Рисунок 3.4 – Екран сторінки зі списком рекомендацій щодо спортивних активностей

Для вибору тренування, зі списку необхідно обрати поточну рекомендацію, яка наведена зі списку. На даній сторінці наведені рекомендації щодо спортивних активностей, які були сформовані на основі характеристик та рівнів складності, активності та натренованості користувача.

Створення нових рекомендацій виконується на запит користувача через сторінку на рисунку 3.5.

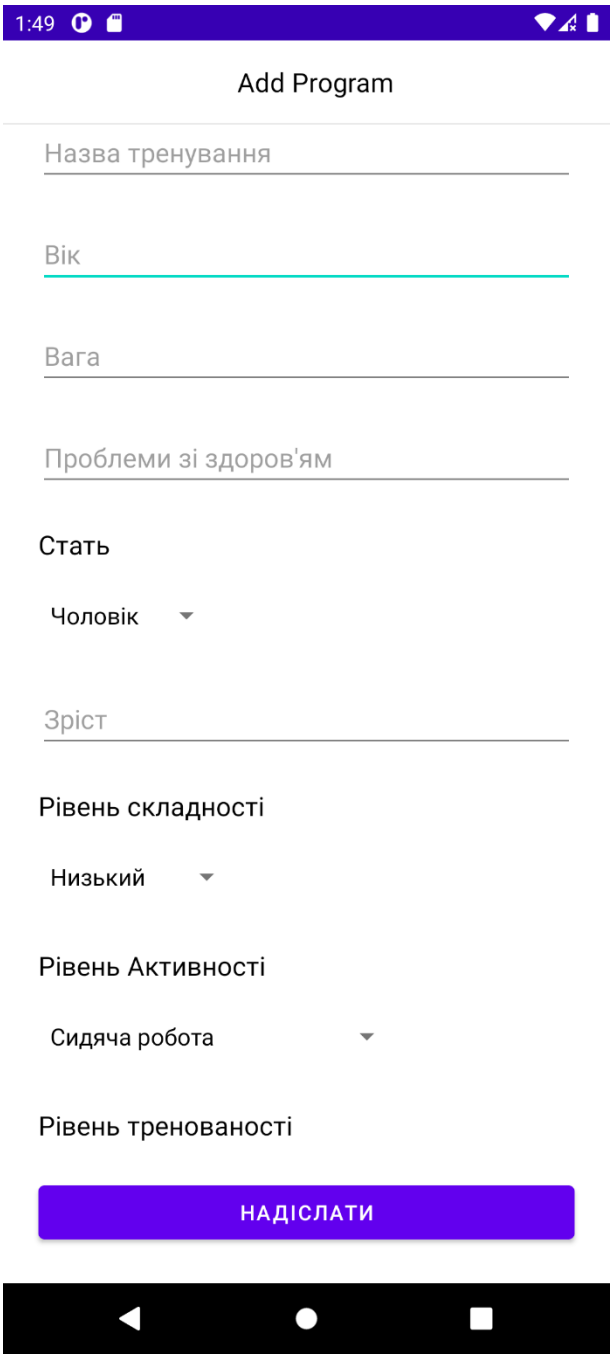

Рисунок 3.5 – Екран сторінки створення запиту на отримання рекомендацій щодо спортивних активностей

Для отримання рекомендацій щодо спортивних активностей вказуються наступні параметри такі як назва тренування, вік, вага, проблеми зі здоров’ям, стать, зріст, рівень складності тренування, рівень активності людини та рівень тренованості. На основі наведених параметрів формується більш точна рекомендація, у якій розписані подальші інструкції для виконання тренування. Приклад відповіді на такий запит подано на рисунку 3.6

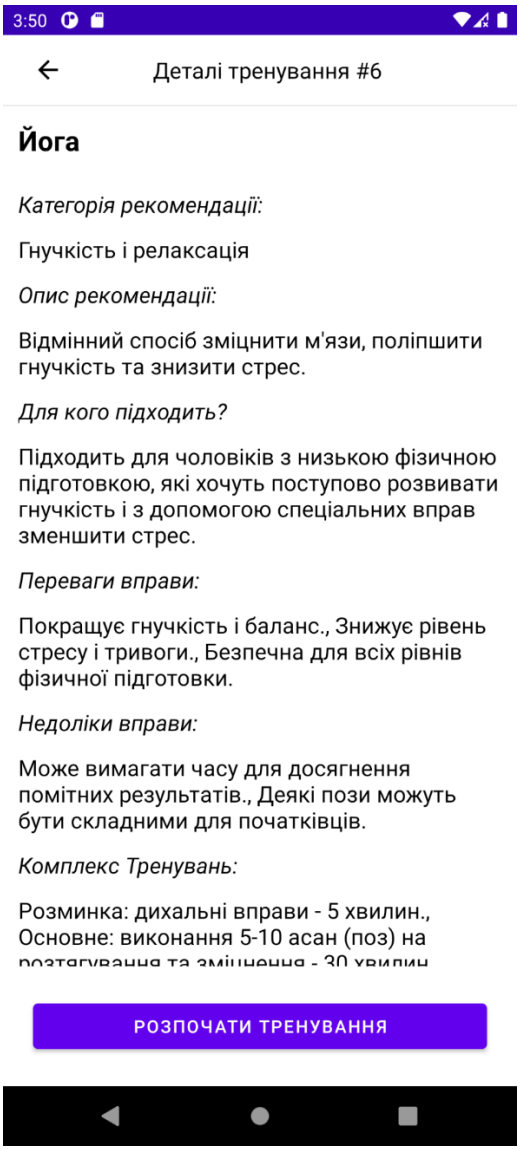

Рисунок 3.6 – Екран сторінки списку з наданими, по запиту на отримання рекомендацій щодо спортивних активностей

В результаті відповіді на запит, маємо можливість отримати саме ту рекомендацію, яку може зацікавити користувача.

3.6 Висновок до розділу 3

У рамках даного розділу розглянуто переваги та недоліки середовища для розробки інформаційної технології, а також було обґрунтовано вибір мови програмування Kotlin для розробки додатків під Android та обґрунтовано її вибір для реалізації інформаційної технології рекомендацій щодо спортивних активностей. У результаті було розроблено програму на мові Kotlin з використанням TensorFlow Lite для обробки моделей машинного навчання. Було здійснено тестування розробленої інформаційної технологія.

Було розглянуто, як реалізовано функцію надання рекомендацій щодо спортивних активностей.

Також було розроблено інтерфейс інформаційної технології та аргументовано підхід до розробки інтерфейсу користувача.

Було здійснено тестування розробленої інформаційної технології, яке показало її повну працездатність та ефективність.

На відміну від своїх програм-аналогів, інформаційня технологія орієнтована на широкий спектр спортивних активностей, додатково надає точні рекомендації на основі характеристик користувача.

4 ЕКОНОМІЧНА ЧАСТИНА

Ефективне впровадження науково-технічної розробки стає можливим, якщо вона відповідає поточним вимогам науково-технічного прогресу та враховує економічні аспекти. Надання оцінки економічної ефективності отриманих результатів науково-дослідної роботи є важливою частиною цього процесу.

Магістерська робота, що присвячена розробці та дослідженню «Інформаційна технологія надання рекомендацій щодо спортивних активностей», віднесена до науково-технічних робіт, спрямованих на введення на ринок. Рішення про комерціалізацію розробки може бути прийняте протягом самої роботи, дозволяючи реалізувати можливість виведення її на ринок. Цей напрямок розглядається як пріоритетний, оскільки розроблені результати можуть бути корисними для різних зацікавлених сторін і приносити економічні вигоди. Однак для успішного втілення цього процесу важливо знайти зацікавленого інвестора, який був би зацікавлений у реалізації цього проекту, і переконати його в обґрунтованості таких інвестицій.

Для цього визначені наступні етапи виконання робіт:

1. Проведено комерційний аудит науково-технічної розробки, що включає в себе визначення науково-технічного рівня та комерційного потенціалу.

2. Розраховані витрати на реалізацію науково-технічної розробки.

3. Проведено розрахунок економічної ефективності науково-технічної розробки в разі її впровадження та комерціалізації потенційним інвестором, а також обґрунтовано економічну доцільність комерціалізації для інвестора.

4.1 Проведення комерційного та технологічного аудиту науково-технічної розробки

Метою проведення комерційного і технологічного аудиту дослідження за темою «Інформаційна технологія надання рекомендацій щодо спортивних активностей» 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 таблиці 4.1

Таблиця 4.1 – 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 влас-тивості 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

Продовження таблиці 4.1

Бали (за 5-ти бальною шкалою)					
	0	1	2	3	4
Ринкові перспективи					
7	Активна конкуренція великих компаній на ринку	Активна конкуренція	Помірна конкуренція	Незначна конкуренція	Конкурентів немає
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	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 використовуються у військово промисловому комплексі	Потрібні дорогі матеріали	Потрібні досяжні та дешеві матеріали	Всі матеріали для реалізації ідеї відомі та давно використовуютьс я у виробництві
11	Термін реалізації ідеї більший за 10 років	Термін реалізації ідеї більший за 5 років. Термін окупності інвестицій більше 10-ти	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	Необхідно отримання великої кількості дозвільних документів на виробництво та	Процедура отримання дозвільних документів для виробництва та реалізації	Необхідно тільки повідомлення відповідним органам про виробництво та реалізацію	Відсутні будь- які регламентні обмеження на виробництво та реалізацію продукту

Для проведення комерційного та технологічного аудиту залучають не менше 3-х незалежних експертів, якими можуть бути провідні викладачі випускової або спорідненої кафедри чи інші відомі фахівці.

Для опитування було залучено експертів з ВНТУ, кафедри комп'ютерних наук: к.т.н., доцент Арсенюк Ігор Ростиславович, к.т.н., доцент Озеранський Володимир Сергійович, к.т.н., проф. Колесницький Олег Костянтинович.

Оцінювання науково-технічного рівня розробки та її комерційного потенціалу рекомендується здійснювати із застосуванням п'ятибальної системи оцінювання за 12-ма критеріями, що було використано у даному опитуванні.

Результати оцінювання науково-технічного рівня та комерційного потенціалу науково-технічної розробки потрібно зведені до таблиці 4.2.

Таблиця 4.2 – Результати оцінювання науково-технічного рівня і комерційного потенціалу розробки експертами

Критерії	Експерт (ПІБ, посада)		
	Арсенюк І.Р.	Озеранський В.С.	Колесницький О. К.
	Бали:		
1. Технічна здійсненність концепції	4	4	4
2. Ринкові переваги (наявність аналогів)	3	3	4
3. Ринкові переваги (ціна продукту)	4	4	4
4. Ринкові переваги (технічні властивості)	4	4	4
5. Ринкові переваги (експлуатаційні витрати)	4	3	3
6. Ринкові перспективи (розмір ринку)	3	3	3
7. Ринкові перспективи (конкуренція)	3	3	4

Продовження таблиці 4.2

Критерії	Бали:		
8. Практична здійсненність (наявність фахівців)	4	4	4
9. Практична здійсненність (наявність фінансів)	3	3	3
10. Практична здійсненність (необхідність нових матеріалів)	4	4	4
11. Практична здійсненність (термін реалізації)	4	4	4
12. Практична здійсненність (розробка документів)	3	3	3
Сума балів	$СБ_1$=43	$СБ_2$=42	$СБ_3$=44
Середньоарифметична сума балів $СБ_c$	(43+42+44)/3 = 43		

За результатами розрахунків, наведених в таблиці 4.2, зробимо висновок щодо науково-технічного рівня і рівня комерційного потенціалу розробки. При цьому використаємо рекомендації, наведені в таблиці 4.3 [39].

Таблиця 4.3 – 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 «Інформаційна технологія надання рекомендацій щодо спортивних активностей» становить 43 бали, що, відповідно до таблиці 4.3, свідчить про

комерційну важливість проведення даних досліджень оскільки рівень комерційного потенціалу розробки високий.

Дана розробка призначена для користувачів, які цікавляться криптовалютами та мають доступ до мережі інтернет. Для використання технології їм знадобиться перейти на сайт ресурсу та купити продукт за фіксованою ціною.

4.2 Визначення рівня конкурентоспроможності розробки

В процесі визначення економічної ефективності науково-технічної розробки також доцільно провести прогноз рівня її конкурентоспроможності за сукупністю параметрів, що підлягають оцінюванню.

Одиничний параметричний індекс розраховуємо за формулою [40]:

$$q_i = \frac{P_i}{P_{баз\,i}}. \qquad (4.1)$$

де q_i – одиничний параметричний індекс, розрахований за i-м параметром;

P_i – значення i-го параметра виробу;

$P_{базі}$ – аналогічний параметр базового виробу-аналога, з яким проводиться порівняння.

Загальні технічні та економічні характеристики розробки представлено в таблиці 4.4.

Таблиця 4.4 – Основні техніко-економічні показники аналога та розробки, що проектується

Показник	Варіанти		Відносний показник якості	Коефіцієнт вагомості параметра
	Базовий (товар-конкурент)	Новий (інноваційне рішення)		
Точність	65	>70	1,08	70%
Горизонт прогнозування	1	4	4	30%

Нормативні параметри оцінюємо показником, який отримує одне з двох значень: 1 – пристрій відповідає нормам і стандартам; 0 – не відповідає.

Груповий показник конкурентоспроможності за нормативними параметрами розраховуємо як добуток частинних показників за кожним параметром за формулою [40]:

$$I_{НП} = \prod_{i=1}^{n} q_i \,, \tag{4.2}$$

де $I_{нп}$ – загальний показник конкурентоспроможності за нормативними параметрами;

q_i – одиничний (частинний) показник за i-м нормативним параметром;

n – кількість нормативних параметрів, які підлягають оцінюванню.

За нормативними параметрами розроблюваний пристрій відповідає вимогам ДСТУ, тому $I_{нп} = 1$.

Значення групового параметричного індексу за технічними параметрами визначаємо з урахуванням вагомості (частки) кожного параметра [40]:

$$I_{ТП} = \sum_{i=1}^{n} q_i \cdot \alpha_i \ , \qquad (4.3)$$

де $I_{тп}$ – груповий параметричний індекс за технічними показниками (порівняно з виробом-аналогом);

q_i – одиничний параметричний показник i-го параметра;

α_i – вагомість i-го параметричного показника, $\sum_{i=1}^{n} \alpha_i = 1$;

n – кількість технічних параметрів, за якими оцінюється конкурентоспроможність.

Проведемо аналіз параметрів згідно даних таблиці 4.4.

$$I_{mn} = 1{,}08 \cdot 0{,}7 + 4 \cdot 0{,}3 = 1{,}96.$$

Груповий параметричний індекс за економічними параметрами розраховуємо за формулою [40]:

$$I_{ЕП} = \sum_{i=1}^{m} q_i \cdot \beta_i \ , \qquad (4.4)$$

де $I_{ЕП}$ – груповий параметричний індекс за економічними показниками;

q_i – економічний параметр i-го виду;

β_i – частка i-го економічного параметра, $\sum_{i=1}^{m} \beta_i = 1$;

m – кількість економічних параметрів, за якими здійснюється оцінювання.

Проведемо аналіз параметрів згідно даних таблиці.

$$I_{ЕП}=0,76 \cdot 0,5 + 0,84 \cdot 0,5 = 0,80.$$

На основі групових параметричних індексів за нормативними, технічними та економічними показниками розрахуємо інтегральний показник конкурентоспроможності за формулою [40]:

$$K_{ІНТ} = I_{НП} \cdot \frac{I_{ТП}}{I_{ЕП}}, \tag{4.5}$$

$$K_{ІНТ} = 1 \cdot 1,96 / 0,80 = 2,45.$$

Інтегральний показник конкурентоспроможності $К_{ІНТ} > 1$, отже розробка переважає відомі аналоги за своїми техніко-економічними показниками.

4.3 Розрахунок витрат на проведення науково-дослідної роботи

Витрати, пов’язані з проведенням науково-дослідної роботи на тему «Інформаційна технологія надання рекомендацій щодо спортивних активностей», під час планування, обліку і калькулювання собівартості науково-дослідної роботи групуємо за відповідними статтями.

4.3.1 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 [40]:

$$З_о = \sum_{i=1}^{k} \frac{М_{пі} \cdot t_і}{Т_р}, \quad (4.6)$$

де k – кількість посад дослідників залучених до процесу досліджень;

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

t_i – число днів роботи конкретного дослідника, дн.;

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

$$З_о = 42000 \cdot 10 / 21 = 19091 \text{ (грн.)}.$$

Проведені розрахунки зведемо до таблиці 4.5.

Таблиця 4.5 – 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	42000	1909,1	10	19091
Інженер-програміст	39000	1772,7	55	97500
Всього				116591

Основна заробітна плата робітників. Витрати на основну заробітну плату робітників ($З_р$) за відповідними найменуваннями робіт НДР на тему «Інформаційна технологія надання рекомендацій щодо спортивних активностей» розраховуємо за формулою:

$$З_р = \sum_{i=1}^{n} C_i \cdot t_i , \tag{4.7}$$

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:

$$C_i = \frac{М_М \cdot К_і \cdot К_с}{Т_р \cdot t_{зм}} , \tag{4.8}$$

де $М_М$ – розмір прожиткового мінімуму працездатної особи, або мінімальної місячної заробітної плати (в залежності від діючого законодавства), приймемо $Мм$=6500 грн;

$К_і$ – коефіцієнт міжкваліфікаційного співвідношення для встановлення тарифної ставки робітнику відповідного розряду [40]:

$К_с$ – 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.

$T_р$ – середнє число робочих днів в місяці, приблизно $T_р$ = 21 дн;

$t_{зм}$ – тривалість зміни, год.

$$С_1 = 8000 \cdot 1 \cdot 1{,}65 / (21 \cdot 8) = 78{,}6 \text{ (грн.)}.$$

$$З_{р1} = 78{,}6 \cdot 2 = 157{,}2 \text{ (грн.)}.$$

Проведені розрахунки зведемо до таблиці 4.6.

Таблиця 4.6 – Величина витрат на основну заробітну плату робітників

Найменування робіт	Тривалість роботи, год	Розряд роботи	Погодинна тарифна ставка, грн	Тарифний коефіцієнт	Величина оплати на робітника грн
1.Підготовчі	2	1	78,6	1	157,2
2.Налагоджувальні	10	2	93,4	1,09	934
3.Випробувальні	2	4	99.8	1,27	199,6
Всього					1290,8

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

$$З_{дод} = (З_о + З_р) \cdot \frac{Н_{дод}}{100\%}, \quad (4.9)$$

де $Н_{дод}$ – норма нарахування додаткової заробітної плати. Приймемо 11%.

$$Здод = (116591 + 1290{,}8) \cdot 11 / 100\% = 12966{,}99 \text{ (грн.)}.$$

4.3.2 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:

$$З_н = (З_о + З_р + З_{дод}) \cdot \frac{Н_{зп}}{100\%}, \quad (4.10)$$

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

$$\text{Зн} = (116591 + 1290{,}8 + 12966{,}99) \cdot 22 / 100\% = 28786{,}73 \text{ (грн.)}.$$

4.3.3 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 за темою «Інформаційна технологія надання рекомендацій щодо спортивних активностей».

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

$$М = \sum_{j=1}^{n} Н_j \cdot Ц_j \cdot К_j - \sum_{j=1}^{n} В_j \cdot Ц_{в\,j}, \quad (4.11)$$

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

n – кількість видів матеріалів;

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

K_j – коефіцієнт транспортних витрат, (K_j = *1,1 … 1,15*);

B_j – маса відходів *j*-го найменування, кг;

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

Проведені розрахунки зведемо до таблиці 4.7.

Таблиця 4.7 – Витрати на матеріали

Найменування матеріалу, марка, тип, сорт	Ціна за 1 кг, грн	Норма витрат, кг	Вартість витраченого матеріалу, грн
Папір А 4	146	1	146
Ручка	14	1	14
Диск оптичний OPTIMA CD	15	1	15
Flesh-пам'ять GOODRAM 64 C10A	410	1	410
Всього			585
З врахуванням коефіцієнта транспортування			643,5

4.3.4 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:

$$B_{спец} = \sum_{i=1}^{k} Ц_i \cdot C_{пр.i} \cdot K_i \ , \qquad (4.12)$$

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

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

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

k – кількість найменувань устаткування.

$$В_{спец} = 40000,00 \cdot 1 \cdot 1,11 = 44000 \text{ (грн.)}.$$

Отримані результати зведемо до таблиці 4.8:

Таблиця 4.8 – Витрати на придбання спецустаткування по кожному виду

Найменування устаткування	Кількість, шт	Ціна за одиницю, грн	Вартість, грн
1. Ноутбук Lenovo	1	40000	44000
2. Клавіатура Zero	1	1270	1397
3. Миша Razer Deathadder v2	1	1399	1538,9
4. Крісло Офісне	1	3076	3383,6
5. Стіл	1	9000	9900
Всього			60219,5

4.3.5 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:

$$B_{прг} = \sum_{i=1}^{k} Ц_{iпрг} \cdot C_{прг.i} \cdot K_i \ , \qquad (4.13)$$

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.

$$B_{прг} = 600 \cdot 1 \cdot 1,11 = 660 \text{ (грн.)}.$$

Отримані результати зведемо до таблиці 4.9:

Таблиця 4.9 – Витрати на придбання програмних засобів по кожному виду

Найменування програмного засобу	Кількість, шт	Ціна за одиницю, грн	Вартість, грн
Підписка на хмарне сховище даних	1	600	660
Оренда серверу	1	720	792
Всього			1452

4.3.6 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:

$$A_{обл} = \frac{Ц_б}{T_в} \cdot \frac{t_{вик}}{12}, \quad (4.14)$$

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.

$$A_{обл} = (45000 \cdot 1) / (2 \cdot 12) = 1875 \text{ (грн.)}.$$

Проведені розрахунки зведемо до таблиці 4.10.

Таблиця 4.10 – Амортизаційні відрахування по кожному виду обладнання

Найменування обладнання	Балансова вартість, грн	Строк корисного використання, років	Термін використання обладнання, місяців	Амортизаційні відрахування, грн
Комп’ютер	45000	2	1	1875,00

Продовження таблиці 4.10

Приміщення лабораторії	190000	20	1	791,67
Всього				2666,67

4.3.7 Паливо та енергія для науково-виробничих цілей

Витрати на силову електроенергію (B_e) розраховуємо за формулою:

$$B_e = \sum_{i=1}^{n} \frac{W_{yi} \cdot t_i \cdot Ц_e \cdot K_{вni}}{\eta_i}, \quad (4.15)$$

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 $Ц_e$ = 7,5 грн;

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

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

$$B_e = 0,25 \cdot 250,0 \cdot 7,5 \cdot 0,5 / 0,8 = 292,97 \text{ (грн.)}.$$

4.3.8 Службові відрядження

До статті «Службові відрядження» дослідної роботи на тему «Інформаційна технологія надання рекомендацій щодо спортивних

активностей» 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:

$$B_{св} = (З_о + З_р) \cdot \frac{Н_{св}}{100\%}, \quad (4.16)$$

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

$$В_{св} = (116591 + 1290{,}8) \cdot 20 / 100\% = 23576{,}36 \text{ (грн.)}.$$

4.3.9 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:

$$І_в = (З_о + З_р) \cdot \frac{Н_{ів}}{100\%}, \quad (4.17)$$

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

$$І_в = (116591 + 1290,8) \cdot 50 / 100\% = 58940,9 \text{ (грн.)}.$$

4.3.10 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:

$$В_{нзв} = (З_о + З_р) \cdot \frac{Н_{нзв}}{100\%}, \quad (4.18)$$

де $Н_{нзв}$ – норма нарахування за статтею «Накладні (загальновиробничі) витрати», приймемо $Н_{нзв}$ = 100%.

$$В_{нзв} = (116591 + 1290,8) \cdot 100 / 100\% = 117881,8 \text{ (грн.)}.$$

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

$$B_{заг} = З_о + З_р + З_{дод} + З_н + М + К_в + В_{спец} + В_{прг} + А_{обл} + В_е + В_{св} + В_{сп} + І_в + В_{нзв}. \quad (4.19)$$

$$\text{В}_{\text{заг}} = 116591 + 1290{,}8 + 12966{,}99 + 28786{,}73 + 643{,}5 + 60219{,}5 + 1452 + 2666{,}67 + 292{,}97 + 23576{,}36 + 58940{,}9 + 117881{,}8 = 425309{,}22 \text{ (грн.)}.$$

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

$$ЗВ = \frac{B_{заг}}{\eta}, \quad (4.20)$$

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

$$\text{ЗВ} = 425309{,}22 / 0{,}9 = 472565{,}8 \text{ (грн.)}.$$

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

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

Результати дослідження проведені за темою «Інформаційна технологія надання рекомендацій щодо спортивних активностей»» передбачають комерціалізацію протягом 3-х років реалізації на ринку.

В цьому випадку основу майбутнього економічного ефекту будуть формувати:

ΔN – збільшення кількості споживачів яким надається відповідна інформаційна послуга у періоди часу, що аналізуються;

N – кількість споживачів яким надавалась відповідна інформаційна послуга у році до впровадження результатів нової науково-технічної розробки, приймемо 1 особа

$Ц_б$ – вартість послуги у році до впровадження інформаційної системи, приймемо 2000,00 грн;

$\pm\Delta Ц_о$ – зміна вартості послуги від впровадження результатів, приймемо зростання на 500,00 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:

$$\Delta П_i = (\pm\Delta Ц_о \cdot N + Ц_о \cdot \Delta N)_i \cdot \lambda \cdot \rho \cdot (1 - \frac{\vartheta}{100}), \qquad (4.21)$$

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 =40%;

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

Збільшення чистого прибутку 1-го року:

$\Delta П_1$ =(1·500+2000·1500)·0,83·0,4·(1-0,18/100%)=640684,79 грн.

Збільшення чистого прибутку 2-го року:

$\Delta П_2$ =(1·500+2000·(1500+1200))·0,83·0,4·(1-0,18/100%)=1153578,9грн.

Збільшення чистого прибутку 3-го року:

$\Delta П_3$ =(1·500+2000·(1500+1200+850))·0,83·0,4·(1-0,18/100%)=1516585,2 грн.

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

$$ПП = \sum_{i=1}^{T} \frac{\Delta П_i}{(1+\tau)^t}, \qquad (4.22)$$

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 =18%;

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

$$ПП = 640684,79/(1+0,18)^1+1153578,9\ /(1+0,18)^2+1516585,2\ /(1+0,18)^3= 2216736,01 \text{ (грн.)}.$$

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

$$PV = k_{інв} \cdot ЗВ, \tag{4.23}$$

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 472565,8 (грн.).

$$PV = k_{інв} \cdot ЗВ = 2 \cdot 472565,8 = 945131,6 \text{ (грн.)}.$$

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

$$E_{абс} = ПП - PV \tag{4.24}$$

де $ПП$ – приведена вартість зростання всіх чистих прибутків від можливого впровадження та комерціалізації науково-технічної розробки, 2216736,01 (грн.);

PV – теперішня вартість початкових інвестицій, 945131,6 (грн.).

$$E_{абс} = ПП - PV = 2216736{,}01 - 945131{,}6 = 1271604{,}41 \text{ (грн.)}.$$

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

$$\mathrm{E_в} = \sqrt[Тж]{1 + \frac{\mathrm{E_{абс}}}{PV}} - 1, \qquad (4.25)$$

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

PV – теперішня вартість початкових інвестицій, грн;

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

$$\mathrm{E_в} = (1 + 1271604{,}41 / 945131{,}6)1/3 - 1 = 0{,}42.$$

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

$$\tau_{мін} = d + f, \qquad (4.26)$$

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.

$$\tau_{мін} = 0,1+0,25 = 0,35 < 0,42 ,$$

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 «Інформаційна технологія онтологічного моделювання бази знань з організації бібліотеки» доцільно.

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

$$T_{ок} = \frac{1}{E_в}, \qquad (4.27)$$

де $E_в$ – внутрішня економічна дохідність вкладених інвестицій.

$T_{ок} = 1 / 0,42 = 2,38$ р.

$T_{ок} <$ 3-х років, що може спонукати потенційного інвестора профінансувати впровадження даної розробки та виведення її на ринок.

4.5 Висновок до розділу 4

Згідно проведених досліджень рівень комерційного потенціалу розробки за темою «Інформаційна технологія надання рекомендацій щодо спортивних активностей» становить 43 бали, що свідчить про значну перспективність проєкту, оскільки система надання рекомендацій з фізичної активності може знайти широке застосування серед користувачів, які прагнуть покращити своє здоров’я та фізичну форму.

При оцінюванні рівня конкурентоспроможності, згідно узагальненого коефіцієнту конкурентоспроможності розробки, науково-технічна розробка переважає існуючі аналоги приблизно в 2,45 рази.

Також термін окупності інвестицій становить 2,38 роки, що є меншим за типовий інвестиційний горизонт у 3 роки. Це свідчить про високу комерційну привабливість проєкту, яка може спонукати потенційних інвесторів до його підтримки та впровадження на ринку. Отже можна зробити висновок про доцільність проведення науково-дослідної роботи за темою «Інформаційна технологія надання рекомендацій щодо спортивних активностей».

ВИСНОВКИ

У рамках виконання магістерської роботи успішно реалізовано інформаційну технологію надання рекомендацій щодо спортивних активностей. Всі поставлені завдання були виконані повністю. Було здійснено аналіз сучасних технологій надання рекомендацій та проведено огляд аналогічних програмних рішень.

Розроблено математичну модель, що описує алгоритми функціонування системи, а також UML-діаграму класів, яка відображає основні етапи створення рекомендацій у рамках інформаційної технології. Діаграма використовується для візуалізації процесів та взаємодії компонентів системи.

Створена інформаційна технологія демонструє високу ефективностість в порівнянні з існуючими аналогами. Результати тестування підтвердили надійність системи, що забезпечує розширення існуючого функціоналу, відповідно до сучасних вимог ринку.

З економічної точки зору, доцільність розробки обґрунтовується аналізом витрат та прогнозуванням прибутковості проєкту. Також термін окупності інвестицій становить 2,38 роки, що є меншим за типовий інвестиційний горизонт у 3 роки. Результати показали, що впровадження цієї технології має значний потенціал для комерційного успіху та розширення ринкових можливостей.

Таким чином, реалізована інформаційна технологія надання рекомендацій щодо спортивних активностей має перспективи подальшого розвитку та вдосконалення в майбутньому.

ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Лесков С. Д., Арсенюк І. Р. Підхід щодо реалізації системи рекомендацій зі спортивних активностей // Світ наукових досліджень. Випуск 34: матеріали Міжнародної мультидисциплінарної наукової інтернет-конференції (м. Тернопіль, Україна, м. Ополе, Польща, 22–23 жовтня 2024 р.) / за ред. : О. Патряк та ін. ГО “Наукова спільнота”, WSZIA w Opolu. Тернопіль: ФОП Шпак В. Б., 2024. – С. 159–162. [Електронний ресурс]. – Режим доступу: https://www.economy-confer.com.ua/full-article/5783/.

2. Машинне навчання – Що таке машинне навчання, як працює та де використовується [Електронний ресурс]. – Режим доступу: https://gigacloud.ua/blog/navchannja/scho-take-mashinne-navchannja-jak-pracjue-ta-de-vikoristovuetsja;

3. Nike Training Club [Електронний ресурс]. – Режим доступу: https://www.nike.com/ntc-app;

4. MyFitnessPal [Електронний ресурс]. – Режим доступу: https://www.myfitnesspal.com/;

5. Freeletics [Електронний ресурс]. – Режим доступу: https://www.freeletics.com/en/;

6. Інтеграція з API – Google Fit API [Електронний ресурс]. – Режим доступу: https://uk.wikipedia.org/wiki/Google_Fit;

7. Mahyari A., Jelinek H., Lu J. Physical Exercise Recommendation and Success Prediction Using Interconnected Recurrent Neural Networks [Електронний ресурс]. – Режим доступу: https://doi.org/10.48550/arXiv.2010.00482.

8. Gaspar P., Martins M. SNPERS: A Physical Exercise Recommendation System Integrating Statistical Principles and NLP [Електронний ресурс]. – Режим доступу: https://www.mdpi.com/2076-3417/11/5/2069.

9. University of California – San Diego. A deep learning tool for personalized workout recommendations from fitness tracking data [Електронний ресурс]. – Режим доступу: https://www.sciencedaily.com/releases/2019/04/190422151023.htm.

10. TensorFlow Lite Documentation. Running TensorFlow Lite on Android [Електронний ресурс]. – Режим доступу: https://www.tensorflow.org/lite/guide/android.

11. Kotlin Documentation. Why Kotlin? [Електронний ресурс]. – Режим доступу: https://kotlinlang.org/docs/why-kotlin.html.

12. Model-View-Controller [Електронний ресурс]. – Режим доступу: https://medium.com/@zorbeytorunoglu/mvc-in-android-c731f2e2b844.

13. Model-View-Presenter [Електронний ресурс]. – Режим доступу: https://uk.wikipedia.org/wiki/Model-View-Presenter.

14. Model-View-Intent [Електронний ресурс]. – Режим доступу: https://www.geeksforgeeks.org/model-view-intent-mvi-pattern-in-reactive-programming-a-comprehensive-overview/.

15. Android Studio Documentation. Building your first app with Android Studio [Електронний ресурс]. – Режим доступу: https://developer.android.com/studio.

16. Google Fit API Documentation [Електронний ресурс]. – Режим доступу: https://developers.google.com/fit.

17. Ivanov D. Data Preprocessing for Fitness Applications: A Practical Guide [Електронний ресурс]. – Режим доступу: https://arxiv.org/abs/2005.12345.

18. Johnson M. Personalized Training Programs through Machine Learning [Електронний ресурс]. – Режим доступу: https://www.springer.com/neural-computing.

19. TensorFlow Lite. Model Optimization [Електронний ресурс]. – Режим доступу: https://www.tensorflow.org/lite/performance/model_optimization.

20. Google AI Blog. Improving Mobile Machine Learning with TensorFlow Lite [Електронний ресурс]. – Режим доступу: https://ai.googleblog.com/2021/06/improving-mobile-machine-learning-with.html.

21. ML Kit for Firebase. Getting Started [Електронний ресурс]. – Режим доступу: https://developers.google.com/ml-kit/guides.

22. Firebase Test Lab Documentation [Електронний ресурс]. – Режим доступу: https://firebase.google.com/docs/test-lab.

23. Android Profiler [Електронний ресурс]. – Режим доступу: https://developer.android.com/studio/profile.

24. Методичні вказівки до виконання магістерських кваліфікаційних робіт для студентів спеціальності 122 «Комп'ютерні науки» [Електронний ресурс] / уклад.: А. А. Яровий, О. К. Колесницький. – Вінниця : ВНТУ, 2023. – 58 с.

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

Додаток А (обов'язковий)

Протокол перевірки кваліфікаційної роботи на наявність текстових запозичень

ПРОТОКОЛ ПЕРЕВІРКИ КВАЛІФІКАЦІЙНОЇ РОБОТИ
НА НАЯВНІСТЬ ТЕКСТОВИХ ЗАПОЗИЧЕНЬ

Назва роботи: Інформаційна технологія надання рекомендацій щодо спортивних активностей

Тип роботи: магістерська кваліфікаційна робота
(БДР, МКР)

Підрозділ кафедра комп'ютерних наук, ФІТА
(кафедра, факультет)

Показники звіту подібності Turnitin

Оригінальність 95,00% Схожість 5,00%

Аналіз звіту подібності (відмітити потрібне):

✓ 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 виконання автором. Роботу направити на розгляд експертної комісії кафедри.

□ 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.

Ознайомлені з повним звітом подібності, який був згенерований системою Turnitin щодо роботи.

Автор роботи Лесков С.Д.

Керівник роботи Арсенюк І.Р.

Опис прийнятого рішення

Магістерську кваліфікаційну роботу допущено до захисту

Особа, відповідальна за перевірку Озеранський В.С.

Додаток Б (обов'язковий)

Лістинг програмного коду

```
class FragmentViewBindingDelegate<T : ViewBinding>(
    val fragment: Fragment,
    val viewBindingFactory: (LayoutInflater) -> T
) : ReadOnlyProperty<Fragment, T> {

    private var binding: T? = null

    init {
        fragment.lifecycle.addObserver(object : DefaultLifecycleObserver {
            val viewLifecycleOwnerLiveDataObserver =
                Observer<LifecycleOwner?> {
                    val viewLifecycleOwner = it ?: return@Observer

                    viewLifecycleOwner.lifecycle.addObserver(object :
DefaultLifecycleObserver {
                        override fun onDestroy(owner: LifecycleOwner) {
                            binding = null
                        }
                    })
                }

            override fun onCreate(owner: LifecycleOwner) {

fragment.viewLifecycleOwnerLiveData.observeForever(viewLifecycleOwnerLiveDataObs
erver)
            }

            override fun onDestroy(owner: LifecycleOwner) {

fragment.viewLifecycleOwnerLiveData.removeObserver(viewLifecycleOwnerLiveDataObs
erver)
            }
        })
    }

    override fun getValue(thisRef: Fragment, property: KProperty<*>): T {
        val binding = binding
        if (binding != null) {
            return binding
        }

        val lifecycle = fragment.viewLifecycleOwner.lifecycle
        if (!lifecycle.currentState.isAtLeast(Lifecycle.State.INITIALIZED)) {
            throw IllegalStateException("Should not attempt to get bindings when
Fragment views are destroyed.")
        }

        return viewBindingFactory(thisRef.layoutInflater).also { this.binding =
it }
    }
}

fun <T : ViewBinding> Fragment.viewBinding(viewBindingFactory: (LayoutInflater)
-> T) =
    FragmentViewBindingDelegate(this, viewBindingFactory)

inline fun <T : ViewBinding> AppCompatActivity.viewBinding(
```

```
    crossinline bindingInflater: (LayoutInflater) -> T) =
    lazy(LazyThreadSafetyMode.NONE) {
        bindingInflater.invoke(layoutInflater)
    }
data class SportRecommendationRequest(
    val messages: List<Message>
)


data class Message(
    val content: String
)


data class SportRecommendationResponse(
    val choices: List<Choice>
)


data class Choice(
    val message: Message
)
@Module
@InstallIn(SingletonComponent::class)
object RoomModule {

    @Provides
    @Singleton
    fun provideSportDatabase(@ApplicationContext context: Context):
SportDatabase =
        Room
            .databaseBuilder(context, SportDatabase::class.java,
"sport_activities_db")
            .fallbackToDestructiveMigration()
            .build()

    @Provides
    @Singleton
    fun provideSportDao(sportDatabase: SportDatabase): SportDao =
sportDatabase.sportDao()

@Provides
@Singleton
fun provideSportService(retrofit: Retrofit): SportApi =
    retrofit.create(SportService::class.java)

@Provides
@Singleton
fun provideGson(): Gson =
    Gson()
        .newBuilder()
        .setFieldNamingPolicy(FieldNamingPolicy.IDENTITY)
        .create()
}
@Module
@InstallIn(SingletonComponent::class)
object UseCasesModule {

    @Provides
```

```
    @Singleton
    fun provideSendUserParamsUseCase(sportService: SportService, sportDao:
SportDao) =
        SendUserParamsUseCase(sportService, sportDao)
}
```

```
<?xml                       version="1.0"                       encoding="utf-8"?>
<RelativeLayout       xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical">

    <TextView
        android:id="@+id/tv_title"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentTop="true"
        android:layout_marginStart="12dp"
        android:layout_marginTop="12dp"
        android:layout_marginEnd="12dp"
        android:layout_marginBottom="12dp"
        android:textColor="@color/black"
        android:textSize="22sp"
        android:textStyle="bold"
        tools:text="Yoga"                                                    />

    <ScrollView
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_above="@+id/btn_start_training"
        android:layout_below="@+id/tv_title">

        <LinearLayout
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:orientation="vertical">

            <TextView
                android:id="@+id/tv_category_title"
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:layout_marginStart="12dp"
                android:layout_marginTop="12dp"
                android:layout_marginEnd="12dp"
                android:textColor="@color/black"
                android:textSize="18sp"
                android:textStyle="italic"
                android:text="Категорія            рекомендації:"            />

            <TextView
                android:id="@+id/tv_category_value"
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:layout_marginStart="12dp"
                android:layout_marginTop="12dp"
                android:layout_marginEnd="12dp"
                android:textColor="@color/black"
                android:textSize="18sp"
                tools:text="Категорія"                                       />

            <TextView
                android:id="@+id/tv_description_title"
```

```
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_marginStart="12dp"
        android:layout_marginTop="12dp"
        android:layout_marginEnd="12dp"
        android:textColor="@color/black"
        android:textSize="18sp"
        android:textStyle="italic"
        android:text="Опис               рекомендації:"               />

    <TextView
        android:id="@+id/tv_description_value"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_marginStart="12dp"
        android:layout_marginTop="12dp"
        android:layout_marginEnd="12dp"
        android:textColor="@color/black"
        android:textSize="18sp"
        tools:text="tkrewoptwreojtirpewipttrewp[to"                    />

    <TextView
        android:id="@+id/tv_suitable_title"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_marginStart="12dp"
        android:layout_marginTop="12dp"
        android:layout_marginEnd="12dp"
        android:textColor="@color/black"
        android:textSize="18sp"
        android:textStyle="italic"
        android:text="Для               кого               підходить?"/>

    <TextView
        android:id="@+id/tv_suitable_value"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_marginStart="12dp"
        android:layout_marginTop="12dp"
        android:layout_marginEnd="12dp"
        android:textColor="@color/black"
        android:textSize="18sp"
        tools:text="Підходить                                    для..."/>

    <TextView
        android:id="@+id/tv_advantages_title"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_marginStart="12dp"
        android:layout_marginTop="12dp"
        android:layout_marginEnd="12dp"
        android:textColor="@color/black"
        android:textSize="18sp"
        android:textStyle="italic"
        android:text="Переваги                                   вправи:"/>

    <TextView
        android:id="@+id/tv_advantages_value"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_marginStart="12dp"
        android:layout_marginTop="12dp"
        android:layout_marginEnd="12dp"
        android:textColor="@color/black"
```

```
        android:textSize="18sp"
        tools:text="Переваги"/>

    <TextView
        android:id="@+id/tv_disadvantages_title"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_marginStart="12dp"
        android:layout_marginTop="12dp"
        android:layout_marginEnd="12dp"
        android:textColor="@color/black"
        android:textSize="18sp"
        android:textStyle="italic"
        android:text="Недоліки                                    вправи:"/>

    <TextView
        android:id="@+id/tv_disadvantages_value"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_marginStart="12dp"
        android:layout_marginTop="12dp"
        android:layout_marginEnd="12dp"
        android:textColor="@color/black"
        android:textSize="18sp"
        tools:text="Недоліки"/>

    <TextView
        android:id="@+id/tv_exercise_complex_title"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_below="@+id/tv_description"
        android:layout_marginStart="12dp"
        android:layout_marginTop="12dp"
        android:layout_marginEnd="12dp"
        android:textColor="@color/black"
        android:textAlignment="textStart"
        android:gravity="start"
        android:textSize="18sp"
        android:textStyle="italic"
        android:text="Комплекс                                Тренувань:"/>

    <TextView
        android:id="@+id/tv_exercise_complex_value"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_marginStart="12dp"
        android:layout_marginTop="12dp"
        android:layout_marginEnd="12dp"
        android:textColor="@color/black"
        android:textAlignment="textStart"
        android:gravity="start"
        android:textSize="18sp"                                  />

    <TextView
        android:id="@+id/tv_training_trackers_title"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_marginStart="12dp"
        android:layout_marginTop="12dp"
        android:layout_marginEnd="12dp"
        android:textColor="@color/black"
        android:textAlignment="textStart"
        android:gravity="start"
        android:textSize="18sp"
```

```
                android:textStyle="italic"
                android:text="Зареєстровано                    тренувань:"/>

            <TextView
                android:id="@+id/tv_training_trackers_value"
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:layout_below="@+id/tv_description"
                android:layout_marginStart="12dp"
                android:layout_marginTop="12dp"
                android:layout_marginEnd="12dp"
                android:textColor="@color/black"
                android:textAlignment="textStart"
                android:gravity="start"
                android:textSize="18sp"/>

        </LinearLayout>

    </ScrollView>

    <Button
        android:id="@+id/btn_start_training"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_alignParentBottom="true"
        android:layout_margin="24dp"
        android:text="Розпочати              тренування"              />

</RelativeLayout>
```

Додаток В (обов'язковий)

ІЛЮСТРАТИВНА ЧАСТИНА

«ІНФОРМАЦІЙНА ТЕХНОЛОГІЯ ПРОГНОЗУВАННЯ РЕНТАБЕЛЬНОСТІ, ВАЛОВОГО ПРИБУТКУ ТА ВАЛОВОГО ЗБИТКУ ПІДПРИЄМСТВА»

Виконав: студент 2 курсу,групи ЗКН-23м
спеціальності 122 – Комп'ютерні науки
(шифр і назва напряму підготовки, спеціальності)
Лесков С.Д.
(прізвище та ініціали)

Керівник: к.т.н., доцент каф. КН
Арсенюк І.Р.
(прізвище та ініціали)

«05» 12 2024 р.

Вінниця ВНТУ - 2024 рік

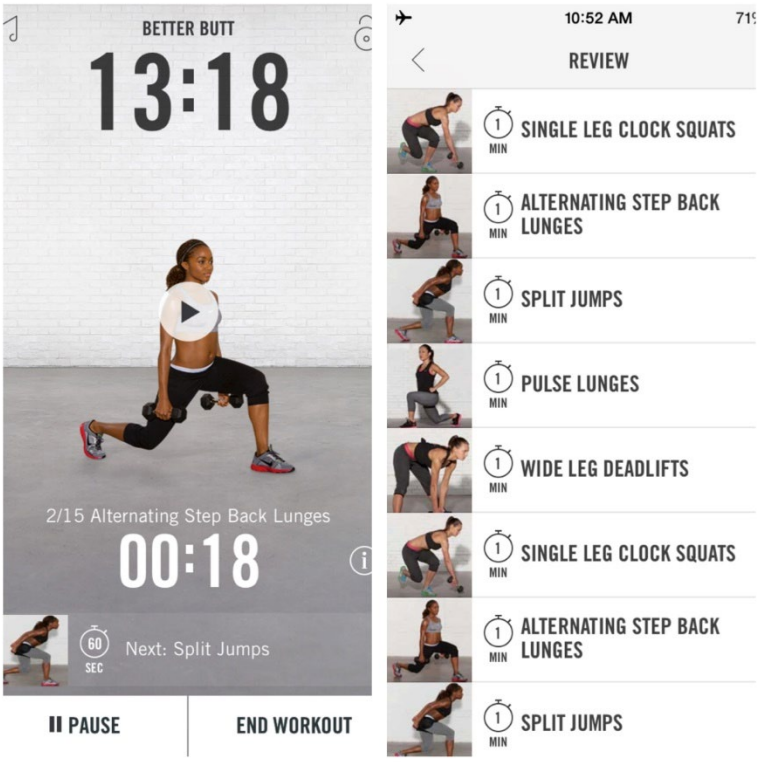

Рисунок В.1 – Приклад роботи програми Nike Training Club (NTC)

Рисунок В.2 – Приклад роботи MyFitnessPal

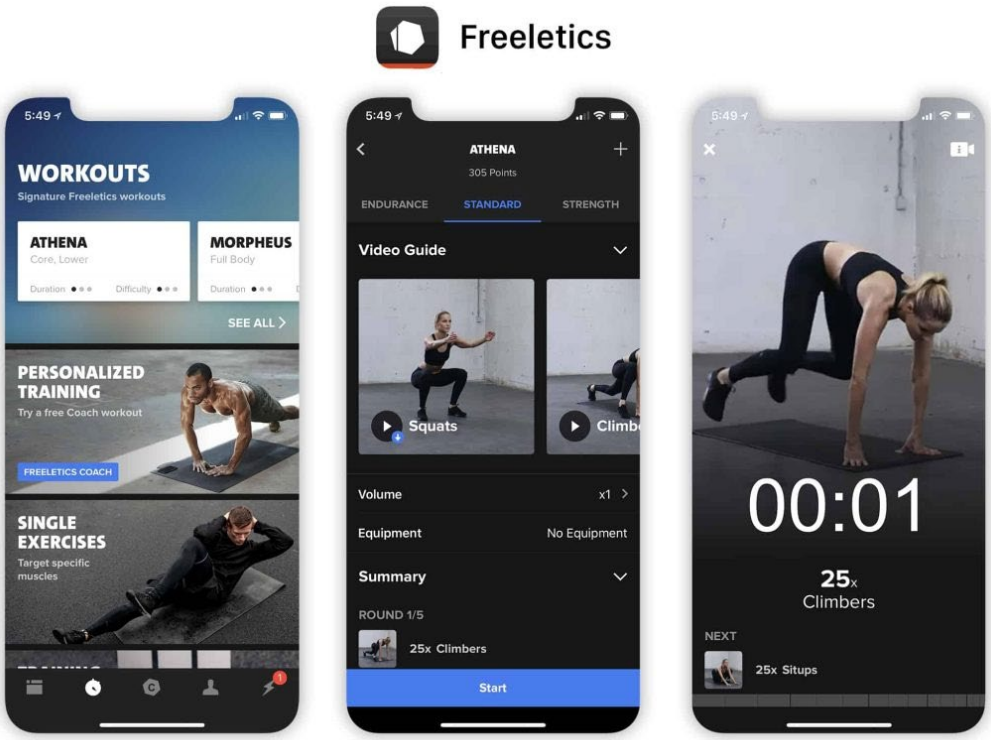

Рисунок В.3 – Приклад роботи Freeletics

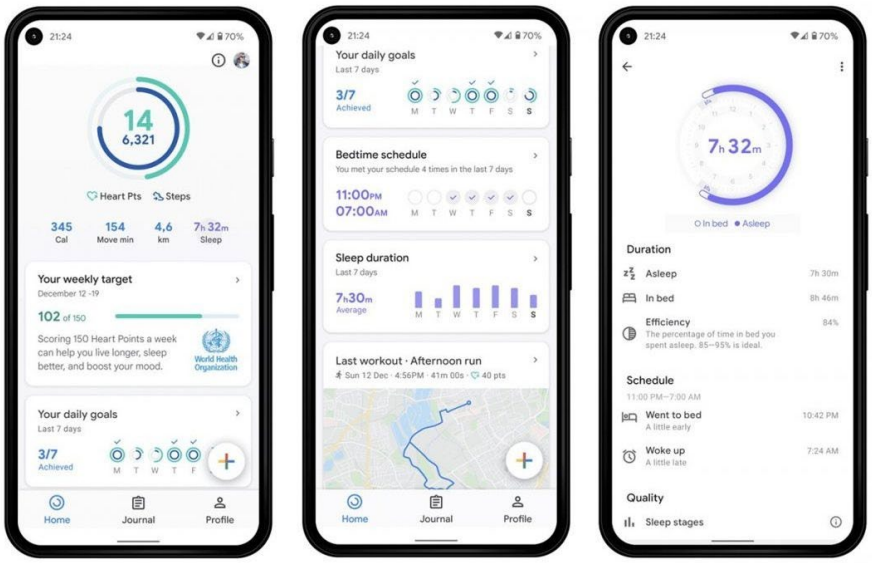

Рисунок В.4 – Приклад роботи Google Fit

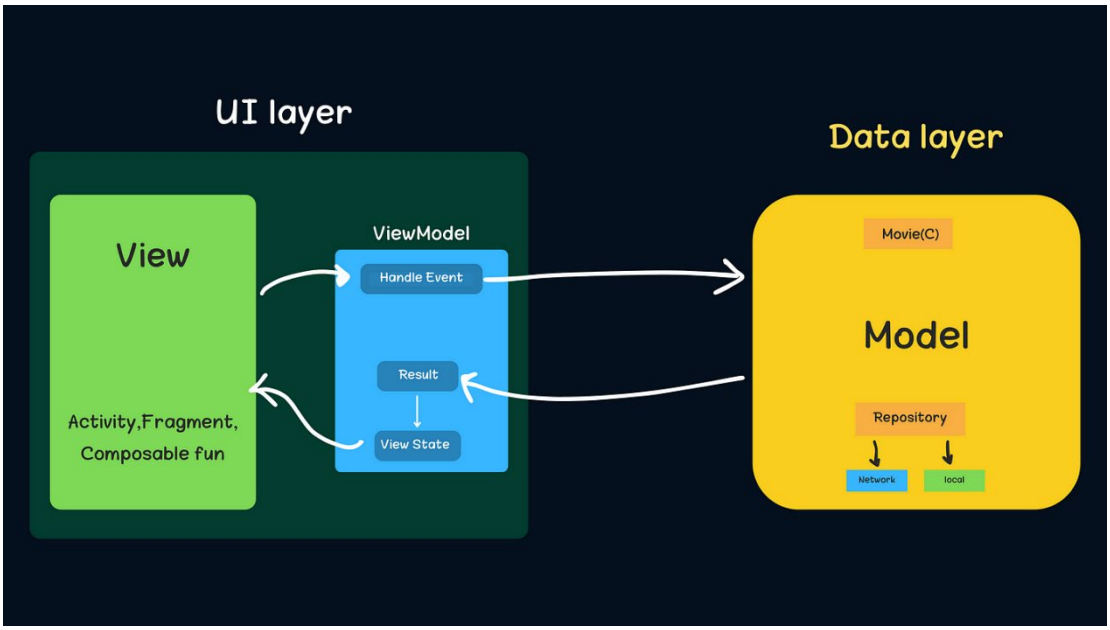

Рисунок В.5 – Структурна схема архітектури розробки мобільного додатку Model-View-ViewModel

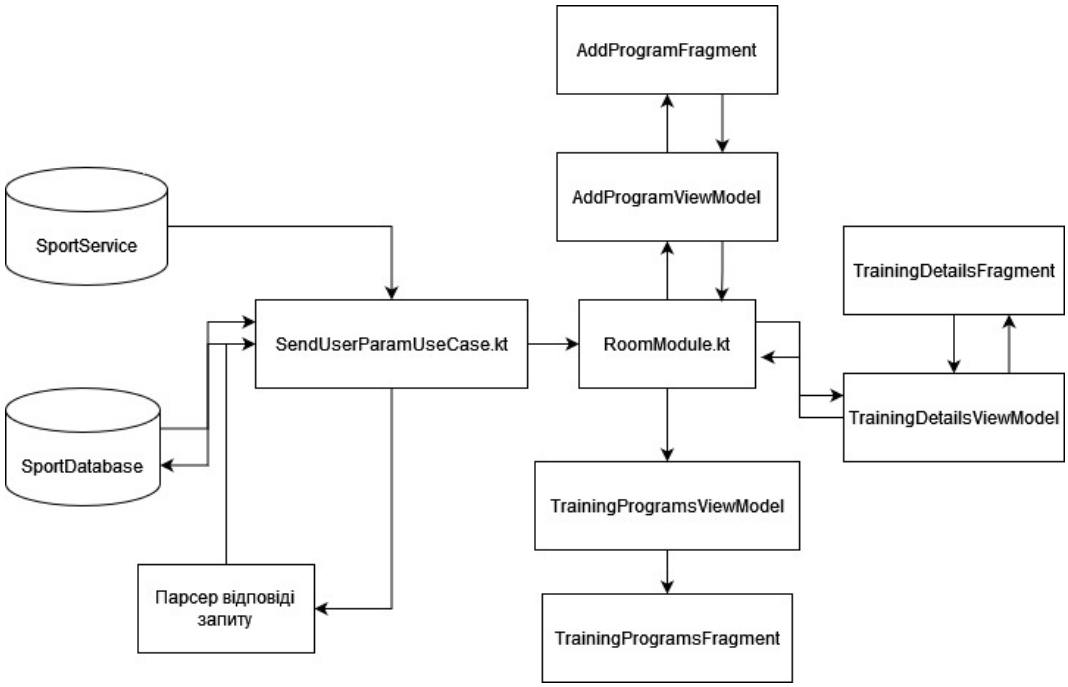

Рисунок В.6 – Діаграма компонентів програмного додатку для надання рекомендацій

Рисунок В.7 – Структурна схема інформаційної технології рекомендацій щодо спортивних активностей

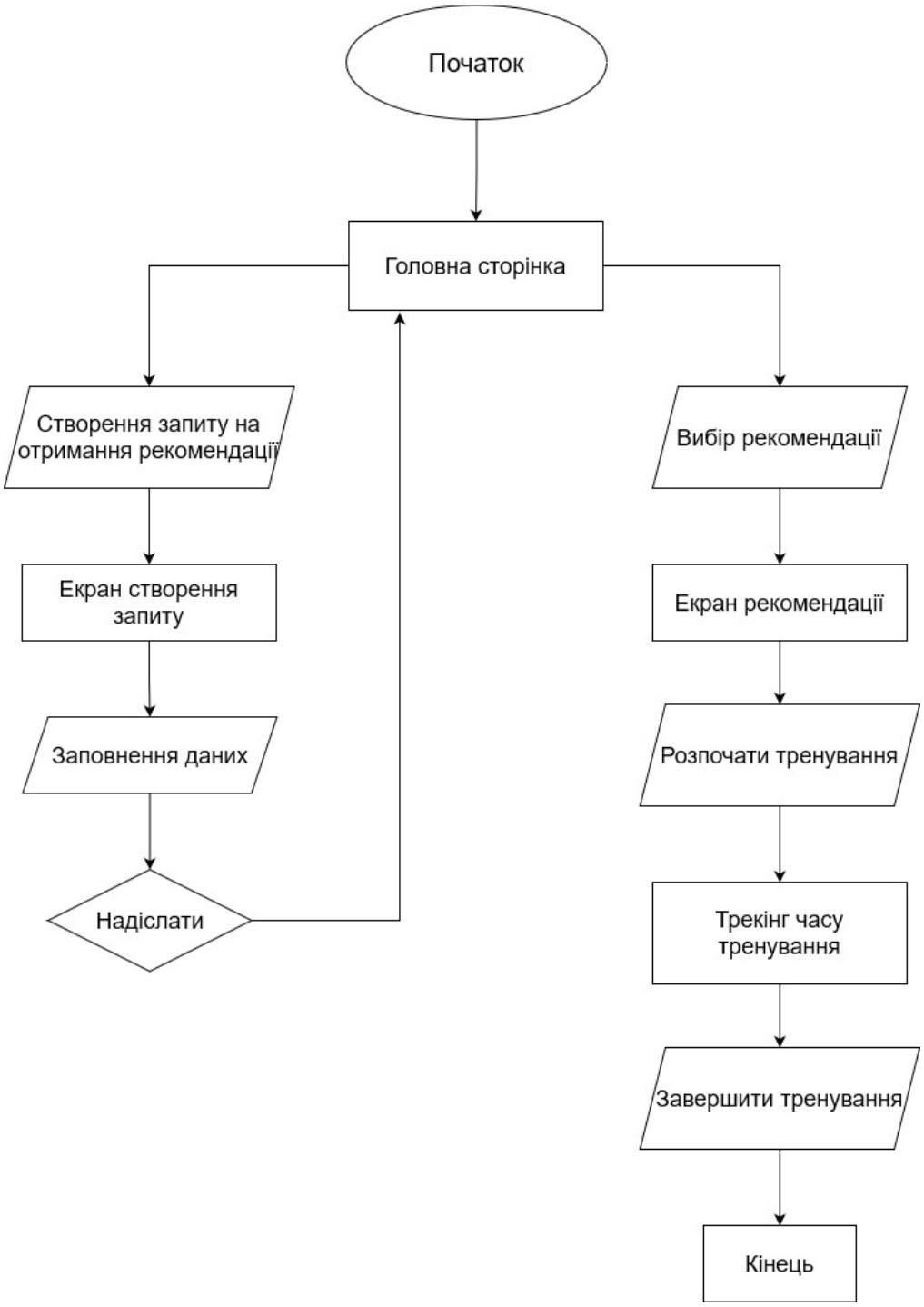

Рисунок В.8 – Схема алгоритму функціонування мобільного додатку

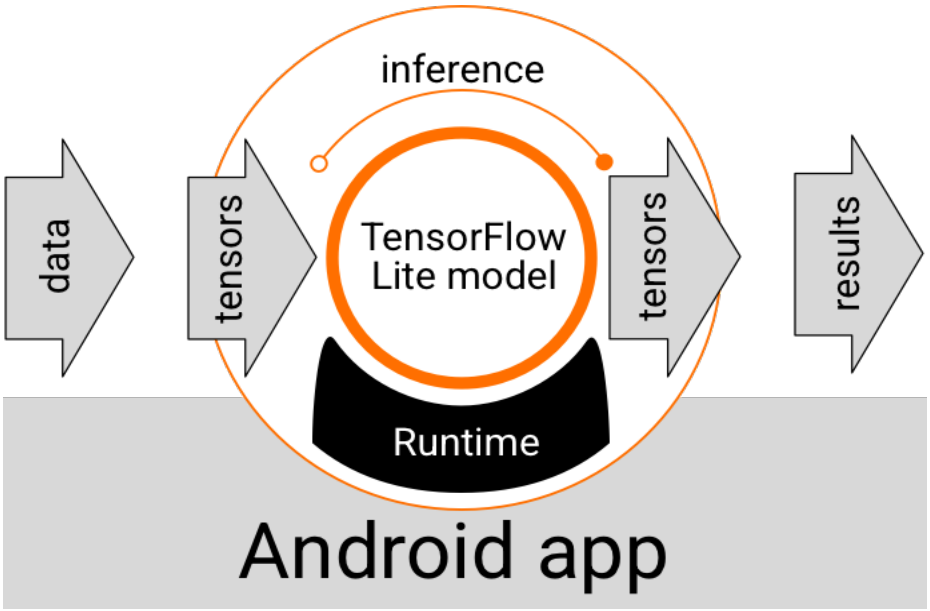

Рисунок В.9 – Схема роботи TensorFlow Lite

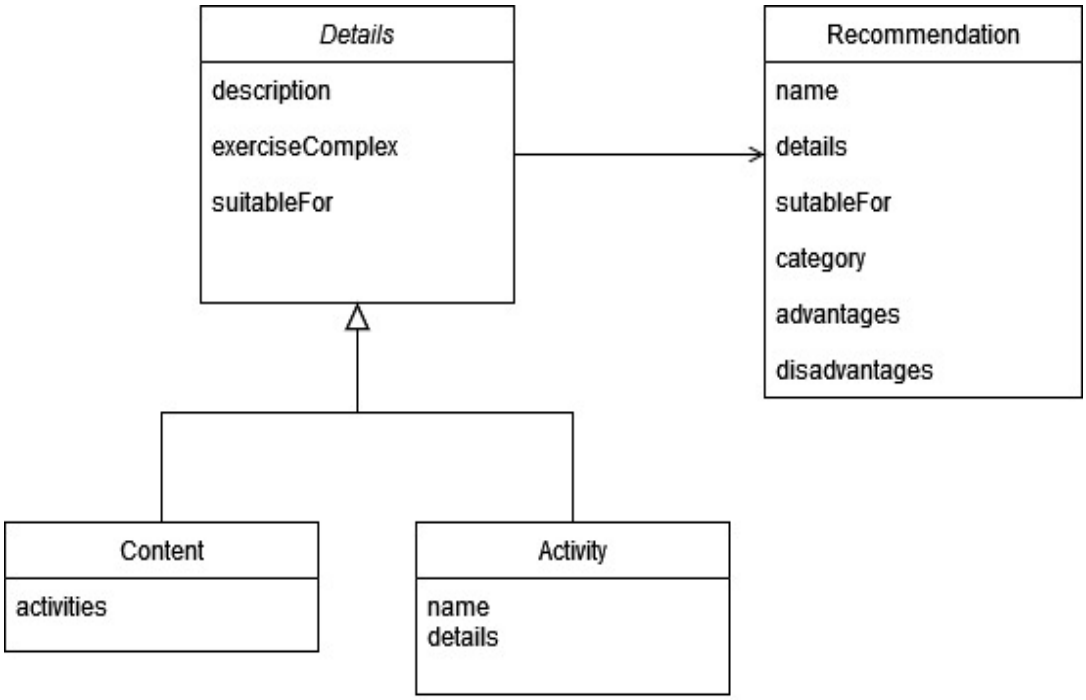

Рисунок В.10 – UML-діаграма класів інформаційної технології розробки рекомендацій щодо спортивних активностей

Додаток Г (довідниковий)

ІНСТРУКЦІЯ КОРИСТУВАЧА

Для початку роботи з інформаційною технологією необхідно відкрити програмний додаток, який встановлений в пристрої, після чого опинимось на головній сторінці додатку (рис. Г.1).

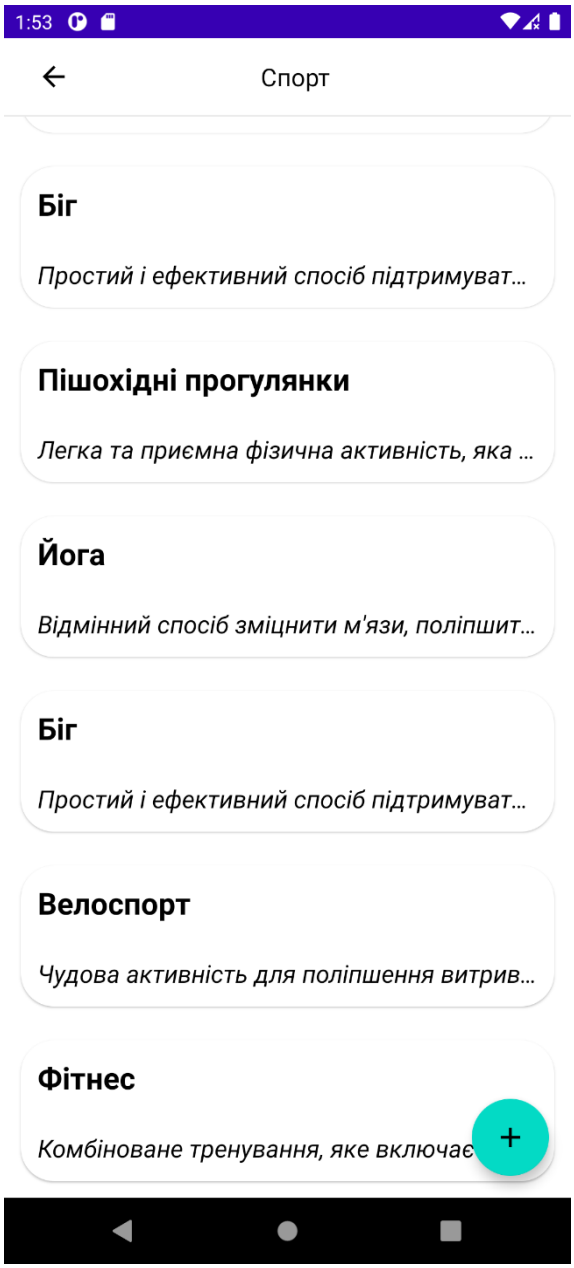

Рисунок Г.1 – Сторінка зі списком рекомендацій щодо спортивних активностей

Після запуску програмного додатку потрібно створити запит на отримання рекомендацій. Для цього, необіхдно натиснути кнопку “+”, для того щоб перейти на сторінку формування запиту (рис. Г.2).

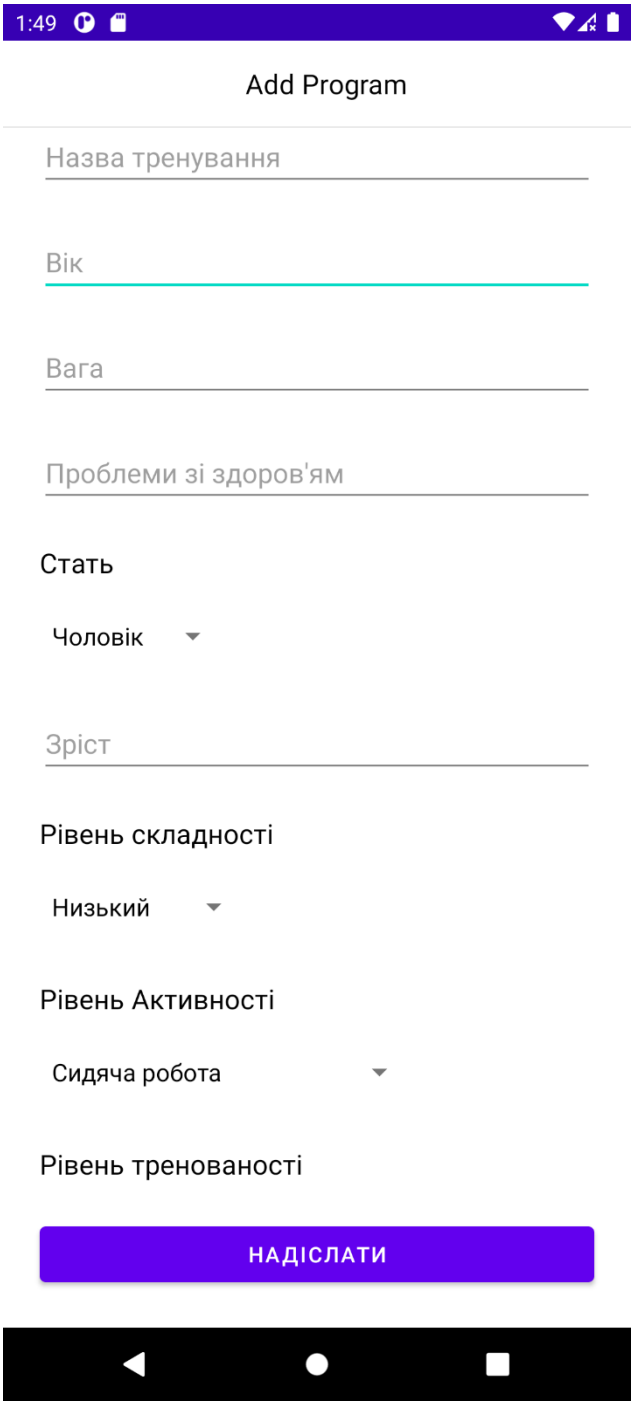

Рисунок Г.2 – Екран сторінки формування запиту для отримання рекомендацій

Після заповнення всіх необхідних даних для отримання рекомендації, натискаємо кнопку “Надіслати”, для відправки запиту, на отримання рекомендацій, на основі введених даних. Далі відбуватиметься обробка даних (рис. Г.3).

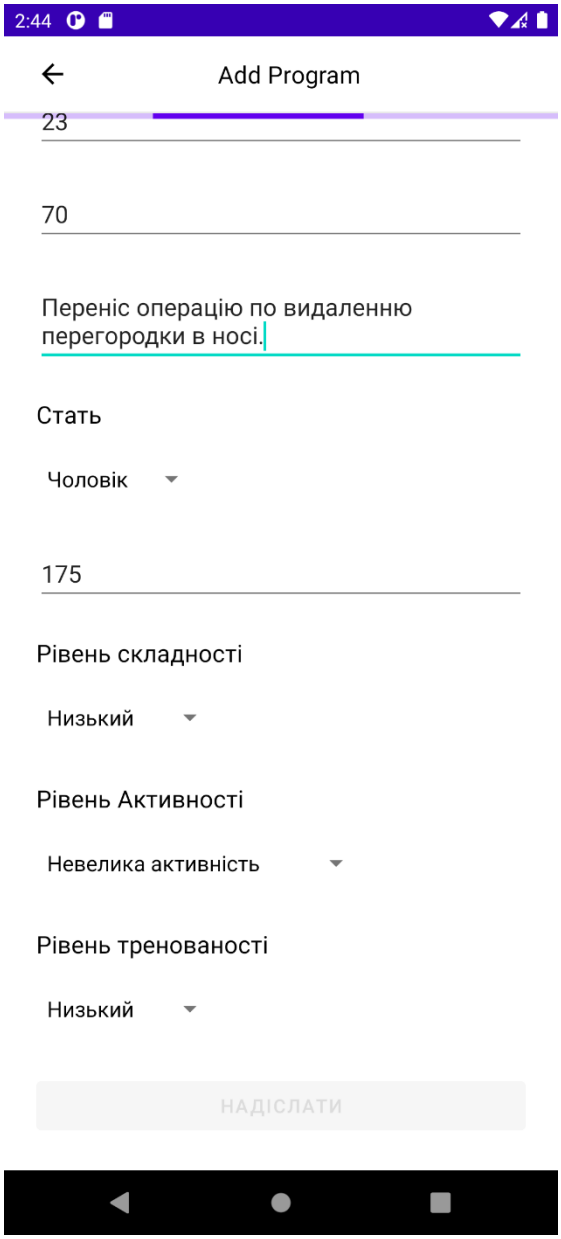

Рисунок Г.3 – Формування запиту на отримання рекомендацій щодо спортивних активностей

Після формування запиту повертаємося на сторінку зі списком рекомендацій, де отримуємо результат надісланого запиту. На даній сторінці, в кінці списку виявлено нові рекомендації, які були надані після сформованого запиту, після чого, обримаємо необхідну рекомендацію (рис. Г.4).

Рисунок Г.4 – Сторінка зі списком нових рекомендацій, які були надані після формування попереднього запиту

На сторінці з детальною інформацію описана категорія, опис рекомендації, для кого підходить, недоліки та комплекс тренувань (рис. Г.5).

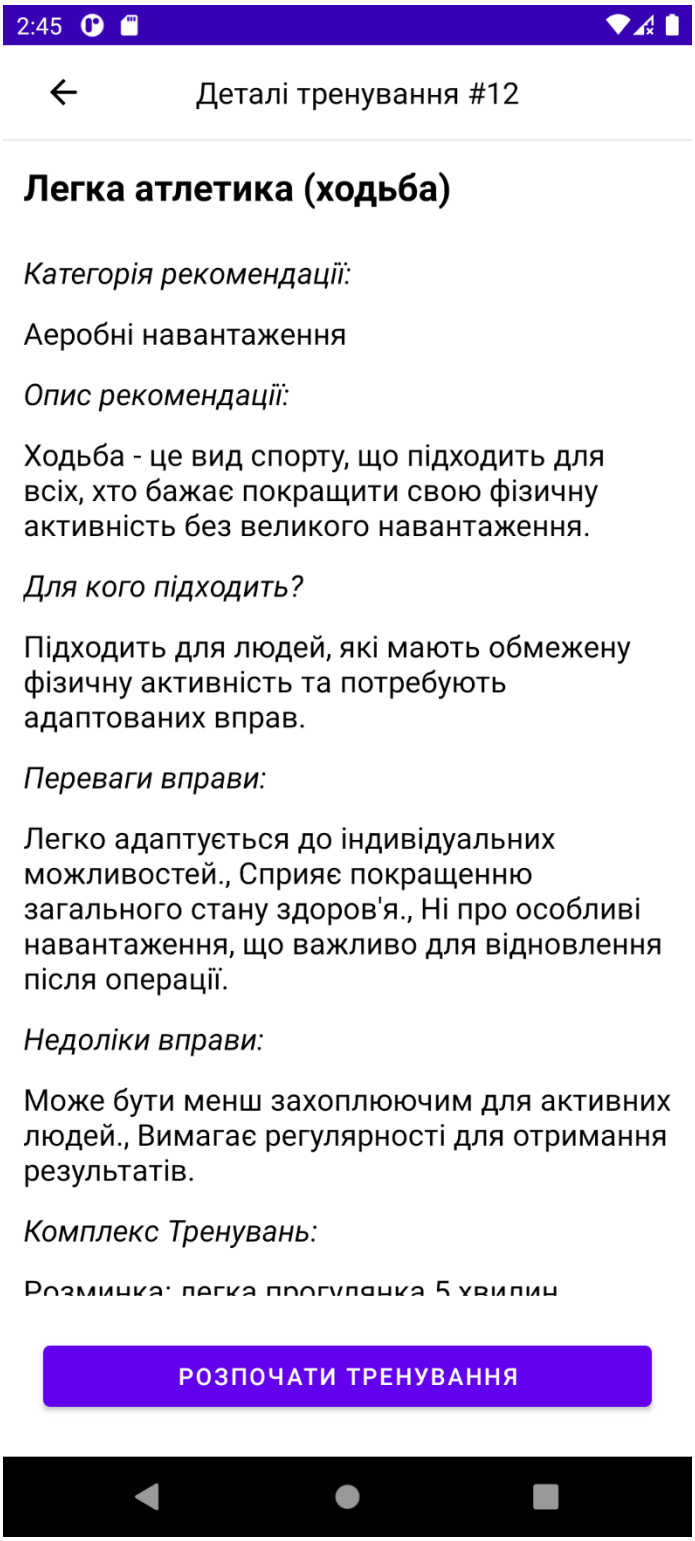

Рисунок Г.5 – Екран сторінки деталей рекомендації програмного додатка

Для того щоб розпочати тренування, на екрані сторінки з деталями рекомендації необхідно натиснути на кнопку “Розпочати тренування”.

Після цього, здійснюється перехід на екран сторінки з трекінгом часу тренування (рис. Г.6).

Рисунок Г.6 – Екран сторінки з трекінгом часу тренування

Для завершення тренування необхідно натистунти на кнопку “Стоп”, яка змінюється з кнопкою “Старт”. Результат проведеного тренування відображається на сторінці екрана рекомендації, в кінці сторінки (рис. Г.7).

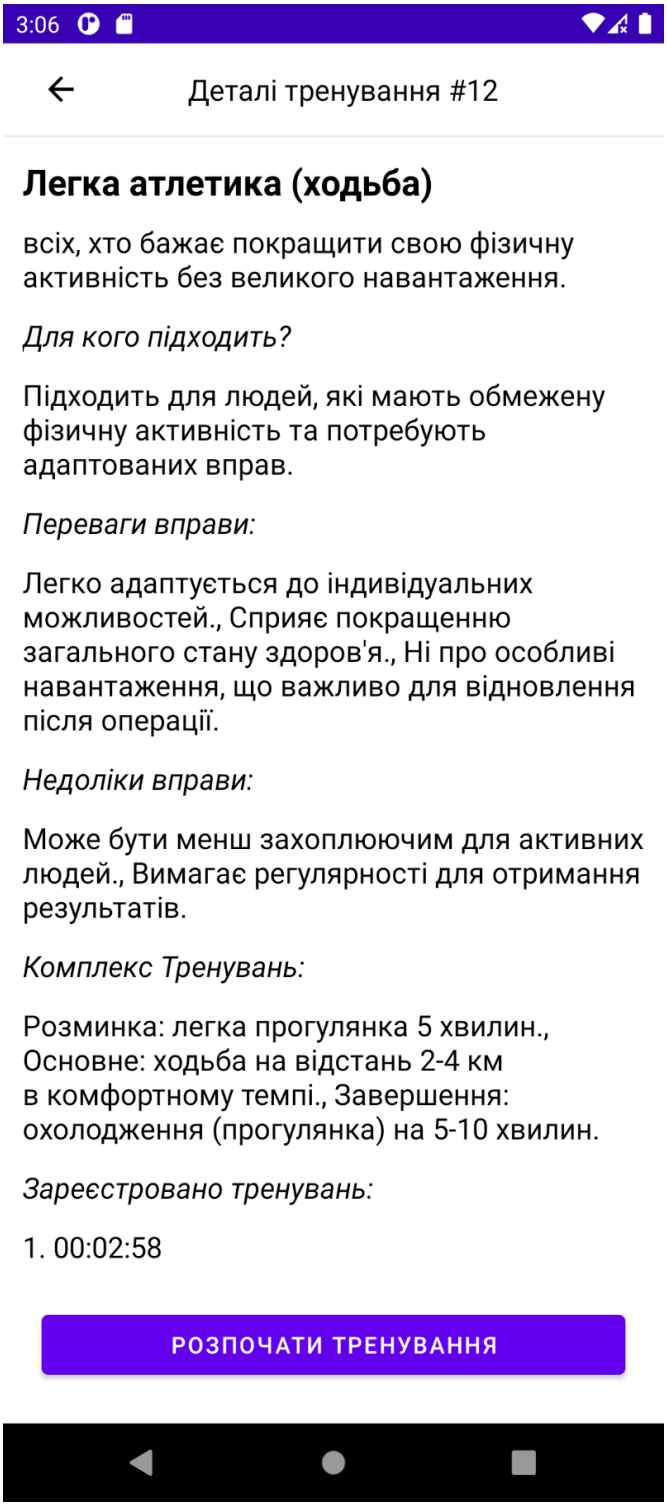

Рисунок Г.7 – Екран сторінки деталей рекомендації, де зображується час який було виділено на тренування