

**Міністерство освіти і науки України
Луцький національний технічний університет
Факультет комп'ютерних та інформаційних технологій
Кафедра інженерії програмного забезпечення**

**КВАЛІФІКАЦІЙНА РОБОТА
ЗА СТУПЕНЕМ ВИЩОЇ ОСВІТИ «БАКАЛАВР»**

**РОЗРОБКА МОБІЛЬНОГО ЗАСТОСУНКУ ОБЛІКУ КЛІЄНТІВ
ДЛЯ ФІТНЕС-ТРЕНЕРІВ НА IOS ТА ANDROID**

**DEVELOPMENT OF A MOBILE CLIENT ACCOUNTING
APPLICATION FOR FITNESS TRAINERS ON IOS AND ANDROID**

спеціальність 121 «Інженерія програмного забезпечення»
освітня програма «Інженерія програмного забезпечення»

Виконав: здобувач вищої освіти
групи ІПЗ-41
Шило Олександр Богданович

Керівник:
Бойко Лев Степанович

Кваліфікаційну роботу
допущено до захисту
«__» _____ 20__р.
Гарант освітньої програми:
к.т.н., доцент
Ліщина Наталія Миколаївна

Луцьк – 2026 року

ЛУЦЬКИЙ НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ

Факультет комп'ютерних та інформаційних технологій

Кафедра інженерії програмного забезпечення

Ступінь вищої освіти бакалавр

Галузь знань: 12 «Інформаційні технології»

Спеціальність: 121 «Інженерія програмного забезпечення»

Освітня програма: «Інженерія програмного забезпечення»

ЗАТВЕРДЖУЮ

Завідувач кафедри

«__» _____ 20__ р.

ЗАВДАННЯ НА КВАЛІФІКАЦІЙНУ РОБОТУ ЗДОБУВАЧУ ВИЩОЇ ОСВІТИ

Шилу Олександрю Богдановичу

(прізвище, ім'я, по батькові)

1. Тема кваліфікаційної роботи «Розробка мобільного застосунку обліку клієнтів для фітнес - тренерів на iOS та Android»

Керівник роботи: доцент Бойко Л. С.

затверджені наказом закладу вищої освіти від «31» грудня 2025 р. № 541/01-02

2. Строк подання здобувачем вищої освіти кваліфікаційної роботи бакалавра «29» травня 2026 р.

3. Вихідні дані до роботи: методичні матеріали з розробки мобільних застосунків, принципи побудови кросплатформених систем, методи організації локального збереження даних, підходи до проектування користувацького інтерфейсу та засоби реалізації функціоналу для обліку клієнтів, тренувань і відстеження прогресу у сфері фітнесу.

4. Зміст розрахунково-пояснювальної записки (перелік питань, що потрібно розробити аналіз сучасного стану проблеми автоматизації обліку клієнтів фітнес-тренера, дослідження існуючих методів і програмних засобів її розв'язання, аналіз і вибір інструментів розробки мобільного застосунку, опис функціонального наповнення об'єкта проектування, розробка й обґрунтування структури системи, реалізація локального збереження даних, оцінка зручності використання та надійності роботи застосунку, розробка функціонально-структурної схеми роботи об'єкта проектування.

5. Перелік графічного матеріалу: 19 рисунків, 3 таблиці, 7 лістингів, 2 додатки.

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис	
		завдання видав	завдання прийняв
Аналіз предметної області	Бойко Л. С.		
Специфікація вимог до розробленої системи	Бойко Л. С.		
Розробка об'єкта проектування	Бойко Л. С.		
Нормоконтроль	Суринович О. М.		
Гарант ОП	Бойко Л. С.		
Показник запозичень тексту	___%		
Академічна доброчесність	Бойко Л. С.		

7. Дата видачі завдання «3» лютого 2026 р.

№	Назва етапів кваліфікаційної роботи бакалавра	Строк виконання етапів роботи	Примітка
1	Огляд літературних джерел по темі кваліфікаційної роботи бакалавра	до 14.02.2026 р.	
2	Аналіз проблеми розробки та впровадження об'єкту проектування	до 03.03.2026 р.	
3	Обґрунтування вибору шляхів, технологій і засобів вирішення поставленого завдання	до 14.03.2026 р.	
4	Розробка функціонально-структурної схеми роботи об'єкта проектування та проектування бази даних	до 14.04.2026 р.	
5	Практична реалізація об'єкта проектування та розробка бази даних	до 20.04.2026 р.	
6	Тестування та налагодження об'єкта проектування	до 04.05.2026 р.	
7	Здача чистового варіанту кваліфікаційної роботи бакалавра на кафедру	до 29.05.2026 р.	

Здобувач вищої освіти _____

Олександр ШИЛО

Керівник кваліфікаційної роботи _____

Лев БОЙКО

АНОТАЦІЯ

Шило О. Б. Розробка мобільного застосунку обліку клієнтів для фітнес-тренерів на iOS та Android. Рукопис. Кваліфікаційна робота бакалавра ОП «Інженерія програмного забезпечення» спеціальності «Інженерія програмного забезпечення». Луцький національний технічний університет. Луцьк, 2026.

Кваліфікаційна робота бакалавра складається зі вступу, трьох розділів, висновків, списку використаних джерел та додатків.

У першому розділі здійснено аналіз сучасного стану ринку мобільних застосунків для обліку клієнтів фітнес-тренерів, було виявлено їхні недоліки та визначено проблеми, які зустрічаються при розробці таких рішень. Було сформульовано завдання на розробку власного мобільного застосунку. У другому розділі обґрунтовано вибір, технологій та засобів реалізації, визначено функціональні, нефункціональні та технічні вимоги до продукту. У третьому розділі реалізовано практичну частину застосунку: розроблено інтерфейс, логіку взаємодії застосунку, базу даних, а також проведено тестування та формування інсталяційного APK- пакета.

Ключові слова: Flutter, фітнес, мобільний застосунок, Apache Hive, інтерфейс, Dart, Hive, кросплатформенність.

ABSTRACT

Shylo O. B. Development of a Mobile Client Accounting Application for Fitness Trainers on iOS and Android. Manuscript. Bachelor's qualification work of the Educational Program «Software Engineering» in the specialty «Software Engineering». Lutsk National Technical University. Lutsk, 2026.

The bachelor's qualification work consists of an introduction, three chapters, conclusions, a list of references, and appendices.

The first chapter analyzes the current state of the mobile application market for client management by fitness trainers, identifies their shortcomings, and defines the problems that arise during the development of such solutions. The task of developing a custom mobile application was formulated. The second chapter substantiates the choice of technologies and implementation tools and defines the functional, non-functional, and technical requirements for the product. The third chapter implements the practical part of the application: the interface, application interaction logic, and database were developed, and testing was carried out, along with the creation of the installation APK package.

Keywords: Flutter, fitness, mobile application, Apache Hive, interface, Dart, Hive, cross-platform.

ЗМІСТ

ВСТУП.....	7
РОЗДІЛ 1 АНАЛІЗ СУЧАСНИХ РІШЕНЬ У СФЕРІ МОБІЛЬНИХ ДОДАТКІВ ДЛЯ ФІТНЕС-ТРЕНЕРІВ І ПОСТАНОВКА ЗАВДАНЬ НА КВАЛІФІКАЦІЙНУ РОБОТУ	10
1.1 Аналіз сучасного стану проблеми.....	10
1.2 Постановка завдання на кваліфікаційну роботу бакалавра	15
РОЗДІЛ 2 СПЕЦИФІКАЦІЯ ВИМОГ ДО РОЗРОБЛЮВАНОЇ СИСТЕМИ	16
2.1 Аналіз, визначення вимог до розроблюваного програмного забезпечення та проєктування програмного забезпечення	16
2.2 Вибір засобів, методів і алгоритмів вирішення поставленого завдання....	17
РОЗДІЛ 3 РОЗРОБКА МОБІЛЬНОГО ЗАСТОСУНКУ ОБЛІКУ КЛІЄНТІВ ДЛЯ ФІТНЕС-ТРЕНЕРІВ НА IOS ТА ANDROID.....	22
3.1 Практична реалізація об'єкта проєктування.....	22
3.2 Реалізація інтерфейсу.....	33
3.3 Тестування та налагодження інформаційно-комп'ютерної системи.....	34
3.4 Розробка інсталяційного пакета	35
ВИСНОВКИ.....	37
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ.....	39
ДОДАТКИ.....	41

ВСТУП

Актуальність теми. У ХХІ столітті все більше людей приділяє увагу своєму фізичному та психічному здоров'ю. Така тенденція сприяє стрімкому розвитку фітнес-індустрії, зростанню популярності здорового способу життя та збільшенню кількості фітнес-клубів і персональних тренерів. Внаслідок цього розширюється ринок праці у даній сфері та зростає попит на якісні інструменти для організації тренувального процесу.

Попри активний розвиток фітнес-індустрії, сфера спеціалізованого програмного забезпечення для тренерів залишається недостатньо розвиненою. Багато фахівців змушені використовувати загальні або незручні інструменти для ведення обліку клієнтів, складання тренувальних програм та відстеження прогресу підопічних, що ускладнює їхню роботу та знижує ефективність взаємодії з клієнтами.

У цьому контексті актуальним є створення сучасного мобільного застосунку для фітнес-тренерів, який поєднуватиме інтуїтивно зрозумілий інтерфейс, можливість формування адаптивних програм тренування, а також функцію моніторингу прогресу клієнтів. Використання технології Flutter дозволяє реалізувати кросплатформений додаток, що забезпечить його стабільну роботу як на пристроях з операційною системою Android, так і iOS, а також спростить процес розробки та подальшої підтримки продукту.

Метою кваліфікаційної роботи бакалавра є розробка мобільного застосунку обліку клієнтів для фітнес-тренерів на iOS та Android.

Об'єкт дослідження – процес розробки мобільних застосунків у сфері спорту та фітнесу.

Предмет дослідження – особливості проєктування та розробки мобільного застосунку для фітнес-тренерів із використанням технології Flutter, зокрема технічні, функціональні та UX-рішення, спрямовані на підвищення ефективності взаємодії користувачів із системою.

Для досягнення поставленої мети в межах кваліфікаційної роботи необхідно виконати такі завдання:

- здійснити аналіз предметної області та визначити основні вимоги до розроблюваного мобільного застосунку;
- провести аналіз існуючих технологій і обґрунтувати вибір засобів розробки;
- впровадити адаптивний інтерфейс, що коректно відображається на мобільних пристроях з різними розмірами екранів;
- забезпечити надійне локальне збереження даних користувачів;
- реалізувати мобільний застосунок із використанням обраних інструментів та технологій розробки;
- провести комплексне тестування застосунку, включаючи перевірку функціональності, продуктивності та стабільності роботи;
- сформувати інсталяційний APK-файл для подальшого розповсюдження застосунку.

Практичне значення розробки пролягає у використанні застосунку фітнес-тренерами, для автоматизації процесу ведення клієнтів, тренувань та обліку фізичних навантажень. Розроблений застосунок забезпечить зручне зберігання інформації про клієнтів, формування тренувань, перегляд історії занять і аналіз прогресу за допомогою графіків. Це дозволить підвищити ефективність роботи фітнес-тренера та спростить організацію тренувального процесу.

Таким чином, розробка мобільного застосунку для фітнес-тренерів здійснюватиметься з використанням фреймворку Flutter, що забезпечить створення ефективного кросплатформеного рішення яке матиме сучасний дизайн та високу продуктивність. Обраний підхід дозволяє оптимізувати процес розробки, зменшити витрати ресурсів та оптимізувати процес розробки, зменшити витрати ресурсів та забезпечити стабільну роботу застосунку на різних мобільних платформах.

У межах кваліфікаційної роботи передбачається реалізація повного циклу

розробки програмного продукту: від аналізу предметної області, формування вимог і вибору інструментів – до безпосередньої розробки, тестування оптимізації та підготовки мобільного застосунку до подальшого розповсюдження.

РОЗДІЛ 1

АНАЛІЗ СУЧАСНИХ РІШЕНЬ У СФЕРІ МОБІЛЬНИХ ДОДАТКІВ ДЛЯ ФІТНЕС-ТРЕНЕРІВ І ПОСТАНОВКА ЗАВДАНЬ НА КВАЛІФІКАЦІЙНУ РОБОТУ

1.1 Аналіз сучасного стану проблеми

Сучасний ринок фітнес-додатків, зокрема орієнтованих на тренерів, інтенсивно розвивається, що зумовлено зростанням попиту на індивідуальні тренування та персоніфіковані підходи до організації тренувального процесу.

Така динаміка свідчить про підвищення ролі цифрових технологій у сфері фітнесу та необхідність впровадження інноваційних рішень для підвищення ефективності роботи тренерів.

Водночас аналіз функціональних можливостей наявних програмних продуктів дозволяє виявити низку проблем, які стримують подальший розвиток даного сегменту. До них належать обмежені можливості персоналізації, складність інтерфейсу, недостатня інтеграція інструментів для відстеження прогресу клієнтів, а також відсутність комплексного підходу до організації взаємодії між тренером та підопічним.

Серед ключових продуктів, що закріпились на ринку фітнес-застосунків для тренерів, доцільно виокремити низку найбільш популярних які відзначаються широким функціоналом, високою якістю контенту та ефективними підходами до організації взаємодії з користувачами. Аналіз таких програмних продуктів дозволяє визначити сучасні тенденції розвитку галузі, а також визначити їхні переваги та недоліки з метою подальшого вдосконалення власного програмного рішення.

Trainerize – це мобільний застосунок для фітнес-тренерів, який дозволяє створювати індивідуальні програми тренувань, вести облік клієнтів, відстежувати їхній прогрес та підтримувати комунікацію через чат, що робить його зручним інструментом для онлайн-коучингу та масштабування бізнесу [1].

У результаті проведеного аналізу застосунку Trainerize було визначено його ключові переваги та недоліки, які наведено в таблиці 1.1 для подальшого порівняння та оцінювання.

Таблиця 1.1 – Переваги та недоліки Trainerize

Переваги Trainerize	Недоліки Trainerize
Багатофункціональність	Складність освоєння для новачків
Інтеграція з популярними сервісами	Велика частина функцій є тільки у платних тарифах
Можливість автоматизації процесів	Обмежена аналітика
Гнучке налаштування тренувань	Відсутність української мови
	Програма заточена для онлайн тренувань
	Інколи має нестабільну роботу

Аналіз продемонстрував що навіть з великою кількістю недоліків Trainerize є платформою з великою кількістю переваг для ведення бізнесу тренера (рис. 1.1).

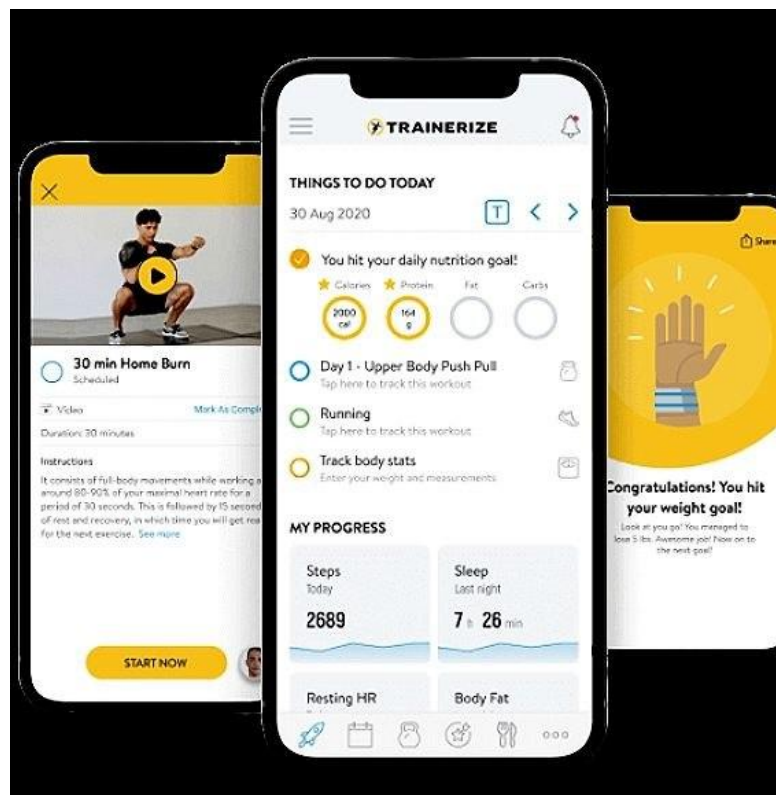


Рисунок 1.1 – Вигляд застосунку Trainerize [2]

Окрім розглянутого застосунку, доцільно відзначити інші програмні рішення, які закріпились на ринку фітнес-індустрії та відіграють важливу роль у формуванні сучасних підходів до організації тренувального процесу. Одним з найпопулярніших застосунків які мають своїх прихильників серед спільноти фітнес-тренерів можна вважати застосунок TrueCoach. Застосунок отримав своїх прихильників під час пандемії COVID-19, коли всі тренувальні зали були зачинені і в людей було багато вільного часу.

TrueCoach – це мобільний застосунок для фітнес-тренерів (рис. 1.2), орієнтований на індивідуальний онлайн-коучинг, який надає змогу проводити персоналізовані тренування, відстежувати прогрес клієнтів, обмінюватись повідомленнями та використовувати відео вправ що робить його зручним для роботи 1-на-1 з підопічним [3].

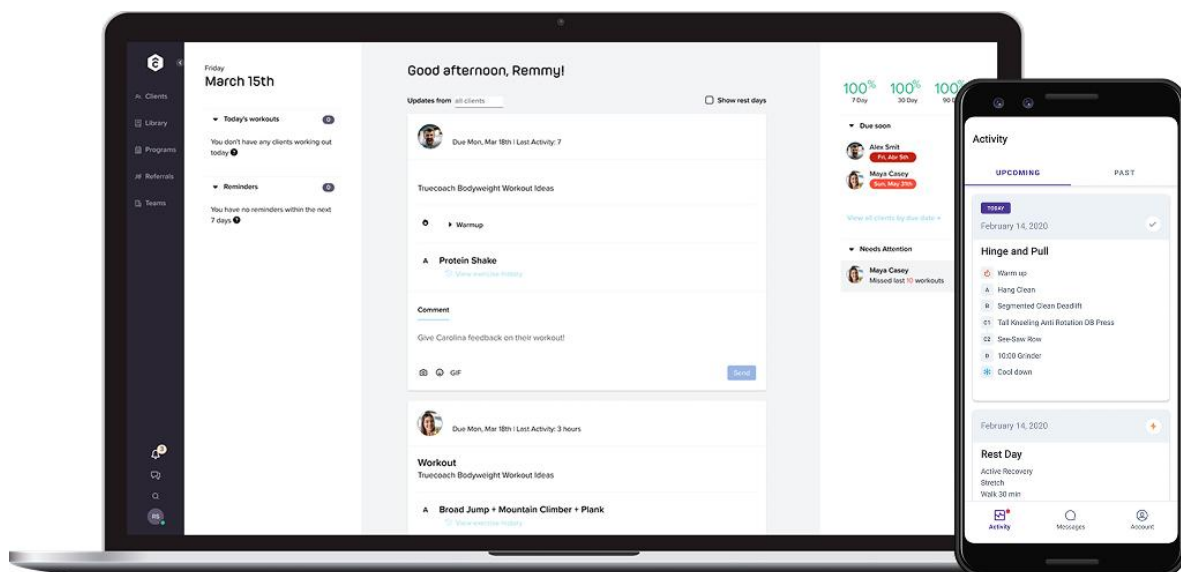


Рисунок 1.2 – Вигляд застосунку TrueCoach [4]

У результаті детального аналізу функціональних можливостей даного застосунку, було виявлено його ключові переваги та недоліки, які представлені в таблиці 1.2, що дозволяє здійснити об'єктивне оцінювання його ефективності.

Таблиця 1.2 – Переваги та недоліки TrueCoach

Переваги TrueCoach	Недоліки TrueCoach
Простота використання	Обмежений функціонал
Швидке створення персональних тренувань	Орієнтування тільки на 1 клієнта
Зручний трекінг	Відсутність української мови
	Незручний інтерфейс

Незважаючи на високий рівень зручності та орієнтованості на користувача, застосунок має низку недоліків, які впливають на ефективність його використання.

Аналізуючи ринок фітнес-застосунків, доцільно також розглянути програмне рішення, розроблене одним із провідних світових брендів у сфері спорту, яке суттєво вплинуло на формування сучасних підходів до організації тренувального процесу.

Nike Training Club – це безкоштовний мобільний застосунок для тренувань, який пропонує багато готових програм і вправ (силові, кардіо, йога) з відео інструкціями та рекомендаціями від професійних тренерів і спортсменів, що робить його одним з найкращих варіантів для самостійних занять [5].

Можна виділити наступні переваги Nike Training Club:

- наявність повністю безкоштовного доступу до функціоналу, що забезпечує охоплення широкої аудиторії потенційних користувачів та сприяє підвищенню рівня їх залученості;
- наявність масштабної бібліотеки тренувальних матеріалів, що охоплює понад 100 000 (сто тисяч) вправ та відеоуроків, забезпечуючи широкий вибір програм та різноманітність тренувального процесу;
- простий та інтуїтивно зрозумілий користувацький інтерфейс, що забезпечує зручність взаємодії із застосунком (рис. 1.3);
- Підтримка одною з провідних компаній в сфері спорту Nike.

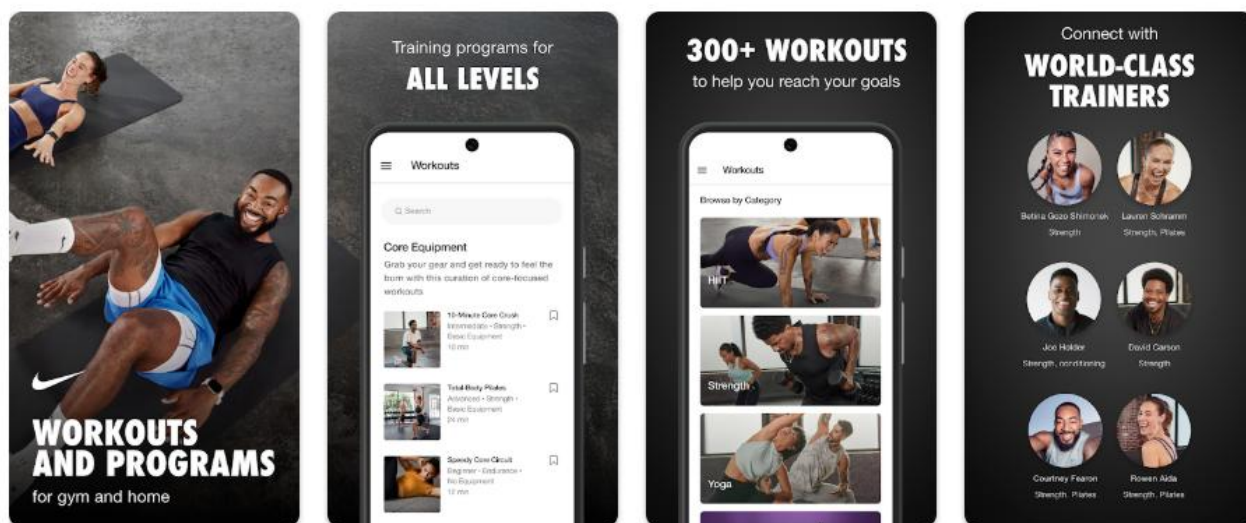


Рисунок 1.3 – Вигляд застосунку Nike Training Club [6]

Проте не зважаючи на значні переваги Nike Training Club даний програмний продукт має і ряд недоліків.

Недостатній рівень персоналізації, відсутність індивідуального підходу. Зокрема застосунок не забезпечує гнучкого налаштування тренувальних програм відповідно до рівня фізичної підготовки, індивідуальних цілей, стану здоров'я або особистих вподобань користувача. Це в свою чергу, може негативно впливати на мотивацію до регулярних занять, рівень залученості та загальні результати тренувального процесу.

Залежність від стабільного підключення до мережі Інтернет, що обмежує можливість повноцінного використання функціоналу застосунку в умовах відсутності або нестабільності мережі. Зокрема, частина функцій, таких як доступ до тренувальних, відеоматеріалів або синхронізацію даних, може бути недоступною без активного інтернет-з'єднання, що знижує зручність використання та негативно впливає на безперервність тренувального процесу.

Nike Training Club є дуже хорошим застосунком для індивідуальних тренувань, але в контексті спеціалізованого програмного забезпечення для фітнес-тренерів він далеко не найкращий варіант.

1.2 Постановка завдання на кваліфікаційну роботу бакалавра

У межах даної роботи було проведено аналіз предметної області та досліджено сучасний стан розв'язання поставленої проблеми.

Для досягнення поставленої мети в межах кваліфікаційної роботи бакалавра необхідно виконати такі технічні та функціональні завдання:

- здійснити аналіз предметної області та визначити основні вимоги до розроблюваного мобільного застосунку;
- провести аналіз існуючих технологій і обґрунтувати вибір засобів розробки;
- впровадити адаптивний інтерфейс, що коректно відображається на мобільних пристроях з різними розмірами екранів;
- забезпечити надійне локальне збереження даних користувачів;
- реалізувати мобільний застосунок із використанням обраних інструментів та технологій розробки;
- провести комплексне тестування застосунку, включаючи перевірку функціональності, продуктивності та стабільності роботи;
- сформувати інсталяційний APK-файл для подальшого розповсюдження застосунку.

РОЗДІЛ 2

СПЕЦИФІКАЦІЯ ВИМОГ ДО РОЗРОБЛЮВАНОЇ СИСТЕМИ

2.1 Аналіз, визначення вимог до розроблюваного програмного забезпечення та проєктування програмного забезпечення

У процесі розробки мобільного застосунку для фітнес-тренерів з використанням Flutter з'являється потреба у чіткому плануванні архітектури програмного забезпечення та визначення основних принципів побудови системи. Грамотно спроектована архітектура (рис.2.1), забезпечує стабільність роботи застосунку, зручність масштабування функціоналу, повторне використання компонентів, а також спростить процес подальшого тестування та підтримки програмного продукту.

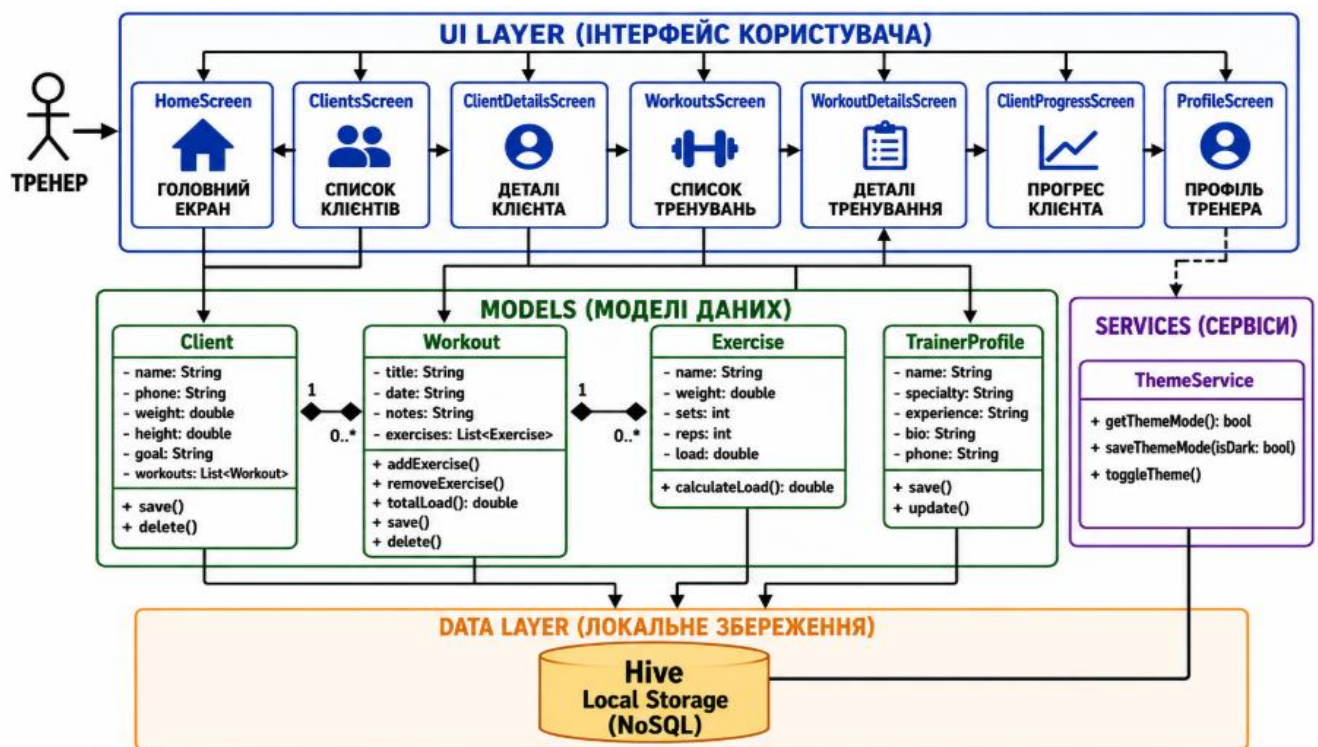


Рисунок 2.1 – Архітектура мобільного застосунку

Особливо важливо це є для мобільних застосунків, які повинні коректно працювати на різних мобільних пристроях, підтримувати адаптивний інтерфейс та забезпечувати швидку взаємодію користувача з системою. Використання

сучасних підходів до проєктування дозволяє розділити логіку роботи на окремі модулі, що позитивно впливає на читабельність коду, зменшує складність розробки та полегшує внесення змін в майбутньому.

На основі попереднього аналізу предметної області, сучасних програмних рішень та потреб користувачів було сформульовано комплекс функціональних та нефункціональних і технічних вимог до мобільного застосунку для фітнес-тренерів. До функціональних вимог належить можливість ведення клієнтів, створення тренувань відстеження прогресу та збереження інформації у локальній базі даних. Не функціональні вимоги охоплюють стабільність роботи, швидкодію, адаптивність інтерфейсу та зручність використання. Технічні вимоги визначають використання сучасних інструментів розробки, підтримки мобільних платформ та можливість подальшого розширення функціоналу системи.

2.2 Вибір засобів, методів і алгоритмів вирішення поставленого завдання

Для вирішення завдання створення мобільного застосунку для фітнес-тренерів необхідно було визначити оптимальний інструментарій, який би забезпечував зручну реалізацію функціоналу, відповідність сучасним технологічним стандартам, а також забезпечував потенціал для масштабування системи. З цією метою було проведено аналіз існуючих підходів до розробки мобільних додатків, зокрема розглянуто як нативні, так і кросплатформенні технології.

Нативний підхід передбачає створення окремих мобільних застосунків для кожної платформи – Android та iOS. Основними перевагами даного підходу є максимальна продуктивність, високий рівень стабільності роботи, а також повний доступ до всіх можливостей операційної системи та апаратних ресурсів пристрою. Серед розглянутих нативних рішень для системи Android можна виділити: Java, Kotlin Swift.

Java – об’єктно-орієнтована мова програмування, випущена 1995 року компанією «Sun Microsystems» як основний компонент платформи Java. У 2009 році компанія «Oracle» придбала «Sun Microsystems». Java-програми компілюються у байт-код, який при виконанні інтерпретується віртуальною машиною для конкретної платформи.

Синтаксис Java дуже подібний до мов програмування C і C++. Він набагато стійкіший до помилок що значно полегшено процес розробки об’єктно-орієнтованих програм. Віртуальна машина Java виконує низку дій, які в C/C++ повинні були здійснювати програмісти самостійно, зокрема керування пам’яттю, автоматичне очищення невикористаних об’єктів, контроль безпечного доступу до пам’яті. Java розроблялась як платформо-незалежна мова, це призвело до того що вона має менше низькорівневих можливостей для роботи з апаратним забезпеченням. Також Java дозволяє викликати підпрограми, що були написані за допомогою інших мов програмування [7].

Хоча мова програмування Java тривалий час залишалась основним компонентом розробки мобільних додатків для платформи Android, останнім часом спостерігається поява більш сучасних і зручних рішень, зокрема одним з найпопулярніших рішень є мова Kotlin, яка офіційно підтримується компанією Google як пріоритетна мова для Android-розробки.

Kotlin – строго типізована мова програмування що розробляється компанією JetBrains, Kotlin працює на базі JVM. Автори хотіли створити більш лаконічнішу та типо-безпечнішу мову, за Java [8].

Оскільки розроблювальний застосунок передбачає підтримку не лише платформи Android, але й операційної системи iOS, необхідно також приділити увагу відповідним технологічним рішенням для iOS-розробки.

Найпопулярнішою мовою розробки для iOS є мова програмування Swift на якій створюється більшість застосунків.

Swift – компільована мова програмування яка підтримує декілька парадигм програмування, Зокрема об’єктно-орієнтовану, протокольно-орієнтовану та функціональну. Вона розроблена компанією Apple спеціально для того, щоб

замінити Objective-C. Мова побудована одразу з LLVM компілятором, включеного в Xcode 6 beta [9].

Нативний підхід передбачає необхідність використання кількох мов програмування для реалізації проєкту, що ускладнює процес розробки та потребує значних витрат часу та ресурсів. Крім того підтримка та подальший супровід таких рішень є більш складним, оскільки потребує внесення змін у кілька окремих кодових баз, що підвищує ризик виникнення помилок та ускладнює процес оновлення програмного продукту.

Натомість альтернативою нативним підходами є кросплатформені технології, які дозволяють створювати мобільні застосунки з використанням уніфікованої кодової бази. Це забезпечує можливість запуску розроблених рішень на різних операційних системах, зокрема Android та iOS, що значно спрощує процес, розробки, скорочує витрати часу та ресурсів, а також полегшує подальший супровід програмного продукту.

Одним з найпопулярніших рішень на ринку кросплатформених технологій є фреймворк React Native.

React Native – це кросплатформений фреймворк який має відкритий вихідний код, був створений компанією Meta. Він використовується для розробки програм які використовуються в Android, iOS, macOS, Windows та інших. Також даний фреймворк можна використовувати з іншими нативними рішеннями для кращої роботи [10].

Окрім React Native доцільно відзначити кросплатформений фреймворк Flutter розроблений компанією Google, який є одним з найсучасніших і найбільш перспективних інструментів для створення мобільних додатків. Використання даної технології дозволяє розробляти високопродуктивні застосунки з єдиною кодовою базою для різних платформ.

Flutter – це SDK який має відкритий вихідний код що використовується для створення додатків на різних платформах таких як Android, iOS, а також вебсередовищі. Він є основним для створення додатків для Google Fuchsia.

Даний SDK відрізняється своєю архітектурою на відміну від інших

кросплатформених рішень тим що використовує власний рушій для рендерингу.

Цей фреймворк не потребує використання сторонніх мов програмування для створення основного функціоналу застосунку, оскільки основна розробка здійснюється за допомогою Dart [11].

Dart – мова програмування, від компанії Google, яка позиціонувала її як мову структурованого програмування для Веб. Розробники хотіли, щоб в довгостроковій перспективі мова Dart стане прогресивною заміною мові програмування JavaScript. Проте на сьогодні Dart переважно використовується саме у Flutter. Її використання забезпечує високу продуктивність застосунків, та спрощує процес розробки завдяки використанню єдиної мови програмування [12]. Механізм Hot Reload дозволяє швидко вносити зміни у весь проєкт.

Серед розглянутих варіантів реалізації програмного продукту було обрано кросплатформені фреймворки як найбільш доцільний підхід для досягнення поставленої мети. З метою обґрунтування остаточного вибору було проведено порівняльний аналіз Flutter React та Native. Результати даного порівняння представлені в таблиці 2.1.

Таблиця 2.1 – Порівняння Flutter React та Native

Критерій	Flutter	React та Native
Мова програмування	Dart	JavaScript
Кодова база	Єдина для всіх платформ	Єдина (з частковими нативними модулями)
Продуктивність	Висока (компіляція в нативний код)	Середня-висока (через bridge)
UI/UX можливості	Повний контроль над UI	Використання нативних компонентів
Доступ до API пристрою	Через плагіни	Через bridge/нативні модулі
Підтримка та масштабування	Зручне	Середнє
Кількість бібліотек	Велика	Дуже велика
Кросплатформеність	Android, iOS, Web	Android, iOS
Офіційна підтримка	Google	Meta

У результаті проведеного аналізу було встановлено, що саме фреймворк Flutter демонструє оптимальне співвідношення продуктивності, гнучкості та зручності розробки, і є найбільш доцільним вибором для реалізації поставлених завдань. Дана SDK технологія забезпечує високу продуктивність, єдину кодову базу для різних платформ, гнучкість у створенні користувацького інтерфейсу, а також значно спрощує процес розробки та подальшого підтримання програмного продукту.

Як середовище розробки було обрано IDE Android Studio, яке надає розширені можливості для написання коду, налагодження, тестування та оптимізації мобільних застосунків, а також забезпечує зручну інтеграцію з необхідними інструментами.

Android Studio – являє собою інтегроване середовище розробки, яке призначене для Android платформи.

До його складу входять засоби для тестування програм на сумісність з різними версіями платформи та інструменти для проєктування застосунків, які працюють на різних пристроях з різною роздільністю екранів [13].

Для локального збереження даних в мобільному застосунку було використано HIVE – швидку та легку NoSQL-систему сховища даних, яка написана на мові програмування Dart, вона працює на мобільних пристроях, desktop та web [14].

Ця база даних легко інтегрується з фреймворком Flutter та забезпечує надійне локальне збереження інформації в пам'яті пристрою. Використання даного дозволяє забезпечити швидкий доступ до даних без необхідності підключення до мережі Інтернет, що підвищує стабільність роботи застосунку та зручність його використання для користувачів.

РОЗДІЛ 3

РОЗРОБКА МОБІЛЬНОГО ЗАСТОСУНКУ ОБЛІКУ КЛІЄНТІВ ДЛЯ ФІТНЕС-ТРЕНЕРІВ НА IOS ТА ANDROID

3.1 Практична реалізація об'єкта проєктування

Розробка мобільного застосунку обліку клієнтів для фітнес-тренерів здійснювалася на базі SDK Flutter з використанням мови програмування Dart. У процесі реалізації було побудовано архітектуру, де кожен окремий dart-файл відповідає певній частині функціоналу або користувацького інтерфейсу.

Файл `main.dart` є точкою входу в застосунок. У ньому виконується ініціалізація Flutter, локальної бази даних Hive, реєстрація моделей, відкриття сховищ, створення початкового профілю тренера (ліст. 3.1), налаштування світлої та темної теми, а також запуск головного екрану застосунку `home_screen.dart`.

Лістинг 3.1 – Початковий профіль тренера

```

if (trainerBox.isEmpty) {
  await trainerBox.add(
    TrainerProfile(
      name: 'Ваше ім'я',
      email: 'Ваша електронна адреса @example.com',
      specialization: 'Персональні тренування',
      experience: 'Ваш досвід роботи',
      description:
        'Опишіть чим ви займаєтесь.',
    ),
  );
}

```

кінець лістингу 3.1

Файл `home_screen.dart` відповідає за перший екран застосунку, у якому реалізовано навігацію між основними розділами: «Клієнти», «Тренування» та «Профіль». Для перемикання між вкладками використовується нижня навігаційна панель (рис. 3.1).

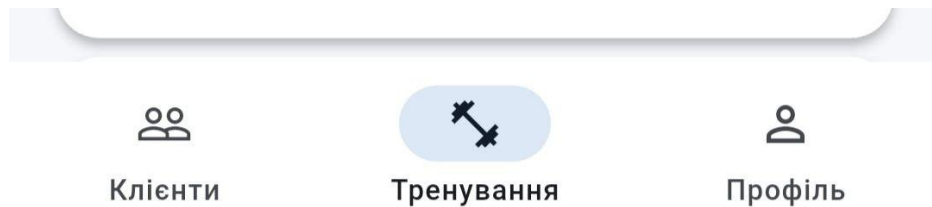


Рисунок 3.1 – Нижня навігаційна панель

Також на цьому екрані реалізовано кнопку зміни теми оформлення, що дозволяє користувачеві перемикатись між темним та світлим режимом інтерфейсу (рис. 3.2).

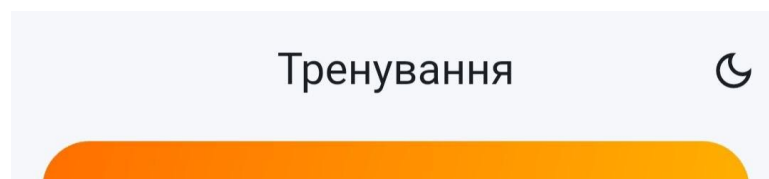


Рисунок 3.2 – Кнопка перемикання теми

Файл `profile_screen.dart` відповідає за реалізацію екрану профілю тренера, який забезпечує створення, перегляд та редагування особистої інформації користувача. Також автоматично відображаються різноманітні дані, що включають загальну кількість клієнтів і тренувань, які зберігаються в локальній базі даних Hive (ліст. 3.2). Реалізований функціонал дозволяє користувачу централізовано керувати власним профілем та швидко отримувати доступ до ключових показників роботи застосунку. На даному екрані тренер має можливість змінювати особисті дані, зокрема ім'я, спеціалізацію, стаж роботи та опис діяльності (рис. 3.3).

Лістинг 3.2 – Статистика клієнтів та тренувань

```
// Підрахунок загальної кількості клієнтів у базі
int getClientsCount() {
  final box = Hive.box<Client>('clientsBox');
  return box.length;
}

// Підрахунок загальної кількості тренувань по всіх клієнтах
int getWorkoutsCount() {
  final box = Hive.box<Client>('clientsBox');
```

```
// Підрахунок загальної кількості клієнтів у базі

int total = 0;

for (final client in box.values) {
    total += client.workouts.length;
}

return total;
```

кінець лістингу 3.2

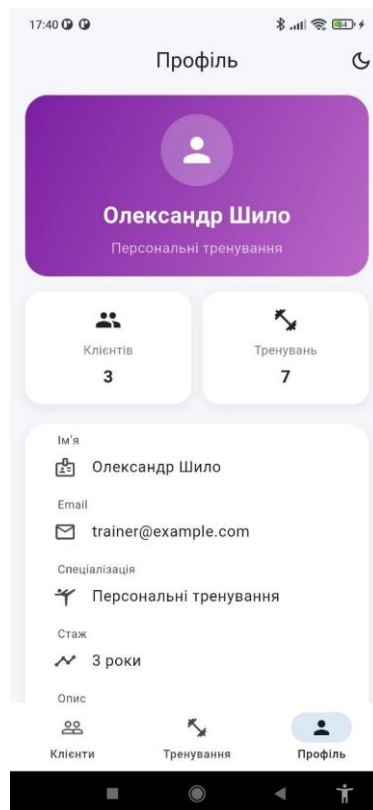


Рисунок 3.3 – Екран тренера

Файл `workouts_screen.dart` реалізує екран журналу тренувань, у якому відображаються всі тренування клієнтів, що забезпечує швидкий доступ до тренувального процесу. На даному екрані реалізовано функціонал пошуку тренувань за назвою тренування, датою проведення або ім'ям клієнта (рис. 3.4). Реалізований підхід до керування вправами забезпечує зручність взаємодії користувача із застосунком під час проведення тренування. Після відкриття тренування тренер має можливість безпосередньо на екрані тренування створювати, редагувати, або видаляти вправи без необхідності повторного

переходу до профілю клієнта. Такий підхід спрощує процес ведення тренувального процесу, скорочує кількість переходів між екранами та забезпечує швидкий доступ до функціоналу навіть після блокування або вимкнення мобільного пристрою.

Одночасно може тривати будь-яка кількість тренувань. Що дає змогу робити групові заняття.

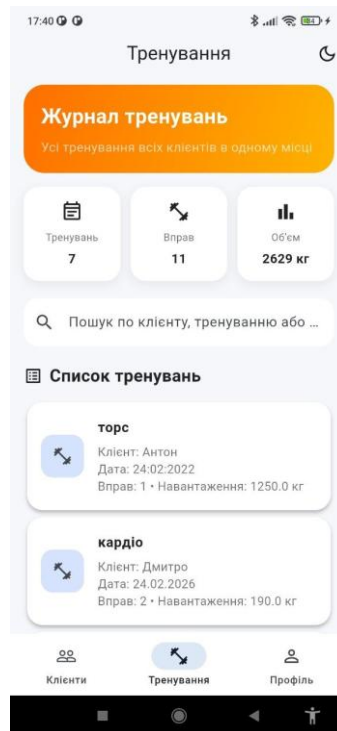


Рисунок 3.4 – Екран тренування

Екран забезпечує отримання даних із локальної бази Hive. Оскільки всі дані зберігаються в пам'яті пристрою можна реалізувати швидке отримання даних без підключення до мережі Інтернет. Для отримання даних всіх тренувань клієнтів використовується код (ліст. 3.3).

Лістинг 3.3 – Отримання всіх тренувань клієнта

```
// Отримання всіх тренувань усіх клієнтів
List<WorkoutItem> getAllWorkouts() {
    final List<WorkoutItem> allWorkouts = [];

    for (final client in clientsBox.values) {
        for (final workout in client.workouts) {
```

```

        allWorkouts.add(_WorkoutItem(client: client, workout:
workout));
    }
}

    allWorkouts.sort((a, b) =>
b.workout.date.compareTo(a.workout.date));

    if (searchQuery.trim().isEmpty) {
        return allWorkouts;
    }
}

```

кінець лістингу 3.3

Для пошуку тренувань у workouts_screen.dart реалізовано фільтрацію за ім'ям клієнта, датою проведення або назвою тренування. Даний підхід дозволяє швидко знаходити необхідне тренування серед великої кількості записів. Цей функціонал спрощує навігацію та підвищує ефективність роботи користувача із застосунком (ліст. 3.4).

Лістинг 3.4 – Пошук тренувань

```

return allWorkouts.where((item) {
    return item.client.name.toLowerCase().contains(query) ||
        item.workout.title.toLowerCase().contains(query) ||
        item.workout.date.toLowerCase().contains(query);
}).toList();

```

кінець лістингу 3.4

На екрані «Клієнти» (рис. 3.5), користувач застосунку має можливість переглядати список клієнтів, створювати нові облікові записи підопічних та видаляти непотрібні записи. Для додавання нового клієнта передбачено кнопку, розташовану в правому нижньому куті екрана та позначено символом «+». Використання такого елемента інтерфейсу відповідає сучасним принципам UI/UX-дизайну та є що є інтуїтивно зрозумілим для більшості користувачів. Даний функціонал спрощує взаємодію із застосунком і забезпечує швидкий доступ до функції створення нового клієнта.

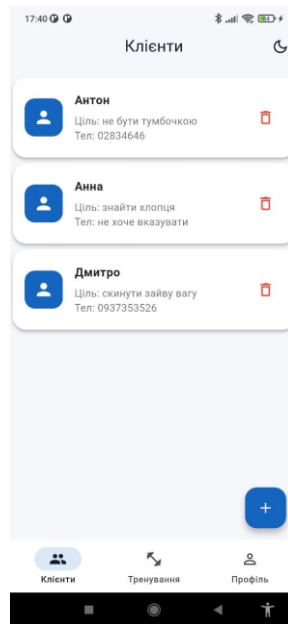


Рисунок 3.5 – Екран «Клієнти»

Програмний код який відповідає за реалізацію функціоналу створення нового клієнта, наведено в лістингу 3.5.

Лістинг 3.5 – Додавання нових клієнтів

```
final newClient = Client(
    name: nameController.text.trim(),
    phone: phoneController.text.trim(),
    weight: double.tryParse(weightController.text.trim()) ?? 0,
    height: double.tryParse(heightController.text.trim()) ?? 0,
    goal: goalController.text.trim(),
    workouts: [],
);
```

кінець лістингу 3.5

Для додавання нового клієнта необхідно заповнити форму «Новий клієнт» призначену для введення основної інформації про підопічного (рис. 3.6). У формі передбачено поля для введення імені клієнта, номера телефону, ваги, зросту та цілі тренування. Ім'я використовується для ідентифікації клієнта у системі, номер телефону – для забезпечення можливості зв'язку, а показники ваги та зросту дозволяють оцінити фізичний стан підопічного. Поле цілі тренувань використовується для визначення основного напрямку тренувального процесу та формування індивідуальної програми занять. Програмний код, що реалізує форму

додавання нового клієнта, наведено в додатку А.

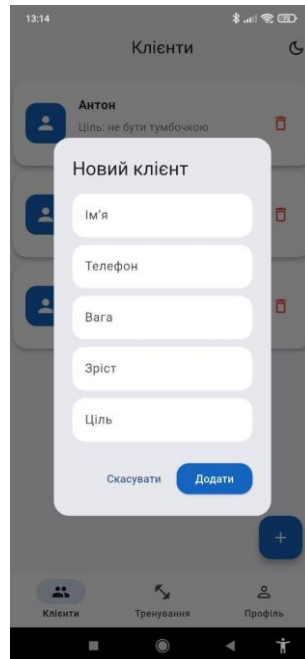


Рисунок 3.6 – Форма для даних клієнта

Після заповнення форми введені дані зберігаються у локальній базі даних застосунку та відображаються екрані «Клієнти».

Для видалення клієнтів використовується фрагмент коду, що продемонстровано в лістингу 3.6.

Лістинг 3.6 – Видалення клієнтів

```
void deleteClient(int index) async {
    await clientsBox.deleteAt(index);
    setState(() {});
}
```

кінець лістингу 3.6

Всі введені дані в файлі client.dart який зберігає їх в локальній базі даних код файлу наведено в лістингу 3.7.

Лістинг 3.7 – Збереження даних

```
import 'package:hive/hive.dart';
```

```

import 'workout.dart';

// Підключення згенерованого файлу (адаптера для Hive)
part 'client.g.dart';
// typeId – унікальний ідентифікатор
@HiveType(typeId: 0)
class Client extends HiveObject {
  @HiveField(0)
  String name;

  @HiveField(1)
  String phone;

  @HiveField(2)
  double weight;

  @HiveField(3)
  double height;

  @HiveField(4)
  String goal;

  @HiveField(5)
  List<Workout> workouts;
  //всі поля обов'язкові при створенні об'єкта

  Client({
    required this.name,
    required this.phone,
    required this.weight,
    required this.height,
    required this.goal,
    required this.workouts,
  });
}

```

кінець лістингу 3.7

Щоб отримати детальну інформацію про клієнта, користувачу необхідно натиснути на відповідний елемент у списку клієнтів, після чого відкриється екран детальної інформації про підопічного (рис. 3.7). За реалізацію даного екрана відповідає файл `client_details_screen.dart`, у якому забезпечується відображення основних даних клієнта, інформації про тренування та доступ до функцій керування тренувальним процесом. На цьому екрані користувач має змогу переглядати основну інформацію про клієнта.

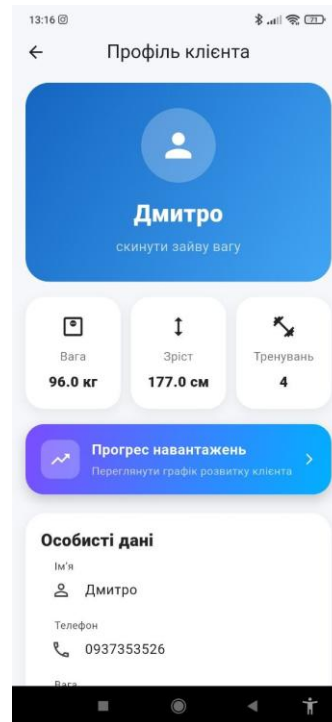


Рисунок 3.7 – Детальна інформація про клієнта

Даний екран надає користувачу можливість за потреби редагувати основні дані про клієнта. Реалізація такого функціоналу є важливою, оскільки тренувальний процес може тривати значний проміжок часу, упродовж якого можуть змінюватись фізичні показники підопічного, зокрема вага та зріст, а також контактна інформація. Можливість оперативного редагування даних дозволяє підтримувати актуальність інформації у системі та забезпечує коректне ведення обліку клієнтів.

Також екран забезпечує функціонал перегляду та додавання тренувань, що дозволяє тренеру формувати індивідуальні програми занять для кожного підопічного. Окрім цього, реалізовано можливість відстеження прогресу клієнта, що дає змогу аналізувати результати тренувального процесу та оцінювання зміни фізичних показників упродовж певного періоду часу.

Для призначення нового тренування користувачеві застосунку необхідно перейти до нижньої частини екрану та натиснути кнопку «Додати тренування» (рис. 3.8).

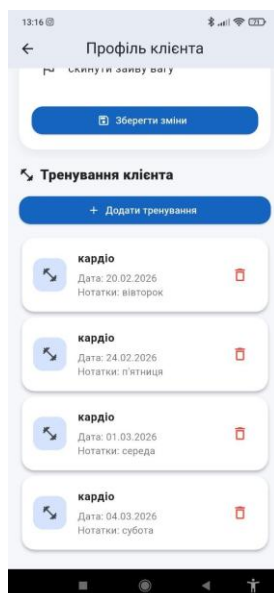


Рисунок 3.8 – Тренування клієнта

Після цього відкривається форма створення тренування, у якій потрібно заповнити основні дані, зокрема назву тренування, дату проведення та додаткові нотатки (рис. 3.9).

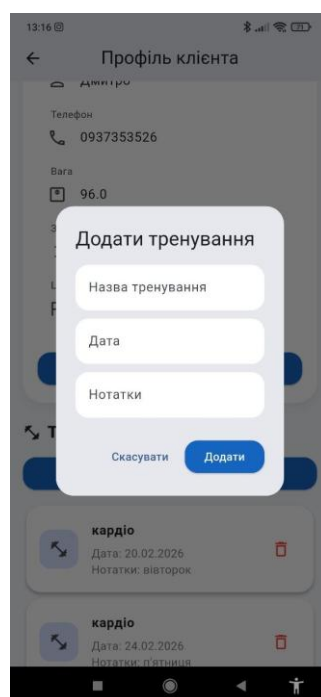


Рисунок 3.9 – Дані для тренування

Поле назви дозволяє коротко описати тип або мету тренування, дата

використовується для планування тренувального процесу, а нотатки дозволяють тренеру залишити коментар для тренування. Заповнені дані зберігаються у базі даних та відображаються у списку тренувань клієнта.

При натисканні на створене тренування відбувається перехід на екран тренування де реалізовано функціонал керування вправами. На даному екрані тренер має можливість створювати записи про вправи, редагувати вже існуючі або видаляти їх у разі необхідності (рис. 3.10).

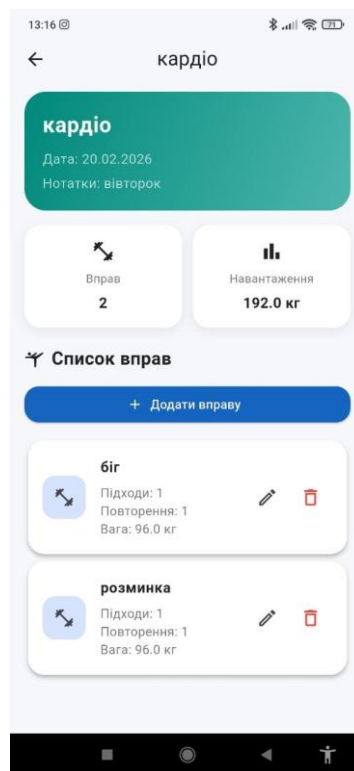


Рисунок 3.10 – Дані для тренування

Для кожної вправи передбачено введення основних параметрів, зокрема назви, необхідної кількості підходів, повторень та ваги навантаження. На основі введених даних система автоматично формує графік прогресу. Що забезпечує візуалізацію успіхів підопічного. Це дозволяє вести облік навантаження під час тренувального процесу та контролювати прогрес клієнта.

Якщо у всіх тренуваннях клієнта наявні більше трьох однакових вправ, розроблений застосунок отримує достатню кількість даних для побудови графіка навантажень (рис. 3.11), який відображається екрані детальної інформації про

клієнта. Даний графік дозволяє візуалізувати зміну тренувального навантаження впродовж певного періоду часу та забезпечує можливість аналізу процесу підопічного.



Рисунок 3.11 – Графік навантажень вправи біцепс

Для побудови графіка було використано пакет `fl_chart` [15], яка дозволяє створювати лінійні графіки у Flutter. Дані тренувань перетворюються у список точок `FlSpot`, після чого відображаються за допомогою віджета `LineChart`.

3.2 Реалізація інтерфейсу

Реалізація користувацького інтерфейсу (UI) мобільного застосунку для обліку клієнтів фітнес-тренерів була спрямована на створення інтуїтивно зрозумілого, візуально привабливого та функціонального середовища, яке забезпечує ефективну взаємодію користувача із програмою. Основний акцент під час розробки було зроблено на забезпеченні послідовного стилю оформлення, адаптивності елементів інтерфейсу та зручної навігації між основними функціями застосунку.

Під час проєктування інтерфейсу враховувались сучасні принципи UI/UX-дизайну, зокрема мінімалістичний стиль, логічне розташування елементів керування та швидкий доступ до ключового функціоналу. Для забезпечення комфортної роботи користувача було реалізовано структуру екранів, яка дозволяє легко переміщатись між списком клієнтів, тренуваннями, вправами та сторінками детальної інформації.

Також для всіх екранів було розроблено темний режим, він продемонстрований на рисунку 3.12.

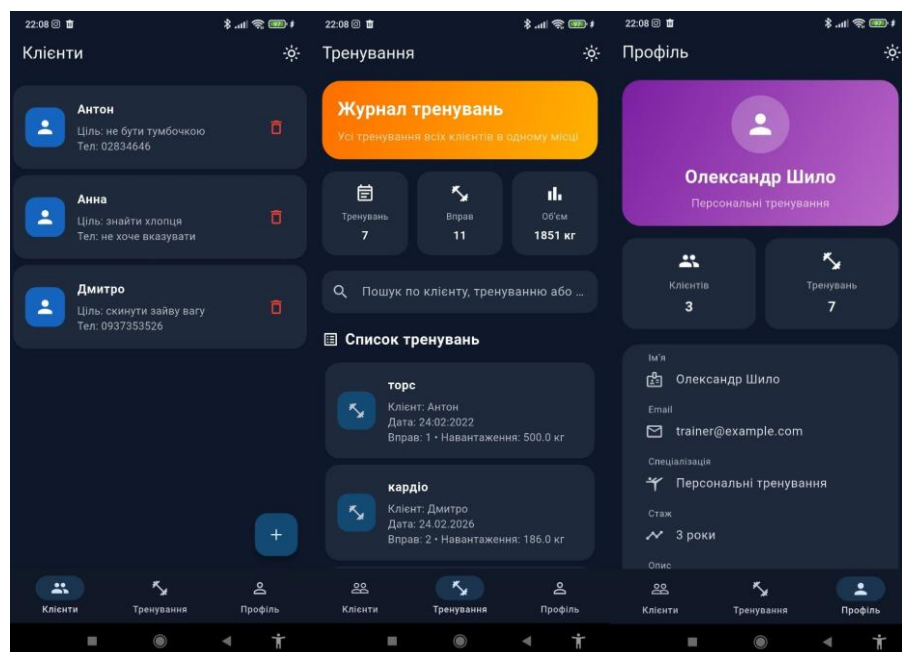


Рисунок 3.12 – Вигляд екранів в темному режимі

Темну тему реалізовано в файлі `main.dart` допомогою програмного коду, який наведено в додатку Б.

3.3 Тестування та налагодження інформаційно-комп'ютерної системи

Під час розробки мобільного застосунку для обліку клієнтів фітнес-тренерів було проведено комплексне тестування з метою перевірки коректності роботи функціоналу, стабільності користувацького інтерфейсу було виявлено помилки з темним режимом інтерфейсу рисунок 3.13.

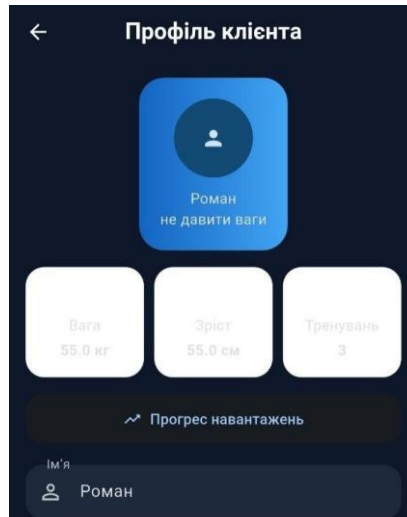


Рисунок 3.13 – Помилка з відображенням інтерфейсу

Багато часу було приділено правильності взаємодії з локальною базою даних, а також сумісності застосунку з різними мобільними пристроями. Велику увагу надано тестуванню навігації між екранами, роботі механізмів локального збереження даних Nive.

У межах тестування перевірялась коректність створення, редагування та видалення клієнтів, тренувань і вправ, а також правильність відображення інформації у списках та деталізованих екранах. Додатково було протестовано стабільність роботи застосунку під час повторного запуску, збереження даних після закриття програми та коректність побудови графіків навантаження на основі введених тренувальних даних.

Окремо здійснювалась перевірка адаптивності інтерфейсу на пристроях із різними розмірами екранів та версіями операційної системи Android. Проведене тестування підтвердило стабільність роботи мобільного застосунку, коректність реалізації основного функціоналу та відповідність програмного продукту поставленим вимогам.

3.4 Розробка інсталяційного пакета

Для розгортання мобільного застосунку, створеного за допомогою

фреймворку Flutter, потрібно сформувати інсталяційний пакет у форматі APK для операційної системи Android або IPA для iOS. У межах даного проєкту було реалізовано генерацію Android-пакета як приклад.

Процес створення інсталяційного пакета виконується за допомогою вбудованих засобів Flutter, зокрема із використанням команди «flutter build apk -release» [16]. Дана команда забезпечує створення оптимізованої release-версії мобільного застосунку, призначеної для подальшого встановлення на мобільні пристрої користувачів (рис.3.14).

```
PS C:\Users\sasha\StudioProjects\fitness_trainer_app> flutter build apk --release
Waiting for another flutter command to release the startup lock...
Running Gradle task 'assembleRelease'... 52,8s
✓ Built build\app\outputs\flutter-apk\app-release.apk (53.8MB)
PS C:\Users\sasha\StudioProjects\fitness_trainer_app> █
```

Рисунок 3.14 – Збірка APK-файлу

У результаті виконання процесу збірки формується APK-файл, який містить усі необхідні компоненти застосунку та може бути використаний для поширення, встановлення й тестування програмного продукту (рис.3.15).

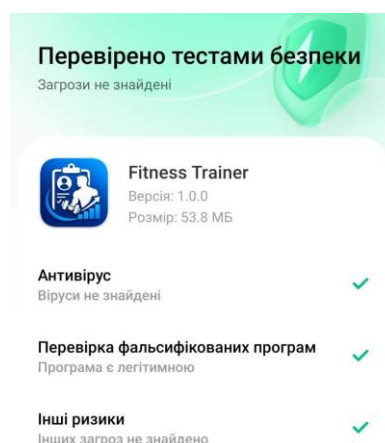


Рисунок 3.15 – Встановлений застосунок на пристрої Android

ВИСНОВКИ

У результаті виконання кваліфікаційної роботи було розроблено мобільний застосунок для обліку клієнтів фітнес-тренерів, та пройдено повний цикл розробки з використанням фреймворку Flutter, та локальної бази даних Hive. В проведеній роботі було охоплено, всі етапи від аналізу предметної області до формування інсталяційного пакета, та продемонстровано практичне застосування сучасних методів розробки програмного забезпечення.

Для досягнення мети, було проведено глибокий аналіз існуючих технічних рішень, а саме Trainerize, TrueCoach та Nike Training Club. Визначено всі функціональні, нефункціональні та технічні вимоги. Даний аналіз дозволив сформулювати бачення кінцевого продукту, врахувати можливі потреби користувачів щоб забезпечити відповідність системи існуючим стандартам якості та безпеки.

В результаті аналізу нативних рішень Java, Kotlin та Swift було обґрунтовано вибір кросплатформних технологій, серед них були розглянуті React Native та Flutter. Використання фреймворку Flutter та мови програмування Dart забезпечило кросплатформне рішення, високу продуктивність та ефективність продукту. Разом з базою даних Hive забезпечить модульність, гнучкість та легкість подальшого вдосконалення та супроводу системи.

Реалізація застосунку була здійснена з використанням обраних технологій. Завдяки модульній структурі проекту можна чітко розділити відповідальність між файлами Dart. Було реалізовано основні функціональні елементи, такі як персональний кабінет, відстеження тренувань клієнтів та сам список клієнтів.

Також було реалізовано адаптивний користувацький інтерфейс, який підтримує світлу та темну тему та має коректне відображення на різних розмірах екранів пристроїв, Було зроблено зручну навігацію між основними розділами застосунку. Візуальний дизайн характеризується мінімалізмом, з акцентом на інтуїтивно зрозумілу навігацію.

Значну увагу було приділено локальному збереженню інформації. Усі дані про клієнтів, тренування, вправи та профіль тренера зберігаються у локальній базі даних, що забезпечує швидкий доступ до інформації без необхідності підключення до мережі Інтернет.

Під час виконання роботи було проведено комплексне тестування функціоналу мобільного застосунку. Перевірено коректність роботи функцій додавання, редагування та видалення даних, стабільність роботи локальної бази даних Hive, правильність переходів між екранами, роботу графіків та адаптивність інтерфейсу. Результати тестування підтвердили працездатність програмного продукту та коректне функціонування основних можливостей системи.

Для розповсюдження застосунку був сформований інсталяційний APK-пакет, для подальшого розповсюдження застосунку на платформі Android. Що свідчить про повну готовність продукту для використання кінцевими користувачами.

В результаті виконання кваліфікаційної роботи бакалавра було повністю реалізовано мобільний застосунок для обліку клієнтів фітнес-тренерів, який повністю відповідає всім заявленим вимогам та демонструє високий рівень практичних навичок в сфері розробки мобільних рішень з використанням фреймворку Flutter.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Trainerize. URL: <https://www.trainerize.com/> (дата звернення: 10.02.2026).
2. Down Dog Official Website. URL: <https://checkmarkfitness.ca/onlinecoaching> (дата звернення: 11.02.2026).
3. TrueCoach. URL: <https://truecoach.com/> (дата звернення 12.02.2026).
4. TrueCoach. URL: <https://truecoach.co/wp-content/uploads/sites/2/2025/05/Second-Hero-1.jpg> (дата звернення 12.02.2026).
5. Nike Training Club. URL: <https://www.nike.com/gb/ntc-app> (дата звернення: 12.02.2026).
6. Nike Training Club. URL: <https://play.google.com/store/apps/details?id=com.nike.ntc&hl=uk> (дата звернення: 12.02.2026).
7. Java – Вікіпедія. URL: <https://uk.wikipedia.org/wiki/Java> (дата звернення: 12.02.2026).
8. Kotlin – Вікіпедія. URL: <https://uk.wikipedia.org/wiki/Kotlin> (дата звернення: 12.02.2026).
9. Свіфт – Вікіпедія. URL: <https://uk.wikipedia.org/wiki/Swift> (дата звернення: 12.02.2026).
10. React Native · Learn once, write anywhere. URL: <https://reactnative.dev/> (дата звернення: 12.02.2025).
11. Flutter – Build apps for any screen. URL: <https://flutter.dev/> (дата звернення: 05.03.2026).
12. Dart programming language. URL: <https://dart.dev/> (дата звернення: 07.03.2026).
13. Download Android Studio & App Tools – Android Developers. URL: <https://developer.android.com/studio> (дата звернення: 07.05.2025).
14. Hive – Dart API docs. URL: <https://pub.dev/documentation/hive/latest/>

(дата звернення: 10.04.2025).

15. Fl_chart – Flutter package. URL: https://pub.dev/packages/fl_chart (дата звернення: 10.04.2026).

16. Build and release an Android app. URL: <https://docs.flutter.dev/deployment/android> (дата звернення: 14.04.2026).

ДОДАТКИ

Додаток А

Код форми для заповнення даних клієнта

```

showDialog(
  context: context,
  builder: (context) {
    return AlertDialog(
      title: const Text('Новий клієнт'),
      shape: RoundedRectangleBorder(
        borderRadius: BorderRadius.circular(20),
      ),
      content: SingleChildScrollView(
        child: Column(
          mainAxisAlignment: MainAxisAlignment.min,
          children: [
            TextField(
              controller: nameController,
              decoration: const InputDecoration(labelText:
'Ім'я'),
            ),
            const SizedBox(height: 10),
            TextField(
              controller: phoneController,
              decoration: const InputDecoration(labelText:
'Телефон'),
            ),
            const SizedBox(height: 10),
            TextField(
              controller: weightController,
              keyboardType: TextInputType.number,
              decoration: const InputDecoration(labelText:
'Вага'),
            ),
            const SizedBox(height: 10),
            TextField(
              controller: heightController,
              keyboardType: TextInputType.number,
              decoration: const InputDecoration(labelText:
'Зріст'),
            ),
            const SizedBox(height: 10),
            TextField(
              controller: goalController,
              decoration: const InputDecoration(labelText:
'Ціль'),

```

```

        ),
      ],
    ),
  ),
  actions: [
    TextButton(
      onPressed: () => Navigator.pop(context),
      child: const Text('Скасувати'),
    ),
    ElevatedButton(
      onPressed: () async {
        if (nameController.text.trim().isEmpty) {
          final newClient = Client(
            name: nameController.text.trim(),
            phone: phoneController.text.trim(),
            weight:
double.tryParse(weightController.text.trim()) ?? 0,
            height:
double.tryParse(heightController.text.trim()) ?? 0,
            goal: goalController.text.trim(),
            workouts: [],
          );

          await clientsBox.add(newClient);
          setState(() {});
        }

        Navigator.pop(context);
      },
      child: const Text('Додати'),
    ),
  ],
);
},
);
}

```

Додаток Б

Код Світлої та темної теми інтерфейсу

```

ThemeData _buildLightTheme() {
  const seedColor = Color(0xFF1565C0);

  return ThemeData(
    useMaterial3: true,
    colorScheme: ColorScheme.fromSeed(
      seedColor: seedColor,
      brightness: Brightness.light,
    ),
    scaffoldBackgroundColor: const Color(0xFFFF5F7FB),

    appBarTheme: const AppBarTheme(
      centerTitle: true,
      elevation: 0,
      backgroundColor: Colors.transparent,
      foregroundColor: Color(0xFF1B1F24),
    ),

    navigationBarTheme: NavigationBarThemeData(
      backgroundColor: Colors.white,
      indicatorColor: seedColor.withOpacity(0.15),
    ),

    cardTheme: CardThemeData(
      elevation: 3,
      color: Colors.white,
      shape: RoundedRectangleBorder(borderRadius:
BorderRadius.circular(18)),
    ),

    inputDecorationTheme: InputDecorationTheme(
      filled: true,
      fillColor: Colors.white,
      border: OutlineInputBorder(
        borderRadius: BorderRadius.circular(16),
        borderSide: BorderSide.none,
      ),
    ),

    elevatedButtonTheme: ElevatedButtonThemeData(
      style: ElevatedButton.styleFrom(
        backgroundColor: seedColor,
        foregroundColor: Colors.white,
        shape: RoundedRectangleBorder(
          borderRadius: BorderRadius.circular(16),

```

```

        ),
    ),
),

floatingActionButtonTheme: const
FloatingActionButtonThemeData(
    backgroundColor: seedColor,
    foregroundColor: Colors.white,
),
);
}

// TEMHA TEMA
ThemeData _buildDarkTheme() {
    const seedColor = Color(0xFF42A5F5);

    return ThemeData(
        useMaterial3: true,
        colorScheme: ColorScheme.fromSeed(
            seedColor: seedColor,
            brightness: Brightness.dark,
        ),
        scaffoldBackgroundColor: const Color(0xFF0F172A),

        appBarTheme: const AppBarTheme(
            backgroundColor: Colors.transparent,
            foregroundColor: Colors.white,
            elevation: 0,
        ),

        navigationBarTheme: NavigationBarThemeData(
            backgroundColor: const Color(0xFF111827),
            indicatorColor: seedColor.withOpacity(0.2),
        ),

        cardTheme: CardThemeData(
            color: const Color(0xFF1E293B),
            shape: RoundedRectangleBorder(borderRadius:
BorderRadius.circular(18)),
        ),

        inputDecorationTheme: InputDecorationTheme(
            filled: true,
            fillColor: const Color(0xFF1E293B),
            border: OutlineInputBorder(
                borderRadius: BorderRadius.circular(16),
                borderSide: BorderSide.none,
            ),
        ),

        elevatedButtonTheme: ElevatedButtonThemeData(

```

```
        style: ElevatedButton.styleFrom(  
            backgroundColor: const Color(0xFF1976D2),  
            foregroundColor: Colors.white,  
        ),  
    ),  
);  
}
```