Міністерство освіти і науки України
Тернопільський національний технічний університет імені Івана Пулюя

Комп'ютерно-інформаційних систем і програмної інженерії
(повна назва факультету)
Програмної інженерії
(повна назва кафедри)

КВАЛІФІКАЦІЙНА РОБОТА

на здобуття освітнього ступеня

бакалавр
(назва освітнього ступеня)

на тему: Розробка та тестування мобільного Android-додатку для моніторингу показників стану серцево-судинної системи людини

Виконав: студент IV курсу, групи СП-43
спеціальності 121 – Інженерія програмного забезпечення
(шифр і назва спеціальності)

(підпис) (прізвище та ініціали)

Керівник
(підпис) (прізвище та ініціали)

Нормоконтроль
(підпис) (прізвище та ініціали)

Завідувач кафедри
(підпис) (прізвище та ініціали)

Рецензент
(підпис) (прізвище та ініціали)

Тернопіль 2026

Міністерство освіти і науки України
Тернопільський національний технічний університет імені Івана Пулюя

Факультет комп'ютерно-інформаційних систем і програмної інженерії
(повна назва факультету)

Кафедра програмної інженерії
(повна назва кафедри)

ЗАТВЕРДЖУЮ
Завідувач кафедри

(підпис) (прізвище та ініціали)

« » 2026 р.

З А В Д А Н Н Я

НА КВАЛІФІКАЦІЙНУ РОБОТУ

на здобуття освітнього ступеня бакалавр
(назва освітнього ступеня)

за спеціальністю 121 інженерія програмного забезпечення
(шифр і назва спеціальності)

студенту Попович Микола Андрійович
(прізвище, ім'я, по батькові)

1. Тема роботи Розробка та тестування мобільного Android-додатку для моніторингу показників стану серцево-судинної системи людини

Керівник роботи Мудрик Іван Ярославович, док. Філософії
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

Затверджені наказом ректора від « 6 » квітня 2026 року № 4/9-171

2. Термін подання студентом завершеної роботи

3.Вихідні дані для роботи
Технічне завдання на розробку мобільного Android-додатку; документація Android Developers, Kotlin та Jetpack Compose; наукова та навчальна література за темою роботи

4. Зміст роботи (перелік питань, які потрібно розробити)
Вступ
Аналіз предметної області та постановка задачі
Проєктування та розробка мобільного Android-додатку для моніторингу показників стану серцево-судинної системи людини.
Тестування, впровадження та підтримка програмної системи.
Безпека життєдіяльності основи охорони праці
Висновки

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень, слайдів)
Діаграма варіантів використання (Use Case Diagram) мобільного додатку CardioMonitor.
Діаграма класів програмної системи.
Архітектура програмного забезпечення на основі шаблону MVVM.
Структура бази даних мобільного додатку.

Блок-схеми основних алгоритмів роботи системи.
Інтерфейс користувача мобільного додатку.
Графіки аналізу показників стану серцево-судинної системи.
Результати тестування програмного продукту.

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв

7. Дата видачі завдання 6 квітня 2026

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів роботи	Термін виконання етапів роботи	**Примітка**
1	Ознайомлення з завданням кваліфікаційної роботи		
2	Збір та аналіз інформації за темою дослідження		
3	Формування структури пояснювальної записки		
4	Розробка технічного завдання та вибір методів реалізації		
5	Проєктування мобільного Android-додатку для моніторингу показників серцево-судинної системи		
6	Розробка користувацького інтерфейсу та бази даних додатку		
7	Реалізація функціоналу моніторингу показників серцево-судинної системи		
8	Тестування функціоналу мобільного додатку		
9	Написання розділів пояснювальної записки (1–3 розділи)		
10	Написання розділу 4: «Охорона праці та безпека життєдіяльності»		
11	Перевірка роботи керівником, внесення правок		
12	Нормоконтроль		
13	Перевірка кваліфікаційної роботи на плагіат		
14	Попередній захист кваліфікаційної роботи		
15	Захист кваліфікаційної роботи		

Студент _____ Попович М. А.
(підпис) (прізвище та ініціали)

Керівник роботи _____ Мудрик І. Я.
(підпис) (прізвище та ініціали)

АНОТАЦІЯ

Кваліфікаційна робота бакалавра за спеціальністю «Інженерія програмного забезпечення» на тему: «Розробка та тестування мобільного Android-додатку для моніторингу показників стану серцево-судинної системи людини».

Пояснювальна записка містить 64 сторінок, 33 рисунки, 6 таблиць, 3 додатки та 22 використаних джерела.

У даній кваліфікаційній роботі розглянуто процес проєктування та розробки мобільного Android-додатку для моніторингу показників серцево-судинної системи людини. Розроблений додаток призначений для введення, збереження та аналізу основних показників стану здоров’я користувача, зокрема артеріального тиску та пульсу.

Актуальність роботи обумовлена необхідністю контролю стану серцево-судинної системи та зростанням популярності мобільних засобів моніторингу здоров’я. Використання мобільного додатку дозволяє користувачам оперативно отримувати інформацію про власні показники та зберігати історію вимірювань.

Метою роботи є проєктування та розробка мобільного Android-додатку для моніторингу показників серцево-судинної системи людини.

Очікуваним результатом є створення функціонального мобільного додатку з можливістю введення, збереження та перегляду показників стану серцево-судинної системи, а також відображення історії вимірювань у зручному для користувача вигляді.

Ключові слова: Android-додаток, мобільний додаток, серцево-судинна система, моніторинг здоров’я, артеріальний тиск, пульс, Android Studio, база даних.

ABSTRACT

Bachelor's qualification work in the specialty "Software Engineering" on the topic: "Development and testing of a mobile Android application for monitoring indicators of the human cardiovascular system".

The explanatory note contains 64 pages, 33 figures, 6 tables, 3 appendices and 22 references.

This qualification work considers the process of designing and developing a mobile Android application for monitoring indicators of the human cardiovascular system. The developed application is intended for input, storage and analysis of the main health indicators of the user, in particular blood pressure and pulse rate.

The relevance of the work is determined by the necessity of monitoring the condition of the cardiovascular system and the growing popularity of mobile health monitoring tools. The use of a mobile application allows users to quickly obtain information about their health indicators and store the history of measurements.

The purpose of the work is the design and development of a mobile Android application for monitoring indicators of the human cardiovascular system.

The expected result is the creation of a functional mobile application with the ability to input, store and review indicators of the cardiovascular system, as well as display the history of measurements in a convenient form for the user.

Keywords: Android application, mobile application, cardiovascular system, health monitoring, blood pressure, pulse, Android Studio, database

ПЕРЕЛІК СКОРОЧЕНЬ ТЕРМІНІВ

Android – операційна система для мобільних пристроїв, розроблена компанією Google.

Android Studio – офіційне інтегроване середовище розробки (IDE) для створення Android-додатків.

SDK (Software Development Kit) – набір засобів для розробки програмного забезпечення.

API (Application Programming Interface) – інтерфейс програмування застосунків, який забезпечує взаємодію між програмними компонентами.

UI (User Interface) – користувацький інтерфейс програмного забезпечення.

UX (User Experience) – взаємодія користувача з програмним продуктом та загальний досвід використання.

SQLite – вбудована реляційна база даних для зберігання інформації в мобільному додатку.

Room Database – бібліотека для роботи з базою даних SQLite в Android.

BPM (Beats Per Minute) – кількість ударів серця за хвилину, показник пульсу.

Артеріальний тиск – показник тиску крові на стінки судин під час роботи серця.

Серцево-судинна система – система органів людини, що забезпечує циркуляцію крові в організмі.

Моніторинг – процес спостереження, збору та аналізу показників стану організму.

ЗМІСТ

ВСТУП

Інформаційні технології сьогодні активно застосовуються в різних напрямах діяльності людини, серед яких важливе місце займає сфера охорони здоров'я. Використання сучасних мобільних пристроїв і програмного забезпечення відкриває можливості для створення засобів контролю фізичного стану людини, накопичення медичних даних та спостереження за показниками здоров'я в повсякденних умовах.

Однією з найбільш актуальних проблем сучасної медицини залишаються захворювання серцево-судинної системи, які належать до основних причин смертності населення у світі [1]. Для своєчасного виявлення можливих порушень важливим є регулярне спостереження за такими показниками, як артеріальний тиск і частота серцевих скорочень. Традиційний спосіб фіксації результатів вимірювань часто пов'язаний із використанням паперових записів або декількох окремих програмних засобів, що не завжди є зручним для користувача.

Актуальність теми роботи обумовлена потребою у створенні мобільного програмного засобу, який забезпечуватиме зручний моніторинг показників стану серцево-судинної системи людини. Використання такого додатку дає можливість оперативно вносити результати вимірювань, накопичувати їх у цифровому форматі, переглядати історію змін та отримувати доступ до збереженої інформації у будь-який зручний момент.

Метою кваліфікаційної роботи є розробка мобільного Android-додатку для моніторингу показників стану серцево-судинної системи людини.

Для досягнення поставленої мети необхідно виконати такі завдання:

– дослідити предметну область та особливості контролю показників серцево-судинної системи людини;

– проаналізувати існуючі мобільні рішення для моніторингу стану здоров'я;

– сформувати функціональні та нефункціональні вимоги до програмного продукту;

– обґрунтувати вибір інструментів і технологій розробки;

– виконати проєктування архітектури додатку та структури бази даних;

– реалізувати мобільний додаток для введення, збереження та аналізу показників стану серцево-судинної системи;

– провести тестування створеного програмного продукту та оцінити результати його роботи.

Об’єктом дослідження є процес моніторингу показників стану серцево-судинної системи людини.

Предметом дослідження є методи, засоби та технології створення мобільних Android-додатків для накопичення, обробки та відображення показників стану здоров’я користувача. Практична цінність отриманих результатів полягає у створенні мобільного програмного засобу для ведення історії вимірювань артеріального тиску та пульсу, що може використовуватися для підвищення ефективності самоконтролю стану здоров’я користувачів.

Структура кваліфікаційної роботи включає вступ, чотири розділи, висновки, список використаних джерел та додатки. У першому розділі виконано аналіз предметної області та розглянуто існуючі програмні рішення. Другий розділ присвячено проєктуванню програмного продукту та обґрунтуванню вибору технологій розробки. У третьому розділі описано процес реалізації та тестування мобільного додатку. У четвертому розділі розглянуто питання охорони праці та безпеки життєдіяльності. У висновках наведено основні результати виконаної роботи та окреслено напрями подальшого розвитку програмного продукту.

1. АНАЛІТИЧНА ЧАСТИНА

У даному розділі виконано аналіз предметної області та досліджено особливості моніторингу показників стану серцево-судинної системи людини. Розглянуто існуючі програмні рішення, сучасні технології мобільної розробки та сформовано основні вимоги до програмного продукту.

1.1 Актуальність теми та опис предметної області

Стрімкий розвиток інформаційних технологій зумовлює їх широке застосування в різних сферах діяльності, зокрема в галузі охорони здоров’я. Значного поширення набули мобільні додатки, які надають можливість здійснювати контроль за станом здоров’я, зберігати результати вимірювань та швидко отримувати доступ до необхідних даних. Завдяки масовому використанню смартфонів мобільні технології стали зручним і доступним засобом для спостереження за різними показниками фізичного стану людини.

Серцево-судинна система є однією з найважливіших систем організму людини, оскільки забезпечує транспортування кисню та поживних речовин до всіх органів і тканин. Порушення її функціонування можуть призводити до виникнення серйозних захворювань, які негативно впливають на якість життя людини. Саме тому регулярний контроль показників стану серцево-судинної системи є важливою складовою профілактики та раннього виявлення можливих відхилень [1,2].

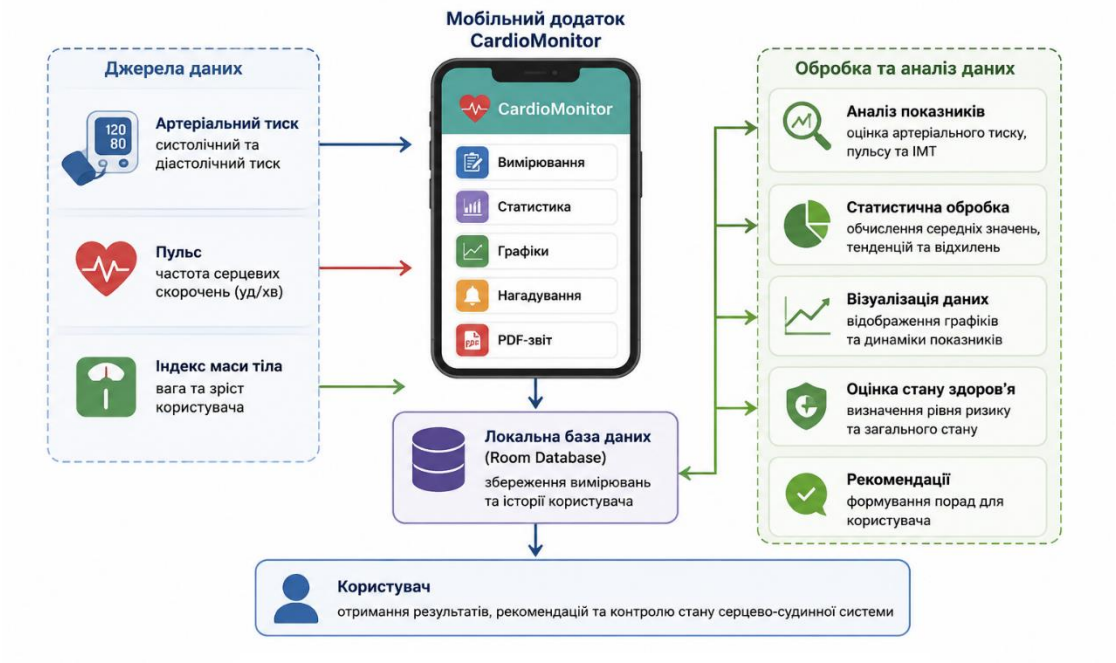

Рисунок 1.1 – Загальна схема моніторингу показників серцево-судинної системи за допомогою додатку CardioMonitor

До основних показників, які характеризують стан серцево-судинної системи людини, належать артеріальний тиск та частота серцевих скорочень (пульс). Регулярне вимірювання та аналіз цих показників дозволяють своєчасно виявляти зміни у функціонуванні організму та приймати відповідні заходи щодо підтримання здоров'я.

Традиційний підхід до фіксації результатів вимірювань передбачає їх запис на паперових носіях або зберігання без належного впорядкування, що ускладнює подальшу обробку та аналіз накопиченої інформації. Використання мобільних застосунків дає змогу спростити процес збереження отриманих даних, забезпечити зручний перегляд попередніх результатів та підвищити ефективність роботи з історією вимірювань.

Операційна система Android є однією з найпоширеніших мобільних платформ у світі. Вона надає широкі можливості для створення сучасних програмних продуктів, які можуть працювати на великій кількості мобільних пристроїв. Це робить платформу доцільним вибором для реалізації мобільного

додатку, призначеного для моніторингу показників стану серцево-судинної системи людини [5].

Таким чином, розробка мобільного додатку для моніторингу показників стану серцево-судинної системи людини є актуальним завданням, спрямованим на підвищення зручності контролю стану здоров’я користувачів та покращення процесу зберігання й аналізу медичних показників.

1.2 Аналіз серцево-судинної системи людини та основних показників стану здоров'я

Серцево-судинна система людини являє собою сукупність органів, які забезпечують циркуляцію крові в організмі. Основними елементами цієї системи є серце та кровоносні судини. Їх основним завданням є постачання органів і тканин киснем, поживними речовинами та виведення продуктів обміну речовин.

Серце виконує функцію насоса, який забезпечує безперервний рух крові по судинах. Під час скорочення серцевого м'яза кров надходить до артерій, а під час розслаблення серце наповнюється новою порцією крові. Від ефективності роботи серця залежить нормальне функціонування всіх органів людського організму [1,3].

Для оцінки стану серцево-судинної системи застосовується низка показників, серед яких найбільш інформативними та поширеними є артеріальний тиск і частота серцевих скорочень. Саме ці параметри найчастіше вимірюються користувачами в домашніх умовах та використовуються як базові критерії для попереднього контролю стану здоров’я людини.

Артеріальний тиск характеризує силу тиску крові на стінки кровоносних судин під час роботи серця. Під час вимірювання визначають два значення: систолічний та діастолічний тиск. Систолічний тиск відображає тиск у момент скорочення серця, а діастолічний — у момент його розслаблення. Для дорослої людини нормальним вважається артеріальний тиск близько 120/80 мм рт. ст., хоча допустимі значення можуть змінюватися залежно від віку та індивідуальних особливостей організму [1].

Таблиця 1.1 - Орієнтовні нормальні показники серцево-судинної системи

Показник	Нормальне значення
Систолічний тиск	90–120 мм рт. ст.
Діастолічний тиск	60–80 мм рт. ст.
Пульс у стані спокою	60–90 уд./хв

Іншим важливим показником є частота серцевих скорочень або пульс. Пульс визначає кількість ударів серця за одну хвилину [3]. У стані спокою нормальним для більшості дорослих людей вважається показник від 60 до 90 ударів за хвилину. Відхилення від нормальних значень можуть свідчити про фізичне навантаження, емоційний стан або наявність певних захворювань.

Регулярний контроль артеріального тиску та пульсу дає змогу своєчасно виявляти зміни стану здоров’я. Для цього дедалі частіше застосовуються сучасні програмні засоби зберігання та аналізу даних.

Рисунок 1.2 - Приклад вимірювання артеріального тиску та пульсу

Використання мобільного додатку дозволяє користувачеві вести електронний журнал вимірювань, переглядати історію показників та аналізувати їх зміни протягом певного періоду часу. Це підвищує зручність контролю стану

здоров'я та сприяє більш відповідальному ставленню до профілактики серцево-судинних захворювань.

1.3 Огляд існуючих мобільних додатків для моніторингу стану здоров’я

На сьогодні розроблено велику кількість мобільних застосунків, орієнтованих на контроль фізичної активності, спостереження за станом здоров’я та накопичення медичних даних користувачів. Подібні програмні рішення надають можливість фіксувати різні показники організму, відстежувати їх зміни в динаміці та отримувати статистичні відомості за обраний проміжок часу.

Одним із найпоширеніших рішень є Google Fit. Даний додаток розроблений компанією Google та призначений для відстеження активності користувача. Google Fit дозволяє збирати інформацію про кількість пройдених кроків, фізичні навантаження, спалені калорії та інші показники. Крім того, додаток підтримує інтеграцію з різними пристроями та сервісами для моніторингу здоров’я [5].

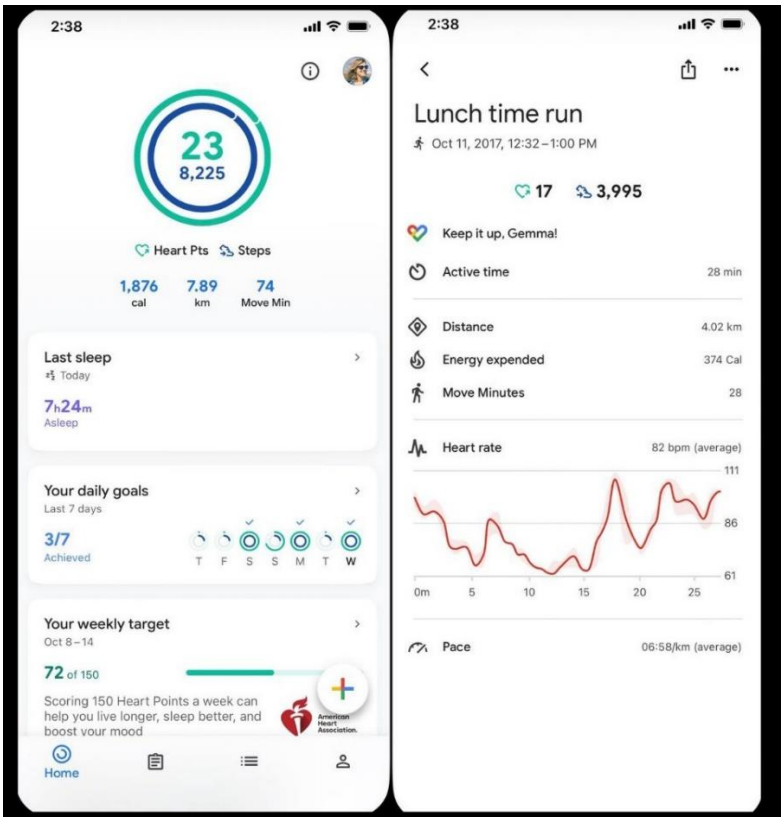

Рисунок 1.3 - Інтерфейс мобільного додатку Google Fit

Ще одним популярним рішенням є Samsung Health. Даний програмний продукт надає користувачам можливість контролювати фізичну активність, показники сну, рівень стресу, пульс та інші параметри організму. Перевагою додатку є широкий функціонал та інтеграція з мобільними пристроями компанії Samsung [2].

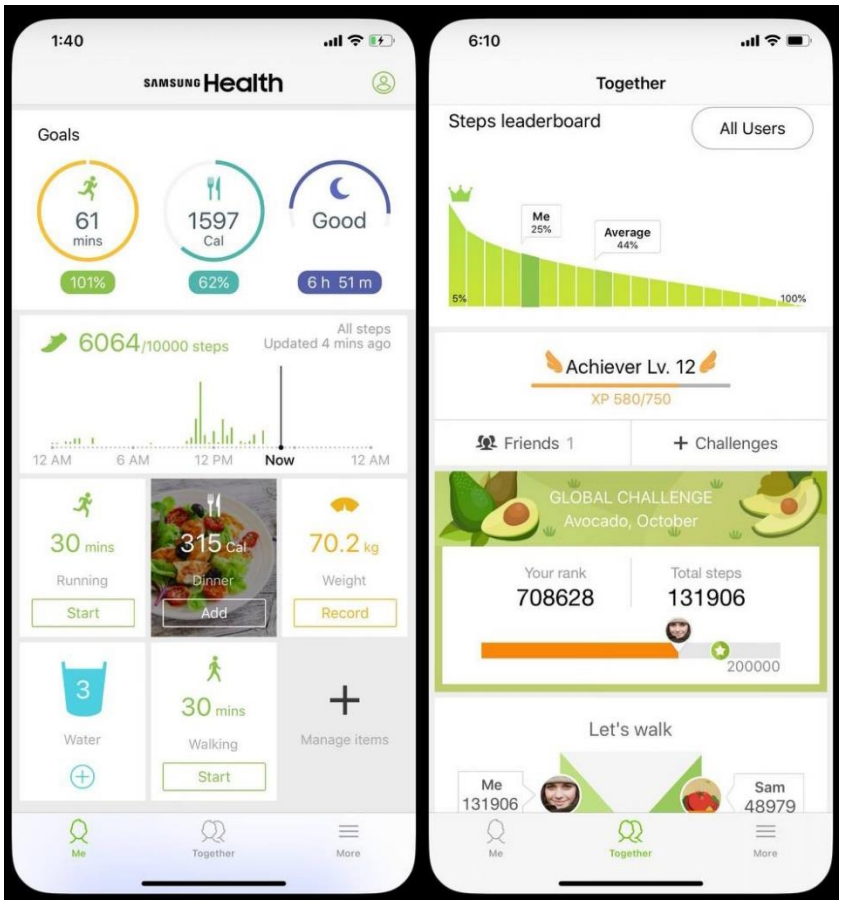

Рисунок 1.4 - Інтерфейс мобільного додатку Samsung Health

Для спостереження за показниками артеріального тиску застосовуються спеціалізовані мобільні додатки, серед яких можна виділити Blood Pressure Tracker. Головною функцією цього програмного продукту є накопичення результатів вимірювань артеріального тиску та надання користувачу доступу до історії збережених даних. Завдяки цьому користувач має можливість відстежувати динаміку змін показників за тривалий проміжок часу та переглядати узагальнену статистичну інформацію.

Також існують додатки для моніторингу частоти серцевих скорочень, які дозволяють зберігати інформацію про пульс та аналізувати його зміни. Такі рішення орієнтовані на вузьке коло завдань та не завжди забезпечують комплексний контроль показників серцево-судинної системи.

Проведений аналіз існуючих програмних продуктів показав, що більшість додатків або орієнтовані на загальний моніторинг фізичної активності, або призначені для контролю окремих показників здоров’я. Тому доцільним є створення мобільного додатку, який забезпечить зручне введення, зберігання та перегляд основних показників стану серцево-судинної системи людини в межах єдиного програмного продукту.

Таблиця 1.2 - Порівняння існуючих мобільних додатків

Функція	Google Fit	Samsung Health	Blood Pressure Tracker	Розроблюваний додаток
Збереження тиску	Частково	Так	Так	Так
Збереження пульсу	Так	Так	Частково	Так
Історія вимірювань	Так	Так	Так	Так
Графіки показників	Так	Так	Так	Так
Простий інтерфейс	Середній	Середній	Так	Так

1.4 Аналіз сучасних технологій мобільної розробки

Сучасний ринок мобільних додатків активно розвивається та пропонує широкий вибір технологій для створення програмного забезпечення. Найбільш популярними платформами для мобільної розробки є Android та iOS. Кожна з них має власні інструменти розробки, програмні бібліотеки та середовища виконання.

Операційна система Android є однією з найпоширеніших мобільних платформ, що пояснюється її відкритою архітектурою, підтримкою великої кількості пристроїв та широким набором інструментів для розробки програмного забезпечення. Більшість сучасних смартфонів функціонує на базі Android, тому дана платформа є доцільним вибором для створення та впровадження мобільних застосунків.

Розробка мобільних застосунків для операційної системи Android виконується за допомогою інтегрованого середовища Android Studio. Це програмне середовище надає повний набір інструментів, необхідних для створення, тестування та налагодження мобільного програмного забезпечення. Серед його можливостей варто відзначити засоби проєктування користувацького інтерфейсу, емуляцію роботи мобільних пристроїв, а також підтримку взаємодії із системами контролю версій [13].

Під час розробки мобільних застосунків для платформи Android переважно використовуються мови програмування Java та Kotlin. Мова Kotlin є сучасним інструментом розробки, який офіційно рекомендований компанією Google для створення додатків. Її перевагами є компактність програмного коду, підвищена надійність та зручність у процесі розробки. З огляду на це, для реалізації розроблюваного програмного продукту було обрано Kotlin як основну мову програмування [6].

Для зберігання даних у мобільних додатках можуть використовуватися локальні бази даних. Одним із найпоширеніших рішень є SQLite. Для спрощення роботи з базою даних у середовищі Android використовується бібліотека Room, яка забезпечує зручний доступ до даних та дозволяє працювати з ними за допомогою

об'єктно-орієнтованого підходу [9, 10]

Отже, використання Android Studio, мови програмування Kotlin та локальної бази даних Room є доцільним рішенням для розробки мобільного додатку моніторингу показників стану серцево-судинної системи людини.

1.5 Постановка задачі

Проведений аналіз предметної області та існуючих програмних рішень показав актуальність створення мобільного додатку для моніторингу показників стану серцево-судинної системи людини.

Основним призначенням розроблюваного програмного продукту є забезпечення користувача можливістю зберігати результати вимірювань артеріального тиску та пульсу, переглядати історію змін показників і отримувати наочне відображення накопичених даних.

Під час виконання кваліфікаційної роботи необхідно реалізувати мобільний додаток, який забезпечуватиме:

- введення показників артеріального тиску;
- введення показників пульсу;
- збереження результатів вимірювань у базі даних;
- перегляд історії вимірювань;
- відображення статистичної інформації;
- побудову графіків зміни показників;
- зручний користувацький інтерфейс.

До програмного продукту висуваються такі основні вимоги:

- простота використання;
- швидке введення інформації;
- надійне збереження даних;
- коректне відображення результатів вимірювань;

- можливість роботи без підключення до мережі Інтернет.

Підсумком виконання кваліфікаційної роботи має бути створення працездатного мобільного застосунку для моніторингу показників стану серцево-судинної системи людини. Розроблений програмний продукт повинен надавати користувачеві можливість здійснювати контроль основних показників здоров’я та виконувати їх подальший аналіз у зручній формі.

2 ПРОЄКТУВАННЯ МОБІЛЬНОГО ANDROID-ДОДАТКУ

У даному розділі описано процес проєктування мобільного додатку CardioMonitor. Також наведено обрану архітектуру системи, структуру бази даних та основні технології, використані під час реалізації програмного продукту.

2.1 Аналіз вимог до мобільного додатку

Перед початком реалізації мобільного додатку було проведено аналіз вимог, які висуваються до майбутнього програмного продукту. Основною метою розробки є створення зручного засобу для контролю показників стану серцево-судинної системи людини, що забезпечуватиме накопичення, обробку та відображення результатів вимірювань артеріального тиску і частоти серцевих скорочень.

Функціональні можливості додатку повинні передбачати ведення особистого журналу вимірювань, доступ до історії збережених показників, аналіз змін стану користувача в часі та формування статистичної інформації на основі накопичених результатів.

Для реалізації поставлених завдань було сформовано функціональні та нефункціональні вимоги до програмного забезпечення.

2.1.1 Функціональні вимоги

Функціональні вимоги визначають набір можливостей, які повинні бути реалізовані в мобільному додатку.

Розроблюваний програмний продукт повинен забезпечувати:

- відображення екрану завантаження (Splash Screen);
- перегляд головної інформаційної сторінки;
- введення показників систолічного та діастолічного артеріального тиску;

- введення показників пульсу;
- додавання текстових приміток до вимірювань;
- збереження результатів вимірювань у локальній базі даних;
- перегляд історії вимірювань;
- фільтрацію вимірювань за часовими періодами;
- редагування раніше збережених записів;
- видалення вимірювань;
- розрахунок середніх показників;
- побудову графіків зміни артеріального тиску та пульсу;
- автоматичний аналіз динаміки показників;
- прогнозування змін показників на основі накопичених даних;
- експорт статистичної інформації у PDF-файл;
- збереження персональних даних користувача;
- розрахунок індексу маси тіла;
- налаштування щоденних нагадувань про проведення вимірювань;
- відображення рівня ризику та поточного стану здоров’я користувача.

2.1.2 Нефункціональні вимоги

Нефункціональні вимоги визначають характеристики якості програмного продукту та умови його експлуатації.

До основних нефункціональних вимог належать:

- зручність та простота користувацького інтерфейсу;
- швидка робота додатку без помітних затримок;
- стабільність роботи на мобільних пристроях під керуванням операційної системи Android;
- коректне збереження даних користувача;

- можливість роботи без підключення до мережі Інтернет;
- надійність зберігання інформації в локальній базі даних;
- масштабованість програмного забезпечення для подальшого розширення функціоналу;
- відповідність сучасним принципам Material Design 3;
- безпечне зберігання персональних даних користувача.

Врахування зазначених вимог дозволяє забезпечити стабільну роботу програмного продукту та створити комфортні умови для його використання.

2.2 Вибір засобів та технологій розробки

Під час розробки мобільного додатку CardioMonitor було використано набір сучасних технологій та програмних засобів, які забезпечують ефективну роботу застосунку, спрощують процес розробки та створюють можливості для подальшого вдосконалення його функціоналу. Як основну мову програмування для реалізації проєкту було обрано Kotlin. Ця мова рекомендована компанією Google для створення Android-застосунків та активно використовується під час розробки сучасного мобільного програмного забезпечення. Серед її переваг варто відзначити компактність програмного коду, підвищену надійність під час роботи з даними та підтримку актуальних підходів до побудови програмних систем [6].

Рисунок 2.1 – Логотип мови програмування Kotlin

Для створення користувацького інтерфейсу використано технологію Jetpack Compose. На відміну від традиційної XML-розмітки, Jetpack Compose дозволяє створювати інтерфейс безпосередньо засобами мови Kotlin. Це спрощує підтримку проєкту, зменшує кількість програмного коду та підвищує швидкість розробки [7].

Дизайн інтерфейсу реалізовано відповідно до принципів Material Design 3. Використання даного підходу дозволяє забезпечити єдиний стиль оформлення елементів керування та покращити зручність взаємодії користувача з програмним продуктом [12].

Для навігації між екранами додатку використовується бібліотека Navigation Compose [8]. Вона забезпечує організацію переходів між сторінками програми та централізоване керування маршрутизацією в межах застосунку. Для зберігання даних використовується Room Database. Room є надбудовою над SQLite та надає зручний механізм роботи з локальною базою даних через об'єктно-орієнтований підхід. У базі даних зберігаються результати вимірювань артеріального тиску, пульсу, примітки користувача та службова інформація [9].

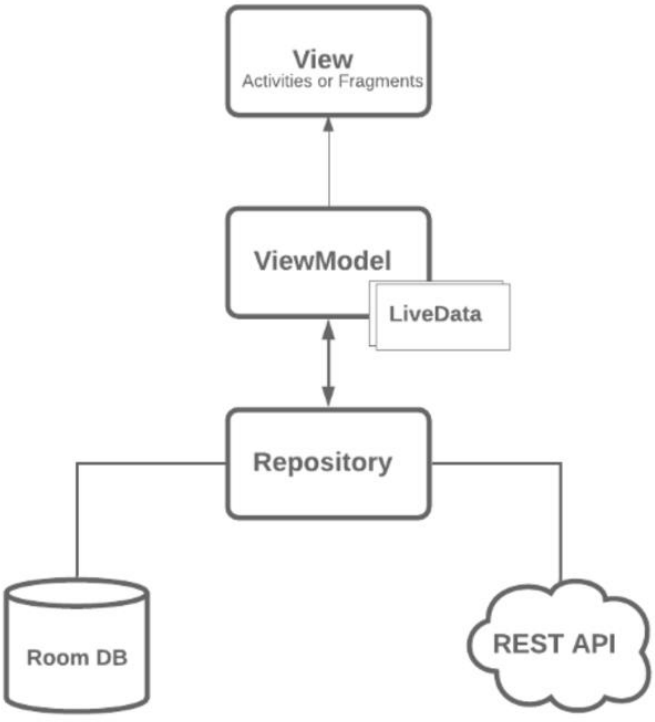

Рисунок 2.2 – Схема взаємодії Room Database та SQLite

Для реалізації асинхронної обробки даних у додатку застосовано Kotlin Coroutines. Використання даної технології забезпечує можливість виконання операцій взаємодії з базою даних та обробки інформації у фоновому режимі, не впливаючи на швидкодію та відгук користувацького інтерфейсу [11].

Передача даних між окремими рівнями архітектури реалізована за допомогою StateFlow. Використання даного інструменту дозволяє синхронізувати стан даних з елементами інтерфейсу та забезпечує їх автоматичне оновлення відповідно до змін, що відбуваються в системі. Для відображення статистики та графіків використовується бібліотека MPAndroidChart. Вона забезпечує побудову лінійних графіків зміни артеріального тиску та частоти серцевих скорочень на основі накопичених вимірювань користувача [14].

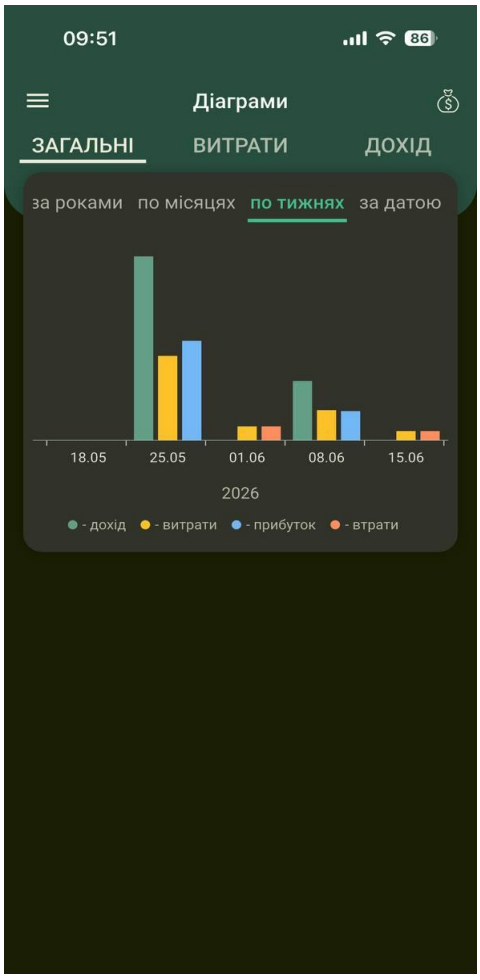

Рисунок 2.3 – Приклад використання графіків у мобільному додатку

Функція формування звітів реалізована за допомогою Android PDFDocument API. Даний програмний інтерфейс дозволяє автоматично створювати PDF-документи, до складу яких входять результати вимірювань та узагальнена статистична інформація щодо стану користувача [5].

Для реалізації механізму нагадувань використано Android Notification System. Завдяки системі сповіщень користувач отримує повідомлення про заплановані вимірювання, що сприяє регулярному контролю показників серцево-судинної системи.

Таким чином, обраний стек технологій повністю відповідає вимогам проєкту та забезпечує реалізацію всіх необхідних функцій мобільного додатку CardioMonitor.

Таблиця 2.1 – Використані технології розробки

Технологія	Призначення
Kotlin	Мова програмування
Jetpack Compose	Побудова інтерфейсу
Material Design 3	Дизайн інтерфейсу
Navigation Compose	Навігація між екранами
Room Database	Локальна база даних
SQLite	Зберігання інформації
Kotlin Coroutines	Асинхронні операції
StateFlow	Управління станом
MPAndroidChart	Побудова графіків
PDFDocument API	Формування PDF
Android Notifications	Система нагадувань

2.3 Проєктування архітектури програмного забезпечення

Під час розробки мобільного додатку CardioMonitor було обрано архітектурний шаблон MVVM (Model–View–ViewModel). Даний підхід є одним із найбільш поширених у сучасній Android-розробці та рекомендований компанією Google [22], для створення мобільних застосунків.

Використання архітектури MVVM дозволяє розділити логіку роботи програмного продукту на окремі рівні, що спрощує підтримку коду, його тестування та подальше розширення функціональних можливостей додатку.

Архітектура MVVM складається з трьох основних компонентів:

- Model;
- View;
- ViewModel.

Компонент Model відповідає за зберігання та обробку даних. У розробленому додатку до цього рівня належать сутності бази даних, DAO-інтерфейси, репозиторії та механізми взаємодії з Room Database [9, 10].

Компонент View відповідає за відображення інформації та взаємодію користувача з програмним продуктом, а його реалізація виконана за допомогою Jetpack Compose. До цього рівня належать усі екрани додатку, зокрема головний екран, форма додавання вимірювань, історія показників, статистичні дані та профіль користувача. Компонент ViewModel виконує роль посередника між користувацьким інтерфейсом та рівнем даних. Його основною функцією є підготовка даних для подальшого відображення на екрані, а також обробка дій користувача без безпосередньої взаємодії з елементами інтерфейсу.

Застосування архітектури MVVM дозволяє мінімізувати залежності між окремими компонентами системи та забезпечує більш структуровану організацію програмного коду.

Для передачі даних між рівнями архітектури використовуються Kotlin Coroutines та StateFlow. Завдяки цьому забезпечується реактивне оновлення користувацького інтерфейсу при зміні інформації в базі даних.

Окрім використання архітектури MVVM, у процесі розробки програмного продукту було враховано сучасні підходи до побудови програмних систем. Особливу увагу приділено розмежуванню функцій між окремими компонентами додатку, що відповідає принципу Single Responsibility Principle (SRP), який входить до складу принципів SOLID. У межах реалізованої архітектури кожен компонент виконує власні завдання: користувацький інтерфейс забезпечує відображення даних, ViewModel відповідає за їх обробку та координацію взаємодії між рівнями системи, а модель даних здійснює збереження та отримання необхідної інформації.

Використання такого підходу сприяє підвищенню масштабованості програмного забезпечення, спрощує його супровід та дозволяє ефективніше виконувати модифікацію окремих модулів без впливу на інші компоненти системи.

Схема взаємодії основних компонентів архітектури MVVM наведена на рисунку 2.4.

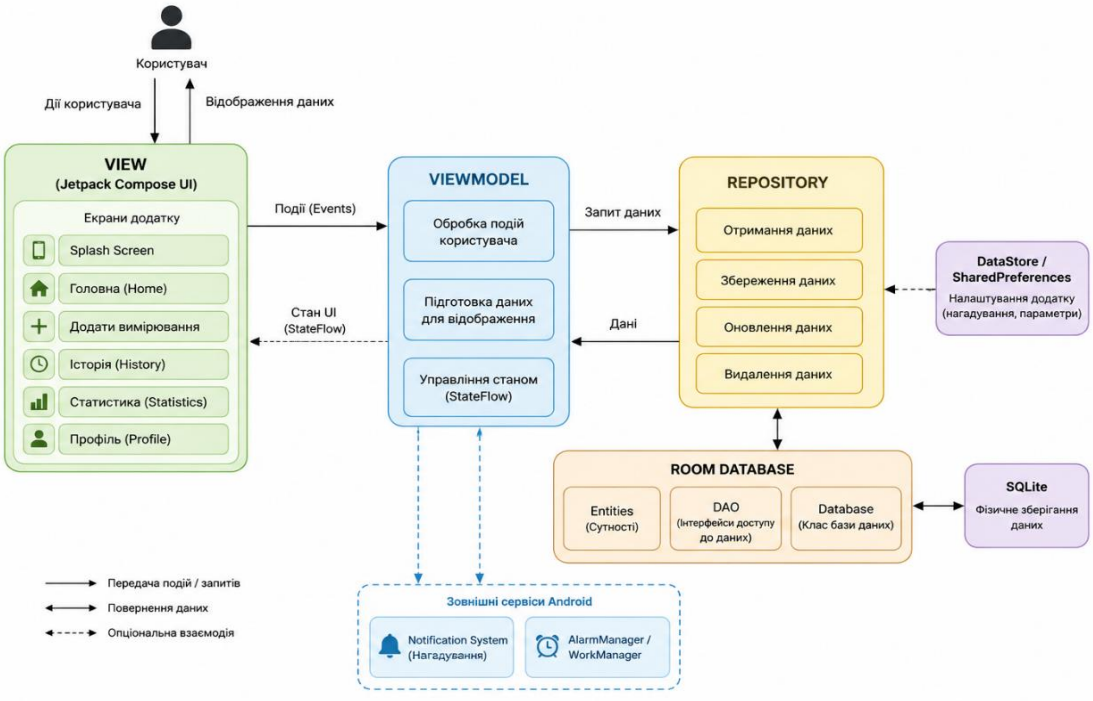

Рисунок 2.4 – Архітектура MVVM мобільного додатку CardioMonitor

Обмін даними між компонентами системи здійснюється за визначеною послідовністю. Спочатку користувач виконує певну дію через елементи інтерфейсу додатку. Отримані дані надходять до ViewModel, де виконуються їх обробка та перевірка. За потреби ViewModel звертається до репозиторію для отримання або оновлення інформації. Репозиторій забезпечує взаємодію з Room Database та повертає необхідні дані до ViewModel. Після завершення обробки актуальний стан системи передається до інтерфейсу користувача і відображається на екрані.

Застосування архітектури MVVM дозволило забезпечити чітке розмежування відповідальності між окремими компонентами програмної системи та підвищити якість програмного забезпечення.

2.4 Проєктування бази даних

Для зберігання даних у мобільному додатку CardioMonitor використовується локальна база даних Room Database. Використання цього рішення забезпечує впорядковане збереження результатів вимірювань користувача та надає можливість оперативного доступу до необхідної інформації незалежно від наявності підключення до мережі Інтернет.

Room Database входить до складу бібліотек Android Jetpack та використовується для організації роботи з локальною базою даних SQLite. Однією з її ключових переваг є можливість взаємодії з даними через об’єкти Kotlin, що суттєво спрощує процес розробки та зменшує потребу у використанні великої кількості SQL-запитів.

У процесі проєктування бази даних було визначено основні сутності, необхідні для функціонування додатку.

Головною сутністю системи є вимірювання показників серцево-судинної системи користувача. Для кожного запису зберігаються значення артеріального тиску, пульсу, дата проведення вимірювання та додаткова примітка користувача.

Структура таблиці вимірювань наведена в таблиці 2.2.

Таблиця 2.2 – Структура таблиці Measurement

Поле	Тип даних	Призначення
Id	Long	Унікальний ідентифікатор запису
systolicPressure	Int	Систолічний тиск
diastolicPressure	Int	Діастолічний тиск
pulse	Int	Частота серцевих скорочень
note	String	Примітка користувача
timestamp	Long	Дата та час вимірювання

Для збереження персональної інформації використовується окрема сутність профілю користувача. У профілі зберігаються дані, необхідні для розрахунку індексу маси тіла та формування персоналізованих рекомендацій.

Таблиця 2.3 – Структура таблиці UserProfile

Поле	Тип даних	Призначення
Name	String	Ім'я користувача
Age	Int	Вік
Gender	String	Стать
Height	Float	Зріст
weight	Float	Вага

На основі даних профілю виконується автоматичний розрахунок індексу маси тіла (IMT), який використовується для формування загальної оцінки стану користувача.

Для взаємодії з базою даних використовуються DAO-інтерфейси (Data Access Object). Вони забезпечують виконання операцій додавання, редагування, видалення та отримання даних із локального сховища.

Загальна структура бази даних мобільного додатку CardioMonitor наведена на рисунку 2.5

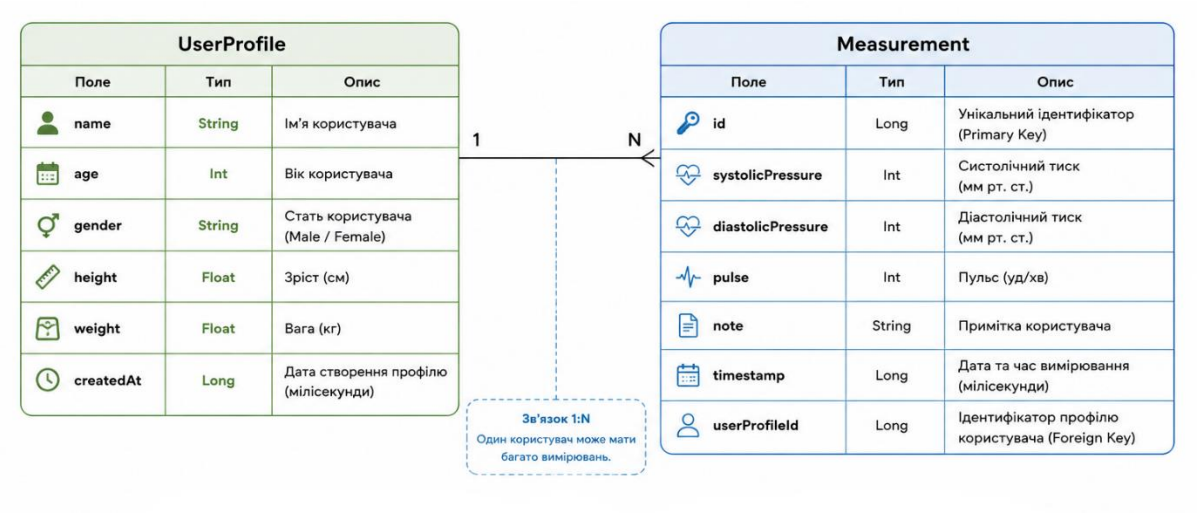

Рисунок 2.5 – Структура бази даних мобільного додатку CardioMonitor

Отже, спроєктована структура бази даних забезпечує ефективне зберігання інформації про результати вимірювань та персональні дані користувача, необхідні для подальшого аналізу стану серцево-судинної системи.

2.5 Проєктування користувацького інтерфейсу

Важливим етапом створення мобільного додатку стало проєктування користувацького інтерфейсу. Під час його розробки основна увага приділялася забезпеченню швидкого доступу до функціональних можливостей додатку, зручності введення даних та зрозумілому представленню результатів аналізу показників стану серцево-судинної системи користувача.

Під час проєктування інтерфейсу використовувалися принципи Material Design 3, які забезпечують сучасний зовнішній вигляд програмного продукту та покращують взаємодію користувача із системою [12].

Структура навігації додатку побудована на основі нижнього навігаційного

меню, яке забезпечує швидкий перехід між основними розділами системи.

До складу мобільного додатку входять такі основні екрани:

- Splash Screen;
- Головна сторінка;
- Додавання вимірювання;
- Історія вимірювань;
- Статистика;
- Профіль користувача.

Початковим екраном мобільного додатку є Splash Screen, який відображається одразу після його запуску. На даному етапі користувачу демонструється логотип програмного продукту, а система виконує завантаження та підготовку необхідних компонентів для подальшої роботи..

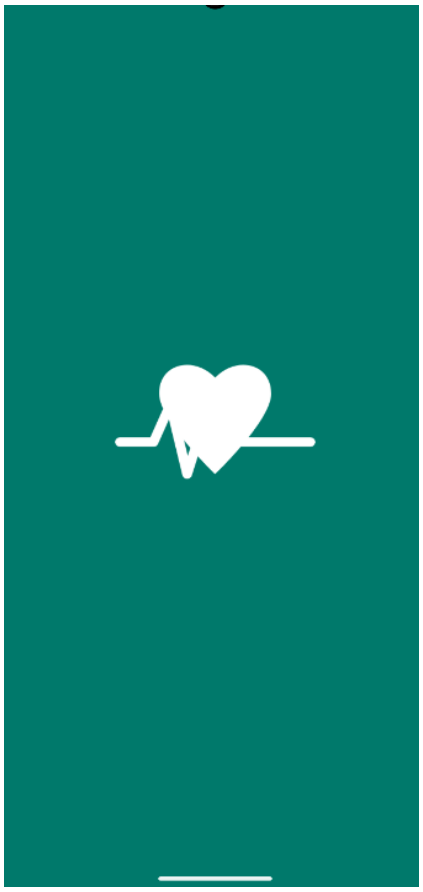

Рисунок 2.6 – Екран завантаження Splash Screen

Головний екран є основною сторінкою мобільного додатку та забезпечує швидкий доступ до найбільш важливої інформації. На ньому відображаються відомості про користувача, дані останнього вимірювання, поточний стан здоров’я та узагальнена оцінка показників організму.

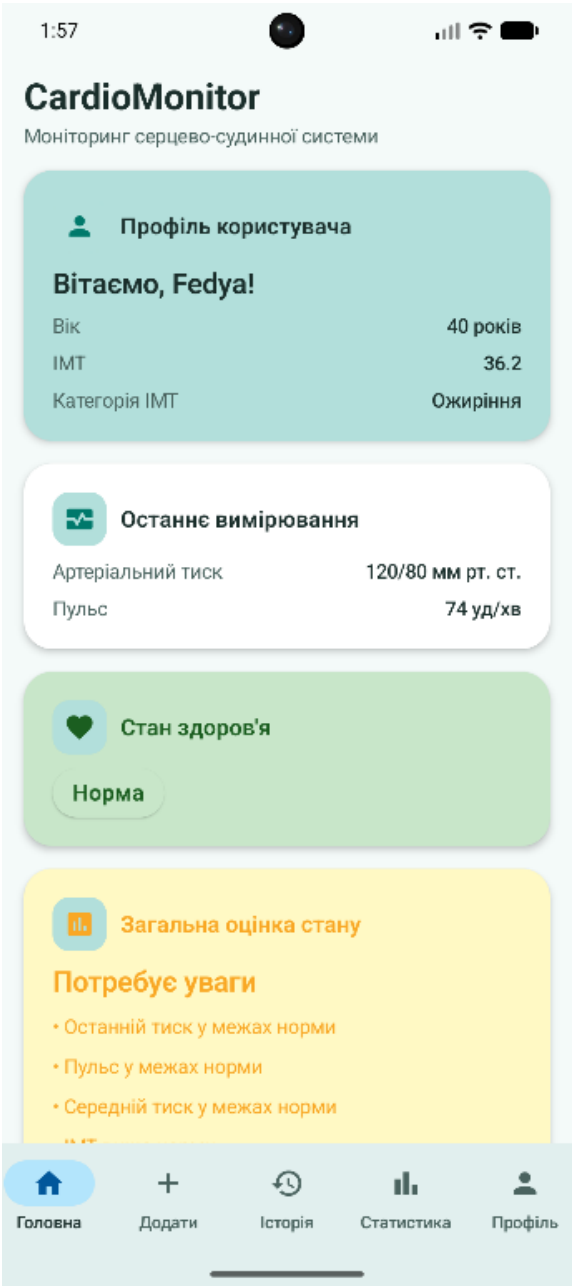

Рисунок 2.7 – Головний екран додатку

Введення нових даних здійснюється за допомогою спеціального екрана додавання вимірювань. На даній сторінці користувач може зберегти показники систолічного та діастолічного артеріального тиску, значення пульсу та за потреби додати примітку з додатковою інформацією.

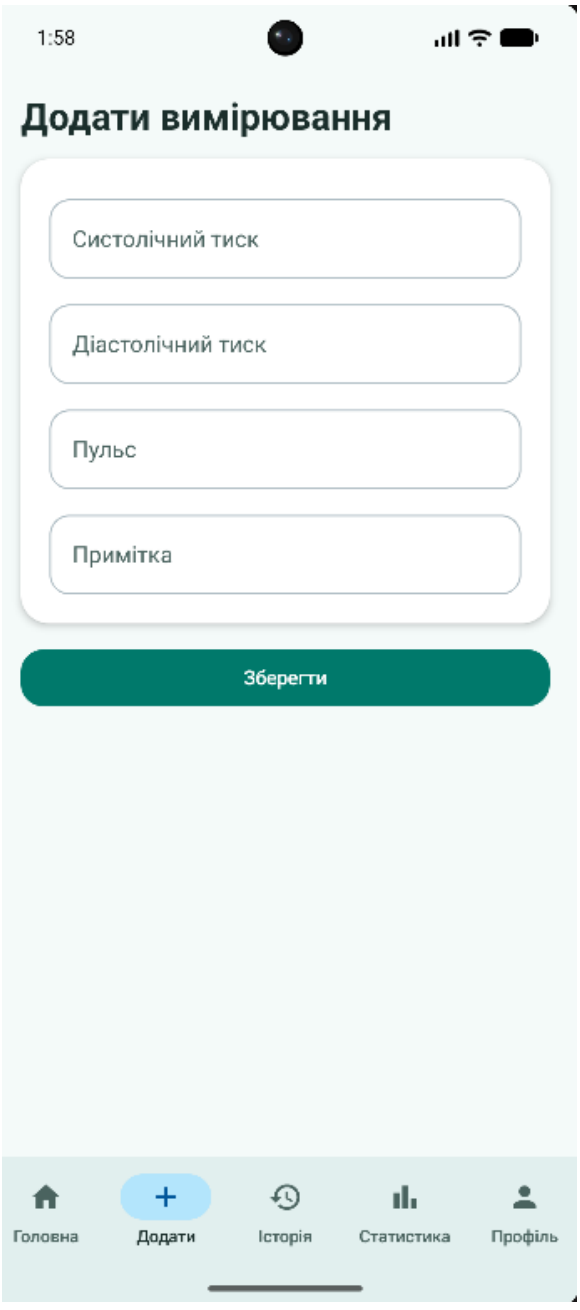

Рисунок 2.8 – Екран додавання вимірювання

Для роботи з накопиченими результатами вимірювань у додатку передбачено окремий екран історії. Він дозволяє переглядати збережені записи, виконувати пошук за вмістом приміток, застосовувати фільтрацію за визначеними часовими інтервалами, а також редагувати або видаляти окремі записи.

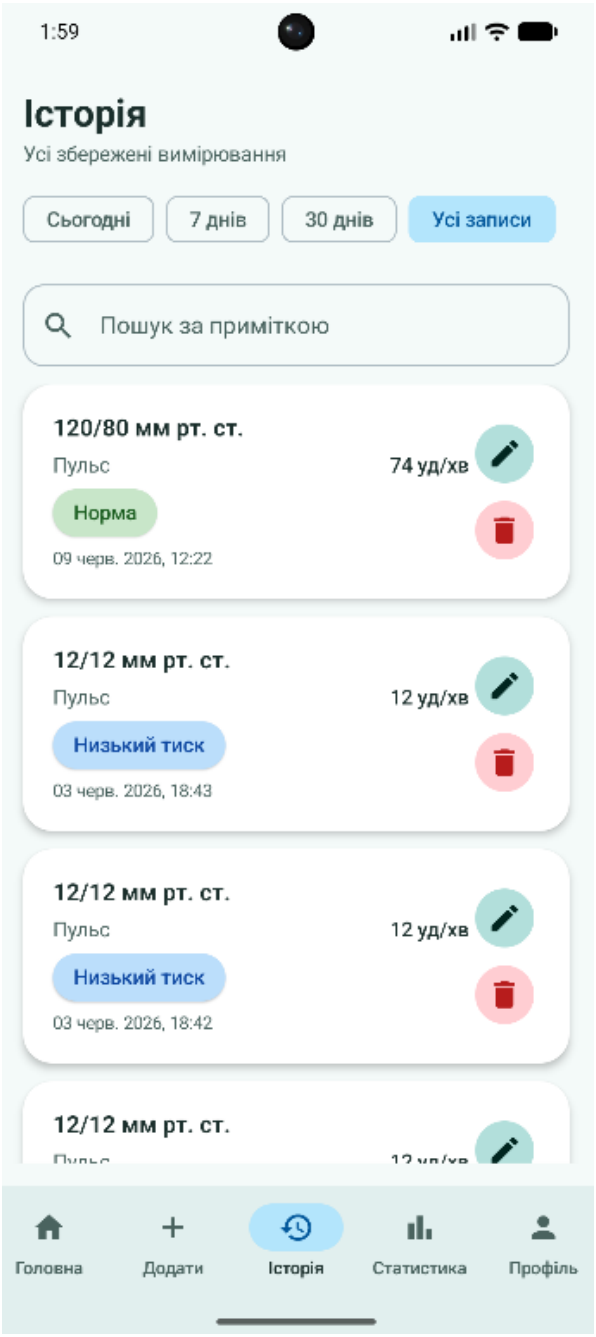

Рисунок 2.9 – Екран історії вимірювань

Для опрацювання та оцінювання накопичених результатів вимірювань у додатку передбачено екран статистики. На цій сторінці користувачу доступна інформація про середні значення артеріального тиску та пульсу, розрахований рівень ризику, результати аналізу змін показників у часі, а також сформований висновок і прогноз щодо стану здоров’я.

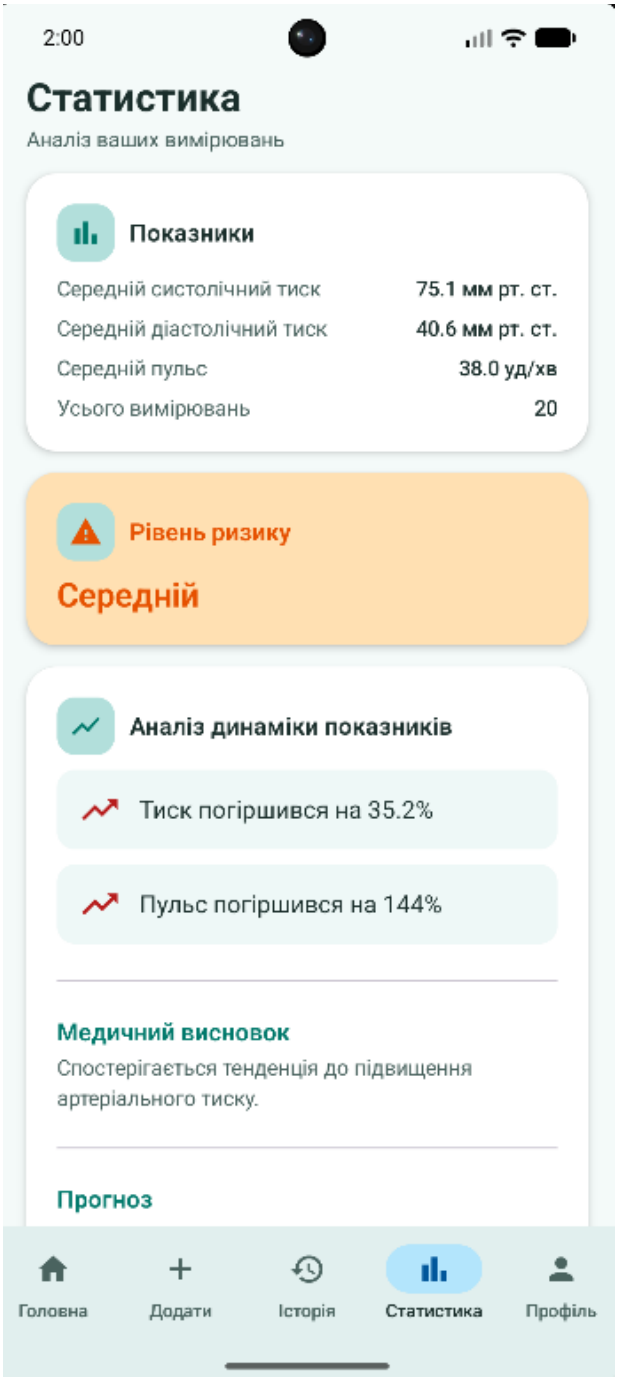

Рисунок 2.10 – Екран статистики

Для підвищення зручності аналізу результатів вимірювань у додатку реалізовано графічне відображення накопичених показників. Використання графіків дає змогу наочно простежувати зміни артеріального тиску та пульсу протягом певного періоду часу, що спрощує оцінювання динаміки стану користувача.

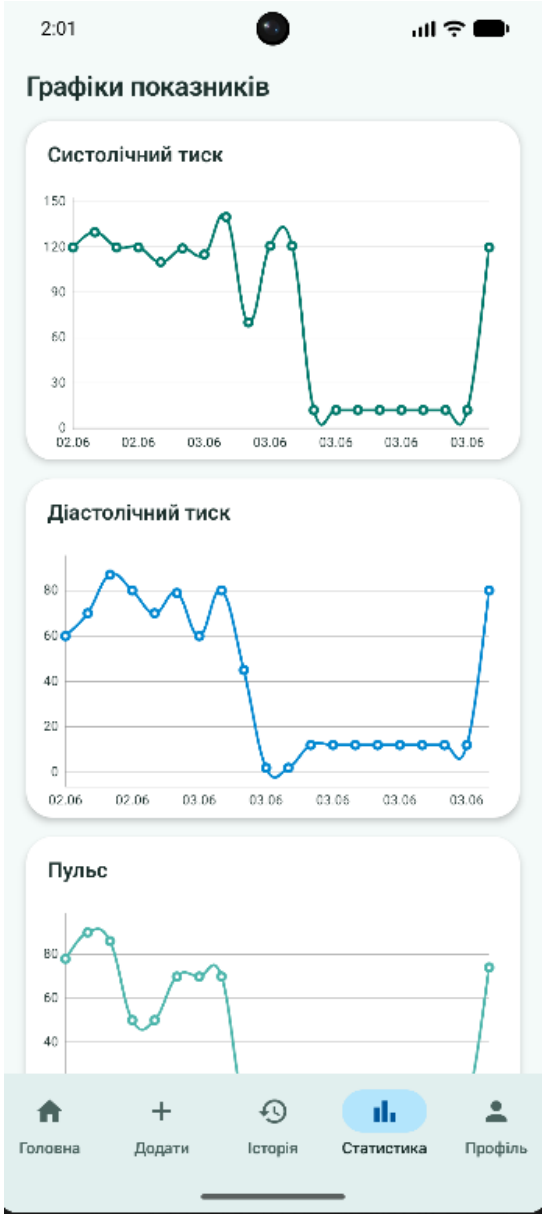

Рисунок 2.11 – Графіки показників стану здоров'я

Для зберігання персональних даних використовується екран профілю користувача. На даній сторінці користувач може редагувати особисті дані, переглядати індекс маси тіла та налаштовувати систему нагадувань.

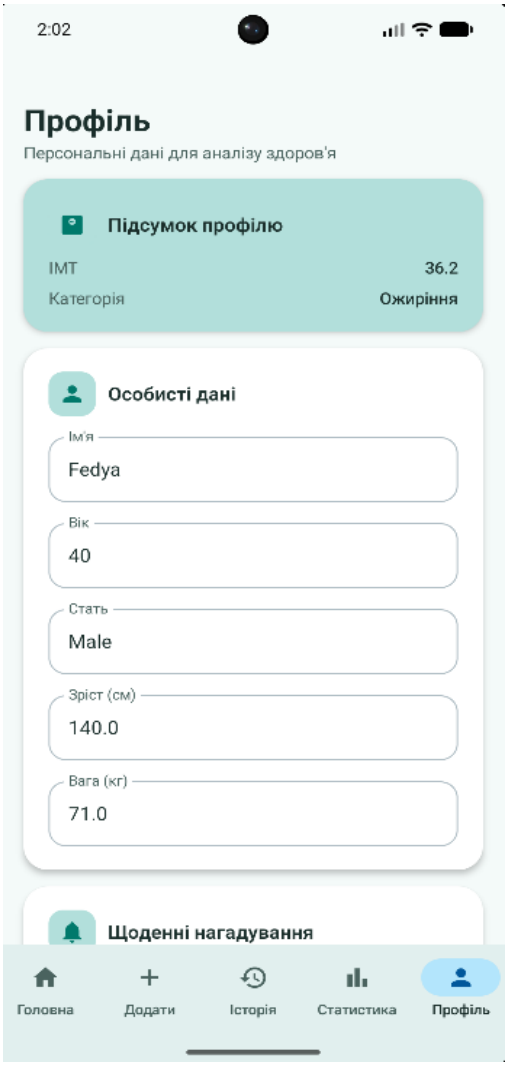

Рисунок 2.12 – Екран профілю користувача

Отримані результати свідчать про те, що розроблений користувацький інтерфейс забезпечує зручну взаємодію з функціональними можливостями мобільного додатку CardioMonitor. Реалізовані рішення сприяють комфортному

використанню програмного продукту та відповідають актуальним вимогам, що висуваються до сучасних мобільних застосунків.

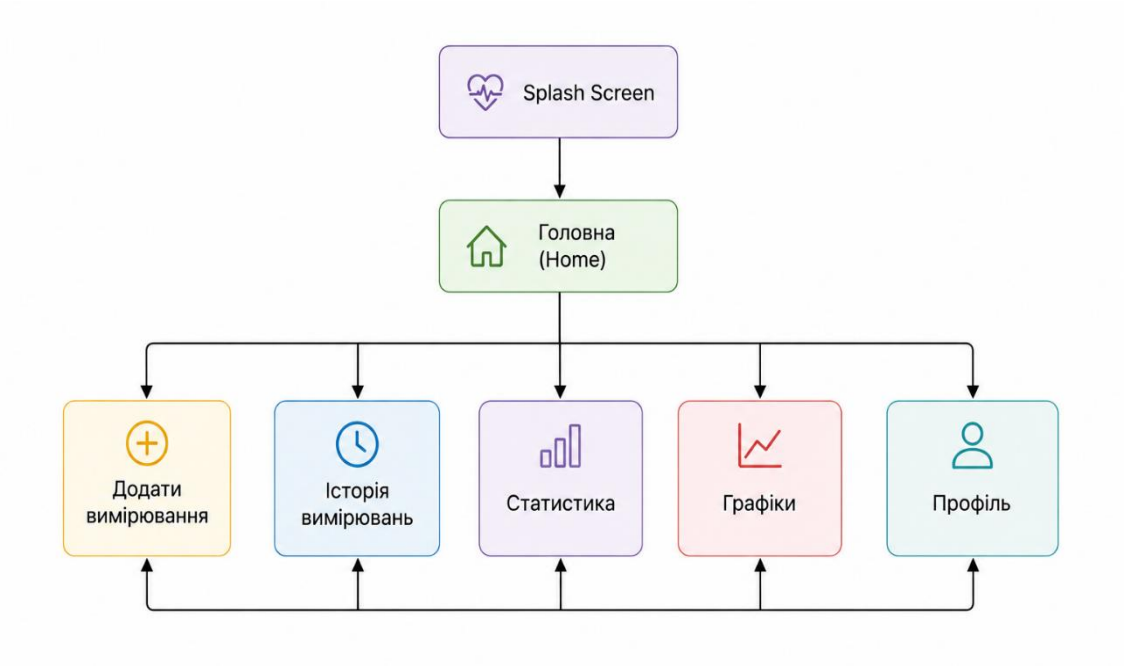

Рисунок 2.13 – Структура навігації між екранами додатку

2.6 UML-моделювання системи

Під час проєктування мобільного Android-додатку CardioMonitor для опису його структури та функціональних можливостей було використано UML-моделювання. Діаграми UML забезпечують графічне представлення взаємодії користувача із системою, взаємозв’язків між програмними компонентами та послідовності виконання основних операцій у додатку [15].

У процесі проєктування програмного забезпечення було побудовано діаграму варіантів використання (Use Case Diagram), діаграму класів (Class Diagram) та діаграму діяльності (Activity Diagram) [15].

2.6.1 Діаграма варіантів використання

Діаграма варіантів використання описує взаємодію користувача з функціональними можливостями системи.

Основним актором системи є користувач мобільного додатку. Він може виконувати введення показників здоров'я, переглядати історію вимірювань, аналізувати статистичні дані, редагувати особистий профіль та налаштовувати систему нагадувань.

На рисунку 2.14 наведено діаграму варіантів використання мобільного додатку CardioMonitor.

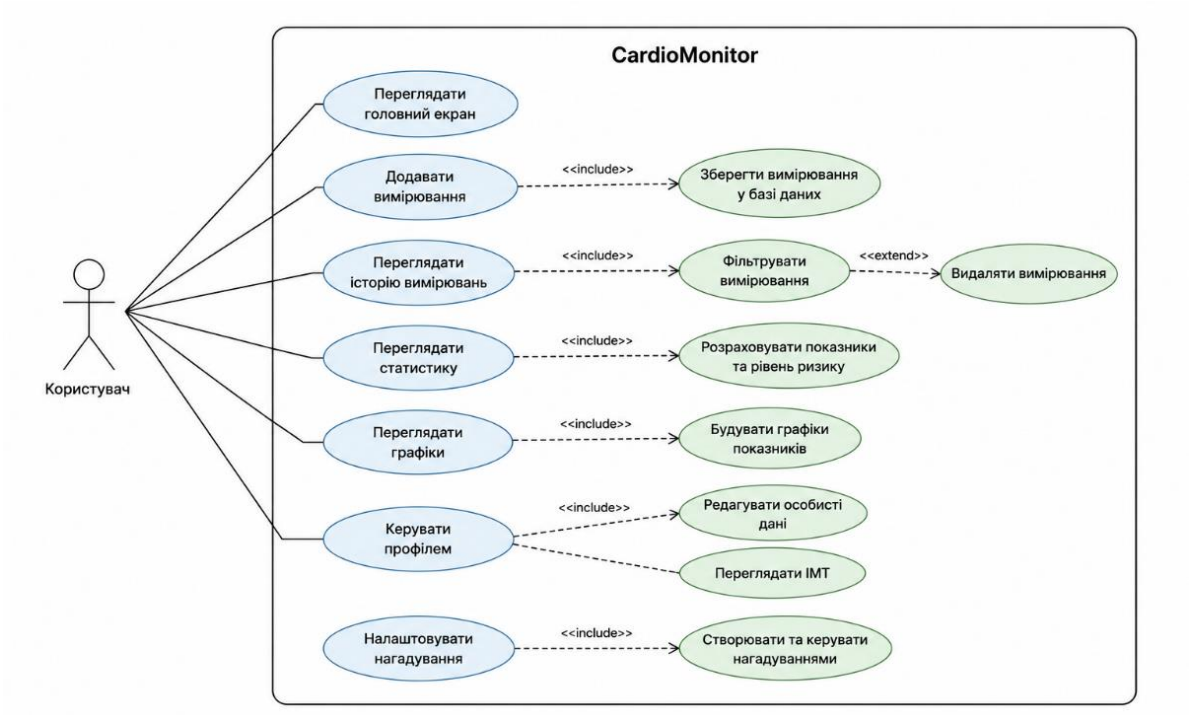

Рисунок 2.14 – UML-діаграма варіантів використання (Use Case Diagram)

2.6.2 Проектування сутностей та класів системи

Діаграма класів відображає структуру програмної системи та взаємозв'язки між її основними компонентами.

Архітектура розробленого додатку включає низку основних класів, серед яких MeasurementEntity, UserProfile, Repository, ViewModel та компоненти користувацького інтерфейсу. Для організації роботи з локальними даними використовуються Room Database і DAO-інтерфейси, що відповідають за збереження інформації та взаємодію із базою даних [15].

На рисунку 2.15 наведено діаграму класів мобільного додатку CardioMonitor.

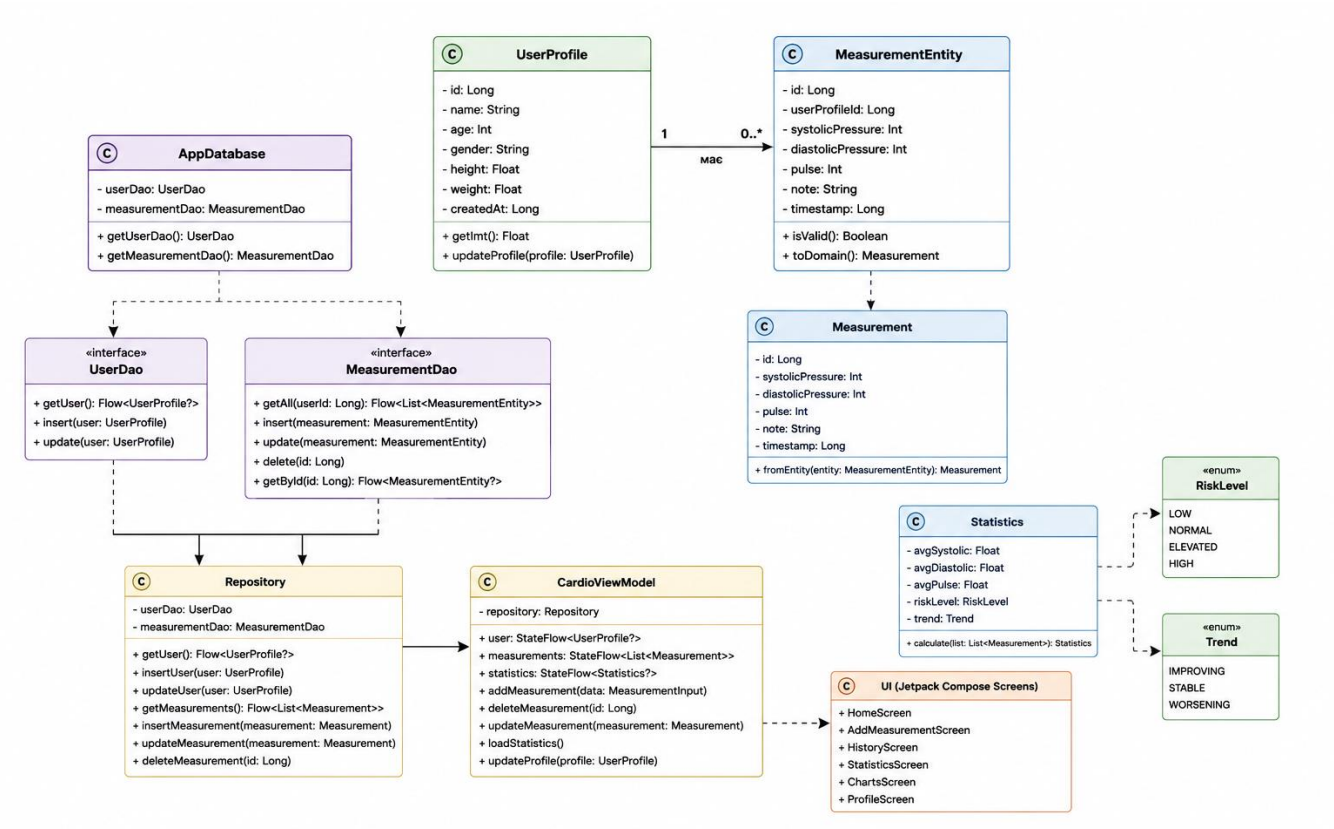

Рисунок 2.15 – UML-діаграма класів (Class Diagram)

2.6.3 Діаграма діяльності

Для відображення процесу додавання нового вимірювання було побудовано діаграму діяльності. Послідовність виконання дій розпочинається з введення користувачем показників артеріального тиску та пульсу. Після перевірки правильності введених значень дані записуються до локальної бази даних. У разі успішного збереження інформації система автоматично оновлює історію вимірювань, статистичні розрахунки та графічне представлення показників..

На рисунку 2.16 наведено діаграму діяльності процесу додавання нового вимірювання.

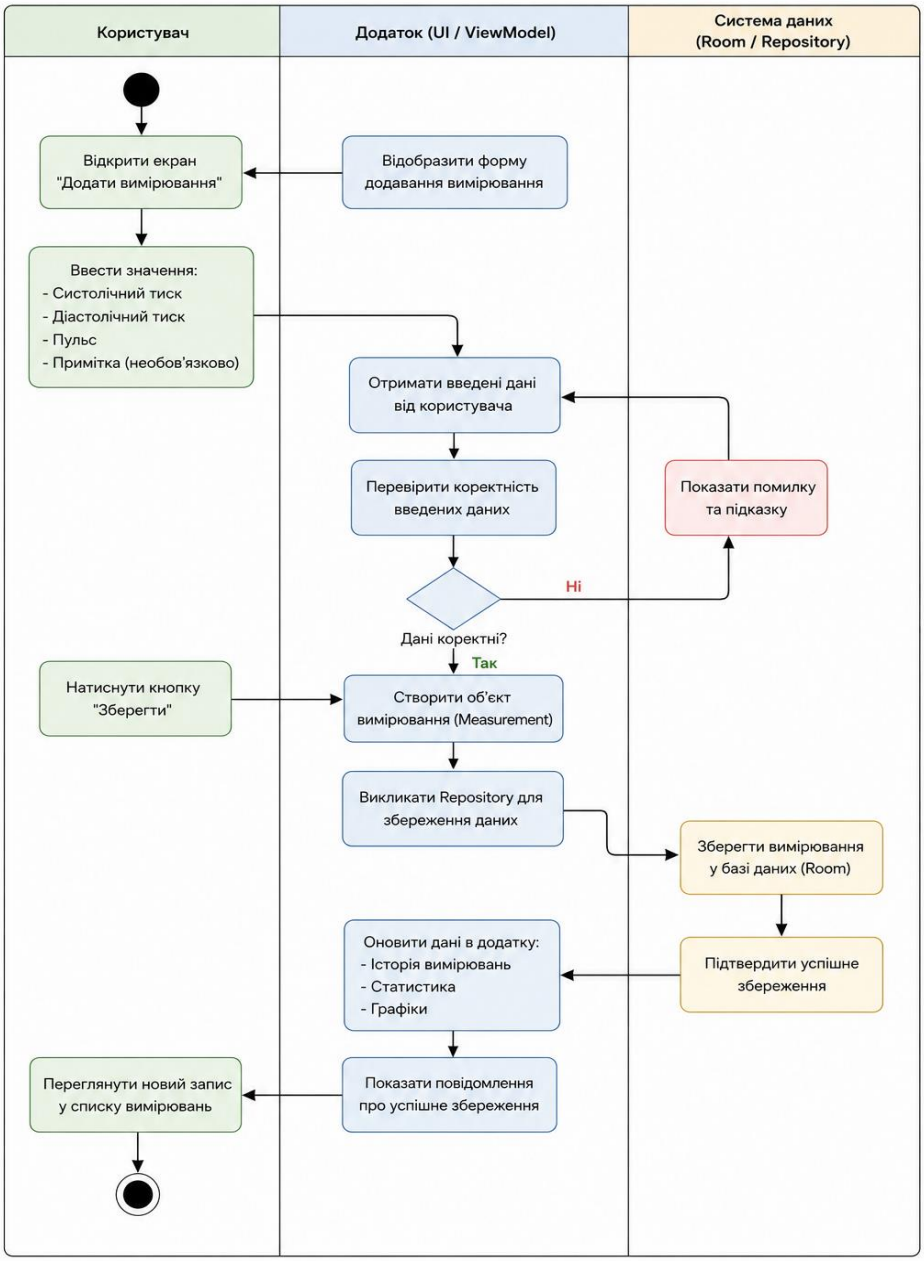

Рисунок 2.16 – UML-діаграма активностей (Activity Diagram)

Таким чином, UML-моделювання дозволило формалізувати структуру програмного забезпечення та наочно представити основні процеси функціонування додатку CardioMonitor.

3 РОЗРОБКА ТА ТЕСТУВАННЯ МОБІЛЬНОГО ДОДАТКУ

У даному розділі розглянуто особливості програмної реалізації мобільного додатку CardioMonitor та його функціональних модулів. Також наведено результати тестування програмного забезпечення та перевірки працездатності реалізованого функціоналу.

3.1 Реалізація структури проєкту

Після завершення робіт, пов’язаних із проєктуванням системи, було здійснено перехід до етапу розробки додатку CardioMonitor. Реалізація програмного продукту виконувалася за допомогою середовища Android Studio [13], мови програмування Kotlin та сучасних інструментів платформи Android Jetpack.

Архітектура програмного забезпечення побудована відповідно до шаблону MVVM (Model–View–ViewModel), що дозволяє розділити інтерфейс користувача, бізнес-логіку та механізми роботи з даними [22].

Структура програмного проєкту складається з декількох логічних модулів:

- модуль користувацького інтерфейсу;
- модуль бізнес-логіки;
- модуль роботи з базою даних;
- модуль статистичного аналізу;
- модуль побудови графіків;
- модуль експорту PDF-звітів;
- модуль системи нагадувань.

Для побудови користувацького інтерфейсу використовувалася технологія Jetpack Compose [7], Кожен екран реалізований у вигляді окремого Compose-компонента, що забезпечує модульність та зручність підтримки програмного коду.

Навігація між екранами реалізована за допомогою Navigation Compose [8]. Даний підхід дозволяє централізовано керувати переходами між сторінками додатку та забезпечує просту інтеграцію нових екранів.

Для роботи з локальною базою даних використовується Room Database [9]. Взаємодія між інтерфейсом користувача та базою даних здійснюється через ViewModel та Repository, що відповідає принципам архітектури MVVM.

Загальна структура програмного проєкту наведена на рисунку 3.1.

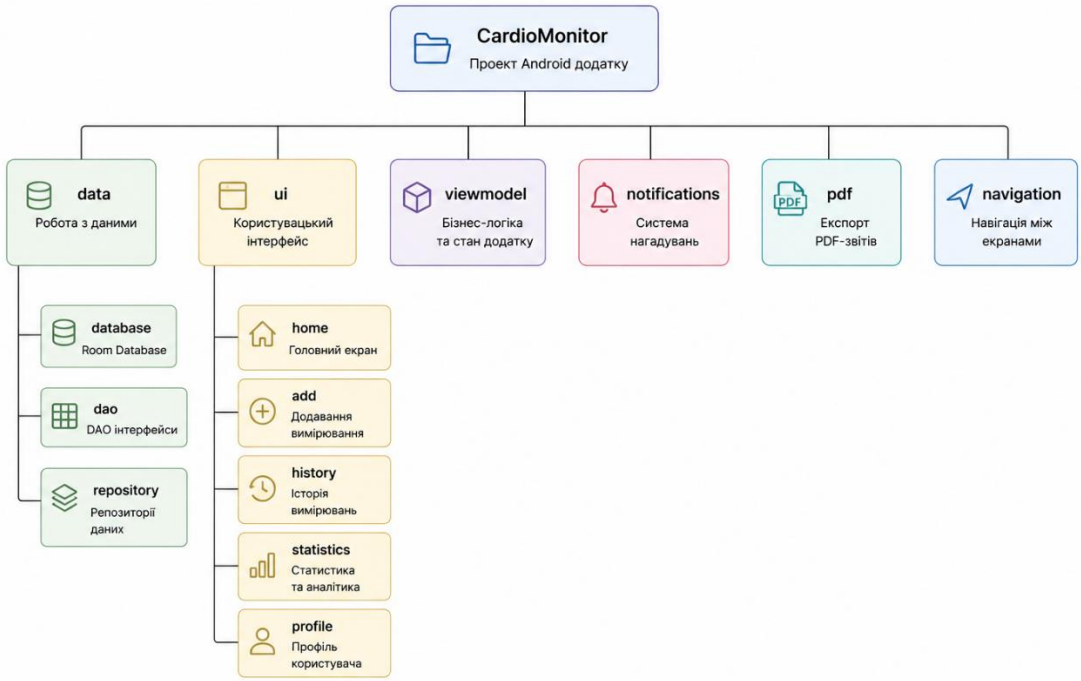

Рисунок 3.1 – Структура програмного проєкту CardioMonitor

Таким чином, реалізована структура програмного забезпечення забезпечує зручність подальшого супроводу, тестування та розширення функціональності мобільного додатку.

3.2 Реалізація функціоналу додавання та збереження вимірювань

Важливою складовою функціоналу мобільного додатку CardioMonitor є можливість реєстрації нових вимірювань. За допомогою цієї функції користувач може вносити показники артеріального тиску та пульсу, які надалі використовуються для накопичення історії спостережень, аналізу динаміки змін і формування статистичних даних.

Для реалізації даного функціоналу було створено окремий екран введення даних, який дозволяє користувачу вносити результати вимірювання артеріального тиску та пульсу. Інтерфейс екрану містить поля для введення систолічного тиску, діастолічного тиску, частоти серцевих скорочень та додаткової примітки.

Після введення необхідних даних користувач натискає кнопку «Зберегти», після чого запускається процес перевірки коректності введеної інформації.

Система виконує перевірку наявності обов’язкових полів та коректності введених значень. Після успішної перевірки формується об’єкт вимірювання, який передається до ViewModel. Подальше збереження даних здійснюється через Repository із використанням Room Database [9]. Для автоматичного оновлення інтерфейсу застосовується StateFlow [11]. Після додавання нового запису оновлюються історія вимірювань, статистичні дані та графіки.

Схему процесу додавання нового вимірювання наведено на рисунку 3.2.

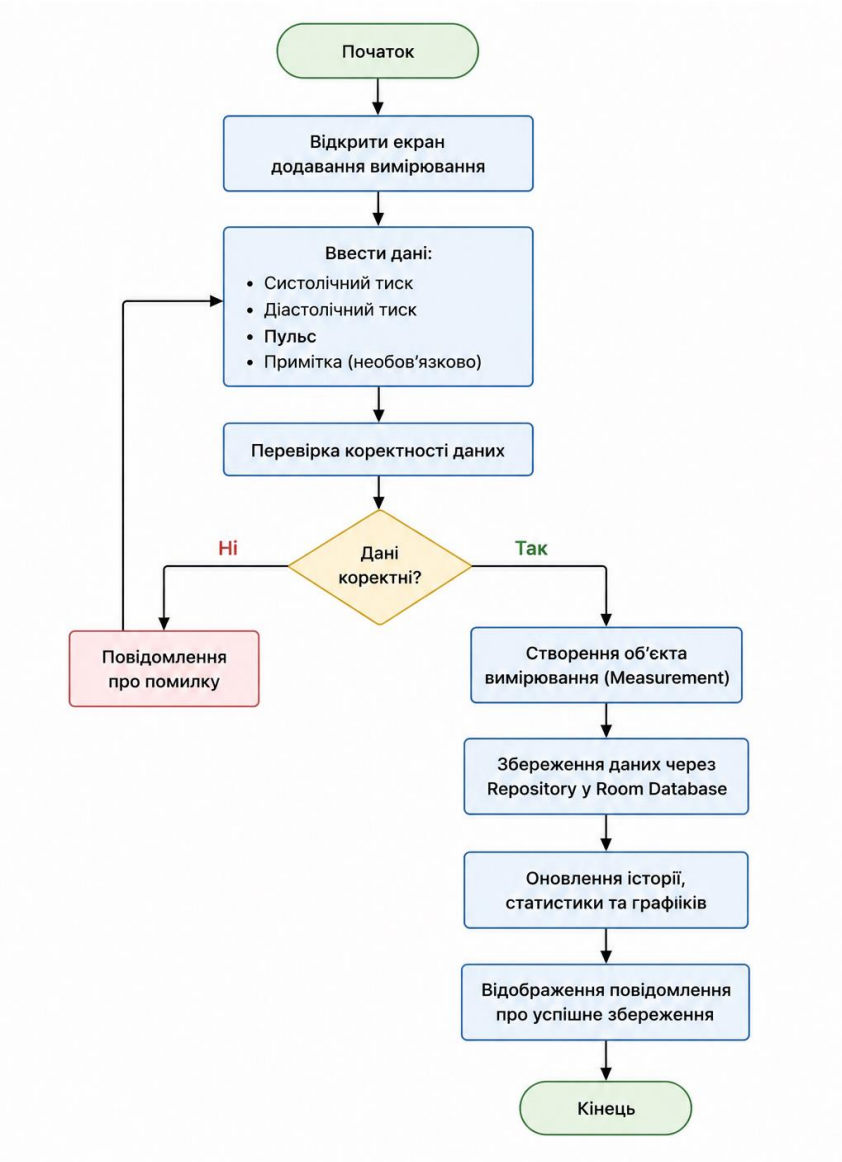

Рисунок 3.2 – Процес додавання та збереження нового вимірювання

Інтерфейс сторінки додавання нового вимірювання наведено на рисунку 3.3.

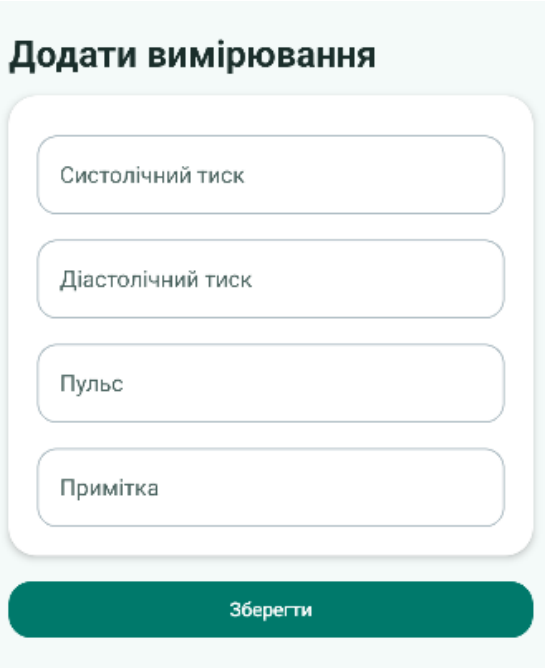

Рисунок 3.3 – Екран додавання нового вимірювання

Завдяки використанню Room Database усі введені користувачем дані зберігаються локально на пристрої та залишаються доступними навіть за відсутності підключення до мережі Інтернет [9].

Для збереження результатів вимірювань використовується DAO-інтерфейс Room Database. Приклад методу додавання нового вимірювання наведено в лістингу 3.1.

Лістинг 3.1 – Фрагмент DAO-інтерфейсу для роботи з вимірюваннями

```
@Insert
suspend fun insertMeasurement(
measurement: MeasurementEntity
)

@Query("SELECT * FROM measurements ORDER BY timestamp DESC")
fun getAllMeasurements(): Flow<List<MeasurementEntity>>
```

Таким чином, реалізований механізм додавання та збереження вимірювань забезпечує швидке введення даних, їх надійне зберігання та подальше використання для аналізу стану здоров'я користувача.

3.3 Реалізація історії вимірювань

Для перегляду збережених показників у мобільному додатку CardioMonitor реалізовано екран історії вимірювань. Він надає можливість переглядати внесені записи, аналізувати попередні результати та керувати наявними даними.

Історія формується на основі інформації, що зберігається в Room Database. Після додавання нового запису дані автоматично відображаються у списку вимірювань без перезапуску додатку. Це забезпечується використанням StateFlow, який підтримує реактивне оновлення користувацького інтерфейсу [11].

На екрані історії для кожного вимірювання відображаються значення артеріального тиску, пульсу, дата та час створення запису, а також коротка оцінка стану користувача. Це дозволяє швидко переглядати попередні результати та виявляти можливі зміни показників.

Для підвищення зручності роботи з великою кількістю записів у додатку реалізовано фільтрацію за періодами. Користувач може переглядати записи за сьогодні, останні 7 днів, 30 днів або всі збережені вимірювання. Також реалізовано пошук за приміткою, що дозволяє швидко знаходити потрібні записи.

Окрім перегляду, користувач має можливість редагувати або видаляти окремі вимірювання. Це забезпечує гнучке керування збереженими даними та дозволяє виправляти помилково введену інформацію.

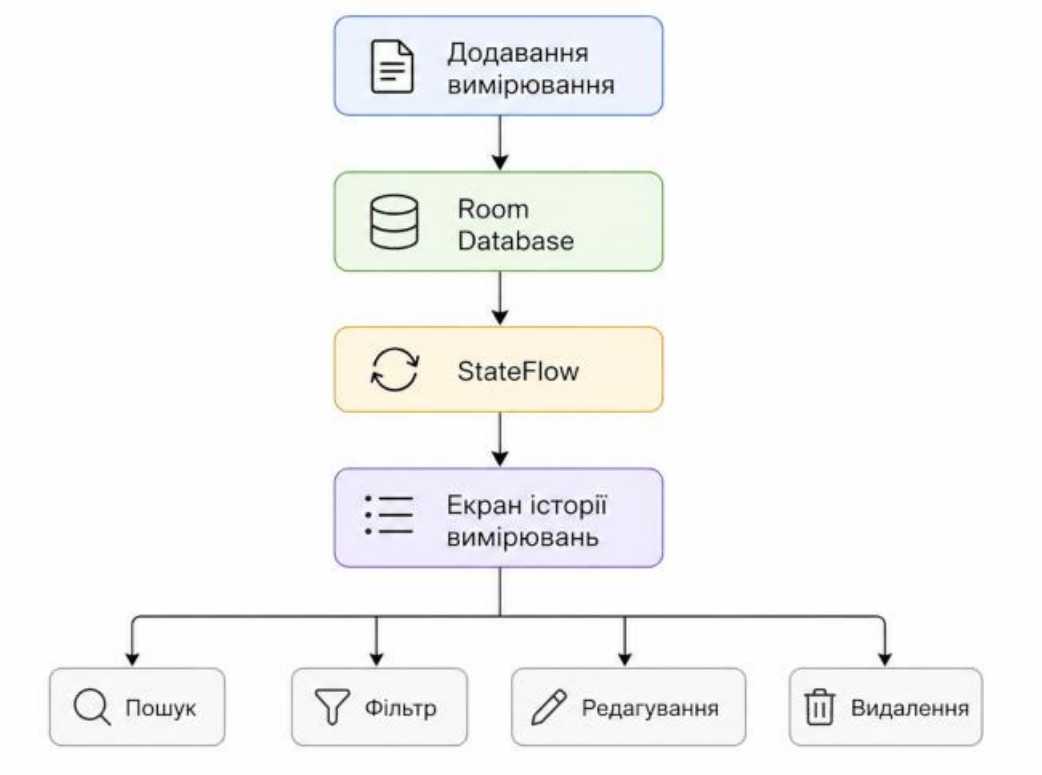

Рисунок 3.4 – Схема роботи модуля історії вимірювань

Схему роботи модуля історії вимірювань наведено на рисунку 3.4. Вона демонструє процес надходження даних до локальної бази Room Database, їх автоматичне оновлення за допомогою StateFlow та подальшу взаємодію користувача із записами через механізми пошуку, фільтрації, редагування та видалення.

Таким чином, реалізований модуль історії вимірювань забезпечує зручний доступ до накопичених даних, підтримує їх сортування та пошук, а також надає можливість керування збереженими записами, що підвищує практичну цінність мобільного додатку.

3.4 Реалізація статистики та графічного відображення показників

Для аналізу збережених результатів вимірювань у додатку реалізовано окремий екран статистики. Він дозволяє отримувати узагальнену інформацію про показники серцево-судинної системи та відстежувати їх зміни на основі накопичених даних.

Статистичний модуль автоматично обробляє всі записи, що зберігаються у базі даних, та виконує розрахунок основних показників. До них належать середнє значення систолічного тиску, середнє значення діастолічного тиску, середній пульс та загальна кількість виконаних вимірювань. Для розрахунку середніх значень показників використовуються стандартні засоби Kotlin. Приклад обчислення середнього значення пульсу наведено в лістингу 3.2.

Лістинг 3.2 – Розрахунок середнього значення пульсу

```
val averagePulse =
measurements
.map { it.pulse }
.average()
```

Для підвищення інформативності системи реалізовано механізм автоматичного визначення рівня ризику. На основі середніх значень артеріального тиску та пульсу додаток формує попередню оцінку стану користувача та відображає її у зрозумілому вигляді.

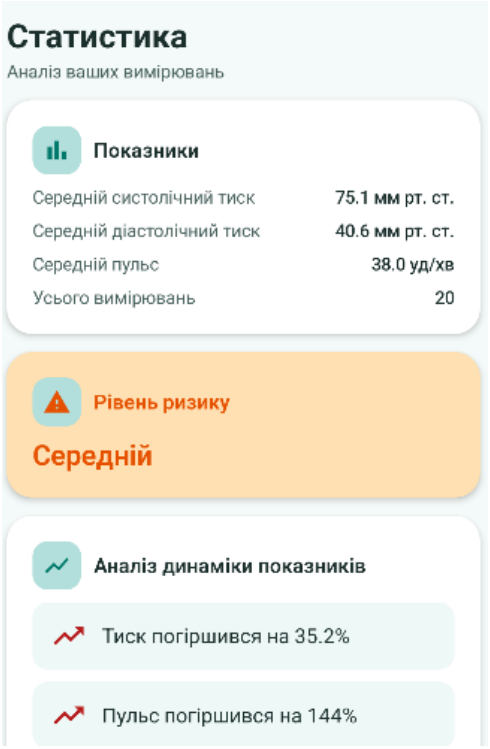

Рисунок 3.5 – Екран статистики мобільного додатку CardioMonitor

Для наочного відображення змін показників використовується бібліотека MPAndroidChart [14]. Дана бібліотека дозволяє будувати лінійні графіки зміни артеріального тиску та частоти серцевих скорочень на основі накопичених даних користувача.

Побудова графіків виконується автоматично після отримання інформації з локальної бази даних. Кожна точка графіка відповідає окремому вимірюванню, що дозволяє користувачу відстежувати зміни показників у часі.

Приклад графічного відображення результатів вимірювань наведено на рисунку 3.6.

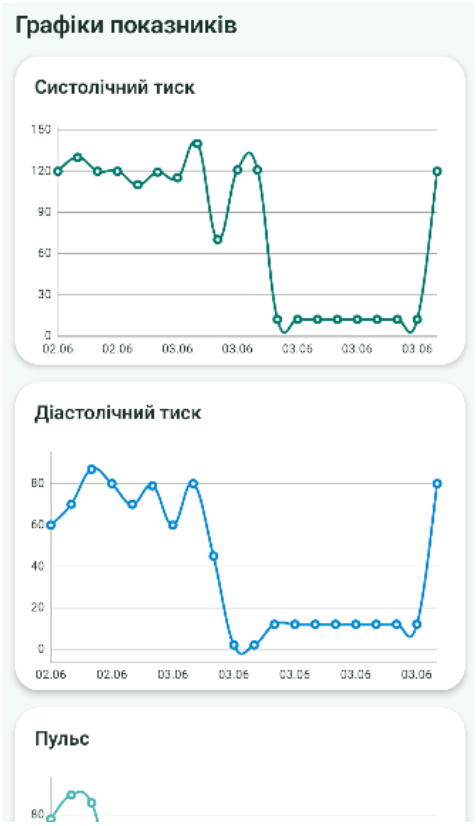

Рисунок 3.6 – Графіки зміни показників артеріального тиску та пульсу

Однією з важливих можливостей додатку є аналіз накопичених результатів вимірювань та їх візуальне представлення. Для реалізації цього функціоналу було розроблено окремий екран статистики, який дозволяє користувачеві отримувати узагальнену інформацію про стан серцево-судинної системи на основі збережених даних.

На основі отриманих даних також генерується медичний висновок та короткостроковий прогноз. Дані повідомлення не є медичним діагнозом, однак можуть допомогти користувачу своєчасно звернути увагу на можливі зміни стану здоров’я.

Таким чином, реалізований модуль статистики та графічного аналізу забезпечує користувачу зручний інструмент для моніторингу динаміки показників серцево-судинної системи та підвищує інформативність мобільного додатку.

3.5 Реалізація профілю користувача та системи нагадувань

З метою персоналізації функціональних можливостей мобільного додатку CardioMonitor розроблено окремий екран профілю користувача. Цей модуль забезпечує збереження персональної інформації, яка використовується для оцінювання стану здоров’я та формування індивідуальних рекомендацій на основі введених даних..

Екран профілю надає користувачеві можливість вносити та змінювати основні особисті відомості, серед яких ім’я, вік, стать, зріст і вага. Уведені дані зберігаються у локальному сховищі пристрою та використовуються додатком під час роботи модуля статистики й аналізу показників.

Інтерфейс сторінки профілю наведено на рисунку 3.7.

Рисунок 3.7 – Екран профілю користувача

На основі значень зросту та ваги система автоматично виконує розрахунок індексу маси тіла (ІМТ). Отриманий результат використовується для визначення відповідної категорії маси тіла та формування рекомендацій щодо загального стану користувача.

Для розрахунку ІМТ використовується наступна формула [3]:

Формула 3.1

$$IMT=m/h^2$$

де:

- ІМТ — індекс маси тіла;
- m — маса тіла користувача, кг;
- h — зріст користувача, м.

Отримане значення автоматично аналізується та відноситься до однієї з категорій: недостатня маса тіла, нормальна маса тіла, надмірна маса тіла або ожиріння. Для визначення індексу маси тіла використовується формула, реалізація якої наведена в лістингу 3.3.

Лістинг 3.3 – Метод обчислення індексу маси тіла

```
fun calculateBmi(
weight: Float,
height: Float
): Float {
return weight / (height * height)
}
```

Для забезпечення регулярного моніторингу показників здоров’я у додатку передбачено систему щоденних нагадувань. Користувач має можливість самостійно керувати її роботою, активуючи або вимикаючи нагадування, а також встановлюючи зручний час для їх відображення.

Для реалізації механізму нагадувань використовується Android Notification System [5]. У визначений користувачем час система автоматично формує та відображає сповіщення з нагадуванням про необхідність проведення вимірювання артеріального тиску та пульсу.

Процес роботи системи нагадувань наведено на рисунку 3.8.

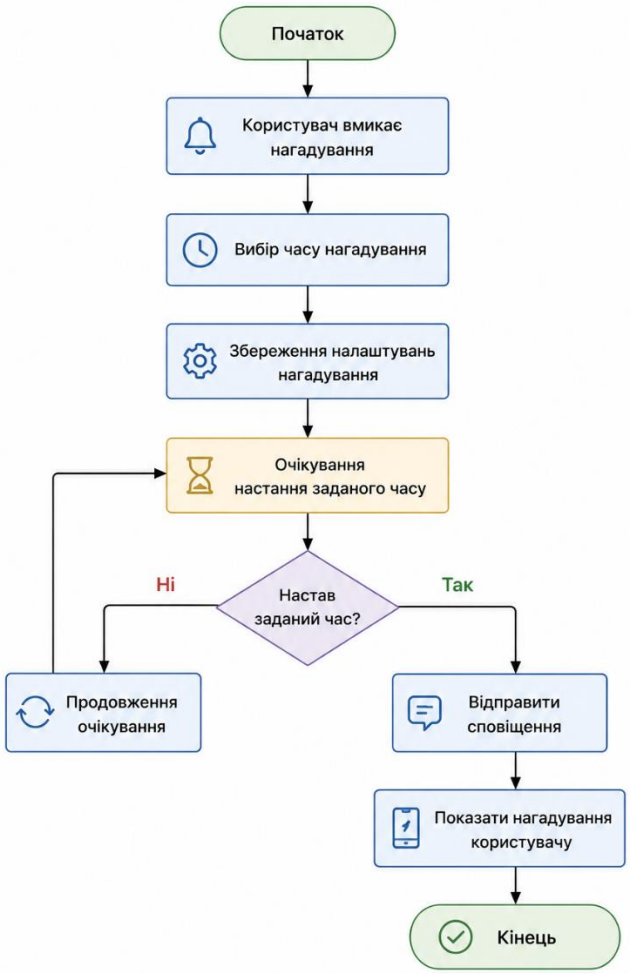

Рисунок 3.8 – Алгоритм роботи системи нагадувань

Завдяки використанню системи сповіщень користувач має можливість регулярно контролювати показники серцево-судинної системи та своєчасно фіксувати зміни стану здоров’я.

На основі наведеного, реалізований модуль профілю користувача та система

нагадувань забезпечують персоналізацію додатку та підвищують ефективність його використання у повсякденному житті.

3.6 Реалізація експорту PDF-звітів

Для забезпечення можливості збереження та подальшого використання результатів вимірювань реалізовано функцію експорту статистичних даних у формат PDF.

Дана функція надає користувачеві можливість формувати звіт, який містить основну інформацію щодо показників стану серцево-судинної системи. Створений документ може використовуватися для особистого зберігання результатів вимірювань або надання відповідної інформації медичним працівникам під час консультацій.

Формування документа реалізовано за допомогою Android PDFDocument API, який входить до складу стандартних засобів розробки Android. Використання даного інструменту дозволяє створювати PDF-файли без необхідності використання сторонніх бібліотек [5].

Під час створення звіту до документа можуть включатися:

- персональні дані користувача;
- середні значення артеріального тиску;
- середній пульс;
- кількість виконаних вимірювань;
- оцінка рівня ризику;
- дата формування документа.

Після завершення процесу формування документа файл зберігається у локальній пам’яті пристрою та може бути відкритий користувачем або переданий іншим особам за допомогою стандартних засобів операційної системи Android.

Загальний алгоритм формування PDF-звіту наведено на рисунку 3.9

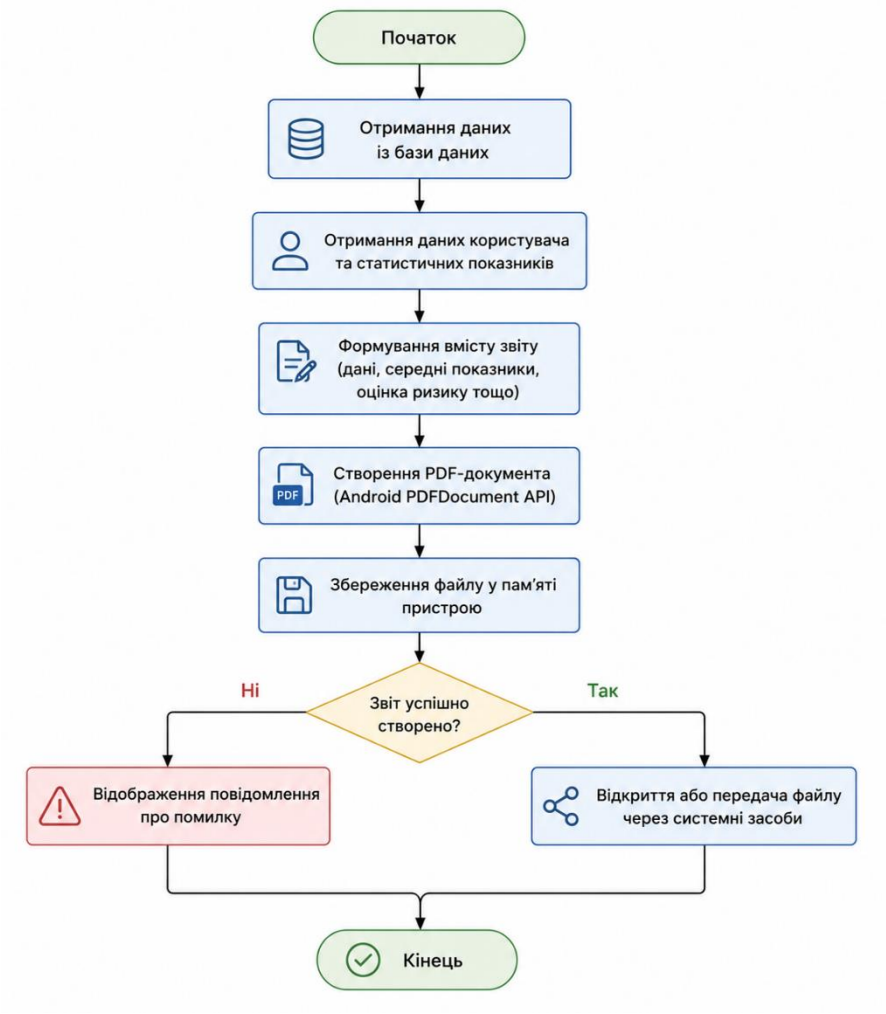

Рисунок 3.9 – Алгоритм формування PDF-звіту

Функція експорту підвищує практичну цінність програмного продукту та дозволяє використовувати накопичені дані не лише в межах мобільного додатку, а й поза ним.

Таким чином, реалізація експорту PDF-звітів забезпечує зручне збереження та подальше використання статистичної інформації про стан серцево-судинної системи користувача.

3.7 Реалізація системи аналізу стану користувача

Однією з важливих функціональних можливостей додатку CardioMonitor є система автоматизованого аналізу показників стану серцево-судинної системи користувача.

На відміну від більшості додатків, які виконують лише збереження результатів вимірювань, розроблена система забезпечує не тільки накопичення даних, а й їх аналіз із подальшим формуванням висновків та рекомендацій для користувача.

Робота аналітичного модуля базується на аналізі значень артеріального тиску, частоти серцевих скорочень та індексу маси тіла користувача. Використовуючи отримані показники, система визначає поточний стан здоров’я та відображає результати оцінювання у зручному й зрозумілому для користувача вигляді.

Для оцінювання стану артеріального тиску використовуються встановлені порогові значення. Залежно від отриманих результатів система може визначати такі категорії:

- низький тиск;
- понижений тиск;
- нормальний тиск;
- підвищений тиск;
- високий тиск.

Результат аналізу автоматично відображається на головному екрані додатку та використовується під час формування загальної оцінки стану користувача.

Крім оцінки окремих вимірювань, система виконує аналіз динаміки показників. Для цього порівнюються поточні результати з попередніми вимірюваннями та обчислюються тенденції зміни стану здоров’я.

На основі проведеного аналізу формується блок «Аналіз динаміки показників», який може містити повідомлення про:

- покращення стану;
- стабільність показників;
- погіршення стану.

Приклад результатів аналізу наведено на рисунку 3.10.

Рисунок 3.10 – Відображення результатів аналізу стану користувача

Додатково система формує медичний висновок та короткостроковий прогноз. Для цього використовуються статистичні показники, отримані на основі накопичених вимірювань користувача.

Згенеровані системою повідомлення не є медичним висновком і використовуються виключно з інформаційною метою. Вони покликані звернути увагу користувача на можливі зміни показників здоров’я та важливість регулярного контролю стану серцево-судинної системи. Прогнозування базується

на аналізі останніх результатів вимірювань і виявленні загальної тенденції зміни показників.

Для підвищення точності загальної оцінки також враховується індекс маси тіла користувача. У разі виявлення надмірної маси тіла або ожиріння система формує додаткові рекомендації щодо контролю ваги та способу життя.

Загальний алгоритм роботи аналітичного модуля наведено на рисунку 3.11.

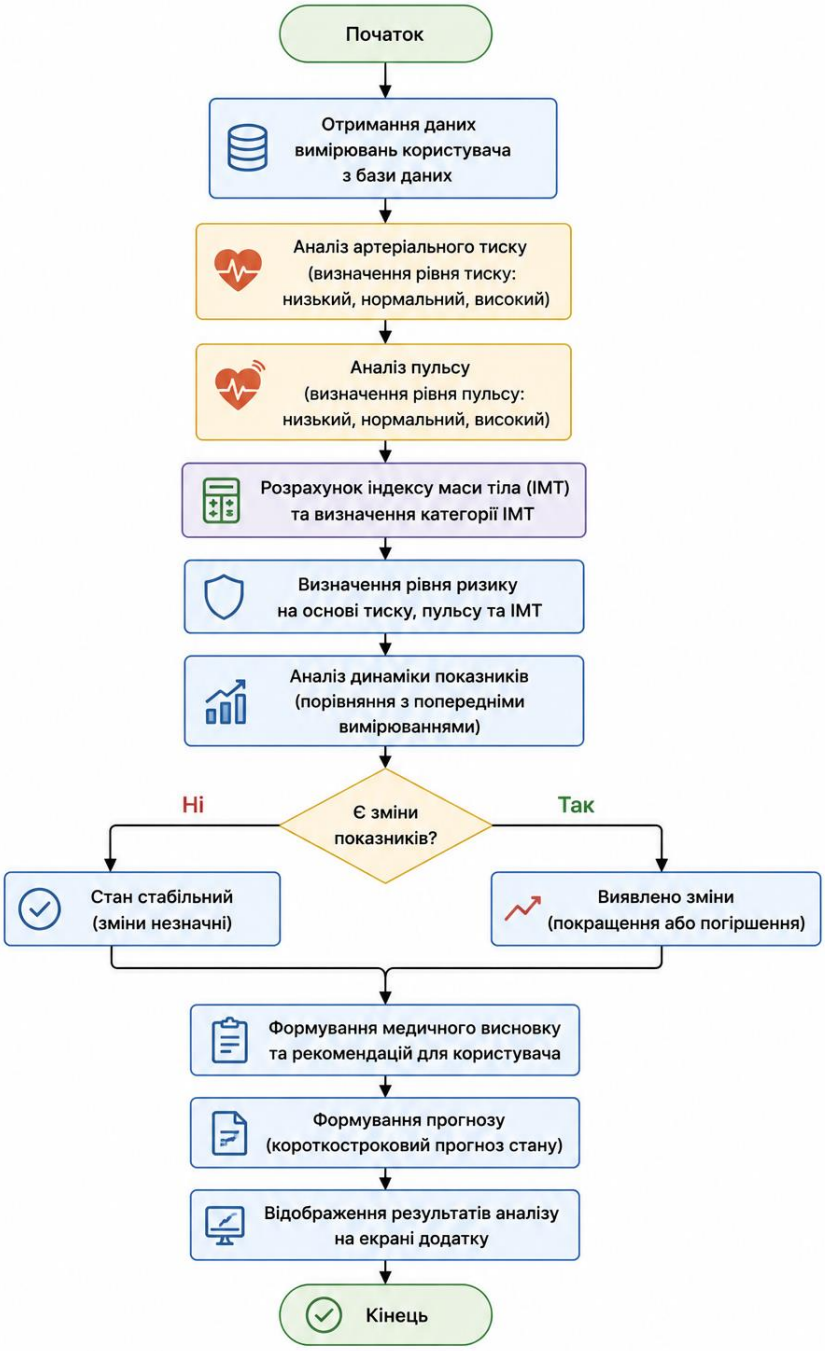

Рисунок 3.11 – Алгоритм аналізу показників користувача

Отже, реалізована система аналізу стану користувача дозволяє автоматизувати оцінювання показників серцево-судинної системи та підвищує практичну цінність додатку.

3.8 Тестування мобільного додатку

Після завершення етапу розробки мобільного додатку проведено перевірку його основних функціональних можливостей. Тестування виконувалося з метою оцінювання коректності роботи програмного забезпечення, виявлення можливих помилок та підтвердження відповідності реалізованих функцій визначеним вимогам.

У процесі тестування перевірялися такі основні компоненти системи:

- додавання нового вимірювання;
- збереження даних у базі даних;
- перегляд історії вимірювань;
- пошук та фільтрація записів;
- редагування вимірювань;
- видалення записів;
- формування статистики;
- побудова графіків;
- розрахунок ІМТ;
- система нагадувань;
- експорт PDF-звітів;
- система аналізу стану користувача.

Тестування проводилося методом функціонального тестування шляхом перевірки роботи кожного програмного модуля окремо.

Результати тестування наведено у таблиці 3.1.

Таблиця 3.1 – Результати функціонального тестування

№	Тестовий сценарій	Очікуваний результат	Результат
1	Додавання вимірювання	Запис успішно збережено	Успішно
2	Перегляд історії	Відображення всіх записів	Успішно
3	Пошук за приміткою	Відображення відповідних записів	Успішно
4	Фільтрація за періодом	Коректне відображення записів	Успішно
5	Редагування запису	Дані оновлено	Успішно
6	Видалення запису	Запис видалено	Успішно
7	Формування статистики	Відображення розрахованих показників	Успішно
8	Побудова графіків	Графіки сформовано	Успішно
9	Розрахунок ІМТ	Коректний результат	Успішно
10	Робота нагадувань	Відображення системного повідомлення	Успішно
11	Формування PDF-звіту	PDF-файл створено	Успішно
12	Аналіз стану користувача	Відображення висновків та прогнозу	Успішно

Під час тестування було встановлено, що всі основні функції додатку працюють відповідно до поставлених вимог. Помилок, які унеможливлюють використання програмного продукту, виявлено не було.

Для додаткової перевірки коректності роботи системи виконувалося тестування на різних наборах вхідних даних. Особлива увага приділялася перевірці граничних значень артеріального тиску та пульсу, а також роботі алгоритмів аналізу стану користувача.

Загальна схема процесу тестування наведена на рисунку 3.12.

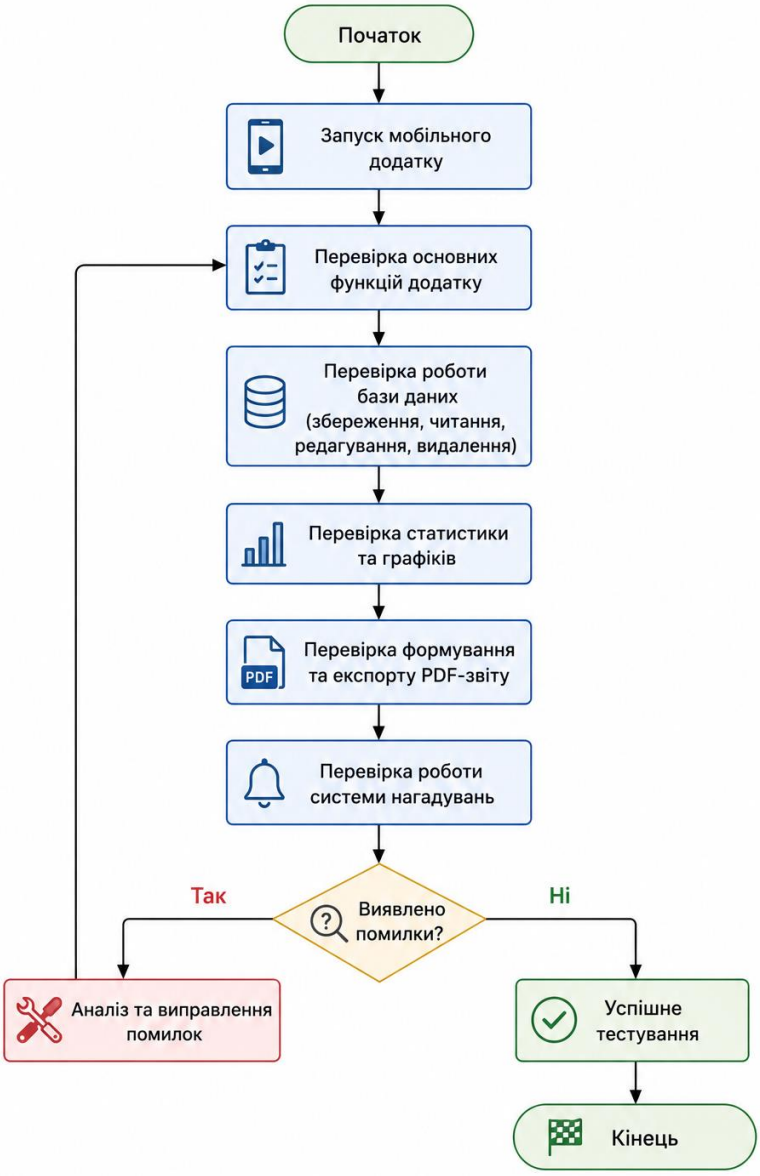

Рисунок 3.12 – Загальна схема тестування мобільного додатку

Таким чином, проведене тестування підтвердило працездатність мобільного додатку та відповідність його функціональних можливостей поставленим вимогам.

РОЗДІЛ 4 БЕЗПЕКА ЖИТТЄДІЯЛЬНОСТІ ОСНОВИ ОХОРОНИ ПРАЦІ

У даному розділі розглянуто питання безпеки життєдіяльності та охорони праці під час роботи з комп'ютерною технікою. Проведено аналіз ризиків виробничого середовища та визначено заходи щодо забезпечення безпечних і комфортних умов праці користувача персонального комп'ютера.

4.1 Ризик як кількісна оцінка небезпек

Безпека життєдіяльності є важливою складовою діяльності людини в сучасному суспільстві. У процесі навчання, праці та повсякденного життя людина постійно стикається з різноманітними небезпечними факторами, які можуть негативно впливати на її здоров'я, працездатність та безпеку. Для оцінювання можливих негативних наслідків використовується поняття ризику.

Ризик є кількісною характеристикою небезпеки, яка дозволяє оцінити ймовірність виникнення небажаної події та можливі наслідки її реалізації. Використання ризик-орієнтованого підходу дає змогу визначити найбільш небезпечні фактори, оцінити рівень їх впливу та розробити заходи щодо зменшення або усунення негативних наслідків [4]. Оцінювання ризиків є важливим елементом системи управління безпекою праці та сприяє прийняттю обґрунтованих рішень щодо запобігання небезпечним ситуаціям. Результати оцінювання ризику використовуються для підвищення рівня безпеки праці.

Під небезпекою розуміють будь-яке явище, процес або фактор, який за певних умов може завдати шкоди людині, матеріальним цінностям чи навколишньому середовищу. Небезпеки можуть бути природного, техногенного, соціального або біологічного походження. До природних небезпек належать стихійні лиха, несприятливі погодні умови та природні катастрофи. Техногенні небезпеки пов'язані з діяльністю людини та функціонуванням технічних систем. Соціальні небезпеки виникають у результаті суспільних процесів, а біологічні

пов’язані з впливом мікроорганізмів, вірусів та інших біологічних факторів.

Кількісна оцінка ризику може бути визначена за формулою [4]:

Формула 4.1

$$R=P\cdot S$$

де:

R – рівень ризику;

P – ймовірність виникнення небезпечної події;

S – тяжкість можливих наслідків.

З наведеної формули видно, що ризик залежить не лише від частоти виникнення певної події, але й від серйозності її наслідків. Навіть малоймовірна подія може становити значну небезпеку, якщо її наслідки є критичними для здоров’я людини або функціонування системи [4].

У сучасних умовах широкого використання інформаційних технологій особливої актуальності набуває оцінювання ризиків, пов’язаних із роботою за персональним комп’ютером. Значна частина професійної діяльності програмістів, інженерів програмного забезпечення та інших фахівців IT-сфери пов’язана з тривалим перебуванням за комп’ютером. Це може призводити до виникнення професійних ризиків, серед яких погіршення зору, порушення постави, захворювання опорно-рухового апарату, психоемоційне перевантаження та швидка втомлюваність [20].

Під час розробки мобільного Android-додатку для моніторингу показників стану серцево-судинної системи людини значна частина робіт виконувалася із використанням персонального комп’ютера та спеціалізованого програмного забезпечення. У процесі розробки застосовувалося середовище Android Studio, здійснювалися проєктування інтерфейсу користувача, програмування

функціональних модулів, тестування та налагодження програмного коду. У зв’язку з цим виникають ризики, характерні для операторів комп’ютерної техніки.

Для зниження рівня ризику необхідно забезпечувати належну організацію робочого місця [17], дотримуватися встановленого режиму праці та відпочинку, використовувати ергономічні меблі та підтримувати оптимальні параметри мікроклімату в приміщенні. Важливим заходом також є регулярне проведення профілактичних перерв під час роботи за комп’ютером, що сприяє зменшенню навантаження на зорову систему та опорно-руховий апарат [18].

Ефективне управління ризиками передбачає своєчасне виявлення небезпечних факторів, оцінювання можливих наслідків їх впливу та впровадження профілактичних заходів. Такий підхід дозволяє забезпечити безпечні умови праці, зберегти здоров’я працівників та підвищити ефективність виконання професійних обов’язків.

Отже, ризик є важливим показником рівня безпеки діяльності людини та дозволяє кількісно оцінити можливість виникнення небезпечних ситуацій. Використання методів оцінювання ризиків сприяє підвищенню рівня безпеки життєдіяльності та створенню безпечних умов праці під час виконання професійної діяльності.

4.2 Вимоги безпеки до робочих місць для виконання робіт з використанням персонального комп’ютера

Робоче місце користувача персонального комп’ютера повинно відповідати вимогам охорони праці та ергономіки, оскільки значна частина професійної діяльності сучасних фахівців виконується із використанням комп’ютерної техніки. Особливо це стосується програмістів та інженерів програмного забезпечення, які протягом робочого дня тривалий час працюють із програмним кодом, документацією та спеціалізованими програмними засобами [16,17].

Під час розробки мобільного додатку використовувався персональний комп’ютер із середовищем Android Studio. Оскільки виконувалися програмування, тестування та підготовка документації, необхідно забезпечувати безпечні та комфортні умови праці.

Робоче місце користувача повинно забезпечувати правильне положення тіла під час роботи. Стіл має бути достатньо просторим для розміщення необхідного обладнання та забезпечувати комфортне розташування користувача.

Важливим елементом робочого місця є правильно підібране крісло [18]. Воно повинно забезпечувати регулювання висоти та підтримку природного положення хребта. Використання ергономічного крісла сприяє зменшенню навантаження на опорно-руховий апарат і підвищує комфорт під час тривалої роботи. Приклад правильної організації робочого місця наведено на рисунку 4.1 [17, 18].

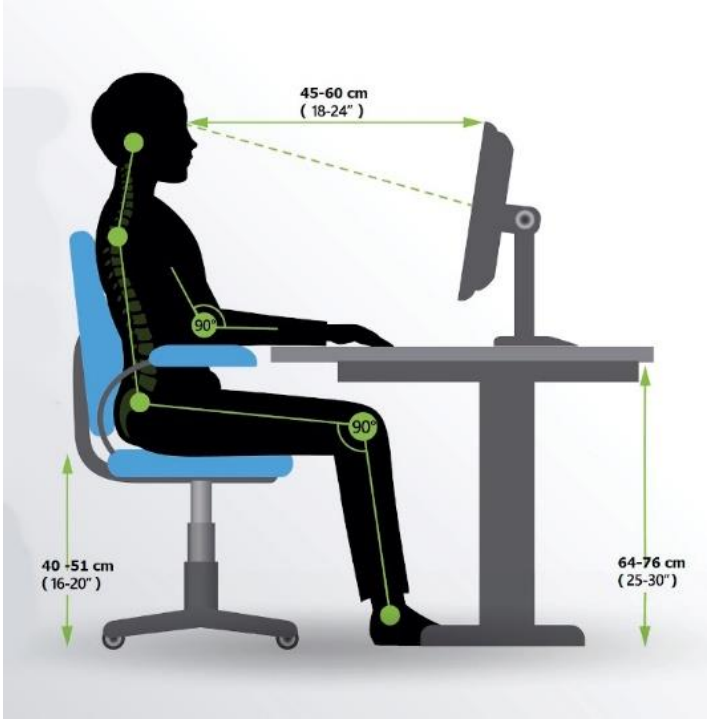

Рисунок 4.1 – Правильна організація робочого місця користувача персонального комп’ютера

Монітор необхідно розташовувати безпосередньо перед користувачем на відстані приблизно 50–70 см від очей. Верхня межа екрана повинна знаходитися на

рівні очей або трохи нижче. Таке розташування сприяє зменшенню навантаження на органи зору та шийний відділ хребта.

Одним із важливих факторів безпеки є забезпечення належного освітлення робочого місця. Освітлення повинно бути рівномірним та достатнім для комфортного сприйняття інформації на екрані монітора і паперових документах. Не допускається утворення відблисків на поверхні екрана, оскільки це призводить до швидкої втоми очей та зниження працездатності [17].

Для забезпечення комфортних умов праці необхідно підтримувати оптимальні параметри мікроклімату в приміщенні [19]. Температура повітря повинна перебувати в межах 20–24 °C, відносна вологість – 40–60 %, а швидкість руху повітря не повинна створювати відчуття протягів. Недотримання цих вимог може негативно впливати на самопочуття працівника та його працездатність.

Під час роботи за комп’ютером необхідно дотримуватися встановленого режиму праці та відпочинку. Тривала безперервна робота за монітором може спричиняти втому, зниження концентрації уваги та погіршення стану здоров’я. Для зменшення навантаження рекомендується робити короткі перерви через кожні 45–60 хвилин роботи та виконувати вправи для очей і м’язів спини [17].

Особливу увагу слід приділяти електробезпеці. Усе комп’ютерне обладнання повинно бути справним, мати надійне заземлення та відповідати вимогам електробезпеки. Забороняється використовувати пошкоджені кабелі живлення, самостійно виконувати ремонт електрообладнання або працювати з обладнанням, що має видимі ознаки несправності [16].

Дотримання вимог охорони праці під час роботи з персональним комп’ютером дозволяє зменшити ризик виникнення професійних захворювань, підвищити продуктивність праці та забезпечити комфортні умови виконання професійних обов’язків.

Отже, правильна організація робочого місця користувача персонального комп’ютера є важливою умовою забезпечення безпечної та ефективної роботи під час розробки програмного забезпечення, зокрема мобільних Android-додатків.

ВИСНОВКИ

У кваліфікаційній роботі бакалавра виконано розробку та тестування мобільного додатку для моніторингу показників стану серцево-судинної системи людини. Актуальність роботи обумовлена необхідністю постійного контролю стану здоров’я населення та зростанням популярності мобільних технологій у сфері охорони здоров’я. Використання мобільних додатків дозволяє спростити процес збору, збереження та аналізу медичних показників, а також забезпечити користувачам швидкий доступ до необхідної інформації.

У першому розділі проведено аналіз предметної області та досліджено особливості моніторингу показників серцево-судинної системи людини. Розглянуто основні показники, які характеризують стан серцево-судинної системи, зокрема артеріальний тиск та частоту серцевих скорочень. Виконано аналіз існуючих програмних рішень для моніторингу стану здоров’я, визначено їх переваги та недоліки, а також сформовано основні вимоги до розроблюваного програмного продукту.

У другому розділі здійснено проєктування мобільного додатку CardioMonitor. Визначено архітектуру програмного забезпечення на основі шаблону MVVM, розроблено структуру бази даних та логіку взаємодії основних компонентів системи. Для реалізації програмного продукту обрано сучасний стек технологій, який включає мову програмування Kotlin, Jetpack Compose, Navigation Compose, Room Database, Kotlin Coroutines та інші інструменти платформи Android.

У третьому розділі виконано практичну реалізацію мобільного додатку. Розроблено функціональні модулі для введення, збереження та перегляду результатів вимірювань артеріального тиску та пульсу. Реалізовано систему ведення історії вимірювань, статистичний модуль, графічне відображення показників, систему аналізу стану користувача, модуль експорту PDF-звітів та механізм нагадувань про необхідність виконання вимірювань. Також проведено тестування програмного продукту, яке підтвердило коректність роботи реалізованого функціоналу та відповідність додатку поставленим вимогам.

У четвертому розділі розглянуто питання безпеки життєдіяльності та охорони праці під час виконання робіт із використанням комп’ютерної техніки. Проведено аналіз ризику як кількісної оцінки небезпек та досліджено основні вимоги до організації безпечного робочого місця користувача персонального комп’ютера. Визначено заходи, спрямовані на зменшення професійних ризиків та забезпечення комфортних умов праці.

Результатом виконання кваліфікаційної роботи є створений мобільний додаток CardioMonitor, який забезпечує моніторинг показників стану серцево-судинної системи людини, збереження історії вимірювань, аналіз отриманих даних та формування рекомендацій для користувача. Розроблений програмний продукт може бути використаний для щоденного контролю стану здоров’я та підвищення обізнаності користувачів щодо власних показників серцево-судинної системи.

Поставлена мета роботи досягнута, а всі визначені завдання виконані в повному обсязі. Перспективою подальшого розвитку програмного продукту є інтеграція з пристроями для автоматичного отримання показників стану здоров’я та використання методів інтелектуального аналізу даних.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. World Health Organization. Cardiovascular diseases (CVDs) [Електронний ресурс]. – Режим доступу: https://www.who.int/news-room/fact-sheets/detail/cardiovascular-diseases-(cvds) (дата звернення: 18.05.2026).
2. European Society of Cardiology. Cardiovascular Health [Електронний ресурс]. – Режим доступу: https://www.escardio.org (дата звернення: 19.05.2026).
3. Центр громадського здоров’я МОЗ України. Артеріальна гіпертензія [Електронний ресурс]. – Режим доступу: https://phc.org.ua (дата звернення: 19.05.2026).
4. Жидецький В.Ц. Охорона праці користувачів комп’ютерів : підручник. Львів : Афіша, 2020. 176 с.
5. Android Developers Documentation [Електронний ресурс]. – Режим доступу: https://developer.android.com (дата звернення: 26.05.2026).
6. Kotlin Documentation [Електронний ресурс]. – Режим доступу: https://kotlinlang.org/docs/home.html (дата звернення: 16.05.2026).
7. Jetpack Compose Documentation [Електронний ресурс]. – Режим доступу: https://developer.android.com/jetpack/compose (дата звернення: 27.05.2026).
8. Navigation Compose Documentation [Електронний ресурс]. – Режим доступу: https://developer.android.com/jetpack/compose/navigation (дата звернення: 27.05.2026).
9. Room Persistence Library Documentation [Електронний ресурс]. – Режим доступу: https://developer.android.com/training/data-storage/room (дата звернення: 28.05.2026).
10. SQLite Documentation [Електронний ресурс]. – Режим доступу: https://www.sqlite.org/docs.html (дата звернення: 28.05.2026).

11. Kotlin Coroutines Guide [Електронний ресурс]. – Режим доступу: https://kotlinlang.org/docs/coroutines-overview.html (дата звернення: 28.05.2026).
12. Material Design 3 [Електронний ресурс]. – Режим доступу: https://m3.material.io (дата звернення: 28.05.2026).
13. Android Studio Documentation [Електронний ресурс]. – Режим доступу: https://developer.android.com/studio (дата звернення: 28.05.2026).
14. MPAndroidChart GitHub Repository [Електронний ресурс]. – Режим доступу: https://github.com/PhilJay/MPAndroidChart (дата звернення: 28.05.2026).
15. Безпека життєдіяльності та охорона праці : підруч. / В. В. Сокуренко, О. М. Бандурка та ін. Харків : ХНУВС, 2021. 308 с.
16. Закон України «Про охорону праці» : Закон України від 14.10.1992 №2694-XII [Електронний ресурс]. – Режим доступу: https://zakon.rada.gov.ua/laws/show/2694-12 (дата звернення: 11.06.2026).
17. Про затвердження Вимог щодо безпеки та захисту здоров'я працівників під час роботи з екранними пристроями : НПАОП 0.00-7.15-18 [Електронний ресурс]. – Режим доступу: https://zakon.rada.gov.ua/laws/show/z0508-18 (дата звернення: 11.06.2026).
18. ДСТУ ISO 6385:2005. Ергономічні принципи проєктування робочих систем. – Київ : Держспоживстандарт України, 2007.
19. ДСН 3.3.6.042-99. Державні санітарні норми мікроклімату виробничих приміщень. – Київ, 1999.
20. Основи охорони праці : навчальний посібник / К. Н. Ткачук, М. О. Халімовський. – Київ : Основа, 2011. – 448 с.
21. Методичні вказівки до виконання кваліфікаційної роботи бакалавра для здобувачів спеціальності 121 Інженерія програмного забезпечення, всіх форм навчання / укладачі: Михалик Д. М., Цуприк Г. Б., Бревус В. М. Тернопіль :

Тернопільський національний технічний університет імені Івана Пулюя, 2024. 45 с.

22. Android Architecture Components [Електронний ресурс]. – Режим доступу: https://developer.android.com/topic/architecture (дата звернення: 24.05.2026).

ДОДАТКИ

ДОДАТОК А

Програмний код головного екрана HomeScreen.kt

```
package com.example.cardiomonitor.ui.screens

import androidx.compose.foundation.layout.Arrangement
import androidx.compose.foundation.layout.Column
import androidx.compose.foundation.layout.Row
import androidx.compose.foundation.layout.fillMaxSize
import androidx.compose.foundation.layout.fillMaxWidth
import androidx.compose.foundation.layout.padding
import androidx.compose.foundation.rememberScrollState
import androidx.compose.foundation.verticalScroll
import androidx.compose.material.icons.Icons
import androidx.compose.material.icons.filled.Add
import androidx.compose.material.icons.filled.Favorite
import androidx.compose.material.icons.filled.MonitorHeart
import androidx.compose.material.icons.filled.Person
import androidx.compose.material3.Button
import androidx.compose.material3.ButtonDefaults
import androidx.compose.material3.FilledTonalButton
import androidx.compose.material3.MaterialTheme
import androidx.compose.material3.Text
import androidx.compose.runtime.Composable
import androidx.compose.ui.Modifier
import androidx.compose.ui.unit.dp
import com.example.cardiomonitor.domain.BmiCalculator
import com.example.cardiomonitor.domain.HealthStatusCalculator
import com.example.cardiomonitor.ui.components.CardSectionHeader
import com.example.cardiomonitor.ui.components.DashboardCard
import com.example.cardiomonitor.ui.components.HealthAssessmentCard
import com.example.cardiomonitor.ui.components.RiskLevelCard
import com.example.cardiomonitor.ui.components.HealthStatusBadge
import com.example.cardiomonitor.ui.components.HealthStatusVisuals
import com.example.cardiomonitor.ui.components.MetricRow
import com.example.cardiomonitor.ui.components.ScreenHeader
```

```
import com.example.cardiomonitor.ui.state.HomeUiState

@Composable
fun HomeScreen(
    uiState: HomeUiState,
    onNavigateToAdd: () -> Unit,
    onNavigateToHistory: () -> Unit,
    onNavigateToStats: () -> Unit
) {
    val latestMeasurement = uiState.latestMeasurement
    val healthStatus = latestMeasurement?.let {
HealthStatusCalculator.resolveStatus(it) }
        ?: "Дані відсутні"
    val healthStyle = HealthStatusVisuals.styleFor(healthStatus)
    val profile = uiState.profile
    val bmi = profile?.let { BmiCalculator.calculate(it.height, it.weight)
}

    Column(
        modifier = Modifier
            .fillMaxSize()
            .verticalScroll(rememberScrollState())
            .padding(horizontal = 20.dp, vertical = 16.dp),
        verticalArrangement = Arrangement.spacedBy(16.dp)
    ) {
        ScreenHeader(
            title = "CardioMonitor",
            subtitle = "Моніторинг серцево-судинної системи"
        )

        if (profile != null && bmi != null) {
            DashboardCard(containerColor =
MaterialTheme.colorScheme.primaryContainer) {
                CardSectionHeader(
                    title = "Профіль користувача",
                    icon = Icons.Default.Person,
                    iconTint = MaterialTheme.colorScheme.primary
                )
```

```
                Text(
                    text = "Вітаємо, ${profile.name}!",
                    style = MaterialTheme.typography.titleLarge
                )
                MetricRow(label = "Вік", value = "${profile.age} років")
                MetricRow(label = "IMT", value =
BmiCalculator.formatBmi(bmi))
                MetricRow(label = "Категорія IMT", value =
BmiCalculator.getCategory(bmi))
            }
        } else {
            DashboardCard(containerColor =
MaterialTheme.colorScheme.surfaceVariant) {
                CardSectionHeader(
                    title = "Профіль користувача",
                    icon = Icons.Default.Person
                )
                Text(
                    text = "Заповніть профіль, щоб отримати
персоналізований аналіз здоров'я.",
                    style = MaterialTheme.typography.bodyMedium
                )
            }
        }

        DashboardCard {
            CardSectionHeader(
                title = "Останнє вимірювання",
                icon = Icons.Default.MonitorHeart
            )
            MetricRow(
                label = "Артеріальний тиск",
                value = "${latestMeasurement?.systolic ?: "--
"}/${latestMeasurement?.diastolic ?: "--"} мм рт. ст."
            )
            MetricRow(
                label = "Пульс",
                value = "${latestMeasurement?.pulse ?: "--"} уд/хв"
```

```
            )
        }

        DashboardCard(
            containerColor = healthStyle.containerColor,
            contentColor = healthStyle.contentColor
        ) {
            CardSectionHeader(
                title = "Стан здоров'я",
                icon = Icons.Default.Favorite,
                iconTint = healthStyle.contentColor
            )
            if (latestMeasurement != null) {
                HealthStatusBadge(measurement = latestMeasurement, large =
true)
            } else {
                HealthStatusBadge(status = healthStatus, large = true)
            }
        }

        uiState.healthAssessment?.let { assessment ->
            HealthAssessmentCard(assessment = assessment)
        }

        uiState.riskLevel?.let { risk ->
            RiskLevelCard(risk = risk)
        }

        DashboardCard {
            CardSectionHeader(title = "Швидкі дії", icon =
Icons.Default.Add)
            Row(
                modifier = Modifier.fillMaxWidth(),
                horizontalArrangement = Arrangement.spacedBy(8.dp)
            ) {
                Button(
                    onClick = onNavigateToAdd,
                    modifier = Modifier.weight(1f),
```

```
                    shape = ButtonDefaults.shape
                ) {
                    Text("Додати")
                }
                FilledTonalButton(
                    onClick = onNavigateToHistory,
                    modifier = Modifier.weight(1f)
                ) {
                    Text("Історія")
                }
                FilledTonalButton(
                    onClick = onNavigateToStats,
                    modifier = Modifier.weight(1f)
                ) {
                    Text("Стат.")
                }
            }
        }
    }
}
```

ДОДАТОК Б

Фрагменти програмного коду роботи з базою даних

```
@Entity(tableName = "measurements")
data class MeasurementEntity(
@PrimaryKey(autoGenerate = true)
val id: Long = 0,
val systolicPressure: Int,
val diastolicPressure: Int,
val pulse: Int,
val note: String,
val timestamp: Long
)

@Entity(tableName = "user_profile")
data class UserProfileEntity(
@PrimaryKey
val id: Int = 1,
val name: String,
val age: Int,
val gender: String,
val height: Float,
val weight: Float
)

@Dao
interface MeasurementDao {
@Insert
suspend fun insertMeasurement(measurement: MeasurementEntity)
@Update
suspend fun updateMeasurement(measurement: MeasurementEntity)
@Delete
suspend fun deleteMeasurement(measurement: MeasurementEntity)
@Query("SELECT * FROM measurements ORDER BY timestamp DESC")
fun getAllMeasurements(): Flow<List<MeasurementEntity>>

@Query("SELECT * FROM measurements WHERE note LIKE '%' || :query || '%'
```

```
ORDER BY timestamp DESC")
fun searchByNote(query: String): Flow<List<MeasurementEntity>>
}

@Database(
entities = [
MeasurementEntity::class,
UserProfileEntity::class
],
version = 1
)
abstract class AppDatabase : RoomDatabase() {
abstract fun measurementDao(): MeasurementDao
abstract fun userProfileDao(): UserProfileDao
}
```

ДОДАТОК В

Фрагменти програмного коду аналітичного модуля

```
object BmiCalculator {

fun calculateBmi(weight: Float, heightCm: Float): Float {
if (heightCm <= 0f) return 0f

val heightM = heightCm / 100f
return weight / (heightM * heightM)
}

fun getBmiCategory(bmi: Float): String {
return when {
bmi < 18.5f -> "Недостатня маса тіла"
bmi < 25f -> "Нормальна маса тіла"
bmi < 30f -> "Надмірна маса тіла"
else -> "Ожиріння"
}
}}
```

```
object HealthStatusCalculator {

fun getPressureStatus(systolic: Int, diastolic: Int): String {
return when {
systolic < 90 || diastolic < 60 -> "Низький тиск"
systolic in 90..99 || diastolic in 60..64 -> "Понижений тиск"
systolic in 100..120 && diastolic in 65..80 -> "Нормальний тиск"
systolic in 121..139 || diastolic in 81..89 -> "Підвищений тиск"
else -> "Високий тиск"
}}}
```

```
object RiskAnalyzer {
fun calculateRisk(
averageSystolic: Double,
averageDiastolic: Double,
averagePulse: Double,
```

```
bmi: Float
): String {
var riskScore = 0

if (averageSystolic > 140 || averageDiastolic > 90) riskScore += 2
if (averageSystolic < 90 || averageDiastolic < 60) riskScore += 1
if (averagePulse > 100 || averagePulse < 60) riskScore += 1
if (bmi >= 30f) riskScore += 2
else if (bmi >= 25f) riskScore += 1

return when {
riskScore <= 1 -> "Низький"
riskScore <= 3 -> "Середній"
else -> "Високий"
}
}
}
```

ДОДАТОК Г

Посилання на GitHub репозиторій проекту

https://github.com/NikolaST8961/cardio-monitor-app