

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ДЕРЖАВНИЙ УНІВЕРСИТЕТ «ЖИТОМИРСЬКА ПОЛІТЕХНІКА»
ФАКУЛЬТЕТ ІНФОРМАЦІЙНО-КОМП'ЮТЕРНИХ ТЕХНОЛОГІЙ
КАФЕДРА ІНЖЕНЕРІЇ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

ПОЯСНЮВАЛЬНА ЗАПИСКА

до кваліфікаційної роботи освітнього ступеня «бакалавр»
за спеціальністю 121 «Інженерія програмного забезпечення»
(освітня програма «Інженерія програмного забезпечення»)

на тему:

**«Застосунок для відстеження фізичної активності
людини з елементами ігрової мотивації»**

Виконав студент групи ІПЗ-21-1
ТАЛЬКО Артем Юзефович

Керівник роботи:
ЛОКТИКОВА Тамара Миколаївна

Рецензент:
ВАКАЛЮК Тетяна Анатоліївна

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ДЕРЖАВНИЙ УНІВЕРСИТЕТ «ЖИТОМИРСЬКА ПОЛІТЕХНІКА»
ФАКУЛЬТЕТ ІНФОРМАЦІЙНО-КОМП'ЮТЕРНИХ ТЕХНОЛОГІЙ
КАФЕДРА ІНЖЕНЕРІЇ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

«ЗАТВЕРДЖУЮ»

Зав. кафедри інженерії програмного
забезпечення

Тетяна ВАКАЛЮК

«26» лютого 2025 р.

ЗАВДАННЯ
на кваліфікаційну роботу

Здобувач вищої освіти: **ТАЛЬКО** Артем Юзефович

Керівник роботи: **ЛОКТИКОВА** Тамара Миколаївна

Тема роботи: **« Застосунок для відстеження фізичної активності людини з елементами ігрової мотивації»**,

затверджена Наказом закладу вищої освіти від **«14» лютого 2025 р., №61/с**

Вихідні дані для роботи: **об'єктом дослідження є процес проєктування та розробки мобільного застосунку для моніторингу та стимулювання фізичної активності користувачів, а предметом дослідження є методи реалізації мобільного застосунку для моніторингу фізичної активності з гейміфікаційними елементами, інтеграція з платформними API (HealthKit для крокоміра й калорій, CoreLocation для маршруту, UserNotifications для нагадувань); архітектурні підходи та патерни (MVVM-C, Clean Architecture) із використанням SwiftUI, Combine; механізми авторизації та зберігання даних (Firebase Auth, Firestore); а також оцінка їх впливу на мотивацію користувачів та зручність взаємодії.**

Консультанти з бакалаврської кваліфікаційної роботи із зазначенням розділів, що їх стосуються:

Розділ	Консультант	Завдання видав	Завдання прийняв
1	Локтікова Т.М.	27.02.2025р.	27.02.2025р.
2	Локтікова Т.М.	27.02.2025р.	27.02.2025р.
3	Локтікова Т.М.	27.02.2025р.	27.02.2025р.

РЕФЕРАТ

Випускна кваліфікаційна робота бакалавра складається з нативного мобільного застосунку для iOS-платформи та пояснювальної записки. Пояснювальна записка містить 61 сторінку текстової частини, 18 ілюстрацій та 10 таблиць.

У межах роботи розроблено мобільний застосунок, який допомагає користувачам підтримувати фізичну активність через мотиваційні механіки та гейміфікацію. Застосунок дозволяє реєструвати активності (ходьба, біг), зберігати дані про тренування, переглядати особисту статистику, брати участь у змаганнях (таблиці лідерів), а також користуватися системою нагадувань.

Інтерфейс програми побудований за допомогою SwiftUI, а для зберігання даних використано Firebase. Проєкт реалізовано з дотриманням принципів UX/UI-дизайну, що забезпечує інтуїтивну взаємодію з основними функціями застосунку. Особливу увагу приділено візуалізації прогресу користувача та легкості використання.

Застосунок є повністю автономним, не потребує підключення до зовнішніх серверів, і може стати основою для подальшої розробки функціоналу спільного доступу, інтеграції з Apple HealthKit та онлайн-синхронізації.

КЛЮЧОВІ СЛОВА: ГЕЙМІФІКАЦІЯ, ЗДОРОВ'Я, МОБІЛЬНИЙ ЗАСТОСУНОК, МОТИВАЦІЯ, ФІЗИЧНА АКТИВНІСТЬ, FIREBASE, IOS, SWIFT.

					ІПЗ.КР.Б-121-25-ІПЗ			
Змн.	Арк.	№ докум.	Підпис	Дата	Застосунок для відстеження фізичної активності людини з елементами ігрової мотивації. Пояснювальна записка	Літ.	Арк.	Аркушів
Розроб.		Талько А.Ю.					3	61
Керівник		Локтікова Т.М				ФІКТ Гр. ІПЗ-21-1		
Н. контр.		Тичина О.П.						
Зав. каф.		Вакалюк Т.А.						

ABSTRACT

The bachelor's thesis consists of a native mobile application for the iOS platform and an explanatory note. The explanatory note contains 61 pages of text, 18 illustrations, and 10 tables.

As part of the work, a mobile application was developed that helps users maintain physical activity through motivational mechanics and gamification. The application allows you to register activities (walking, running), save training data, view personal statistics, participate in competitions (leaderboards), and use the reminder system.

The application interface is built using SwiftUI, and Firebase is used for data storage. The project was implemented in accordance with UX/UI design principles, which ensures intuitive interaction with the main functions of the application. Particular attention was paid to the visualization of user progress and ease of use.

The app is completely autonomous, does not require a connection to external servers, and can serve as a basis for further development of shared access functionality, integration with Apple HealthKit, and online synchronization.

KEYWORDS: GAMIFICATION, HEALTH, MOBILE APPLICATION, MOTIVATION, PHYSICAL ACTIVITY, FIREBASE, IOS, SWIFT.

		Талько А.Ю.			ІІЗ.КР.Б-121-25-ІІЗ	Арк.
		Локтікова Т.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		4

ЗМІСТ

РЕФЕРАТ	3
ABSTRACT	4
ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ	6
ВСТУП.....	7
РОЗДІЛ 1 АНАЛІЗ НАПРЯМКІВ ВИКОРИСТАННЯ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ ДЛЯ РОЗРОБКИ ЗАСТОСУНКУ З ВІДСТЕЖЕННЯ ФІЗИЧНОЇ АКТИВНОСТІ ЛЮДИНИ	9
1.1 Постановка задачі.....	9
1.2 Аналіз аналогів програмного продукту	11
1.3 Вибір архітектури та інструментальних засобів реалізації застосунку, вимоги до апаратного забезпечення	15
Висновки до першого розділу.....	17
РОЗДІЛ 2 ПРОЄКТУВАННЯ ЗАСТОСУНКУ	19
2.1 Визначення варіантів використання.....	19
2.2 Розробка бази даних.....	29
2.3 Проєктування та реалізація алгоритмів роботи застосунку.....	32
2.4 Реалізація функціоналу застосунку	35
Висновки до другого розділу	37
РОЗДІЛ 3 ІНТЕРФЕЙС КОРИСТУВАЧА ТА ПОРЯДОК РОБОТИ ІЗ ЗАСТОСУНКОМ З ВІДСТЕЖЕННЯ ФІЗИЧНОЇ АКТИВНОСТІ ЛЮДИНИ	39
3.1 Порядок встановлення та налаштування параметрів застосунку.....	39
3.2 Загальна структура інтерфейсу користувача	41
Висновки до третього розділу.....	54
ВИСНОВКИ	56
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ.....	58
ДОДАТКИ	61

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

API – прикладний програмний інтерфейс (Application Programming Interface)

CRUD – основні операції з даними (Create, Read, Update, Delete)

GPS – глобальна система позиціонування (Global Positioning System)

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

UI – інтерфейс користувача (User Interface)

UX – користувацький досвід (User Experience)

БД – база даних

СУБД – система управління базами даних

		Талько А.Ю.			ІПЗ.КР.Б-121-25-ПЗ	Арк.
		Локтікова Т.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		6

ВСТУП

Актуальність теми полягає у необхідності створення інноваційних рішень для підвищення фізичної активності серед населення через сучасні мобільні технології. Недостатній рівень фізичної активності є серйозною проблемою, що призводить до погіршення здоров'я, зниження якості життя та підвищення ризику виникнення хронічних захворювань. Використання мобільних застосунків з інтегрованими ігровими елементами може ефективно стимулювати користувачів до регулярної рухової активності, підвищуючи їхню мотивацію та зацікавленість у підтриманні здорового способу життя.

Метою випускної кваліфікаційної роботи бакалавра є розробка мобільного застосунку для відстеження денної активності користувачів, який включає інтерактивний елемент (квіточку), що змінює свій вигляд залежно від кількості пройдених кроків, а також надає можливість перегляду маршрутів і статистики активності.

Для досягнення цієї мети необхідно виконати такі **завдання**:

1. Провести аналіз існуючих мобільних застосунків для моніторингу фізичної активності, визначити їхні переваги та недоліки.
2. Сформулювати основні вимоги до мобільного застосунку, враховуючи точність підрахунку кроків, інтеграцію гейміфікованих елементів та зручність використання.
3. Обґрунтувати вибір технологічного стеку для реалізації проєкту.
4. Розробити мобільний застосунок, що дозволяє користувачам ефективно відстежувати свою активність, використовувати інтерактивні ігрові елементи та переглядати статистику.
5. Провести тестування розробленого застосунку для перевірки його функціональності.

Об'єктом дослідження є процес проєктування та розробки мобільного застосунку для моніторингу та стимулювання фізичної активності користувачів.

Предметом дослідження є методи реалізації мобільного застосунку для моніторингу фізичної активності з гейміфікаційними елементами, інтеграція з

		Талько А.Ю.			ІПЗ.КР.Б-121-25-ІПЗ	Арк.
		Локтікова Т.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		7

платформними API (HealthKit для крокоміра й калорій, CoreLocation для маршруту, UserNotifications для нагадувань); архітектурні підходи та патерни (MVVM-C, Clean Architecture) із використанням SwiftUI, Combine; механізми авторизації та зберігання даних (Firebase Auth, Firestore); а також оцінка їх впливу на мотивацію користувачів та зручність взаємодії.

		Талько А.Ю.			ІПЗ.КР.Б-121-25-ПЗ	Арк.
		Локтікова Т.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		8

РОЗДІЛ 1 АНАЛІЗ НАПРЯМКІВ ВИКОРИСТАННЯ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ ДЛЯ РОЗРОБКИ ЗАСТОСУНКУ З ВІДСТЕЖЕННЯ ФІЗИЧНОЇ АКТИВНОСТІ ЛЮДИНИ

1.1 Постановка задачі

Сучасний спосіб життя характеризується значним зменшенням фізичної активності серед населення через переважно сидячий спосіб життя та поширення дистанційної роботи. Велика кількість людей проводить більшу частину доби перед комп'ютерами чи іншими електронними пристроями, часто не знаходячи часу для активного руху або прогулянок на свіжому повітрі. Така тенденція спостерігається не лише серед дорослого населення, але й серед підлітків і навіть дітей, оскільки навчання й дозвілля дедалі більше переміщуються у цифровий простір.

Це призводить до різноманітних проблем зі здоров'ям, таких як ожиріння, серцево-судинні захворювання, діабет другого типу, опорно-рухові порушення, а також інші хронічні хвороби. Доведено, що недостатня фізична активність є однією з основних причин передчасної смертності в усьому світі. Окрім фізичних аспектів, малорухливий спосіб життя негативно впливає на психоемоційний стан, підвищує рівень стресу та тривожності, сприяє розвитку депресивних розладів.

Важливість підтримки фізичної активності підкреслюється рекомендаціями Всесвітньої організації охорони здоров'я, яка радить здійснювати не менше 10 тисяч кроків на день для підтримки загального здоров'я та профілактики захворювань. Регулярна рухова активність допомагає зменшити ризик виникнення хронічних недуг, покращує роботу серцево-судинної системи, сприяє підтримці здорової ваги та підвищує загальну якість життя. Навіть помірна щоденна активність позитивно впливає на сон, рівень енергії та працездатність людини.

В умовах інформаційного перенавантаження та дефіциту мотивації до спорту надзвичайно важливо знайти прості, зручні та інтерактивні інструменти, які б заохочували людину рухатися частіше, не відчуючи зайвого стресу чи дискомфорту. Особливо це актуально для тих, хто не має можливості або часу

		Талько А.Ю.			ІПЗ.КР.Б-121-25-ПЗ	Арк.
		Локтікова Т.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		9

відвідувати спортивні зали чи брати участь у групових тренуваннях. Саме тому останніми роками зростає популярність цифрових рішень, які допомагають організовувати фізичну активність у повсякденному житті.

Одним із таких інструментів є мобільні застосунки для відстеження фізичної активності, що об'єднують функції крокоміра, трекера маршруту й аналізаторів здоров'я з елементами гейміфікації. Завдяки інтеграції з платформами HealthKit [14] і MapKit [17] користувачі можуть у реальному часі бачити свій маршрут на карті, автоматично фіксувати пульс і кількість кроків, а також відстежувати динаміку змін у загальному стані організму. Такі застосунки дозволяють зручно збирати статистику за тривалий період, порівнювати свої результати, ставити персональні цілі та аналізувати прогрес.

Водночас гейміфіковані механізми — віртуальна квітка, що «росте» пропорційно подоланим крокам, система досягнень і внутрішніх нагород — перетворюють рутинні вправи на захопливу гру, підвищуючи рівень залученості та особистої мотивації до регулярних занять. Додаткові візуальні елементи й інтерактивні завдання допомагають користувачам не втрачати інтерес до підтримки здорового способу життя та досягнення власних фітнес-цілей.

Слід зазначити, що зростає також кількість наукових досліджень, які підтверджують ефективність використання цифрових фітнес-трекерів та мобільних застосунків для формування і закріплення корисних звичок. Саме поєднання технологій і гейміфікації відкриває нові можливості для впливу на поведінку користувачів та стимулювання їх до здорової активності без примусу.

Основною метою випускної кваліфікаційної роботи є розробка мобільного застосунку, що дозволяє користувачам відстежувати денну активність, включає інтерактивний ігровий елемент (квіточку), що змінює вигляд залежно від кількості пройдених кроків, а також надає функції перегляду маршрутів та статистики. Таким чином, користувач отримує не лише інструмент для контролю за власним здоров'ям, а й додаткову мотивацію підтримувати рухову активність, що, у свою чергу, сприяє профілактиці численних захворювань і загальному покращенню якості життя.

		Талько А.Ю.			ІПЗ.КР.Б-121-25-ПЗ	Арк.
		Локтікова Т.М.				10
Змн.	Арк.	№ докум.	Підпис	Дата		

Для досягнення поставленої мети необхідно проаналізувати існуючі мобільні застосунки для відстеження активності, визначити їхні переваги та недоліки, сформулювати основні вимоги до майбутнього застосунку з урахуванням зручності інтерфейсу, точності підрахунку активності та інтеграції гейміфікації, обґрунтувати вибір технологічного стеку для реалізації проєкту, а також розробити власний мобільний застосунок із гейміфікованим функціоналом.

1.2 Аналіз аналогів програмного продукту

Ринок мобільних фітнес-додатків пропонує широкий спектр рішень, що поєднують моніторинг фізичної активності з ігровими механіками задля підвищення мотивації користувачів. Особливу увагу приділено інтерфейсним рішенням, моделям стимулювання, механізмам інтеграції зі сенсорами пристрою та фінансовим і соціальним аспектам взаємодії, що дозволяє окреслити конкурентні переваги та нішу для розроблюваного застосунку.

Першим аналогом буде застосунок StepBet [24]. Основною ідеєю застосунку є стимулювання фізичної активності через фінансові стимули, де користувачі ставлять гроші на досягнення певних цілей щодо кількості кроків. Такий підхід дозволяє створити додатковий рівень відповідальності та змагальності, що значно підвищує ймовірність дотримання обраного плану активності. Крім цього, участь у таких челенджах дозволяє користувачам об'єднуватися у спільноти, де вони можуть порівнювати власні досягнення з іншими та ділитися досвідом.

На рис. 1.1 представлено інтерфейс застосунку StepBet.

Перевагами цього застосунку є висока мотивація користувачів завдяки реальним фінансовим стимулам, однак він не надає інтерактивних візуальних елементів, які могли б додатково захопити увагу користувача і зробити процес більш привабливим. Крім того, відсутність яскравої гейміфікації може зменшувати рівень емоційної залученості для окремих категорій користувачів, які віддають перевагу візуальному зворотному зв'язку.

		Талько А.Ю.			ІПЗ.КР.Б-121-25-ІПЗ	Арк.
		Локтікова Т.М.				11
Змн.	Арк.	№ докум.	Підпис	Дата		

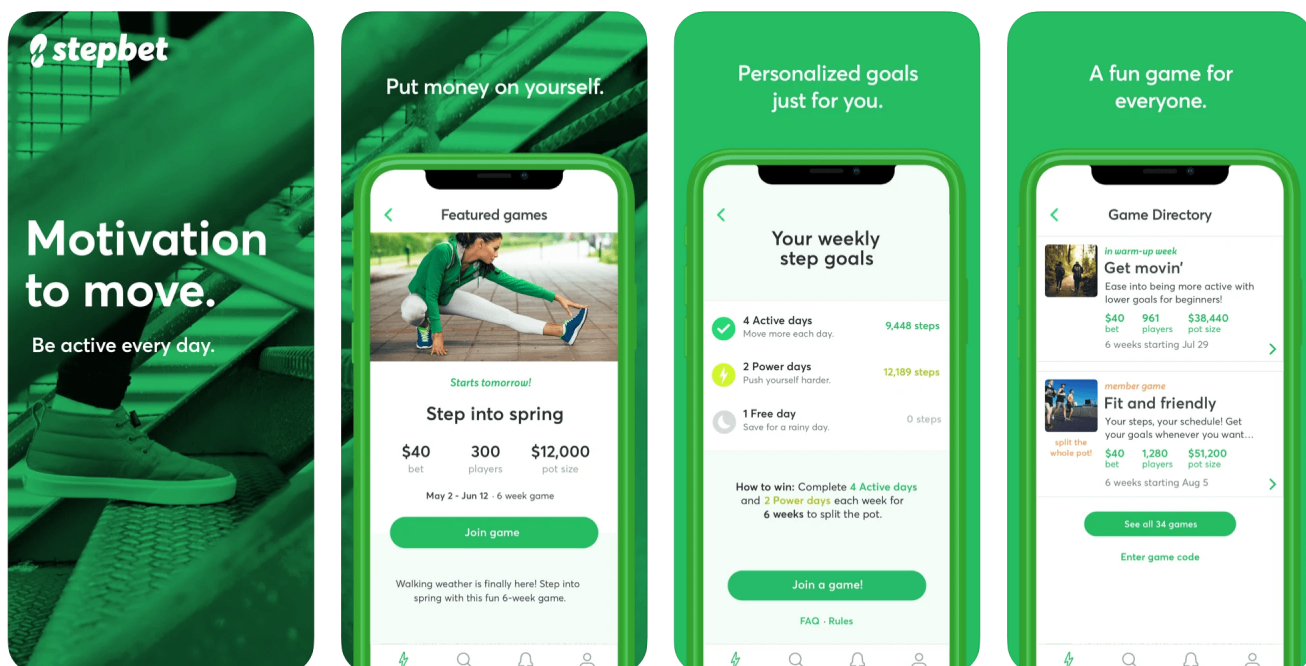


Рисунок 1.1 – Інтерфейс застосунку StepBet

Другим аналогом є мобільний застосунок Forest (рис. 1.2) [10]. Він орієнтований на зменшення часу користування смартфоном через вирощування віртуальних дерев. Основна перевага застосунку полягає в яскравому інтерактивному геймплеї, що забезпечує візуальну мотивацію користувачів. Проте, Forest не спрямований на фізичну активність та не інтегрує функції моніторингу кроків.

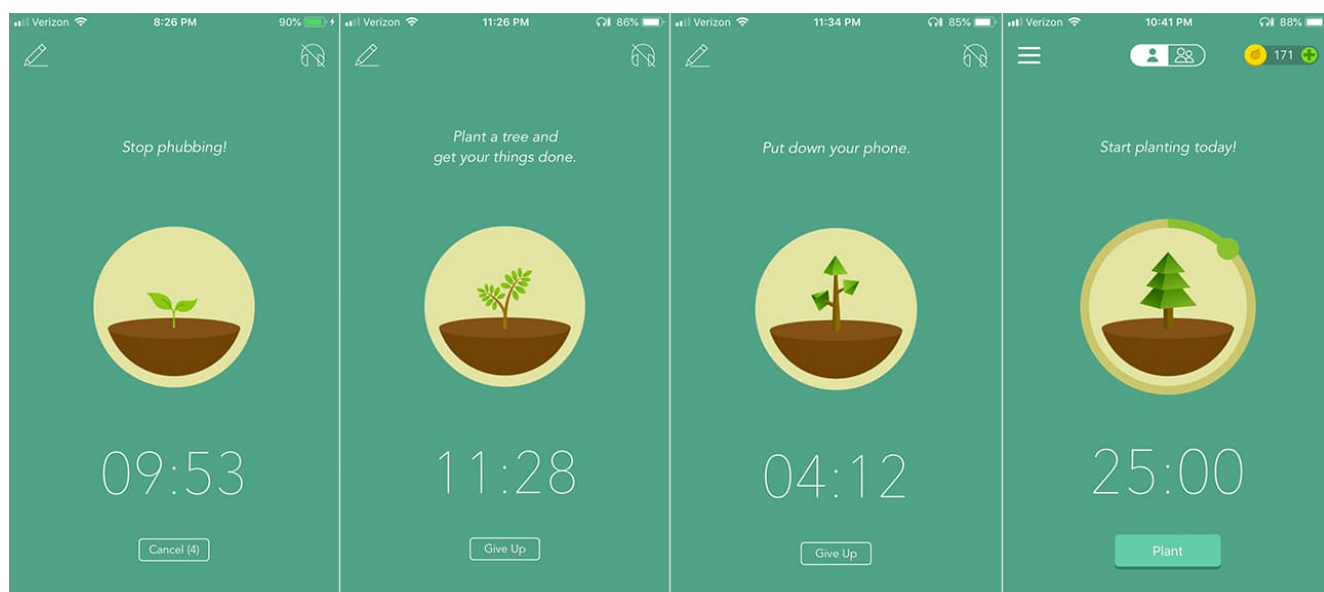


Рисунок 1.2 – Інтерфейс застосунку Forest

		Талько А.Ю.			ІПЗ.КР.Б-121-25-ПЗ	Арк.
		Локтікова Т.М.				12
Змн.	Арк.	№ докум.	Підпис	Дата		

Третім аналогом обрано застосунок Pacer (рис. 1.3) [20]. Це популярний застосунок, що орієнтований на щоденний підрахунок кроків та базову візуалізацію статистики фізичної активності користувача. Він має простий і зрозумілий інтерфейс, що сприяє його широкій популярності серед різних вікових категорій. Проте, цей застосунок не пропонує глибоких гейміфікованих функцій, які могли б додатково мотивувати та утримувати інтерес користувачів на тривалий час.



Рисунок 1.3 – Інтерфейс застосунку Pacer

Порівняльний аналіз аналогів програмного продукту наведено у табл. 1.1.

Таблиця 1.1

Порівняльна характеристика аналогів програмного продукту

Функціонал	StepBet	Forest	Pacer	Розроблюваний застосунок
Моніторинг фізичної активності	+	-	+	+
Візуальна інтерактивність	-	+	частково	+
Гейміфікація	+	+	-	+

Перегляд маршрутів та статистики	-	-	частково	+
Простота та зрозумілість	+	+	+	+
Синхронізація з HealthKit	-	-	+	+
Підтримка щоденних цілей	+	-	+	+
Повідомлення та нагадування	+	+	+	+

Проведений порівняльний аналіз свідчить, що жоден із існуючих рішень одночасно не забезпечує комплексного поєднання всіх ключових елементів: від точного відстеження фізичної активності та інтеграції з HealthKit до динамічної візуалізації, продуманої гейміфікації, відображення маршрутів і детальної статистики, а також підтримки персональних цілей із нагадуваннями. Розроблюваний застосунок заповнює цю нішу, поєднуючи зазначені можливості в єдиному інтерфейсі та створюючи унікальний користувацький досвід, який сприяє як мотивації, так і точному контролю за станом здоров'я.

На основі проведених огляду й аналізу можна сформулювати вимоги до розроблюваного застосунку: (просто текст)

На основі проведених огляду й аналізу можна сформулювати такі вимоги до розроблюваного застосунку: потрібно забезпечити реєстрацію та автентифікацію користувача з використанням Firebase Auth [8], з можливістю відновлення пароля; реалізувати механізми запуску, паузи та зупинки тренування із фіксацією маршруту на карті через MapKit та збором метрик здоров'я (кількість кроків, дистанція, калорії, пульс) через HealthKit; відображати маршрут і поточні дані тренування у реальному часі на інтерактивному екрані WorkoutView; надати окремий екран History/Statistic для перегляду детальної статистики та зведених графіків попередніх сесій (щоденних, тижневих, місячних); впровадити гейміфікаційний елемент — віртуальну квітку, що «росте» та змінює вигляд

		Талько А.Ю.			ІПЗ.КР.Б-121-25-ПЗ	Арк.
		Локтікова Т.М.				14
Змн.	Арк.	№ докум.	Підпис	Дата		

залежно від кількості пройдених кроків і досягнутих цілей; реалізувати таблицю лідерів із рейтингом користувачів за обраними метриками (кроки, дистанція, калорії) та можливістю фільтрації за періодом; забезпечити систему push-сповіщень і локальних нагадувань про початок чи завершення тренувань і досягнення проміжних цілей; а також надати екран налаштувань профілю, де користувач може змінити особисті дані та параметри сповіщень.

1.3 Вибір архітектури та інструментальних засобів реалізації застосунку, вимоги до апаратного забезпечення

Обрана архітектурна модель MVVM-C поєднує декларативний інтерфейс SwiftUI з чітким розділенням обов'язків між шарами View, ViewModel, Model та навігаційним шаром Coordinator, що забезпечує високу модульність і тестованість. View відповідає тільки за відображення стану, ViewModel — за ін'єкцію сервісів і реактивну трансформацію даних (використовуючи @Published/@Bindable), Model — за доменні сутності й бізнес-логіку у вигляді сервісів (WorkoutService, AuthService тощо), а Coordinator — за централізоване керування переходами між екранами та контейнерами вкладок. Такий підхід гарантує, що змінити навігаційний флоу або додати новий екран можна без втручання в бізнес-логіку, а саму бізнес-логіку легко покрити юніт-тестами незалежно від UI. Крім того, розділення навігації й представлення дозволяє уникнути дублювання коду та спрощує подальшу еволюцію проєкту, забезпечуючи зрозумілу ієрархію залежностей і мінімальну зв'язаність між компонентами.

Вибір технологій є важливою складовою процесу розробки мобільного застосунку, оскільки саме від цього залежить ефективність його функціонування, зручність експлуатації, масштабованість системи та безпека даних. У межах реалізації зазначеного проєкту було сформовано технологічний стек, що включає Swift і SwiftUI, HealthKit, MapKit, Firebase, а також підтримку нативних push-сповіщень із можливістю їх індивідуального налаштування користувачем.

Основною мовою програмування, використаною під час розробки, є Swift - сучасна високопродуктивна мова, створена Apple для розробки програмного

		Талько А.Ю.			ІПЗ.КР.Б-121-25-ПЗ	Арк.
		Локтікова Т.М.				15
Змн.	Арк.	№ докум.	Підпис	Дата		

забезпечення під iOS, macOS, watchOS і tvOS [27]. Swift вирізняється відкритим вихідним кодом, активною підтримкою спільноти розробників, високою швидкістю виконання, безпечністю типів та структурованістю синтаксису, що значно оптимізує як процес написання, так і подальшого супроводу коду.

Для побудови інтерфейсу користувача було застосовано фреймворк SwiftUI [28]. Це інноваційне рішення від Apple, що використовує декларативний підхід до розробки, дозволяючи ефективно створювати адаптивні та інтерактивні інтерфейси. SwiftUI забезпечує автоматичне масштабування UI-компонентів під різні типи пристроїв Apple, що сприяє уніфікації користувацького досвіду та скороченню часу розробки.

Механізми відстеження фізичної активності реалізовано через фреймворк HealthKit, який забезпечує доступ до фізіологічних показників користувача (кількість кроків, витрачені калорії, пройдена відстань тощо). Вбудована інтеграція з іншими сервісами Apple Health дозволяє зберігати та обробляти ці дані в межах єдиної екосистеми з дотриманням високих стандартів конфіденційності.

Для візуалізації користувацьких маршрутів було використано MapKit — геоінформаційний компонент, що надає засоби інтеграції картографічних сервісів Apple. Він підтримує кілька режимів відображення (стандартний, супутниковий, гібридний) і дозволяє налаштовувати карти, забезпечуючи інтуїтивно зрозумілу взаємодію з геолокаційними даними.

Обробку даних, аутентифікацію користувачів, та підтримку додаткового функціоналу реалізовано через платформу Firebase. Цей сервіс від Google надає інструменти для зберігання даних, миттєвого оновлення інформації на клієнтських пристроях і гнучкої масштабованості, що особливо важливо для проєктів зі зростаючою базою користувачів.

Важливим доповненням функціональності є реалізація системи нативних push-сповіщень. Користувачі мають змогу самостійно налаштовувати параметри сповіщень відповідно до особистих уподобань, що сприяє підвищенню рівня персоналізації інтерфейсу та загальної залученості.

		Талько А.Ю.			ІПЗ.КР.Б-121-25-ПЗ	Арк.
		Локтікова Т.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		16

Таким чином, обраний стек технологій забезпечує оптимальний баланс між продуктивністю, стабільністю, зручністю розробки та можливістю масштабування, що створює надійну основу для реалізації повноцінного функціоналу мобільного застосунку з орієнтацією на якісний користувацький досвід.

Висновки до першого розділу

У першому розділі було проведено всебічний аналіз предметної області, пов'язаної з проблематикою зниження рівня фізичної активності серед населення, що обумовлено переважно малорухливим способом життя та поширенням дистанційної роботи. Визначено, що одним з ефективних засобів стимулювання активності є використання мобільних застосунків, особливо за умови інтеграції в них елементів гейміфікації. Саме тому основною метою проєкту визначено створення мобільного застосунку, який забезпечує моніторинг щоденної активності користувачів, зокрема підрахунок кроків, та підвищення мотивації завдяки інтерактивній візуалізації прогресу у вигляді ігрового елемента – квітки, що змінює вигляд залежно від досягнутих цілей.

У рамках дослідження було здійснено аналіз функціональних можливостей та недоліків трьох популярних аналогів: StepBet, Forest та Pacer. Порівняльна характеристика виявила, що жоден із цих застосунків не поєднує повного спектру необхідного функціоналу, зокрема одночасного використання механізмів моніторингу активності, динамічної візуалізації, маршрутизації, аналітики та гейміфікованої взаємодії з користувачем. На основі виявлених обмежень аналогів було сформульовано унікальні функціональні вимоги до власного мобільного застосунку.

Крім того, у цьому розділі обґрунтовано вибір технологічного стеку, що відповідає визначеним вимогам до продуктивності, зручності використання, масштабованості та безпеки. Для реалізації клієнтської частини застосунку було обрано мову програмування Swift у поєднанні з фреймворком SwiftUI, що забезпечує високу продуктивність, гнучкий адаптивний інтерфейс та зручність розробки завдяки декларативному підходу до створення інтерфейсів. Для реалізації

		Талько А.Ю.			ІПЗ.КР.Б-121-25-ПЗ	Арк.
		Локтікова Т.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		17

функціоналу відстеження фізичної активності інтегровано фреймворк HealthKit, який дозволяє точно моніторити фізіологічні показники користувача, включаючи кількість кроків, витрачені калорії та пройдену дистанцію, а також забезпечує захист персональних даних і взаємодію з іншими сервісами Apple Health. Компонент MapKit використовується для побудови та візуалізації маршрутів користувача в різних режимах відображення карти, що дозволяє інтерактивно працювати з геолокаційною інформацією. Для зберігання даних, обробки авторизації, реалізації аналітики та підтримки масштабованості застосовано хмарну платформу Firebase, яка гарантує надійне збереження інформації та забезпечує миттєве оновлення даних на пристроях користувачів. Доповненням функціональності є впровадження нативних push-сповіщень із можливістю індивідуального налаштування, що сприяє підвищенню залученості користувачів і персоналізує досвід взаємодії з застосунком.

Отже, у першому розділі ми не тільки розглянули ключові проблеми низької фізичної активності та дослідили можливості існуючих рішень, а й чітко визначили, які саме функції та інструменти стануть основою нашого застосунку. Обраний стек — Swift/SwiftUI, HealthKit, MapKit і Firebase — вже довів свою ефективність у подібних проєктах та відповідає всім вимогам до продуктивності, безпеки та зручності. Це дає впевненість, що в подальших розділах ми зможемо зосередитися на детальній реалізації й отримати працездатний і приємний у користуванні продукт.

		Талько А.Ю.			ІПЗ.КР.Б-121-25-ПЗ	Арк.
		Локтікова Т.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		18

РОЗДІЛ 2 ПРОЄКТУВАННЯ ЗАСТОСУНКУ

2.1 Визначення варіантів використання

Аналіз вимог до мобільного застосунку дозволив чітко виокремити єдиного ключового актора – Користувача – та визначити набір основних сценаріїв його взаємодії з системою, включно з реєстрацією й входом, відображенням головної панелі з персоналізованою інформацією, запуском, паузою й зупинкою тренувань із фіксацією маршруту й метрик здоров'я через MapKit і HealthKit, переглядом історії та статистики тренувань у вигляді таблиць і графіків, участю в рейтинговій таблиці лідерів, налаштуванням push-сповіщень і редагуванням профілю (особисті дані та одиниці вимірювання). На основі цього аналізу була побудована діаграма варіантів використання (рис. 2.1), яка візуалізує функціональні можливості додатку та зв'язки між Користувачем і прецедентами, гарантуючи узгоджене розуміння меж і завдань розроблюваної системи.

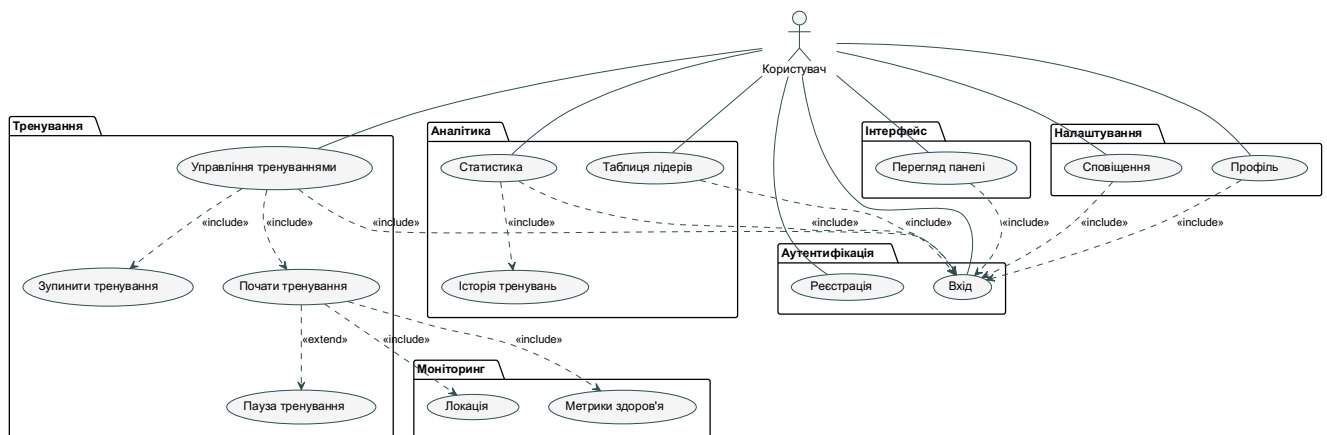


Рисунок 2.1 – Діаграма використання

У табл. 2.1–2.7 наведено деталізований опис прецедентів використання, де в колонці «Назва варіанту використання» подано найменування сценарію (наприклад, «Реєстрація та автентифікація», «Управління тренуваннями», «Перегляд статистики та історії тренувань», «Таблиця лідерів», «Робота з профілем», «Налаштування сповіщень»), а в колонці «Короткий опис» — стисла

		Талько А.Ю.			ІПЗ.КР.Б-121-25-ПЗ	Арк.
		Локтікова Т.М.				19
Змн.	Арк.	№ докум.	Підпис	Дата		

характеристика функціоналу й умов взаємодії користувача із застосунком. Зокрема, табл. 2.1 описує етап створення облікового запису та входу в систему, табл. 2.2 — механізми запуску, паузи та завершення тренування з відстеженням маршруту й метрик здоров'я, табл. 2.3 — відображення зведених даних попередніх сесій у вигляді графіків і таблиць, табл. 2.4 — соціальну функцію рейтингу користувачів, табл. 2.5 і 2.6 — керування персональними даними та доступ до власної статистики, а табл. 2.7 — налаштування умов і часу надсилання push-сповіщень.

Таблиця 2.1

Опис прецеденту «Реєстрація та автентифікація»

Назва варіанту використання	Реєстрація та автентифікація
Короткий опис	Користувач має можливість створити обліковий запис, вказавши адресу електронної пошти та пароль. Після реєстрації можна здійснити вхід до системи. Під час автентифікації перевіряється правильність введених даних, після чого надається доступ до персоналізованих функцій застосунку.

Таблиця 2.2

Опис прецеденту «Управління тренуваннями»

Назва варіанту використання	Реєстрація та автентифікація
Короткий опис	Реалізовано можливість запуску, зупинки та паузи тренування. Додатково можна вказати локацію, на якій фіксуватиметься маршрут, а також збирати дані про фізичний стан користувача через інтеграцію з HealthKit.

Таблиця 2.3

Опис прецеденту «Перегляд статистики та історії тренувань»

Назва варіанту використання	Реєстрація та автентифікація
Короткий опис	Користувач може переглядати зведену інформацію про попередні тренування, включно з тривалістю, маршрутом, кількістю кроків та іншими показниками. Дані відображаються у формі графіків або таблиць для зручного аналізу.

Таблиця 2.4

Опис прецеденту «Таблиця лідерів»

Назва варіанту використання	Реєстрація та автентифікація
Короткий опис	Забезпечено можливість перегляду позиції користувача серед інших учасників у межах таблиці лідерів. Це слугує інструментом гейміфікації та сприяє змагальності між користувачами.

Таблиця 2.5

Опис прецеденту «Робота з профілем»

Назва варіанту використання	Реєстрація та автентифікація
Короткий опис	Користувач має змогу переглядати та змінювати особисті дані, зокрема ім'я, аватар, пароль тощо. Також реалізовано доступ до індивідуальної статистики активності.

Таблиця 2.6

Опис прецеденту «Робота з профілем»

Назва варіанту використання	Реєстрація та автентифікація
Короткий опис	Користувач має змогу переглядати та змінювати особисті дані, зокрема ім'я, аватар, пароль тощо. Також реалізовано доступ до індивідуальної статистики активності.

Таблиця 2.7

Опис прецеденту «Налаштування сповіщень»

Назва варіанту використання	Налаштування сповіщень
Короткий опис	Реалізовано можливість керування push-сповіщеннями: їх увімкнення/вимкнення, налаштування умов, часу надсилання тощо. Це підвищує персоналізацію застосунку та сприяє підтримці щоденної активності користувача.

Розглянемо функціональні вимоги до застосунку.

Застосунок має забезпечувати надійний механізм автентифікації та реєстрації користувачів. Створення облікового запису здійснюється за допомогою електронної пошти та пароля через Firebase Authentication. Користувачі повинні мати змогу входити в обліковий запис, виходити з нього, а також змінювати дані. Усі процеси автентифікації мають відповідати вимогам безпеки та не зберігати облікові дані у відкритому вигляді.

Система має реалізовувати функціональність керування тренуваннями, включаючи запуск, призупинення, відновлення та завершення сесій.

Трекінг маршруту тренування реалізується з використанням MapKit і GPS, що дозволяє фіксувати координати користувача в реальному часі. Додатково система повинна бути інтегрована з Apple HealthKit для збору показників фізичної

		Талько А.Ю.			ІПЗ.КР.Б-121-25-ІЗ	Арк.
		Локтікова Т.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		22

активності, зокрема кількості кроків, пульсу та витрачених калорій. Дані мають синхронізуватися безперервно протягом усієї сесії тренування.

Історія тренувань зберігається з фіксацією ключових метрик, включаючи дату, тривалість, пройдено відстань, кількість кроків та спалені калорії. Ця інформація повинна бути доступною у вигляді списку і графічних візуалізацій, що дозволяє користувачеві оцінювати власний прогрес.

Функціональність таблиці лідерів забезпечує ранжування користувачів за такими критеріями, як кількість кроків. Система повинна формувати рейтинг у режимі реального часу, використовуючи актуальні дані.

Профіль користувача повинен містити можливість перегляду та редагування особистої інформації, включаючи ім'я, електронну адресу та пароль.

Система має підтримувати нативні push-сповіщення, які користувач може налаштовувати самостійно. Передбачено конфігурацію параметрів надсилання нагадувань за часом. Це сприяє підтримці регулярності занять і загальної залученості користувача.

Ключовим елементом гейміфікації є віртуальна квітка, зовнішній вигляд якої змінюється залежно від денної активності користувача. Цей механізм повинен реагувати на досягнення цілей (наприклад, 10 000 кроків на день), забезпечуючи візуальний зворотний зв'язок і підвищуючи мотивацію.

Таким чином, система має поєднувати функції трекінгу активності, аналітики, соціальної взаємодії та гейміфікації в межах захищеного, персоналізованого мобільного середовища.

Нефункціональні вимоги до проєкту охоплюють аспекти продуктивності, безпеки, зручності використання, надійності, масштабованості, сумісності, обслуговування, енергоефективності, локалізації та підтримки офлайн-режиму.

Щодо продуктивності, застосунок повинен запускатися не довше ніж за 3 секунди на пристроях дворічної давності. Відгук інтерфейсу користувача має залишатися в межах 500 мілісекунд, оновлення метрик тренування (відстані, калорій тощо) повинно здійснюватися у реальному часі із затримкою не більше 1

		Талько А.Ю.			ІПЗ.КР.Б-121-25-ІЗ	Арк.
		Локтікова Т.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		23

секунди. Збереження даних про тренування має тривати не більше 2 секунд, а завантаження історії тренувань за останній місяць — не перевищувати 3 секунд.

У сфері безпеки передбачено обов'язкове зберігання облікових даних у зашифрованому вигляді. Доступ до функціональності застосунку можливий виключно після авторизації користувача.

Зручність використання забезпечується дотриманням гайдлайнів платформи iOS, а також наявністю усіх основних дій у межах трьох дотиків. Інтерфейс повинен бути логічно структурованим і демонструвати консистентну поведінку на всіх екранах.

Надійність системи гарантується доступністю сервісів на рівні не менше ніж 99.5% часу, регулярним (кожні 30 секунд) автоматичним збереженням проміжних даних під час тренування, а також стійкістю до помилок, пов'язаних із GPS-сигналом або недоступністю зовнішніх сервісів.

З точки зору масштабованості, система повинна підтримувати до 100 000 активних користувачів та можливість обробки до 10 000 одночасних тренувань. Архітектура застосунку має бути спроектована з урахуванням можливості подальшого розширення функціоналу без суттєвого втручання у базову структуру.

У частині сумісності система має працювати на пристроях з iOS версії 16 і новіших, адаптуватися до різних розмірів екранів - від iPhone SE до iPhone Pro Max - та інтегруватися з Apple HealthKit.

Питання обслуговування вирішується завдяки модульній архітектурі, яка дозволяє легко тестувати та оновлювати окремі компоненти, наявності детального логування для моніторингу поведінки системи та користувачів, а також можливості випускати оновлення без втрати користувацьких даних.

Енергоефективність застосунку повинна гарантувати споживання не більше 10% заряду батареї на годину активного використання під час тренування. GPS має використовуватись з адаптивною частотою оновлення, а також передбачено енергозберігаючий режим для тривалих тренувань.

Система повинна бути локалізована із урахуванням регіональних форматів дати та часу.

		Талько А.Ю.			ІПЗ.КР.Б-121-25-ІЗ	Арк.
		Локтікова Т.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		24

Нарешті, застосунок має підтримувати повну функціональність відстеження тренування в офлайн-режимі, з локальним збереженням необхідних даних для основних функцій і автоматичною синхронізацією після відновлення підключення до інтернету.

Системні вимоги до проєкту охоплюють як клієнтську частину (вимоги до пристроїв користувачів), так і серверну інфраструктуру, засновану на хмарних сервісах.

На стороні клієнта застосунок має працювати на пристроях під управлінням операційної системи iOS 16 або новішої. Підтримуються моделі iPhone, починаючи з iPhone SE (2-го покоління) і вище. Для встановлення застосунку та зберігання локальних даних необхідно мати щонайменше 200 МБ вільної пам'яті. Пристрій повинен бути обладнаний GPS-модулем для забезпечення трекінгу маршруту, а також підтримувати інтеграцію з Apple HealthKit для збору показників активності, таких як кроки, пульс та калорії. Стабільне інтернет-з'єднання не є обов'язковим для базової функціональності, однак необхідне для реєстрації, входу, синхронізації даних та перегляду таблиці лідерів. Оскільки під час активного тренування застосунок використовує GPS та аналітичні функції, рекомендується мати не менше 20% заряду акумулятора або доступ до зовнішнього джерела живлення. Інтерфейс застосунку адаптований під різні розміри екранів, зокрема від моделей iPhone SE до iPhone Pro Max, та підтримує динамічну зміну шрифтів (Dynamic Type).

Серверна частина побудована на основі хмарної платформи Firebase. Для автентифікації та керування обліковими записами використовується Firebase Authentication. Дані користувача, зокрема інформація про тренування, профіль та історію, зберігаються в базі даних Firebase Firestore або Realtime Database, з автоматичним масштабуванням та забезпеченням низької затримки обробки запитів. Система повинна мати змогу обслуговувати до 100 000 активних користувачів, а також підтримувати обробку до 10 000 одночасних сесій тренувань без зниження продуктивності. Архітектура застосунку повинна передбачати можливість подальшого розширення функціональності без необхідності внесення

		Талько А.Ю.			ІПЗ.КР.Б-121-25-ІПЗ	Арк.
		Локтікова Т.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		25

суттєвих змін у її базову структуру. Для забезпечення сумісності проєкт реалізується з урахуванням інтеграції з HealthKit і адаптується до актуальних вимог iOS-платформи.

З точки зору обслуговування, система розробляється з модульною архітектурою, що полегшує оновлення, масштабування та тестування окремих компонентів. Передбачається наявність механізмів детального логування, які дозволяють аналізувати поведінку користувача та ефективно виявляти проблеми. Усі оновлення мають впроваджуватися без втрати користувацьких даних.

Щодо енергоефективності, застосунок не повинен споживати більше 10% заряду акумулятора на годину активного тренування. Для цього передбачене адаптивне використання GPS із регульованою частотою оновлення, а також спеціальний енергозберігаючий режим, що активується під час тривалих сесій.

Система повинна бути локалізована відповідно до регіональних форматів дати та часу, а також підтримувати повноцінну офлайн-роботу. Відстеження тренувань, збереження проміжних результатів і доступ до базового функціоналу мають бути можливими без підключення до інтернету, з автоматичною синхронізацією даних після відновлення мережевого з'єднання.

Для представлення логічної структури застосунку та взаємозв'язків між його основними сутностями використовується діаграма класів (рис. 2.2).

Головний координатор MainCoordinator відповідає за вивантаження початкового вікна й ініціалізацію головного таб-бару, делегуючи подальшу навігацію спеціалізованим підкоординаторам. TabBarCoordinator налаштовує UITabBarController, створює чотири незалежні координатори Home, Statistic, Leaderboard та Settings і забезпечує їх синхронізоване взаємодію в рамках вкладеного UINavigationController. Кожен підкоординатор власноруч формує свій кореневий SwiftUI-екран, обгорнутий у NavigationCoordinatorView, і реалізує конкретні сценарії навігації: від вибору типу тренування та запуску сесії до демонстрації історії активностей або екрану налаштувань користувача.

		Талько А.Ю.			ІПЗ.КР.Б-121-25-ІЗ	Арк.
		Локтікова Т.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		26

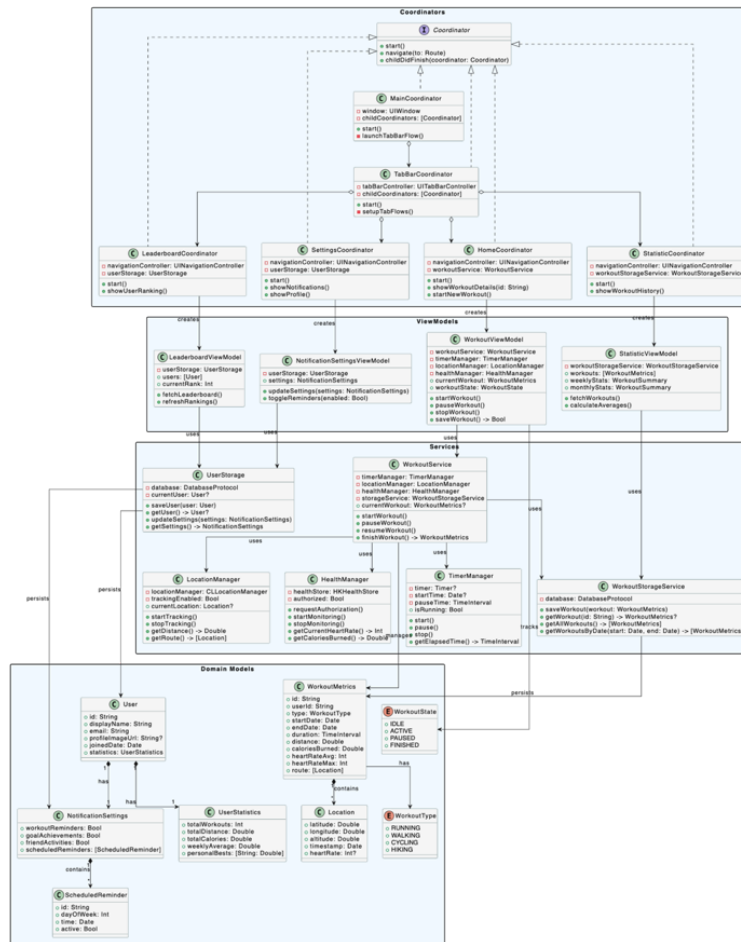


Рисунок 2.2 – UML діаграма класів

Шар представлення представлено набором SwiftUI-компонентів — HomeView, WorkoutSelectionView, RecordWorkoutView, StatisticView, StatisticDetailView, LeaderboardView, ProfileView, NotificationSettingsView тощо. Кожний екран зв'язаний із відповідною ViewModel, яка виступає адаптером між UI та бізнес-логікою. ViewModel використовують властивості з атрибутами @Published або @Bindable, що забезпечують реактивне оновлення інтерфейсу при зміні даних. Прикладом є WorkoutViewModel, який ін'єктується сервісами WorkoutService, TimerManager, LocationManager та HealthManager, містить поточні метрики тренування (WorkoutMetrics) і стан (WorkoutState) та реалізує методи startWorkout(), pauseWorkout(), resumeWorkout(), stopWorkout() із гарантуванням узгодженості даних за допомогою Combine-паблішерів і асинхронних викликів async/await.

		Только А.Ю.			Арк.
		Локтікова Т.М.			ІПЗ.КР.Б-121-25-ІПЗ
Змн.	Арк.	№ докум.	Підпис	Дата	27

Бізнес-логіка інкапсульована в наборі сервісів, котрі виконують взаємодію з платформними API і зовнішніми сховищами. WorkoutService координує роботу таймера (через TimerManager із публікаціями кожної секунди), геолокацію (LocationManager як обгортка над CLLocationManager із фільтрацією точності до 10 метрів), доступ до HealthKit (HealthManager для запиту поточного крокового лічильника, пульсу та розрахунку калорій) і зберігання результатів у WorkoutStorageService. Запуск тренування ініціює зчитування початкової кількості кроків, запуск оновлення маршруту та таймера, а завершення — асинхронний розрахунок різниці кроків, калорій та запис сесії у Firestore з використанням Swift Codable і методів setData()/getDocuments(). AuthService, що реалізує протокол AuthProtocol, використовує FirebaseAuth для входу, реєстрації, оновлення облікових даних та виведення користувача з сесії з підтримкою помилок через резолв Result. CurrentUserService забезпечує централізоване збереження та синхронізацію стану користувача в UserStorage (на UserDefaults або Firestore), тоді як LeaderboardService формує рейтинг активних користувачів за місячними кроками з фільтрацією та сортуванням у Firestore, з обробкою помилок та стейтом завантаження. NotificationService використовує UNUserNotificationCenter для асинхронного запиту авторизації та планування локальних сповіщень за розкладом у NotificationModel.

Шар доступу до даних представлено WorkoutStorageService і ActivityStorageService, які взаємодіють із Firestore через протокол DatabaseProtocol, впроваджують CRUD-операції для WorkoutMetrics і Activity, а також підтримують фільтрацію за датами. UserStorage відповідає за локальне та віддалене зберігання даних профілю та налаштувань (NotificationSettings і ScheduledReminder), використовуючи прототипи, що гарантують атомарність операцій. CachingManager обробляє внутрішні прапори (isFirstAppLoad, isLogged) у UserDefaults, забезпечуючи уникнення дублювання запитів та коректний стан першого запуску.

Доменні моделі включають сутності WorkoutMetrics, User, UserStatistics, NotificationSettings, ScheduledReminder, Location та перелічення WorkoutType і WorkoutState. WorkoutMetrics—value-тип Swift struct із полями id, userId, дати

		Талько А.Ю.			ІПЗ.КР.Б-121-25-ПЗ	Арк.
		Локтікова Т.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		28

старту/фінішу, тривалості, дистанції, калорій, середнього/максимального пульсу й масиву координат Location; User містить profileImageUrl, joinedDate та структуру UserStatistics із накопичувальними показниками. NotificationSettings групує прапорці для різних типів нагадувань і список ScheduledReminder для налаштування повторень за днями тижня.

Взаємодія шарів реалізована через чітко визначені залежності: координатори створюють ViewModel і передають їм потрібні сервіси; ViewModel делегує бізнес-логіку сервісам, які повертають доменні об'єкти; інфраструктурні сервіси зберігають і відновлюють ці дані; UI оновлюється реактивно через Combine та SwiftUI. Така структура забезпечує високу модульність, легкість тестування (юніт-тести для сервісів і ViewModel, інтеграція координаторів), а також масштабованість із можливістю додавання нових екранних флоу чи сервісів без порушення існуючих контрактів між шарами.

2.2 Розробка бази даних

У межах розробки мобільного застосунку для зберігання даних було обрано Firebase Firestore — хмарну NoSQL-базу даних, яка дозволяє зберігати структуровану інформацію у вигляді колекцій та документів із гнучкою схемою. Firestore забезпечує автоматичну синхронізацію між клієнтом і сервером у реальному часі, що є критично важливим для актуалізації статистики активності, таблиці лідерів та інших функцій застосунку.

Основними сутностями у структурі бази даних є тренування користувачів (workouts) та таблиця лідерів (leaderboard). Колекція workouts містить документи з унікальними ідентифікаторами користувачів. У середині кожного з них розміщено вкладену колекцію user_workouts, у якій зберігаються всі сесії активності користувача. Кожне тренування включає ключові поля: кількість кроків (steps), тип тренування (type), тривалість (elapsedTime), дату (date), дистанцію (distance), кількість спалених калорій (calories), середню швидкість (speed) та маршрут (route), представлений масивом GPS-координат. Такий формат дозволяє зберігати

		Талько А.Ю.			ІПЗ.КР.Б-121-25-ІЗ	Арк.
		Локтікова Т.М.				29
Змн.	Арк.	№ докум.	Підпис	Дата		

повноцінні трекінгові дані та згодом візуалізувати маршрути руху користувача на мапі.

Для реалізації гейміфікаційного та змагального функціоналу використовується колекція `leaderboard`, яка організована за місяцями. Кожен місяць (наприклад, 2025-06) є документом, що містить підколекцію `users`. Усередині неї створюються документи з унікальними ID користувачів, у яких зберігається ім'я користувача (`username`), кількість зроблених за місяць кроків (`monthlySteps`), дата останнього оновлення (`lastUpdated`) та мітка періоду (`month`). Така структура забезпечує швидкий доступ до агрегованих статистичних даних, необхідних для побудови таблиці лідерів.

Дані про активність користувача надходять із інтеграції з фреймворком `Apple HealthKit`, який надає доступ до кількості кроків, пройденої дистанції та витраченої енергії. Отримана інформація обробляється на клієнтському пристрої та надсилається до `Firestore` лише за умови авторизації користувача і надання відповідних дозволів. Для уникнення дублювання записів використовується дата фіксації, що дозволяє контролювати період оновлення кожної сесії.

З точки зору організації даних, архітектура бази є модульною: усі основні сутності, зокрема історія тренувань та щомісячна активність, зберігаються окремо, проте мають спільні зв'язки через унікальні ідентифікатори користувачів. Така побудова полегшує масштабування, аналітику та підтримку системи. При цьому кожна колекція та підколекція чітко відповідає окремому функціональному компоненту застосунку — статистиці, гейміфікації чи трекінгу.

Для забезпечення безпеки даних застосовуються правила доступу `Firestore`, які гарантують, що користувач має доступ лише до власних документів. Зміни в базі можливі виключно після автентифікації. Доступ до даних `HealthKit` контролюється на рівні операційної системи `iOS`, що відповідає вимогам до конфіденційності.

Таким чином, обрана структура бази даних `Firestore` забезпечує гнучкість, масштабованість і захищеність при зберіганні та обробці даних, що є важливою

		Талько А.Ю.			ІПЗ.КР.Б-121-25-ІЗ	Арк.
		Локтікова Т.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		30

умовою для безперебійної роботи мобільного застосунку та довіри з боку користувачів.

Опис кожної з колекцій бази даних наведений у табл. 2.8 – 2.9.

Таблиця 2.8

Структура колекції user_workouts

Поле	Тип	Призначення
id	STRING (UUID)	Унікальний ідентифікатор користувача (використовується як ім'я документа)
username	STRING	Ім'я користувача, відображається в таблиці лідерів
month	STRING (формат YYYY-MM)	Період, за який зберігається статистика активності користувача
monthlySteps	INTEGER	Загальна кількість кроків, зроблених користувачем протягом поточного місяця
lastUpdated	TIMESTAMP	Дата і час останнього оновлення статистики

Таблиця 2.9

Структура колекції users

Поле	Тип	Призначення
id	STRING (UUID)	Унікальний ідентифікатор сесії тренування (іменування документа)

date	TIMESTAMP	Дата та час початку тренування
steps	INTEGER	Кількість кроків, виконаних під час тренування
distance	FLOAT	Пройдена дистанція (у метрах)
calories	INTEGER	Орієнтовна кількість витрачених калорій
elapsedTime	FLOAT	Тривалість тренування у секундах
speed	FLOAT	Середня швидкість руху (м/с)
type	STRING	Тип тренування (наприклад: "running", "walking")
route	ARRAY	Масив GPS-координат маршруту у форматі [{latitude: FLOAT, longitude: FLOAT}]

2.3 Проєктування та реалізація алгоритмів роботи застосунку

Функціонування мобільного застосунку побудоване на послідовній взаємодії низки внутрішніх механізмів, які охоплюють повний цикл роботи користувача із системою — від моменту першого запуску до збирання та обробки даних, формування результатів тренувань, синхронізації з хмарними сервісами, управління віртуальним об'єктом (квіткою) та персоналізованої взаємодії через сповіщення. Ключовими принципами, що лежать в основі архітектури, є модульність, реактивність, оптимізація продуктивності та підтримка високого рівня користувацького досвіду. Уся логіка реалізована відповідно до архітектурного підходу MVVM з Coordinator-патерном, що забезпечує чітке розділення відповідальностей між шарами застосунку, спрощує тестування та

		Талько А.Ю.			ІПЗ.КР.Б-121-25-ІПЗ	Арк.
		Локтікова Т.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		32

підтримку коду, а також дозволяє безболісно масштабувати функціональність у майбутньому.

Перший запуск застосунку супроводжується запуском анбордингу — серії вступних екранів, які знайомлять користувача з метою, принципами роботи та концепцією гейміфікованої квітки-домашнього улюбленця. Цей блок виконує не лише інформативну, але й мотиваційну функцію: користувач розуміє, як його фізична активність впливатиме на зовнішній вигляд квітки та прогрес у застосунку. Після завершення анбордингу відбувається перенаправлення на екран реєстрації, де користувач створює обліковий запис за допомогою електронної пошти та пароля. Цей процес реалізовано через Firebase Authentication, з обов'язковим підтвердженням пошти, що додає рівень валідації та безпеки. Лише після підтвердження поштової адреси користувач отримує доступ до функціоналу програми.

Після автентифікації застосунок ініціалізує профіль користувача: виконується підключення до Apple HealthKit (за згодою), завантаження даних з Firestore (включно з історією тренувань, параметрами квітки, рейтингами) та налаштування внутрішніх сервісів. Важливу роль відіграє менеджер локацій — спеціалізований компонент, що відповідає за GPS-трекінг, фільтрацію координат, обробку сигналу у фоновому режимі та адаптацію частоти оновлень залежно від активності. Він реалізує точну фільтрацію сигналів на основі точності та швидкості зміни координат, виключаючи потенційно хибні дані.

Коли користувач натискає кнопку старту тренування, активується WorkoutService — модуль, який керує станом тренування ('idle', 'active', 'paused') та ініціалізує збір метрик. З цього моменту застосунок запускає таймер, зберігає початкове місцезнаходження, активує збір кроків, калорій та відстані через HealthKit, а також запускає запис маршруту. Локації збираються у реальному часі за допомогою Combine, обробляються кожні 0.5 секунди та одразу відображаються на карті, оновлюючи дистанцію та швидкість. Фільтрація GPS-даних відбувається на основі 'horizontalAccuracy', і лише координати з точністю ≤ 10 м додаються до маршруту. Паралельно з цим таймерний менеджер підраховує час, що минув, а

		Талько А.Ю.			ІПЗ.КР.Б-121-25-ІПЗ	Арк.
		Локтікова Т.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		33

логіка розрахунку швидкості та витрати калорій відбувається у кожному оновленні стану. При натисканні паузи таймер та GPS-збір призупиняються, а всі дані залишаються збереженими в оперативній пам'яті. Завершення тренування ініціює серіалізацію зібраних даних, які включають маршрут, дистанцію, кроки, час, тип активності та кількість витрачених калорій. Ця інформація зберігається у Firebase Firestore в окремій колекції `user\_workouts`, прив'язаній до конкретного ідентифікатора користувача.

Після завершення тренування також запускається механізм синхронізації з таблицею лідерів. Алгоритм автоматично обчислює місячну суму кроків та оновлює/створює запис у структурі `leaderboard/{місяць}/users`, що дозволяє порівнювати результати з іншими користувачами. Це формує основний мотиваційний драйвер, який поєднує індивідуальну активність із елементами змагання. Перегляд історії тренувань реалізовано через запити до бази Firestore, де застосунок витягує список попередніх тренувань, відображаючи дані у вигляді списку, графіків або карт із маршрутами. Додатково користувач має можливість фільтрувати сесії за типом, датою, тривалістю та іншими параметрами.

Окрему роль відіграє система гейміфікації, візуально уособлена у вигляді віртуальної квітки. Її стан залежить від щоденної активності користувача: якщо кількість кроків за день не перевищує 3000, здоров'я квітки погіршується, а зовнішній вигляд змінюється на в'янучий. Помірна активність (3000–6000 кроків) підтримує квітку в гарному стані, а висока активність (понад 6000 кроків) приносить бонусне зростання. Усі зміни відбуваються в реальному часі, на основі локальних або синхронізованих з HealthKit даних. Квітка має різні скіни та анімації, які змінюються залежно від настрою, здоров'я та прогресу. Анімації реалізовано через алгоритми процедурної генерації — пелюстки рухаються із застосуванням фізики пружин, мають випадкові кольори та позиції, а оновлення відбувається плавно через механізми SwiftUI та `'drawingGroup()'` для оптимізації рендерингу.

Ще однією важливою частиною є система сповіщень, яка реалізована через Firebase Cloud Messaging. Повідомлення можуть мати нагадувальний, мотиваційний або інформативний характер: початок нового дня, відсутність

		Талько А.Ю.			ІПЗ.КР.Б-121-25-ІПЗ	Арк.
		Локтікова Т.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		34

активності протягом тривалого часу, досягнення певних цілей або змін у рейтингу. Користувач має змогу вмикати/вимикати окремі типи повідомлень та обирати бажану частоту нагадувань.

Для підтримки плавної роботи та енергоефективності в системі реалізовано кілька механізмів оптимізації. Використовується адаптивна частота оновлення GPS, обмеження обсягу фонових операцій, кешування даних у 'UserDefaults', розділення важких обчислень у фонові потоки та обробка помилок зовнішніх сервісів (наприклад, у випадку втрати сигналу або недоступності HealthKit). Усі дані, що зберігаються локально, шифруються та захищаються відповідно до стандартів iOS, а обмін із Firebase відбувається через захищене з'єднання.

У підсумку, алгоритмічна логіка застосунку є комплексною, розгалуженою та узгодженою: вона охоплює не лише основний функціонал трекінгу, але й реалізацію повного життєвого циклу користувача, багатий візуальний інтерфейс, гейміфіковану мотивацію, систему рейтингу та механізми персоналізованої взаємодії. Усі ці компоненти поєднуються в цілісну систему, що стимулює фізичну активність через зв'язок із віртуальним об'єктом, що перетворює звичайне тренування на приємну щоденну рутину.

2.4 Реалізація функціоналу застосунку

На рис. 2.3 зображено діаграму взаємодії компонентів системи.

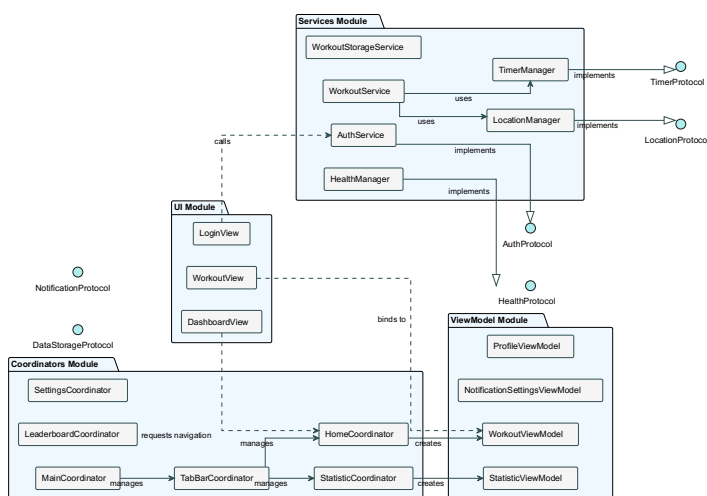


Рисунок 2.3 – Діаграма взаємодії компонентів

У правому нижньому куті знаходиться ViewModel Module, де зібрані класи ProfileViewModel (додаток А), NotificationSettingsViewModel (додаток Б), WorkoutViewModel (додаток В) та StatisticViewModel (додаток Г). Діаграма вказує на те, що кожен координатор створює екземпляри свого ViewModel: HomeCoordinator — WorkoutViewModel, StatisticCoordinator — StatisticViewModel і так далі. Стрілки від Coordinators Module до ViewModel Module означають передачу контролю і залежностей: координатори ініціалізують ViewModels і передають їм контекстні дані.

Вгорі праворуч розташований Services Module, що містить класи, які інкапсулюють бізнес-логіку й комунікацію із зовнішніми сервісами: WorkoutService та WorkoutStorageService для обробки та збереження метрик тренування, TimerManager (додаток Д) і LocationManager (додаток Е) для відліку часу і геолокації, AuthService для авторизації, HealthManager для роботи з HealthKit. Стрілки між цими сервісами показують, що WorkoutService використовує TimerManager та LocationManager, а AuthService й HealthManager реалізують відповідні інтерфейси, які можуть споживатися іншими модулями.

Нарешті, у верхньому лівому куті розміщено UI Module з компонентами LoginView, WorkoutView та DashboardView. Штрихові стрілки від UI Module до Coordinators Module свідчать про те, що події інтерфейсу користувача (натискання кнопок, вибір табу) передаються безпосередньо в координатори, які вирішують, які подальші кроки виконати. Аналогічно, штрихові стрілки від UI Module до ViewModel Module позначають двонаправлену зв'язок: екран прив'язаний до свого ViewModel, споживаючи його дані та реагуючи на їх зміну.

Загалом ця діаграма розгортання відображає архітектурний розподіл компонентів за модулями, ілюструючи, як високорівневі вузли (координатори та UI) взаємодіють із шаром представлення (ViewModels), а ті, у свою чергу, спираються на сервісний шар для виконання бізнес-логіки й доступу до зовнішніх API та сховищ. Такий підхід гарантує слабку зв'язаність між компонентами, полегшує модульне тестування та подальшу еволюцію окремих підсистем.

		Талько А.Ю.			ІПЗ.КР.Б-121-25-ПЗ	Арк.
		Локтікова Т.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		36

Реалізація функціоналу застосунку спирається на чіткий поділ відповідальностей між чотирма модулями (Coordinators, UI, ViewModel і Services), що ілюструє рис. 2.5. Після успішного входу через AuthService та MainCoordinator користувач потрапляє на DashboardView, де за допомогою TabBarCoordinator (додаток Є) обирає потрібний розділ – Workout, Statistics або Profile. На екрані WorkoutView він спочатку вибирає тип активності (ходьба, біг або їзда на велосипеді), після чого HomeCoordinator створює відповідний WorkoutViewModel і передає йому сервіс WorkoutService із ін'єкцією TimerManager та LocationManager.

WorkoutService налаштовано таким чином, що для ходьби, бігу та велопрогулянки використовуються різні режими трекінгу: частота оновлення GPS координат та фільтрація шуму підбираються відповідно до швидкості руху, а HealthManager збирає додаткові метрики (пульс, калорії). ViewModel накопичує ці дані в доменну модель WorkoutMetrics і передає їх у WorkoutStorageService для локального кешування та подальшої синхронізації з Firebase.

Після завершення сесії WorkoutViewModel викликає метод completeWorkout(), у якому агреговані метрики надходять у Firestore через Cloud Functions, а користувач автоматично повертається на Dashboard із показом підсумків — відстані, часу, середнього пульсу й динамічної візуалізації прогресу через анімовану квітку. Паралельно NotificationSettingsViewModel (додаток Ж) забезпечує налаштування нагадувань про початок тренувань і повідомлень про досягнення цілей, використовуючи Apple Push Notifications. Такий підхід гарантує прозору й гнучку реалізацію всього функціоналу з підтримкою декількох видів активності та персональними підписками на біг і вело-тренування.

Висновки до другого розділу

У другому розділі було розглянуто логічну структуру мобільного застосунку та здійснено проектування його основних функціональних складових. Зокрема, було сформовано модель варіантів використання, яка відображає ключові сценарії взаємодії користувача із системою, включаючи реєстрацію, автентифікацію,

		Талько А.Ю.			ІПЗ.КР.Б-121-25-ПЗ	Арк.
		Локтікова Т.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		37

управління тренуваннями, перегляд статистики, роботу з профілем, взаємодію з таблицею лідерів, гейміфіковану систему віртуальної квітки та налаштування персоналізованих сповіщень. На основі діаграм і таблиць описано логіку реалізації кожного з варіантів, що дозволило закласти чітку структурну основу майбутньої реалізації.

Особливу увагу приділено алгоритмам роботи застосунку, зокрема механізмам запуску та завершення тренування, збору метрик через HealthKit, побудові маршруту за GPS-координатами з фільтрацією хибних значень, збереженню результатів у Firebase Firestore та актуалізації лідерборду. Реалізована модель підтримує інтерактивну гейміфікацію: стан віртуальної квітки змінюється залежно від активності користувача, що посилює мотивацію та користувацьку залученість. Окремо розглянуто процес першого запуску та onboarding-послідовність, яка знайомить користувача з основною ідеєю застосунку.

Також було здійснено проектування структури бази даних, що передбачає використання гнучкої NoSQL-схеми на основі Firebase Firestore. Описані сутності та поля забезпечують ефективне збереження історії тренувань і щомісячної статистики кроків для рейтингу. Додатково проведено аналіз компонентної архітектури застосунку, яка реалізована на основі патерну MVVM-C з розподілом відповідальностей між UI, ViewModel, Coordinator та сервісними модулями, що підвищує масштабованість і стабільність системи. Побудована діаграма взаємодії компонентів демонструє взаємозв'язки між шарами та ілюструє загальну структуру проекту.

Таким чином, у межах другого розділу було комплексно спроектовано логічну архітектуру, структуру даних, алгоритмічну базу та внутрішню взаємодію компонентів застосунку, що стало основою для його подальшої технічної реалізації.

		Талько А.Ю.			ІПЗ.КР.Б-121-25-ІЗ	Арк.
		Локтікова Т.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		38

**РОЗДІЛ 3 ІНТЕРФЕЙС КОРИСТУВАЧА ТА ПОРЯДОК РОБОТИ ІЗ
ЗАСТОСУНКОМ З ВІДСТЕЖЕННЯ ФІЗИЧНОЇ АКТИВНОСТІ ЛЮДИНИ**

3.1 Порядок встановлення та налаштування параметрів застосунку

Розгортання програмного комплексу мобільного застосунку передбачає налаштування клієнтської частини, хмарної серверної інфраструктури та зовнішніх сервісів, необхідних для коректної взаємодії між компонентами системи. Важливими аспектами є забезпечення безпечної автентифікації, надійного зберігання даних про фізичну активність, а також підтримка механізмів синхронізації, сповіщень і гейміфікації. Усі ці компоненти повинні працювати узгоджено, забезпечуючи безперебійну роботу мобільного застосунку в реальному часі. Архітектура розгортання, що відображає фізичне розміщення основних елементів системи та характер їхніх взаємозв'язків, наведена на рис. 3.1.

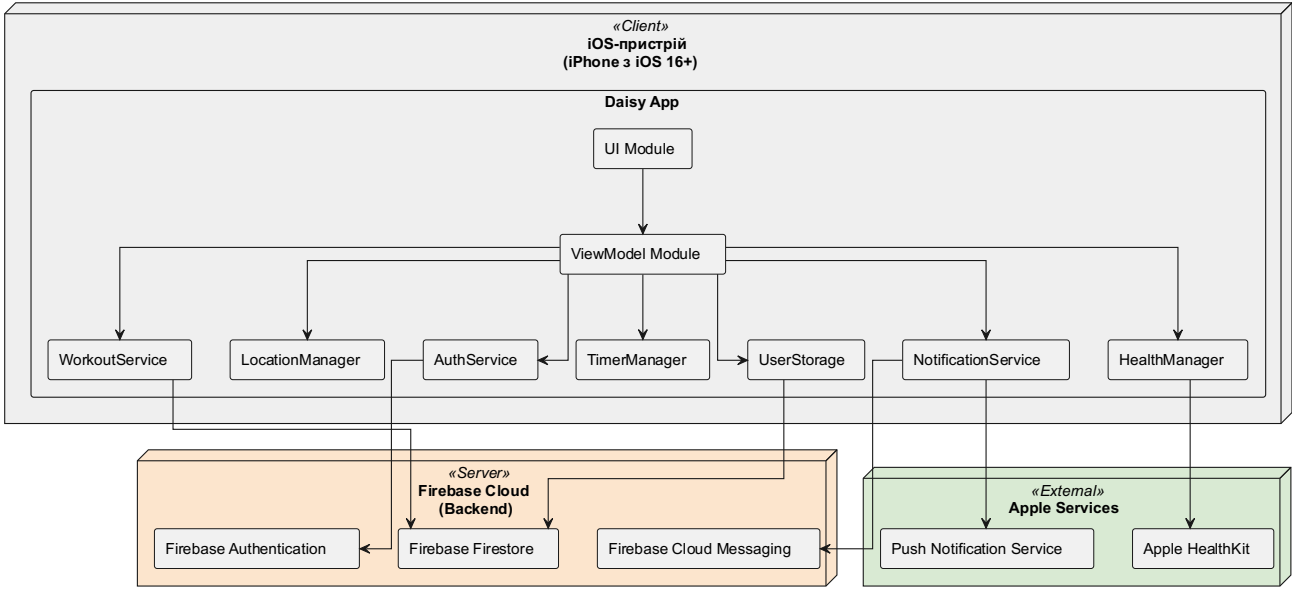


Рисунок 3.1 – Архітектура розгортання застосунку

Архітектура розгортання мобільного застосунку охоплює клієнтську частину, хмарну серверну інфраструктуру та зовнішні сервіси, що забезпечують безперебійну роботу системи. Всі компоненти взаємодіють між собою через безпечні канали зв'язку, гарантуючи збереження, актуальність і захищеність даних користувача.

Клієнтський пристрій представлений мобільним застосунком, що працює на пристроях під управлінням iOS 16 або новішої версії. Для його встановлення використовується App Store, а інтерфейс реалізовано за допомогою SwiftUI з використанням архітектури MVVM-C. Застосунок не потребує локального бекенду, оскільки вся логіка виконання — включаючи обробку тренувань, навігацію, взаємодію з сенсорами (GPS, HealthKit) — реалізована на клієнтському боці через спеціалізовані сервіси: WorkoutService, TimerManager, LocationManager, NotificationService тощо.

Серверна інфраструктура побудована на базі хмарної платформи Firebase, яка виконує роль бекенду застосунку. Сервіс Firebase Authentication обробляє запити реєстрації, входу в обліковий запис та автентифікації користувачів. Firebase Firestore зберігає тренування, статистику, налаштування профілю та інші користувацькі дані в режимі реального часу, синхронізуючи їх між пристроями. Для надсилання push-сповіщень застосовується Firebase Cloud Messaging, який формує та доставляє повідомлення на основі налаштувань користувача.

Зовнішні сервіси Apple виконують роль джерела платформових даних та системної інтеграції. За допомогою Apple HealthKit відбувається збір інформації про кроки, дистанцію, пульс та спалені калорії. Застосунок отримує ці метрики лише за згодою користувача та використовує їх у рамках аналізу фізичної активності. Для доставки повідомлень iOS-система взаємодіє з Push Notification Service, що використовується у поєднанні з FCM для реалізації локальних та віддалених сповіщень.

Комунікація між компонентами здійснюється через захищені канали HTTPS. Дані, зібрані локально, шифруються та зберігаються у Firestore. Всі запити до Firestore та Authentication відбуваються через офіційні SDK, інтегровані у клієнтський застосунок. Завдяки модульній архітектурі застосунок не потребує проміжного API-сервера — уся бізнес-логіка реалізована у ViewModel та сервісах, безпосередньо взаємодіючих з Firebase.

У результаті розгортання система функціонує як повноцінний хмарний застосунок, у якому вся обробка критичних даних (ідентифікація, історія

		Талько А.Ю.			ІПЗ.КР.Б-121-25-ПЗ	Арк.
		Локтікова Т.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		40

тренувань, рейтингова інформація) винесена на серверну частину (Firebase), а обробка фізичної активності та візуалізація — локалізована на мобільному пристрої користувача.

3.2 Загальна структура інтерфейсу користувача

Інтерфейс користувача мобільного застосунку розроблений відповідно до принципів сучасного UX/UI-дизайну з урахуванням рекомендацій Apple Human Interface Guidelines. Основна мета інтерфейсу — забезпечити інтуїтивну, зручну та візуально привабливу взаємодію з функціоналом застосунку, мінімізуючи кількість дій, необхідних для виконання базових операцій.

Першим екраном, який бачить користувач після запуску застосунку, є вітальний онбординг-екран (рис. 3.2). Він слугує початковою точкою знайомства з ідеєю розроблюваного застосунку.



Рисунок 3.2 – Екран онбордингу

Центральне місце займає вітальний текст, який стисло і зрозуміло доносить головну мету застосунку. Візуально текст поділений на заголовок — «Ласкаво

		Талько А.Ю.			ІПЗ.КР.Б-121-25-ПЗ	Арк.
		Локтікова Т.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		41

просимо в Daisy», виділений жирним шрифтом, — та основний опис, який заохочує користувача до активного способу життя.

Фон екрану представлено у вигляді розмитого зображення стилізованих квітів, що задає атмосферу легкості та позитиву й підтримує загальну концепцію застосунку, у центрі якого — віртуальна квітка як символ прогресу. Такий підхід створює емоційний зв'язок між користувачем та візуальним образом застосунку.

У нижній частині екрана розміщена помітна кнопка «Продовжити». Вона спрямовує користувача до наступного етапу — реєстрації або входу.

Наступним етапом взаємодії користувача з мобільним застосунком є екран входу в акаунт (рис. 3.3), який дозволяє авторизованим користувачам отримати доступ до персоналізованого функціоналу системи.

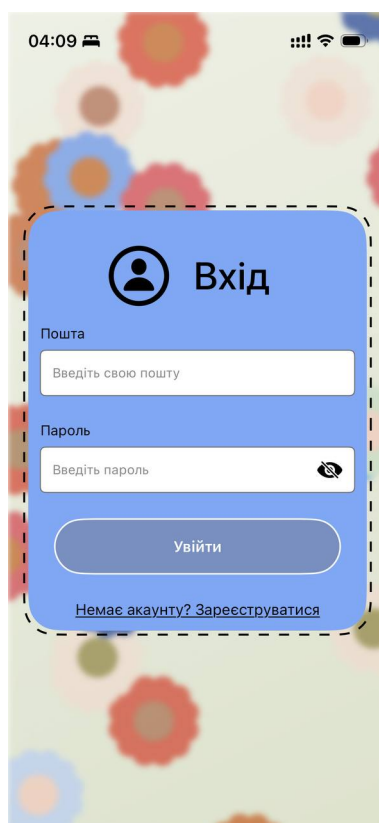


Рисунок 3.3 – Екран входу

Дизайн цього екрана зберігає візуальну цілісність з попереднім (вітальним) екраном – фонове зображення із квітами слугує символом легкості та підтримує загальну тематику застосунку.

Центральний елемент екрана – форма авторизації, оформлена у вигляді напівпрозорої панелі з заокругленими кутами, що розташована по центру. У

		Талько А.Ю.			ІПЗ.КР.Б-121-25-ПЗ	Арк.
		Локтікова Т.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		42

верхній частині форми розміщено іконку користувача та напис «Вхід», які чітко сигналізують про призначення цього екрану.

Форма складається з двох полів введення:

поле "Пошта" – для введення електронної адреси користувача;

поле "Пароль" – із вбудованим перемикачем видимості введеного тексту (іконка ока), що покращує зручність та безпеку при введенні пароля.

Нижче розташована кнопка "Увійти", яка після валідації даних запускає процес автентифікації через сервіс Firebase Authentication. Кнопка стилістично узгоджена з кольоровою палітрою застосунку, має достатній розмір для зручного натискання.

Під формою авторизації користувачеві пропонується альтернатива у вигляді посилання «Немає акаунту? Зареєструватися», яке веде на екран реєстрації (рис. 3.4) нового користувача. Такий підхід забезпечує зручну навігацію між основними етапами роботи з профілем.

Екран логіну демонструє просту, логічну структуру, орієнтовану на мінімальну кількість дій з боку користувача. Він забезпечує швидкий та безпечний вхід до системи, зберігаючи при цьому візуальну привабливість та дружність інтерфейсу.

Рисунок 3.4 – Екран реєстрації

		Талько А.Ю.			ІПЗ.КР.Б-121-25-ПЗ	Арк.
		Локтікова Т.М.				43
Змн.	Арк.	№ докум.	Підпис	Дата		

Головний екран застосунку (рис. 3.5) є центральним елементом взаємодії користувача з системою, що поєднує функціональність трекінгу фізичної активності та візуальне відображення прогресу у гейміфікованій формі. Після успішного входу користувача система автоматично відкриває цей екран, який виконує роль інформаційної панелі. У верхній частині екрана розташовано два ключові віджети, які відображають персоналізовану статистику. Перший блок – «Кроків сьогодні» – показує кількість кроків, зроблених протягом поточного дня, а також прогрес щодо встановленої цілі (наприклад, 10 000 кроків). Порядок виконання цілі візуалізується у вигляді горизонтальної шкали прогресу, що дозволяє користувачу швидко оцінити свою активність. Другий блок — «Пройдена дистанція» — виводить актуальне значення пройденої відстані в метрах на основі синхронізованих даних з Apple HealthKit та GPS. У центрі екрана розміщено стилізовану віртуальну квітку, яка змінює свій вигляд залежно від рівня фізичної активності користувача.

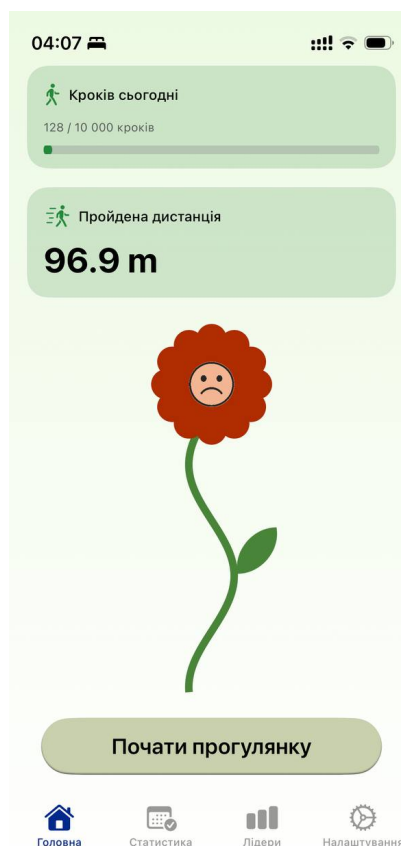


Рисунок 3.5 – Головний екран

		Талько А.Ю.			ІПЗ.КР.Б-121-25-ПЗ	Арк.
		Локтікова Т.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		44

У даному випадку, коли за день зроблено недостатню кількість кроків, квітка має зів'ялий вигляд та сумне обличчя, що символізує потребу в русі. Цей елемент є частиною гейміфікації, яка покликана формувати емоційний зв'язок користувача з застосунком та стимулювати його залишатися активним. Така візуалізація не лише підкреслює важливість регулярної активності, а й делікатно нагадує про необхідність подбати про власне здоров'я без нав'язливих сповіщень.

У нижній частині екрана розташована велика кнопка «Почати прогулянку», яка ініціює запуск сесії тренування. Після натискання активується збір геолокаційних даних, кроків та інших метрик фізичної активності. Цей функціонал реалізує основну ідею застосунку — просте і зручне фіксування щоденних прогулянок, навіть для користувачів без спортивного досвіду чи спеціальної підготовки. Інтерфейс кнопки зроблено максимально помітним та інтуїтивним, що дозволяє швидко розпочати тренування у будь-який момент.

Також у нижній частині екрана реалізовано панель навігації з чотирма вкладками: «Головна», «Статистика», «Лідери» та «Налаштування», що забезпечують швидкий доступ до інших функцій застосунку. Завдяки зручній структурі головний екран є не лише точкою входу для користувача, а й місцем, де поєднується інформативність, естетика та мотиваційна складова. Такий підхід сприяє тому, що користувачеві приємно повертатися до застосунку щодня та стежити за власним прогресом.

Екран вибору типу активності (рис. 3.6) є проміжною ланкою між головним інтерфейсом та безпосереднім запуском сесії тренування, він з'являється після натискання відповідної кнопки. Його мета — надати користувачеві можливість обрати формат фізичної активності, яку він планує виконувати, для коректного збору відповідних метрик та персоналізованого відображення результатів. Такий підхід дозволяє максимально точно врахувати особливості кожного тренування й зробити подальший аналіз ще більш корисним.

У верхній частині екрана розміщено ілюстрацію квітки, що сидить на велосипеді з усміхненим обличчям — це не лише декоративний, а й мотивуючий візуальний елемент, який створює асоціацію з позитивними емоціями від руху.

		Талько А.Ю.			ІПЗ.КР.Б-121-25-ІЗ	Арк.
		Локтікова Т.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		45

Такий графічний прийом підтримує загальну концепцію гейміфікації, додає інтерфейсу емоційної привабливості та надихає користувача до нових досягнень.

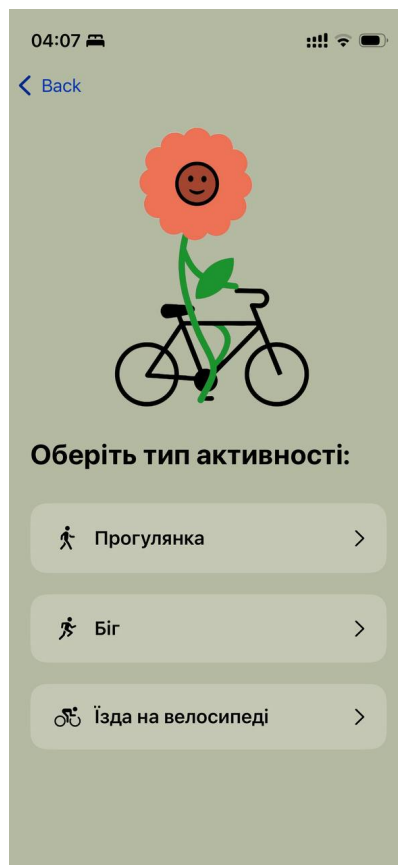


Рисунок 3.6 – Екран вибору активності

Нижче на екрані розміщено заголовок «Оберіть тип активності:», під яким знаходяться три інтерактивні кнопки для вибору формату тренування: «Прогулянка», «Біг» та «Їзда на велосипеді». Кожна кнопка має відповідну іконку, а також стрілку справа, яка підказує користувачу, що можна перейти далі.

Вибір типу активності необхідний для коректного збору фізичних даних, оскільки кожен режим має свої параметри обчислення калорій і відображення прогресу. Окрім цього, вибрана активність впливає на гейміфікаційну частину застосунку — стан квітки та оновлення лідербордів.

Якщо користувач обирає «Біг» або «Їзду на велосипеді» без оформленої підписки, відкривається екран підписки (рис. 3.7), де коротко пояснюються переваги преміум-доступу і пропонується оформити підписку. Лише після підтвердження підписки стає можливим обрати ці типи активностей.

		Талько А.Ю.			ІПЗ.КР.Б-121-25-ПЗ	Арк.
		Локтікова Т.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		46

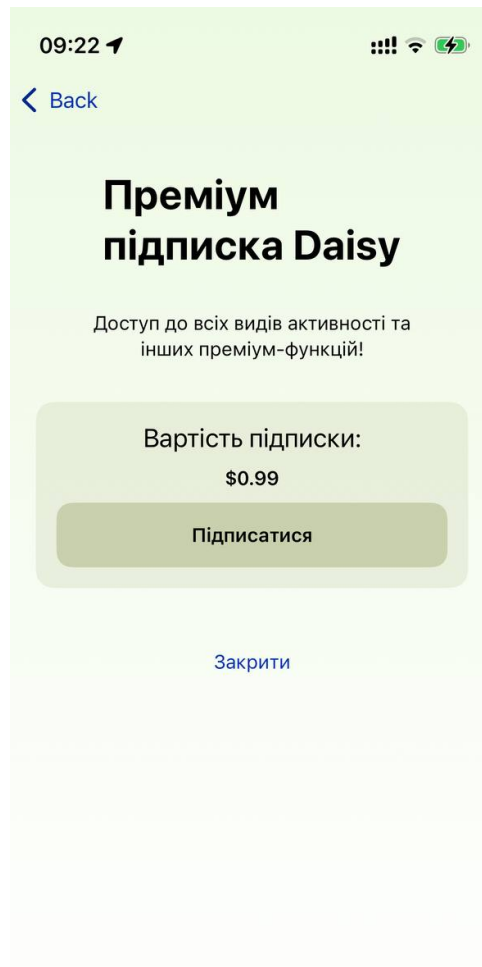


Рисунок 3.7 – Екран вибору активності

Після вибору доступної активності (або оформлення підписки для преміум-режимів) відкривається екран активного тренування. На цьому екрані користувач бачить всю необхідну інформацію про свою сесію в режимі реального часу. Інтерфейс зроблений максимально простим: усі основні показники розташовані на видному місці, а керування сесією інтуїтивно зрозуміле й не відволікає від самого тренування.

Після обрання типу активності відкривається екран активного тренування (рис. 3.8), який є центральним елементом застосунку під час фізичної сесії. Цей екран призначений для надання користувачу повної інформації про хід тренування в режимі реального часу, з акцентом на простоту використання, швидке зчитування даних і легке управління сесією.

		Талько А.Ю.			ІПЗ.КР.Б-121-25-ПЗ	Арк.
		Локтікова Т.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		47

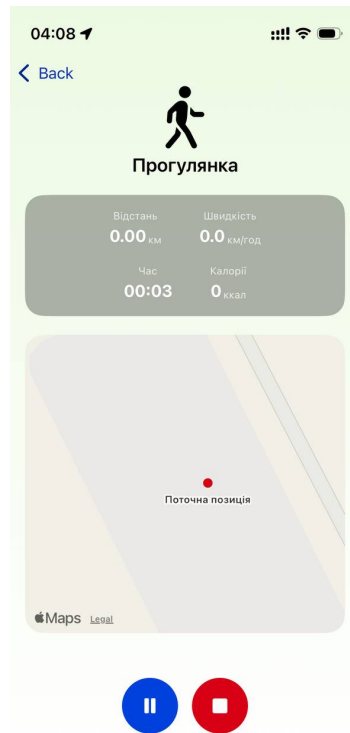


Рисунок 3.8 – Екран активності

У верхній частині інтерфейсу відображається обраний тип активності, наприклад «Прогулянка», разом із відповідною піктограмою. Це дозволяє користувачу одразу ідентифікувати поточний режим тренування, не відволікаючись на зайві деталі. Нижче розташований компактний, але інформативний блок, у якому зібрані основні метрики сесії: загальна пройдена відстань у кілометрах, поточна швидкість руху (в км/год), тривалість тренування (у форматі хвилини: секунди), а також кількість витрачених калорій. Усі ці показники автоматично оновлюються під час тренування, що дає змогу користувачеві в реальному часі контролювати свій прогрес і за потреби коригувати інтенсивність.

Центральне місце на екрані займає інтегрована карта на основі Apple Maps, яка відображає актуальне місцеперебування користувача. Червона точка на карті демонструє його поточну позицію, а весь маршрут фіксується у фоновому режимі з використанням GPS. Завдяки коректній роботі GPS та вбудованих алгоритмів фільтрації координат, карта забезпечує високу точність у відображенні переміщень, що особливо важливо для подальшого аналізу тренувань та перегляду пройденого шляху.

		Талько А.Ю.			ІПЗ.КР.Б-121-25-ПЗ	Арк.
		Локтікова Т.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		48

У нижній частині екрана розміщено елементи керування сесією — дві великі кнопки з чітким кольоровим маркуванням. Синя кнопка «Пауза» дає змогу тимчасово призупинити тренування, а червона кнопка «Стоп» — повністю завершити сесію та зберегти результати. Їхнє розміщення й розмір підібрано таким чином, щоб користувач міг легко натиснути потрібну кнопку навіть під час руху, що робить інтерфейс максимально зручним та інтуїтивно зрозумілим.

Загалом, екран активного тренування вдало поєднує інформативність, функціональність та мінімалістичний дизайн, забезпечуючи користувачеві ефективне відстеження фізичної активності. Особливий акцент зроблено на простоті взаємодії та комфорті під час використання додатка в реальних умовах, незалежно від досвіду користувача.

Наступним елементом панелі навігації є екран «Статистика» (рис. 3.9). Екран «Статистика» у застосунку надає користувачу зручний спосіб переглядати історію тренувань за різні дні, дозволяючи аналізувати власну активність у динаміці. У верхній частині екрана розміщено інтерактивний календар, який відображає поточний місяць і дає змогу швидко обрати будь-яку дату. Дні, коли користувач займався фізичною активністю, виділяються спеціальними кольоровими індикаторами, що одразу сигналізує про наявність тренувань у цей день.

Після вибору конкретної дати під календарем з’являється перелік тренувань, здійснених у цей день. Кожен запис містить детальну інформацію: тип активності (наприклад, «Прогулянка» або «Біг»), подолану дистанцію в кілометрах, а також точний час завершення сесії. Це дає змогу користувачеві легко орієнтуватися в обсязі навантаження за день і відстежувати зміни своїх фізичних показників у довгостроковій перспективі.

Інтерфейс екрана статистики виконано у спокійній кольоровій гамі, що не перевантажує увагу й сприяє зосередженості на вмісті. Завдяки логічній структурі даних і простому, зрозумілому візуальному представленню, цей екран стає для користувача персональним щоденником активності, який не лише допомагає в самоаналізі, а й мотивує до подальшого покращення результатів.

		Талько А.Ю.			ІПЗ.КР.Б-121-25-ПЗ	Арк.
		Локтікова Т.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		49

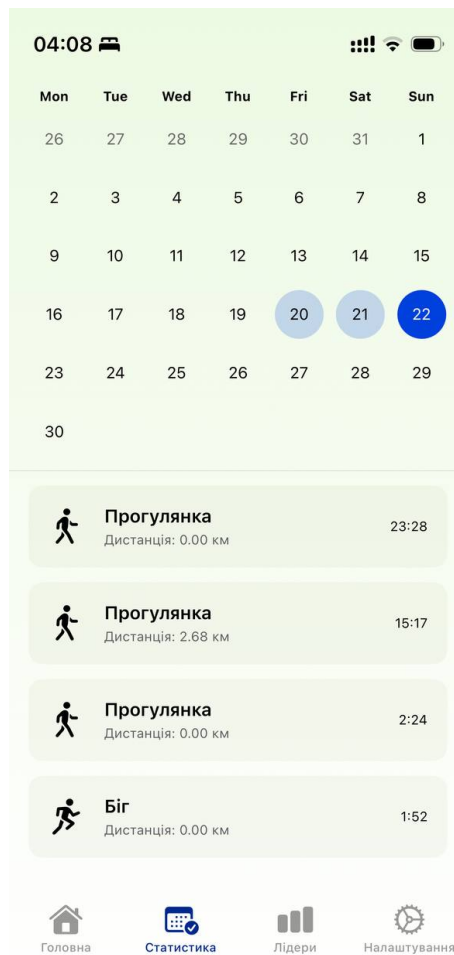


Рисунок 3.9 – Екран статистики

Екран «Лідери місяця» (рис. 3.10) відображає рейтинг користувачів за кількістю кроків, здійснених протягом поточного календарного місяця. Інтерфейс представлений у вигляді таблиці, що містить три основні колонки: ранг (місце в загальному заліку), нікнейм користувача та загальна кількість кроків. Для візуального виділення, кожному з трьох призових місць відповідає спеціальний маркер з кольоровим позначенням (золото, срібло, бронза). Поточний користувач застосунку підсвічується окремим стилем, що дозволяє швидко знайти себе у списку. Такий підхід реалізує елемент змагальності, що позитивно впливає на мотивацію користувачів до фізичної активності та регулярних тренувань.

Окрім того, рейтинг оновлюється автоматично, що стимулює користувачів стежити за своїми результатами і намагатися покращити власні показники. Це створює здорову конкуренцію та допомагає формувати спільноту однодумців, об'єднаних спільною метою — рухатися більше щодня.

		Талько А.Ю.			ІПЗ.КР.Б-121-25-ПЗ	Арк.
		Локтікова Т.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		50

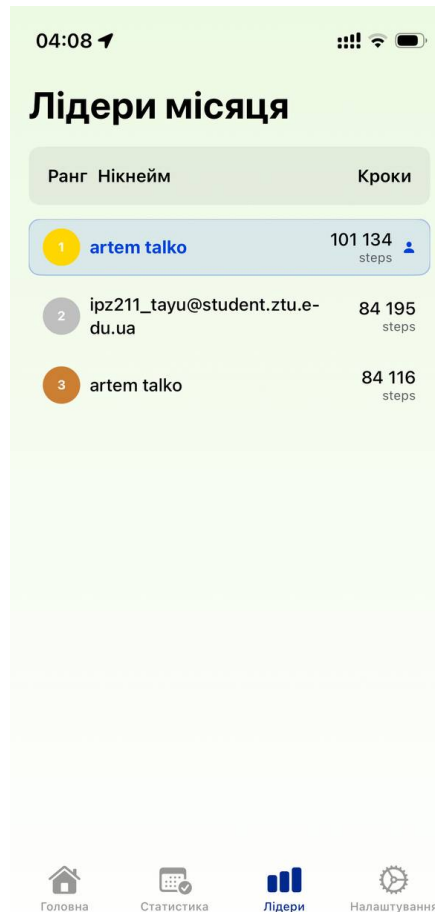


Рисунок 3.10 – Екран «Лідери місяця»

Екран «Налаштування» (рис. 3.11) забезпечує користувачеві доступ до базових функцій персоналізації та керування обліковим записом. У верхній частині екрана розміщені розділи «Профіль» та «Сповіщення», які наразі є активними. У розділі «Профіль» користувач може переглядати й оновлювати свої персональні дані, зокрема нікнейм. Розділ «Сповіщення» дозволяє керувати параметрами push-повідомлень — зокрема, визначати, які типи сповіщень надходитимуть і з якою частотою.

Також інтерфейс налаштувань передбачає подальше розширення, наприклад, додавання нових розділів для зворотного зв'язку чи підтримки користувачів. Продумана структура екрана робить процес керування обліковим записом простим і зрозумілим навіть для новачків, сприяючи персоналізації застосунку під власні потреби.

		Талько А.Ю.			ІПЗ.КР.Б-121-25-ПЗ	Арк.
		Локтікова Т.М.				51
Змн.	Арк.	№ докум.	Підпис	Дата		

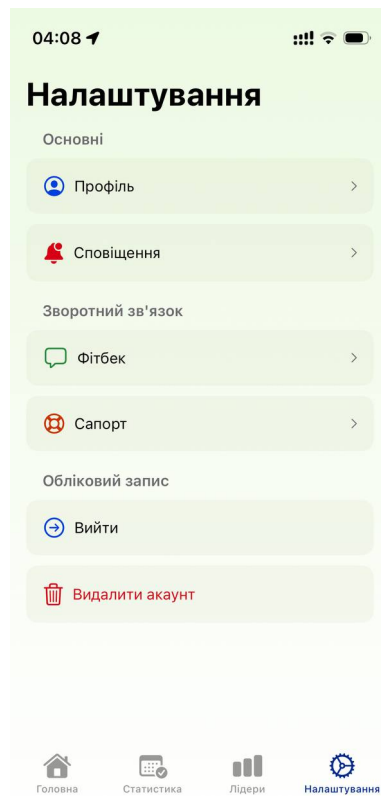


Рисунок 3.11 – Екран «Налаштування»

У секції «Обліковий запис» (рис 3.12) реалізовано дві важливі функції: вихід з акаунту та його повне видалення. Натискання кнопки «Вийти» завершує поточну сесію користувача, очищуючи локальні дані й повертаючи його до екрана входу. Функція «Видалити акаунт» дозволяє повністю видалити обліковий запис користувача разом із пов'язаними даними з бази даних, після підтвердження дії. Такий підхід гарантує користувачеві повний контроль над власними даними й відповідає сучасним стандартам безпеки та конфіденційності.

Інші розділи, такі як «Фітбек» і «Сапорт», наразі не реалізовані, але зарезервовані для подальших оновлень, що розширять можливості зворотного зв'язку та підтримки. Це свідчить про гнучкість архітектури застосунку й готовність до подальшого розвитку функціоналу відповідно до запитів користувачів. Така структура екрана дозволяє зручно керувати налаштуваннями вже зараз і легко масштабувати застосунок у майбутньому, не порушуючи цілісності інтерфейсу.

		Талько А.Ю.			ІПЗ.КР.Б-121-25-ПЗ	Арк.
		Локтікова Т.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		52

Рисунок 3.12 – Екран редагування профілю користувача

Екран редагування профілю користувача надає можливість переглядати й редагувати особисті та облікові дані, що зберігаються в системі. Він логічно поділений на дві секції: «Особисті дані» та «Облікові дані».

У секції «Особисті дані» користувач може змінити своє ім'я та прізвище, які використовуються для відображення в інтерфейсі застосунку, зокрема у лідерборді та профілі. Поля редагування є стандартними текстовими інпутами, що дозволяють вільне введення та редагування даних.

У секції «Облікові дані» відображається електронна адреса користувача, яка водночас слугує логіном для входу в систему. Тут же користувач може оновити пароль — для цього передбачене окреме поле з можливістю приховування/відображення введених символів задля безпеки.

У нижній частині екрана розміщена кнопка «Зберегти», яка активує механізм валідації введених даних та надсилає оновлену інформацію до хмарної бази даних (Firebase Firestore). Таким чином, зміни зберігаються й миттєво відображаються у

		Талько А.Ю.			ІПЗ.КР.Б-121-25-ІЗ	Арк.
		Локтікова Т.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		53

застосунку. Цей екран є ключовим для персоналізації профілю користувача та забезпечення актуальності його даних у системі.

Для забезпечення надійності та стабільності роботи застосунку було проведено комплексне тестування на різних моделях iPhone з версіями iOS від 16 до 17. Застосунок успішно проходив функціональні перевірки як на базових пристроях (iPhone SE 2-го покоління, iPhone 11), так і на сучасних моделях з більшими дисплеями (iPhone 13 Pro Max, iPhone 14). Особлива увага приділялася коректному відображенню інтерфейсу та адаптації під різні розміри екранів, перевірці роботи GPS-трекінгу, інтеграції з HealthKit і обробці push-сповіщень. Тестування проводилося в реальних умовах: під час прогулянок, бігу та їзди на велосипеді, що дозволило перевірити стійкість роботи застосунку до втрати інтернет-з'єднання та зміни геолокації.

Додатково були перевірені сценарії реєстрації, входу, оновлення профілю, роботи з історією тренувань і лідербордом. Проведено тестування під навантаженням — із одночасним використанням кількох сервісів (GPS, HealthKit, зберігання даних у Firestore) для перевірки стабільності та енергоефективності. Усі основні функції застосунку демонстрували високу надійність: запуск і завершення тренування, збереження статистики, зміна стану віртуальної квітки, а також навігація між екранами відбувалася без збоїв та затримок. Виявлені під час тестування незначні помилки були своєчасно виправлені, що дозволило забезпечити позитивний користувацький досвід і відповідність стандартам якості для iOS-додатків.

Висновки до третього розділу

У цьому розділі було детально розглянуто структуру користувацького інтерфейсу мобільного застосунку, спрямованого на відстеження фізичної активності та мотивацію до здорового способу життя. Наведено послідовний опис усіх основних екранів програми, починаючи з онбордингу, що знайомить користувача з можливостями застосунку, та екрана автентифікації, що забезпечує доступ до персоналізованих функцій після входу в систему.

		Талько А.Ю.			ІПЗ.КР.Б-121-25-ІПЗ	Арк.
		Локтікова Т.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		54

Розкрито логіку побудови головного екрана, який містить ключові показники щоденної активності (кількість кроків, пройденої дистанції) та візуальний елемент — квітку, яка змінюється залежно від досягнутих результатів. Детально описано процес початку тренування з вибором типу активності, а також функціонал екрана активного воркауту, де у реальному часі відображаються основні метрики (дистанція, швидкість, час, калорії) та карта з поточною позицією користувача.

Окрему увагу приділено вкладці статистики, де реалізовано календар для перегляду історії тренувань, із можливістю відкриття детальної інформації про кожен сеанс. Також описано інтерфейс екрана лідерів, який відображає щомісячний рейтинг користувачів за кількістю кроків, формуючи елемент соціальної мотивації.

Додатково охарактеризовано розділ налаштувань, що включає зміну персональних даних, керування сповіщеннями, вихід із системи та видалення облікового запису. Зокрема, вказано на наявність функціоналу створення та редагування нагадувань про активність, що сприяє регулярному залученню користувача до фізичної активності.

Опис інтерфейсу дає повне уявлення про логіку роботи із застосунком з точки зору кінцевого користувача, відображаючи акценти на зручності, мотивації та персоналізованому досвіді використання.

		Талько А.Ю.			ІПЗ.КР.Б-121-25-ІПЗ	Арк.
		Локтікова Т.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		55

ВИСНОВКИ

У межах виконання кваліфікаційної роботи бакалавра було розроблено мобільний застосунок, основною метою якого є підтримка фізичної активності користувача через відстеження щоденного руху, ведення статистики тренувань та мотивацію за допомогою візуальних і соціальних механізмів. Особлива увага приділялася створенню інтуїтивного, простого у використанні та естетично привабливого інтерфейсу відповідно до принципів сучасного мобільного дизайну та рекомендацій Apple Human Interface Guidelines.

На початковому етапі було проаналізовано потреби цільової аудиторії та сформульовано вимоги до функціональності застосунку, що охоплює реєстрацію та автентифікацію користувачів, відображення статистики активності, запуск тренувань із визначенням типу активності, побудову маршрутів на карті, перегляд досягнень у вигляді рейтингу серед інших користувачів, а також налаштування профілю та сповіщень.

Другий розділ роботи було присвячено проєктуванню архітектури мобільного застосунку. Було визначено основні компоненти та модулі системи, спроектовано логіку взаємодії між ними, розроблено структуру зберігання та обробки даних, що надходять із сенсорів пристрою та зовнішніх API (зокрема Apple HealthKit). Особливу увагу приділено організації багатoshарового підходу: інтерфейс – логіка – сервіси – локальне сховище, що забезпечує модульність, тестованість та гнучкість розширення функціоналу.

У третьому розділі було детально описано користувацький інтерфейс застосунку. Розглянуто екрани онбордингу, входу в систему, головної панелі з візуалізацією результатів, вибору типу активності, режиму активного тренування з відображенням основних метрик, перегляду статистики за днями, рейтингу користувачів та розділу налаштувань. Визначено логіку переходів між екранами, поведінку інтерфейсних елементів та загальну навігаційну модель.

Розроблений застосунок демонструє успішну реалізацію поставлених цілей — створення зручного інструмента для моніторингу активності, який сприяє підвищенню мотивації до руху та формуванню здорових звичок. Інтерфейс

		Талько А.Ю.			ІПЗ.КР.Б-121-25-ІЗ	Арк.
		Локтікова Т.М.				56
Змн.	Арк.	№ докум.	Підпис	Дата		

орієнтований на комфортну взаємодію, а структура застосунку дозволяє легко розширювати функціонал у майбутньому.

Перспективи подальшого розвитку включають: інтеграцію з більшою кількістю фітнес-сервісів, додавання підтримки групових викликів або тренувань, реалізацію рекомендацій на основі даних користувача, використання машинного навчання для персоналізованих підказок, а також публікацію застосунку в App Store із супровідною маркетинговою кампанією.

		Талько А.Ю.			ІПЗ.КР.Б-121-25-ПЗ	Арк.
		Локтікова Т.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		57

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Apple Developer Forums [Електронний ресурс] – Режим доступу до ресурсу: <https://developer.apple.com/forums/>
2. Apple Human Interface Guidelines [Електронний ресурс] – Режим доступу до ресурсу: <https://developer.apple.com/design/human-interface-guidelines/>
3. Apple Sign In Documentation [Електронний ресурс] – Режим доступу до ресурсу: https://developer.apple.com/documentation/sign_in_with_apple/
4. App Store Review Guidelines [Електронний ресурс] – Режим доступу до ресурсу: <https://developer.apple.com/app-store/review/guidelines/>
5. Core Location Framework [Електронний ресурс] – Режим доступу до ресурсу: <https://developer.apple.com/documentation/corelocation/>
6. Core Motion Framework [Електронний ресурс] – Режим доступу до ресурсу: <https://developer.apple.com/documentation/coremotion/>
7. Firebase Documentation – Notifications [Електронний ресурс] – Режим доступу до ресурсу: <https://firebase.google.com/docs/cloud-messaging>
8. Firebase for iOS – Authentication [Електронний ресурс] – Режим доступу до ресурсу: <https://firebase.google.com/docs/auth/ios/start>
9. Firebase for iOS – Firestore Database [Електронний ресурс] – Режим доступу до ресурсу: <https://firebase.google.com/docs/firestore>
10. Forest: Focus for Productivity [Електронний ресурс] – Режим доступу до ресурсу: <https://play.google.com/store/apps/details?id=cc.forestapp>
11. GitHub – HealthKit Walk Tracking Example [Електронний ресурс] – Режим доступу до ресурсу: <https://github.com/>
12. Google Fit for Android Developers [Електронний ресурс] – Режим доступу до ресурсу: <https://developers.google.com/fit>
13. Hacking with Swift – Health App Integration [Електронний ресурс] – Режим доступу до ресурсу: <https://www.hackingwithswift.com/>
14. HealthKit Framework [Електронний ресурс] – Режим доступу до ресурсу: <https://developer.apple.com/documentation/healthkit/>

		Талько А.Ю.			ІПЗ.КР.Б-121-25-ІЗ	Арк.
		Локтікова Т.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		58

- 15.Human Interface Guidelines for watchOS [Електронний ресурс] – Режим доступу до ресурсу: <https://developer.apple.com/design/human-interface-guidelines/watchos/overview/themes/>
- 16.iOS Dev Weekly [Електронний ресурс] – Режим доступу до ресурсу: <https://iosdevweekly.com/>
- 17.MapKit Framework [Електронний ресурс] – Режим доступу до ресурсу: <https://developer.apple.com/documentation/mapkit/>
- 18.Medium – Building Fitness Apps on iOS [Електронний ресурс] – Режим доступу до ресурсу: <https://medium.com/tag/ios-development>
- 19.NSHipster – Apple Frameworks Overview [Електронний ресурс] – Режим доступу до ресурсу: <https://nshipster.com/>
- 20.Pacer: Pedometer & Step Tracker [Електронний ресурс] – Режим доступу до ресурсу: <https://play.google.com/store/apps/details?id=cc.pacer.androidapp>
- 21.RayWenderlich – SwiftUI Tutorials [Електронний ресурс] – Режим доступу до ресурсу: <https://www.raywenderlich.com/ios/paths/learn>
- 22.SF Symbols [Електронний ресурс] – Режим доступу до ресурсу: <https://developer.apple.com/sf-symbols/>
- 23.Stack Overflow – iOS Development [Електронний ресурс] – Режим доступу до ресурсу: <https://stackoverflow.com/questions/tagged/ios>
- 24.StepBet: Get Active & Stay Fit [Електронний ресурс] – Режим доступу до ресурсу: <https://play.google.com/store/apps/details?id=com.stepbet.app&hl=uk>
- 25.Swift by Sundell [Електронний ресурс] – Режим доступу до ресурсу: <https://www.swiftbysundell.com/>
- 26.Swift Package Manager [Електронний ресурс] – Режим доступу до ресурсу: https://developer.apple.com/documentation/swift_packages/
- 27.Swift Programming Language (Apple Documentation) [Електронний ресурс] – Режим доступу до ресурсу: <https://developer.apple.com/swift/>
- 28.SwiftUI Documentation [Електронний ресурс] – Режим доступу до ресурсу: <https://developer.apple.com/documentation/swiftui/>

		Талько А.Ю.			ІПЗ.КР.Б-121-25-ІЗ	Арк.
		Локтікова Т.М.				59
Змн.	Арк.	№ докум.	Підпис	Дата		

- 29.UIKit Documentation [Електронний ресурс] – Режим доступу до ресурсу:
<https://developer.apple.com/documentation/uikit/>
- 30.UX Collective – Mobile App Onboarding Best Practices [Електронний ресурс] –
 Режим доступу до ресурсу: <https://uxdesign.cc/mobile-app-onboarding-best-practices-2020-83c490e47f2e>

		Талько А.Ю.			ІПЗ.КР.Б-121-25-ПЗ	Арк.
		Локтікова Т.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		60

ДОДАТКИ

		Талько А.Ю.			ІПЗ.КР.Б-121-25-ПЗ	Арк.
		Локтікова Т.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		61

```
//
// ProfileViewModel.swift
// Daisy
//
// Created by Artem Talko on 21.06.2025.
//

import SwiftUI
import SwiftUI

@Observable
final class ProfileViewModel {
    var user: UserModel
    var showSuccessAlert = false
    var errorMessage: String?

    private let authService: AuthProtocol = AuthService()

    init(user: UserModel = UserModel(name: "", surname: "")) {
        self.user = user
    }

    func saveChanges() {
        guard validateAllFields() else {
            errorMessage = "Перевірте правильність введених даних"
            return
        }

        let dispatchGroup = DispatchGroup()
        var updateError: Error?

        dispatchGroup.enter()
        authService.updateDisplayName(name: user.name, surname:
user.surname) { result in
            if case .failure(let err) = result { updateError = err }
            dispatchGroup.leave()
        }

        dispatchGroup.enter()
        authService.updateEmail(to: user.email.value) { result in
            if case .failure(let err) = result { updateError = err }
            dispatchGroup.leave()
        }

        dispatchGroup.enter()
        authService.updatePassword(to: user.password.value) { result in
            if case .failure(let err) = result { updateError = err }
            dispatchGroup.leave()
        }

        dispatchGroup.notify(queue: .main) { [weak self] in
            guard let self else { return }
            if let error = updateError {
```

```

        errorMessage = error.localizedDescription
    } else {
        showSuccessAlert = true
        UserStorage.shared.saveUserData(
            userId: UserStorage.shared.userId ?? "",
            username: "\(user.name) \(user.surname)",
            email: user.email.value
        )
    }
}

private func validateAllFields() -> Bool {
    let emailValid = user.email.onValidate()
    let passwordValid = user.password.onValidate()
    let nameValid = !user.name.trimmingCharacters(in:
.whitespaces).isEmpty
    let surnameValid = !user.surname.trimmingCharacters(in:
.whitespaces).isEmpty

    return emailValid && passwordValid && nameValid && surnameValid
}

```

Додаток Б

```

//
// NotificationSettingsViewModel.swift
// Daisy
//
// Created by Artem Talko on 22.06.2025.
//

import Foundation

@Observable
final class NotificationSettingsViewModel {
    var reminders: [NotificationModel] = []
    var showTitlePrompt = false
    private let storageKey = "user_notifications"
    private var newReminderTitleBuffer: String?

    init() {
        Task {
            await initialize()
        }
    }

    @MainActor
    func initialize() async {
        let permission = await
NotificationService.shared.requestAuthorization()
        if !permission { return }
        await loadNotifications()
    }
}

```

		Талько А.Ю.			ІІЗ.КР.Б-121-25-ІІЗ	Арк.
		Локтікова Т.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		63

```

func loadNotifications() async {
    if let data = UserDefaults.standard.data(forKey: storageKey),
        let decoded = try? JSONDecoder().decode([NotificationModel].self,
from: data) {
        reminders = decoded
    } else {
        let defaultReminder = NotificationModel.defaultMorningReminder()
        reminders = [defaultReminder]
        await NotificationService.shared.schedule(defaultReminder)
        saveToStorage()
    }
}

func toggleReminder(_ reminder: NotificationModel) {
    guard let index = reminders.firstIndex(of: reminder) else { return }
    reminders[index].isActive.toggle()

    Task { @MainActor in
        if reminders[index].isActive {
            await NotificationService.shared.schedule(reminders[index])
        } else {
            NotificationService.shared.remove(id: reminders[index].id)
        }
        saveToStorage()
    }
}

func updateTime(for reminder: NotificationModel, newTime:
DateComponents) {
    guard let index = reminders.firstIndex(of: reminder) else { return }
    reminders[index].time = newTime

    Task { @MainActor in
        NotificationService.shared.remove(id: reminder.id)
        await NotificationService.shared.schedule(reminders[index])
        saveToStorage()
    }
}

func updateTitle(for reminder: NotificationModel, title: String) {
    guard let index = reminders.firstIndex(of: reminder) else { return }
    reminders[index].title = title

    Task { @MainActor in
        NotificationService.shared.remove(id: reminder.id)
        await NotificationService.shared.schedule(reminders[index])
        saveToStorage()
    }
}

func prepareToAddReminder() {
    showTitlePrompt = true
}

```

```

func confirmAddReminder(title: String) {
    let new = NotificationModel(
        id: UUID().uuidString,
        title: title,
        time: DateComponents(hour: 10, minute: 0),
        isActive: true
    )
    reminders.append(new)

    Task { @MainActor in
        await NotificationService.shared.schedule(new)
        saveToStorage()
    }
}

func deleteReminder(_ reminder: NotificationModel) {
    reminders.removeAll { $0.id == reminder.id }
    NotificationService.shared.remove(id: reminder.id)
    saveToStorage()
}

private func saveToStorage() {
    if let encoded = try? JSONEncoder().encode(reminders) {
        UserDefaults.standard.set(encoded, forKey: storageKey)
    }
}
}

```

Додаток В

```

//
// RecordWorkoutViewModel.swift
// Daisy
//

import Foundation
import CoreLocation

@MainActor
@Observable
final class RecordWorkoutViewModel {
    // MARK: - Properties
    private let workoutService = WorkoutService.shared
    private let coordinator: HomeCoordinatorProtocol

    var countdownRemaining: Int? = 3
    var showEndWorkoutAlert = false
    var showErrorAlert = false
    var errorMessage = ""

    // MARK: - Derived State
    var metrics: WorkoutMetrics { workoutService.metrics }
    var isWorkoutActive: Bool { workoutService.isWorkoutActive }
    var state: WorkoutState { workoutService.state }
    var formattedElapsedTime: String {
        let interval = metrics.elapsedTime
    }
}

```

		Талько А.Ю.			ІІЗ.КР.Б-121-25-ІІЗ	Арк.
		Локтікова Т.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		65

```

        let h = Int(interval) / 3600
        let m = Int(interval) % 3600 / 60
        let s = Int(interval) % 60
        return h > 0 ? String(format: "%d:%02d:%02d", h, m, s)
                        : String(format: "%02d:%02d", m, s)
    }

    // MARK: - Initialisation
    init(coordinator: HomeCoordinatorProtocol, workoutType: WorkoutType) {
        self.coordinator = coordinator
        workoutService.configure(workoutType: workoutType)
    }

    func onAppear() {
        workoutService.prepareForWorkout()
        Task { @MainActor [weak self] in
            guard let self else { return }
            await runCountdown()
        }
    }

    // MARK: - Workout Actions
    func beginWorkout() {
        workoutService.beginWorkout()
    }

    func pauseWorkout() {
        workoutService.pauseWorkout()
    }

    func resumeWorkout() {
        workoutService.resumeWorkout()
    }

    func cancelWorkout() {
        workoutService.cancelWorkout()
        coordinator.popToRoot()
    }

    func endWorkout() async {
        do {
            try await workoutService.endWorkout()
            try? await Task.sleep(for: .milliseconds(100))
            coordinator.popToRoot()
        } catch {
            errorMessage = error.localizedDescription
            showErrorAlert = true
        }
    }

    // MARK: - Countdown
    private func runCountdown() async {
        for count in (1...3).reversed() {
            countdownRemaining = count

```

		Талько А.Ю.			ІІЗ.КР.Б-121-25-ІІЗ	Арк.
		Локтікова Т.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		66

```

        try? await Task.sleep(for: .seconds(1))
    }
    countdownRemaining = nil
    beginWorkout()
}
}

```

Додаток Г

```

//
//  StatisticViewModel.swift
//  Daisy
//
//  Created by Artem Talko on 07.11.2024.
//

import SwiftUI
import Foundation
import CoreLocation

@Observable
final class StatisticViewModel {
    // MARK: - Properties
    private let coordinator: StatisticCoordinator
    private let storage = WorkoutStorageService.shared
    private let userStorage = UserStorage.shared

    var allWorkouts: [WorkoutMetrics] = []
    var filteredWorkouts: [WorkoutMetrics] = []
    var workoutsForSelectedDay: [WorkoutMetrics] = []

    var selectedDate: Date? = nil
    var isLoading: Bool = false
    var error: String?

    var highlightedDays: [Date] {
        allWorkouts.map { Calendar.current.startOfDay(for: $0.date) }
    }

    var chartData: [WorkoutType: Int] {
        var result: [WorkoutType: Int] = [:]

        for workout in filteredWorkouts {
            result[workout.type, default: 0] += 1
        }

        return result
    }

    var allRoutes: [[CLLocationCoordinate2D]] {
        filteredWorkouts.map { $0.route }
    }

    // MARK: - Initialisation
    init(coordinator: StatisticCoordinator) {

```

		Талько А.Ю.			ІІЗ.КР.Б-121-25-ІІЗ	Арк.
		Локтікова Т.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		67

```

        self.coordinator = coordinator
        Task { @MainActor [weak self] in
            await self?.loadWorkouts()
        }
    }

    // MARK: - Public Logic
    func loadWorkouts() async {
        guard let userId = userStorage.userId else { return }

        isLoading = true
        defer { isLoading = false }

        do {
            let workouts = try await storage.fetchAllWorkouts(for: userId)
            allWorkouts = workouts
            filterWorkoutsForCurrentWeek()
        } catch {
            self.error = error.localizedDescription
        }
    }

    func filterWorkoutsForCurrentWeek() {
        let calendar = Calendar.current
        let today = Date()
        let startOfWeek = calendar.date(from:
calendar.dateComponents([.yearForWeekOfYear, .weekOfYear], from: today))!
        let endOfWeek = calendar.date(byAdding: .day, value: 7, to:
startOfWeek)!

        filteredWorkouts = allWorkouts.filter {
            $0.date >= startOfWeek && $0.date < endOfWeek
        }
    }

    func selectDay(_ date: Date) {
        selectedDate = date
        let startOfDay = Calendar.current.startOfDay(for: date)
        let endOfDay = Calendar.current.date(byAdding: .day, value: 1, to:
startOfDay)!

        workoutsForSelectedDay = allWorkouts.filter {
            $0.date >= startOfDay && $0.date < endOfDay
        }
    }

    func showWorkoutDetail(_ workout: WorkoutMetrics) {
        coordinator.navigate(to: .workoutDetail(workout))
    }
}

```

Додаток Д

```

//
// TimerManager.swift

```

		Талько А.Ю.			ІПЗ.КР.Б-121-25-ІПЗ	Арк.
		Локтікова Т.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		68

```

// Daisy
//
// Created by Artem Talko on 01.06.2025.
//

import Foundation
import Combine

@Observable
final class TimerManager {
    var elapsedTime: TimeInterval = 0
    var isNil = true
    var state: TimerMode = .stopped

    private var timer: Timer.TimerPublisher?
    private var cancellable: AnyCancellable?

    func start() {
        let startDate = Date()
        isNil = false
        state = .running
        timer = Timer.publish(every: 1, on: .main, in: .common)
        cancellable = timer?
            .autoconnect()
            .map { $0.timeIntervalSince(startDate) }
            .assign(to: \.elapsedTime, on: self)
    }

    func pause() {
        state = .paused
        cancellable?.cancel()
    }

    func resume() {
        let start = Date() - elapsedTime
        state = .running
        timer = Timer.publish(every: 1, on: .main, in: .common)
        cancellable = timer?
            .autoconnect()
            .map { $0.timeIntervalSince(start) }
            .assign(to: \.elapsedTime, on: self)
    }

    func stop() {
        cancellable?.cancel()
        isNil = true
        elapsedTime = 0
        state = .stopped
    }
}

//
// LocationManager.swift
// Daisy

```

Додаток Е

		Талько А.Ю.			ІІЗ.КР.Б-121-25-ІІЗ	Арк.
		Локтікова Т.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		69

```
//

import Foundation
import CoreLocation
import Combine

@Observable
final class LocationManager: NSObject, CLLocationManagerDelegate {
    @ObservationIgnored private let manager = CLLocationManager()

    var userLocation: CLLocation? = nil
    var lastValidLocation: CLLocation? = nil
    var isAuthorized: Bool = false
    var locationPublisher = PassthroughSubject<CLLocation, Never>()

    override init() {
        super.init()
        manager.delegate = self
        manager.desiredAccuracy = kCLLocationAccuracyBest
        manager.distanceFilter = 1
    }

    func startLocationServices() {
        switch manager.authorizationStatus {
        case .authorizedWhenInUse, .authorizedAlways:
            manager.allowsBackgroundLocationUpdates = true
            manager.pausesLocationUpdatesAutomatically = false
            manager.startUpdatingLocation()
            isAuthorized = true
        case .notDetermined:
            manager.requestWhenInUseAuthorization()
        default:
            isAuthorized = false
        }
    }

    func requestCurrentLocation() {
        manager.requestLocation()
    }

    func stopUpdatingLocation() {
        manager.stopUpdatingLocation()
    }

    // MARK: - CLLocationManagerDelegate
    func locationManager(_ manager: CLLocationManager,
    didChangeAuthorization status: CLAuthorizationStatus) {
        if status == .authorizedAlways || status == .authorizedWhenInUse {
            isAuthorized = true
            manager.startUpdatingLocation()
        } else {
            isAuthorized = false
        }
    }
}
```

		Талько А.Ю.			ІІЗ.КР.Б-121-25-ІІЗ	Арк.
		Локтікова Т.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		70

```

func locationManager(_ manager: CLLocationManager, didUpdateLocations
locations: [CLLocation]) {
    for location in locations {
        guard location.horizontalAccuracy >= 0 &&
location.horizontalAccuracy <= 10 else {
            AppPprints.CustomPrint("❌ Відкинута локація: accuracy =
\\(location.horizontalAccuracy), timestamp = \\(location.timestamp)",
withResult: .failure)
            continue
        }

        self.userLocation = location
        self.lastValidLocation = location
        locationPublisher.send(location)
    }
}

func locationManager(_ manager: CLLocationManager, didFailWithError
error: Error) {
    print("📍 Location error: \\(error.localizedDescription)")
}

```

Додаток Є

```

//
// TabBarCoordinator.swift
// Daisy
//
// Created by Artem Talko on 03.11.2024.
//

```

```
import SwiftUI
```

```

@Observable
final class TabBarCoordinator: Coordinatable {
    private(set) var view: AnyView?
    var selection: TabItem = .home

    weak var mainCoordinator: MainCoordinator?

    private let homeCoordinator = HomeCoordinator()
    private let statisticCoordinator = StatisticCoordinator()
    private let leaderboardCoordinator = LeaderboardCoordinator()
    private let settingCoordinator = SettingCoordinator()

    func start() {
        settingCoordinator.mainCoordinator = mainCoordinator

        homeCoordinator.start()
        statisticCoordinator.start()
        leaderboardCoordinator.start()
        settingCoordinator.start()
        setupView()
    }
}

```

		Талько А.Ю.			ІПЗ.КР.Б-121-25-ІЗ	Арк.
		Локтікова Т.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		71

```

func stop() {
    homeCoordinator.stop()
    statisticCoordinator.stop()
    leaderboardCoordinator.stop()
    settingCoordinator.stop()
}

func viewForTab(_ tab: TabItem) -> AnyView {
    switch tab {
    case .home:
        return homeCoordinator.view ?? AnyView(EmptyView())
    case .statistics:
        return statisticCoordinator.view ?? AnyView(EmptyView())
    case .leaderboard:
        return leaderboardCoordinator.view ?? AnyView(EmptyView())
    case .settings:
        return settingCoordinator.view ?? AnyView(EmptyView())
    }
}
}

// MARK: - Private methods
private extension TabBarCoordinator {
    func setupView() {
        self.view = AnyView(MainTabBarView(coordinator: self))
    }
}

```

Додаток Ж

```

//
// NotificationSettingsViewModel.swift
// Daisy
//
// Created by Artem Talko on 22.06.2025.
//

import Foundation

@Observable
final class NotificationSettingsViewModel {
    var reminders: [NotificationModel] = []
    var showTitlePrompt = false
    private let storageKey = "user_notifications"
    private var newReminderTitleBuffer: String?

    init() {
        Task {
            await initialize()
        }
    }

    @MainActor
    func initialize() async {

```

		Талько А.Ю.			ІПЗ.КР.Б-121-25-ІЗ	Арк.
		Локтікова Т.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		72

```

        let permission = await
NotificationService.shared.requestAuthorization()
        if !permission { return }
        await loadNotifications()
    }

    func loadNotifications() async {
        if let data = UserDefaults.standard.data(forKey: storageKey),
            let decoded = try? JSONDecoder().decode([NotificationModel].self,
from: data) {
            reminders = decoded
        } else {
            let defaultReminder = NotificationModel.defaultMorningReminder()
            reminders = [defaultReminder]
            await NotificationService.shared.schedule(defaultReminder)
            saveToStorage()
        }
    }

    func toggleReminder(_ reminder: NotificationModel) {
        guard let index = reminders.firstIndex(of: reminder) else { return }
        reminders[index].isActive.toggle()

        Task { @MainActor in
            if reminders[index].isActive {
                await NotificationService.shared.schedule(reminders[index])
            } else {
                NotificationService.shared.remove(id: reminders[index].id)
            }
            saveToStorage()
        }
    }

    func updateTime(for reminder: NotificationModel, newTime:
DateComponents) {
        guard let index = reminders.firstIndex(of: reminder) else { return }
        reminders[index].time = newTime

        Task { @MainActor in
            NotificationService.shared.remove(id: reminder.id)
            await NotificationService.shared.schedule(reminders[index])
            saveToStorage()
        }
    }

    func updateTitle(for reminder: NotificationModel, title: String) {
        guard let index = reminders.firstIndex(of: reminder) else { return }
        reminders[index].title = title

        Task { @MainActor in
            NotificationService.shared.remove(id: reminder.id)
            await NotificationService.shared.schedule(reminders[index])
            saveToStorage()
        }
    }
}

```

		Талько А.Ю.			ІІЗ.КР.Б-121-25-ІІЗ	Арк.
		Локтікова Т.М.				73
Змн.	Арк.	№ докум.	Підпис	Дата		

```

func prepareToAddReminder() {
    showTitlePrompt = true
}

func confirmAddReminder(title: String) {
    let new = NotificationModel(
        id: UUID().uuidString,
        title: title,
        time: DateComponents(hour: 10, minute: 0),
        isActive: true
    )
    reminders.append(new)

    Task { @MainActor in
        await NotificationService.shared.schedule(new)
        saveToStorage()
    }
}

func deleteReminder(_ reminder: NotificationModel) {
    reminders.removeAll { $0.id == reminder.id }
    NotificationService.shared.remove(id: reminder.id)
    saveToStorage()
}

private func saveToStorage() {
    if let encoded = try? JSONEncoder().encode(reminders) {
        UserDefaults.standard.set(encoded, forKey: storageKey)
    }
}
}

```

		Талько А.Ю.			ІІЗ.КР.Б-121-25-ІІЗ	Арк.
		Локтікова Т.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		74