МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Чорноморський національний університет імені Петра Могили

Факультет комп'ютерних наук

Кафедра інженерії програмного забезпечення

ДОПУЩЕНО ДО ЗАХИСТУ

Завідувач кафедри інженерії

програмного забезпечення

_____ Євген ДАВИДЕНКО

«___» _____ 2025 р.

КВАЛІФІКАЦІЙНА РОБОТА

НА ЗДОБУТТЯ ОСВІТНЬОГО СТУПЕНЯ БАКАЛАВРА

ВЕБЗАСТОСУНОК УПРАВЛІННЯ ФІТНЕС-КЛУБОМ З ФУНКЦІЯМИ ПЕРСОНАЛІЗОВАНОГО ПЛАНУВАННЯ ТРЕНУВАНЬ

Спеціальність 121 Інженерія програмного забезпечення

Освітня програма «Інженерія програмного забезпечення»

Здобувач _____ **Максим МЕЛЬНИКОВ**

«__» _____ 2025 р.

Керівник роботи

PhD,

доцент (б.в.з) _____ **Катерина АНТІПОВА**

«__» _____ 2025 р.

Миколаїв – 2025

Завдання на виконання кваліфікаційної роботи

Чорноморський національний університет імені Петра Могили

Факультет	Комп'ютерних наук
Кафедра	Інженерії програмного забезпечення
Рівень вищої освіти	Перший (бакалаврський)
Освітній ступінь	Бакалавр
Спеціальність	121 Інженерія програмного забезпечення
Освітня програма	Інженерія програмного забезпечення

ЗАТВЕРДЖУЮ
Завідувач кафедри інженерії
програмного забезпечення
_____ Євген ДАВИДЕНКО
«___» _____ 2025 р.

ЗАВДАННЯ

на кваліфікаційну бакалаврську роботу здобувача

Мельникова Максима

1. Тема кваліфікаційної роботи Вебзастосунок управління фітнес-клубом з функціями персоналізованого планування тренувань затверджена наказом ректора ЧНУ ім. Петра Могили № 315 від «13» листопада 2024 р.
2. Строк представлення кваліфікаційної роботи «18» червня 2025 р.
3. Очікуваний результат роботи та початкові дані якщо такі потрібні.

Очікуваним результатом є вебзастосунок управління фітнес-клубом з функціями персоналізованого планування тренувань

4. Перелік питань, що підлягають розробці:

– дослідження предметної галузі та аналіз існуючих рішень;
– проєктування структури вебзастосунку;

– створення бази даних для зберігання інформації про фітнес-клуб;

– імплементування функціоналу для персоналізованого планування тренувань;

– проведення тестування вебзастосунку.

5. Перелік графічних матеріалів:

Презентація

6. Консультанти:

Консультант	**Кафедра (організація)**	**Частина роботи**

Дата видачі завдання «____» _____ 20___ р.

КАЛЕНДАРНИЙ ПЛАН
виконання кваліфікаційної роботи

Тема: **Вебзастосунок управління фітнес-клубом з функціями персоналізованого планування тренувань**

№	Найменування роботи	Початок	Закінчення	Примітки
1.	Розробка та затвердження завдання на виконання КБР	25.11.2024	26.11.2024	Виконано
2.	Огляд літератури за темою роботи	20.01.2025	24.01.2025	Виконано
3.	Складання календарного плану КБР	30.01.2025	31.01.2025	Виконано
4.	Аналіз предметної області та визначення вимог до вебзастосунку	03.02.2025	07.02.2025	Виконано
5.	Розробка проєктних рішень до КБР	10.02.2025	25.02.2025	Виконано
6.	Моделювання та конструювання ПЗ (проєктування бази даних, API)	26.02.2025	14.03.2025	Виконано
7.	Кодування, тестування та апробація розробленого ПЗ, аналіз результатів тестування	17.03.2025	18.04.2025	Виконано
8.	Оформлення КБР та презентації	21.04.2025	21.05.2025	Виконано
9.	Відгук керівника КБР	22.05.2025	22.05.2025	Виконано
10.	Попередній захист КБР	27.05.2025	27.05.2025	Виконано
11.	Завершення оформлення КБР та презентації	28.05.2025	02.06.2025	Виконано
12.	Рецензування	03.06.2025	03.06.2025	Виконано
13.	Захист кваліфікаційної роботи	23.06.2025	23.06.2025	Виконано

Здобувач _____ **Максим МЕЛЬНИКОВ**

«__» _____ 2025 р.

Керівник роботи

PhD,

доцент (б.в.з) _____ **Катерина АНТІПОВА**

«__» _____ 2025 р.

АНОТАЦІЯ

до кваліфікаційної бакалаврської роботи

Вебзастосунок управління фітнес-клубом з функціями персоналізованого планування тренувань

Здобувач 409 гр.: Мельников Максим

Керівник: PhD, доцент (б.в.з) Антіпова Катерина

Сучасний світ відзначається зростаючою популярністю здорового способу життя та фізичної активності серед населення. Все більше людей усвідомлюють важливість підтримки свого фізичного стану через відвідування фітнес-клубів. У зв'язку з цим зростає потреба в індивідуальному підході до тренувань, що враховує специфіку кожного клієнта, зокрема його фізичну підготовленість, цілі, графік та стан здоров'я.

Кваліфікаційна бакалаврська робота присвячена розробці вебзастосунку управління фітнес-клубом з функціями персоналізованого планування тренувань, що дозволить покращити якість обслуговування клієнтів, оптимізувати роботу тренерів та адміністраторів, а також підвищити ефективність тренувального процесу в цілому.

Об'єктом роботи є процес управління фітнес-клубом з послугою персоналізованого планування тренувань.

Предметом кваліфікаційної роботи є інструментарій розробки вебзастосунку управління фітнес-клубом та методи персоналізації тренувальних програм.

Метою кваліфікаційної роботи є розробка вебзастосунку управління фітнес-клубом з функціями персоналізованого планування тренувань для оптимізації роботи фітнес-закладу та покращення якості обслуговування клієнтів.

Для досягнення визначеної мети необхідно вирішити такі завдання:

1) дослідження предметної галузі управління фітнес-клубами та методів персоналізації тренувальних програм;

2) аналіз існуючих рішень та визначення ключових функцій вебзастосунку;

3) проєктування структури вебзастосунку, включаючи розробку інтерфейсу користувача, орієнтованого на зручність взаємодії;

4) створення бази даних для зберігання інформації про клієнтів, тренувальні плани, тренерів та ресурси клубу;

5) імплементування функціоналу для персоналізованого планування тренувань;

6) проведення тестування вебзастосунку з метою перевірки його функціональності та зручності використання.

Кваліфікаційна робота складається із вступу, чотирьох розділів, висновків та переліку джерел посилання.

У вступі обґрунтовано актуальність розробки вебзастосунку для управління фітнес-клубом, визначено мету, завдання, об'єкт та предмет дослідження.

У першому розділі проведено детальний аналіз предметної області управління фітнес-клубами, досліджено існуючі рішення та їх функціональні можливості.

Другий розділ присвячено моделюванню структури вебзастосунку, що розробляється, визначено вимоги до розроблюваного вебзастосунку.

У третьому розділі виконано проєктування системи з представленням низки UML-діаграм.

У четвертому розділі наведено етапи процесу розробки вебзастосунку та представлено результати тестування системи.

У висновках узагальнено результати виконаної роботи та отриманих результатів.

Кваліфікаційна робота викладена на 75 сторінках машинописного тексту, складається із вступу, 4 розділів, загальних висновків, переліку джерел посилання з 14 найменувань. Праця містить 4 таблиці та 39 рисунків.

Ключові слова: персоналізоване планування тренувань, управління фітнес-клубом, штучний інтелект, Laravel, React.

ABSTRACT

to the qualifying bachelor's thesis

Web-based fitness club management application with personalized training planning functions

Student of 409 group: Melnykov Maksym

Supervisor: PhD, Associate Professor Antipova Kateryna

The modern world is marked by the growing popularity of healthy lifestyles and physical activity among the population. More and more people are realizing the importance of maintaining their physical condition by visiting fitness clubs. In this regard, there is a growing need for an individualized approach to training that takes into account the specifics of each client, including their physical fitness, goals, schedule, and health status.

The bachelor's thesis is devoted to the development of a fitness club management web application with personalized training planning functions, which will improve the quality of customer service, optimize the work of trainers and administrators, and increase the efficiency of the training process in general.

The object of work is the process of managing a fitness club with a personalized training planning service.

The subject of the qualification work is the tools for developing a fitness club management web application and methods for personalizing training programs.

The purpose of the qualification work is to develop a web application for managing a fitness club with personalized training planning functions to optimize the operation of the fitness facility and improve customer service quality.

To achieve this goal, it is necessary to solve the following tasks:

1) study of the subject area of fitness club management and methods of personalization of training programs;

2) analyze existing solutions and identify key functions of the web application;

3) designing the structure of the web application, including the development of a user interface focused on ease of interaction;

4) creating a database to store information about clients, training plans, coaches, and club resources;

5) implementing functionality for personalized training planning;

6) testing the web application to check its functionality and usability.

The qualification work consists of an introduction, four chapters, conclusions and a list of references.

The introduction substantiates the relevance of developing a web application for fitness club management, defines the purpose, objectives, object and subject of the study.

The first section provides a detailed analysis of the subject area of fitness club management, examines existing solutions and their functionality.

The second section is devoted to modeling the structure of the web application under development, defining the requirements for the developed web application.

The third section describes the design of the system with the presentation of a number of UML diagrams.

The fourth section describes the stages of the web application development process and presents the results of system testing.

The conclusions summarize the results of the work performed and the results obtained.

The qualification work is presented on 75 pages of typewritten text, consists of an introduction, 4 sections, general conclusions, a list of references with 14 titles. The work contains 4 tables and 39 illustrations.

Keywords: artificial intelligence, fitness club management, Laravel, personalized training planning, React.

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ

БД – база даних

ПЗ – програмне забезпечення

ШІ – штучний інтелект

API – Application Programming Interface

HTTP – Hypertext Transfer Protocol

HTTPS – Hyper Text Transfer Protocol Secure

UML – Unified Modeling Language

ВСТУП

Сучасний світ відзначається зростаючою популярністю здорового способу життя та фізичної активності серед населення. Все більше людей усвідомлюють важливість підтримки свого фізичного стану через відвідування фітнес-клубів. У зв'язку з цим зростає потреба в індивідуальному підході до тренувань, що враховує специфіку кожного клієнта, зокрема його фізичну підготовленість, цілі, графік та стан здоров'я.

Особливо актуальним це стає в умовах зростання кількості людей, які працюють або навчаються дистанційно. Такий спосіб життя часто призводить до зниження фізичної активності, оскільки люди проводять більше часу за комп’ютером у домашніх умовах. Для багатьох із них відвідування фітнес-клубу стає не лише способом підтримки фізичної форми, але й можливістю відпочити, змінити оточення та поспілкуватися з іншими людьми, що позитивно впливає на емоційний стан і якість життя.

Традиційні методи управління клієнтами у фітнес-клубах, які передбачають ручну працю адміністратора та тренера, стають менш ефективними в умовах зростання кількості відвідувачів. Автоматизація таких процесів, як реєстрація клієнтів, створення персоналізованих програм тренувань, моніторинг прогресу та управління розкладом, є критично важливою для підвищення якості послуг та конкурентоспроможності фітнес-клубів.

Розробка вебзастосунку управління фітнес-клубом з функціями персоналізованого планування тренувань є актуальною темою, оскільки вона відповідає сучасним тенденціям у сфері фітнесу та здорового способу життя. Персоналізація тренувань стає дедалі важливішою для досягнення оптимальних результатів, а автоматизація процесів управління фітнес-клубом дозволяє підвищити ефективність роботи закладу.

Практичне значення цієї теми полягає в розробці інноваційного підходу до управління фітнес-клубом, який поєднує сучасні технології веброзробки з методами персоналізації тренувальних програм. Це дозволить покращити якість

обслуговування клієнтів, оптимізувати роботу тренерів та адміністраторів, а також підвищити ефективність тренувального процесу в цілому.

Метою кваліфікаційної роботи є розробка вебзастосунку управління фітнес-клубом з функціями персоналізованого планування тренувань для оптимізації роботи фітнес-закладу та покращення якості обслуговування клієнтів.

Для досягнення визначеної мети необхідно вирішити такі **завдання**:

1) дослідження предметної галузі управління фітнес-клубами та методів персоналізації тренувальних програм;

2) аналіз існуючих рішень та визначення ключових функцій вебзастосунку;

3) проєктування структури вебзастосунку, включаючи розробку інтерфейсу користувача, орієнтованого на зручність взаємодії;

4) створення бази даних для зберігання інформації про клієнтів, тренувальні плани, тренерів та ресурси клубу;

5) імплементування функціоналу для персоналізованого планування тренувань;

6) проведення тестування вебзастосунку з метою перевірки його функціональності та зручності використання.

Об'єктом роботи є процес управління фітнес-клубом з послугою персоналізованого планування тренувань.

Предметом кваліфікаційної роботи є інструментарій розробки вебзастосунку управління фітнес-клубом та методи персоналізації тренувальних програм.

Існуючі системи управління фітнес-клубами здебільшого забезпечують лише базові функції бронювання та моніторингу, але не враховують особисті дані користувача для персоналізації тренувань. Вебзастосунок матиме модульну архітектуру: адаптивний фронтенд на Next.js, RESTful API бекенд на Laravel, Python сервіс для інтеграції ШІ та БД MySQL. Таке рішення підходить для комерційних фітнес-клубів та онлайн-платформ дистанційних тренувань.

1 АНАЛІЗ ПРОЦЕСІВ УПРАВЛІННЯ ФІТНЕС-КЛУБОМ

1.1 Цифровізація процесів управління фітнес-клубом

Управління фітнес-клубом у теперішніх реаліях потребує інтеграції цифрових інструментів, що гарантують ефективну роботу, індивідуальний підхід до клієнтів та оптимізацію внутрішніх операцій. Зростаюча конкуренція, зміна споживчих уподобань та впровадження інноваційних рішень сприяють трансформації традиційних моделей ведення бізнесу у сфері фітнесу. На початок 2025 року світовий ринок здоров'я та фітнесу оцінюється в 5,11 мільярда доларів США, що свідчить про зростання на 13,1% порівняно з попереднім роком [1]. Така динаміка підкреслює зростаючий попит на інноваційні рішення в галузі фітнесу.

Особливу увагу привертає розвиток віртуального фітнесу. За прогнозами, до 2030 року глобальний ринок віртуального фітнесу досягне 106,4 мільярда доларів США, демонструючи середньорічний темп зростання (CAGR) у 26,7% [2]. Це свідчить про зміну підходів до тренувань та зростання популярності онлайн-платформ для занять спортом (рис. 1.1).

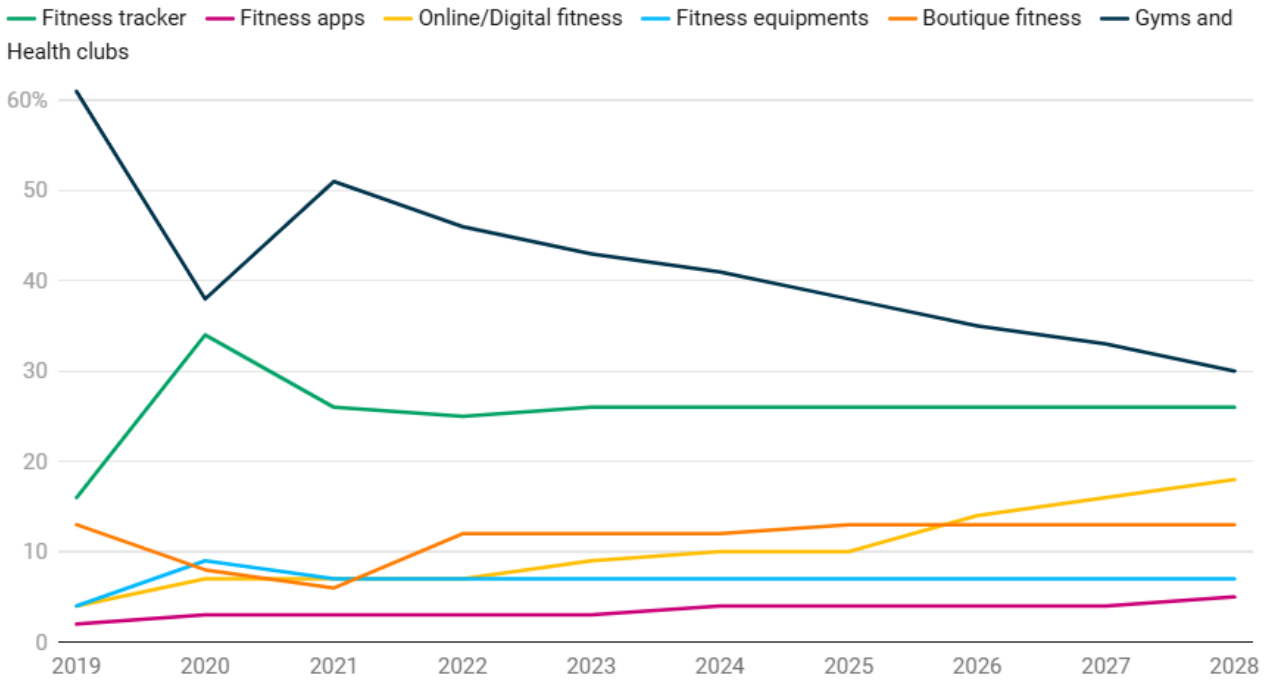

Рисунок 1.1 – Частка ринку фітнес-індустрії за сегментами

Сучасні відвідувачі фітнес-залів бажають персоналізованого підходу до тренувань. Індивідуальні програми, що створені враховуючи фізичні дані, поставлені цілі та особисті переваги, допомагають підвищити мотивацію та результативність тренувань. Реалізація таких програм стає можливою завдяки використанню цифрових платформ, які аналізують дані користувачів та пропонують оптимальні рішення.

Цифрова трансформація стала невід'ємною складовою фітнес-індустрії. У 2023 році приблизно 81% клієнтів застосовували застосунки на своїх гаджетах [3]. Ринок фітнес-трекерів теж показує значне зростання, з прогнозованим обсягом у $66,48 мільярди у 2024 році та передбачуваним збільшенням до $362,96 мільярди до 2034 року [4]. Ці технології дозволяють користувачам моніторити фізичну активність, контролювати свій прогрес та отримувати рекомендації стосовно тренувань.

Крім того, пандемія COVID-19 залишила глибокий відбиток на фітнес-індустрії, особливо прискоривши поширення домашніх тренувань та онлайн-сервісів. Наприклад, у США 33% населення почали віддавати перевагу стрімінговим відео для фітнесу вдома, а 29% придбали обладнання для домашніх тренувань [5]. Така трансформація підкреслює необхідність адаптації фітнес-клубів до нових умов та впровадження гібридних моделей обслуговування.

В Україні фітнес-індустрія теж демонструє позитивну динаміку. Збільшується кількість фітнес-центрів, все більше людей обирає онлайн-курси, відеотренування та застосунки для відстеження прогресу. Ще впроваджуються сучасні технології та розробляються новаторські програми тренувань з урахуванням стану здоров’я, способу життя та цілей клієнтів [6].

Однак, відчувається необхідність у подальшій цифровізації процесів управління, особливо у впровадженні вебзастосунків для оптимізації та ефективності роботи клубів та надання персоналізованих послуг клієнтам.

В цілому, фітнес-індустрія переживає швидкий розвиток, зокрема завдяки інтеграції цифрових технологій та індивідуалізованих підходів до обслуговування клієнтів. Глобальні тенденції, викликані змінами в споживчих вподобаннях та

впливом пандемії, акцентують на необхідності адаптації фітнес-клубів до нових реалій.

1.2 Компоненти у системі управління фітнес-клубом

В сучасних реаліях ринкової економіки, успіх фітнес-бізнесу значною мірою визначається ефективним управлінням та орієнтацією на клієнта. Це набуває особливої ваги в умовах жорсткої конкуренції та підвищення вимог споживачів до якості сервісу. Згідно з інформацією Global Wellness Institute, світовий ринок фітнес-послуг демонструє стабільне щорічне зростання на 5-7%, що підкреслює перспективність цього бізнесу. В цьому контексті, персоналізація послуг, зокрема індивідуальне планування тренувань, стає не просто перевагою, а обов'язковою умовою для утримання та розширення клієнтської бази.

Управління фітнес-клубом, який пропонує персоналізоване планування тренувань, представляє собою комплексну систему, що складається з взаємопов'язаних бізнес-процесів. Їхня мета – забезпечення ефективної роботи закладу та задоволення потреб клієнтів. Дослідження Американської асоціації фітнесу підтверджують, що фітнес-клуби, які впроваджують індивідуальні програми тренувань, мають на 23% вищий рівень утримання клієнтів у порівнянні з тими, хто використовує стандартні програми.

Структурний аналіз об'єкта дослідження дозволяє виокремити декілька ключових компонентів у системі управління фітнес-клубом:

1) Адміністративний блок:
- управління персоналом (підбір, навчання, оцінка ефективності);
- стратегічне планування діяльності клубу;
- маркетинг та просування послуг;
- взаємодія з партнерами та постачальниками.

2) Клієнтський блок:
- залучення клієнтів та продаж абонементів;
- реєстрація та ведення клієнтської бази;
- система лояльності та утримання клієнтів;

- комунікація з клієнтами та зворотний зв'язок.

3) Тренувальний блок:

- розробка і реалізація загальних програм тренувань;
- персоналізоване планування тренувань;
- групові заняття та їх розклад;
- моніторинг прогресу клієнтів.

4) Фінансово-господарчий блок:

- бухгалтерський облік та звітність;
- управління матеріально-технічною базою;
- контроль витрат та доходів;
- розрахунки із співробітниками та податковими органами.

5) Інформаційно-технологічний блок:

- забезпечення функціонування програмних рішень;
- інтеграція з зовнішніми системами;
- захист даних та інформаційна безпека;
- технічна підтримка користувачів.

Особливу роль у системі управління фітнес-клубом відіграє складова персоналізованого планування тренувань, яка слугує водночас як частина тренувального процесу і як окрема функціональна підсистема. За даними досліджень Міжнародної асоціації здоров'я та фітнесу (IHRSA), аж 67% відвідувачів фітнес-клубів визначають індивідуальний підхід до складання плану тренувань як ключовий фактор під час вибору закладу [7].

Функціональний аналіз процесу управління фітнес-клубом розкриває складну систему взаємозалежних операцій, що спільно забезпечують повний цикл обслуговування клієнтів та підтримку бізнес-процесів. Детальніше далі розглянуто основні функціональні блоки.

Адміністративний блок виконує функції планування, організації та контролю діяльності фітнес-клубу в цілому. Керівництво визначає стратегічні напрямки розвитку, формує корпоративну культуру, визначає ключові показники ефективності для різних підрозділів та співробітників. За даними Harvard Business

Review, якість управлінських рішень безпосередньо впливає на фінансові показники фітнес-бізнесу, причому ефективність прийняття рішень збільшується на 68% при використанні спеціалізованих систем управління.

Клієнтський блок забезпечує залучення нових клієнтів, їх реєстрацію в системі, продаж абонементів та інших послуг, а також підтримку постійного зв'язку з клієнтами. В межах цього блоку відбувається збір та аналіз інформації про клієнтів, їхні потреби та вподобання, що є критично важливим для подальшої персоналізації послуг.

Тренувальний блок відповідає за безпосереднє надання основних послуг фітнес-клубу: проведення групових та індивідуальних тренувань, розробку програм занять, моніторинг фізичного стану клієнтів. Особливе значення в цьому блоці має підсистема персоналізованого планування тренувань, яка дозволяє розробляти індивідуальні програми з урахуванням особливостей кожного клієнта: фізичної підготовки, цілей, наявних обмежень, графіка відвідувань тощо.

Процес персоналізованого планування тренувань включає наступні етапи:

1) первинна діагностика фізичного стану клієнта;
2) визначення цілей та обмежень;
3) розробка індивідуальної програми тренувань;
4) періодичний моніторинг прогресу;
5) коригування програми на основі досягнутих результатів.

Фінансово-господарчий блок гарантує економічну стабільність і операційну здатність фітнес-клубу. В рамках цього блоку здійснюється нагляд за надходженнями та витратами, формування цінової стратегії, облік матеріальних цінностей, виплата зарплат працівникам, сплата податків та інших обов'язкових виплат. Згідно з дослідженнями, більше половини малих та середніх підприємств у сфері фітнес-послуг зазнають труднощів через недостатню автоматизацію фінансового обліку.

Інформаційно-технологічний блок відповідає за технічне забезпечення всіх бізнес-процесів фітнес-клубу через впровадження та підтримку програмних рішень, забезпечення захисту інформації, інтеграцію з зовнішніми сервісами

(наприклад, платіжними системами, SMS-повідомленнями тощо). У нинішніх умовах цей блок набуває надзвичайної важливості, оскільки від його ефективності залежить можливість реалізації інших функцій, особливо персоналізованого планування тренувань, що потребує обробки значних обсягів особистих даних.

Вивчення актуальних методів управління фітнес-центрами демонструє певні недоліки та перешкоди, що негативно впливають на загальну продуктивність:

1) значний обсяг ручної роботи та використання паперової документації, що підвищує ризик помилок та сповільнює обробку даних. Наразі, близько 42% фітнес-клубів досі послуговуються паперовими журналами або базовими електронними таблицями для обліку клієнтів та планування тренувань;

2) відсутність налагодженої взаємодії між різними функціональними підрозділами, що зумовлює дублювання інформації та неузгодженість даних;

3) обмежені можливості для ретельної персоналізації тренувальних програм через брак інструментів аналізу даних та підтримки прийняття рішень. Дослідження Journal of Strength and Conditioning Research свідчить, що лише 17% персональних тренерів використовують спеціалізоване ПЗ для розробки індивідуальних програм тренувань [8];

4) складність у розширенні послуг персоналізованого планування тренувань, пов'язана з великою залежністю від кваліфікації та досвіду окремих тренерів;

5) недостатня автоматизація комунікаційних процесів з клієнтами, що знижує ефективність зворотного зв'язку та оперативного реагування на запити.

Враховуючи виявлені проблеми та обмеження, можна сформулювати ключові вимоги до вебзастосунку управління фітнес-клубом з функціями персоналізованого планування тренувань:

1) комплексна автоматизація основних бізнес-процесів фітнес-клубу з можливістю гнучкого налаштування під специфіку конкретного закладу;

2) єдина інтегрована система, що забезпечує узгодженість даних між різними функціональними блоками та мінімізує дублювання інформації;

3) новітні інструменти для персоналізованого планування тренувань, включаючи збір та аналіз даних про фізичний стан клієнтів, їхні цілі та прогрес;

4) засоби аналітики та звітності для оцінки ефективності роботи фітнес-клубу в цілому та окремих напрямків діяльності;

5) масштабованість рішення з можливістю адаптації до зростання бізнесу та розширення спектру послуг.

Особливе значення має інтеграція вебзастосунку з зовнішніми системами, такими як платіжні сервіси, системи електронної пошти, мобільні застосунки для відстеження фізичної активності тощо. Ще одним ключовим фактором є прийняття до уваги трендів, що визначають тенденції розвитку фітнес-індустрії. Зокрема, це персоналізація сервісу, перехід у цифровий формат, застосування штучного інтелекту для обробки інформації та передбачення результативності занять. До того ж, варто враховувати впровадження інтернету речей задля відстеження перебігу тренувань та покращення їх ефективності.

Аналіз структурно-функціональних особливостей процесу управління фітнес-клубом показує, що успішна реалізація вебзастосунку з функціями персоналізованого планування тренувань вимагає комплексного підходу, який враховує всі аспекти діяльності фітнес-клубу та забезпечує інтеграцію різних функціональних блоків у єдину систему. При цьому особлива увага має бути приділена розробці алгоритмів персоналізації тренувань, які дозволять автоматизувати цей процес без втрати якості та індивідуального підходу до кожного клієнта.

Таким чином, розробка вебзастосунку управління фітнес-клубом з функціями персоналізованого планування тренувань представляє собою комплексне завдання, що вимагає глибокого розуміння як бізнес-процесів фітнес-індустрії, так і технологічних аспектів створення сучасних рішень. Успішна реалізація такого застосунку дозволить підвищити ефективність управління фітнес-клубом, покращити якість послуг та збільшити задоволеність клієнтів, що, в свою чергу, позитивно вплине на конкурентоспроможність та фінансові показники бізнесу.

1.3 Огляд застосунків-аналогів

У сучасному фітнес-бізнесі ПЗ для керування клубом – це невід'ємна частина. Воно робить бізнес ефективним, автоматизуючи повсякденні завдання, покращуючи комунікацію з клієнтами, зменшуючи адміністративні турботи та підвищуючи якість послуг. Ринок ПЗ для фітнес-клубів швидко росте, пропонуючи широкий вибір варіантів – від простих систем фіксації відвідувань до складних платформ із функціями персоналізованого планування тренувань, онлайн-запису, фінансового аналізу та інтеграції з мобільними застосунками.

Аналіз існуючих застосунків-аналогів – ключовий етап у процесі створення нового вебзастосунку для керування фітнес-клубом із функціями персоналізованого планування тренувань. Він дає можливість визначити переваги та недоліки вже наявних рішень, виокремити функціонал, який найбільше затребуваний власниками фітнес-клубів та їх відвідувачами, а також відшукати незаповнені ніші чи слаборозвинені функціональні компоненти, котрі здатні стати конкурентною перевагою нового продукту.

З-поміж численних доступних варіантів на ринку, на особливу увагу заслуговують такі платформи, як Mindbody (табл. 1.1), Glofox (табл. 1.2) і GymMaster (табл 1.3). Кожна з них пропонує унікальний набір функцій, розроблених для задоволення потреб фітнес-бізнесу різного масштабу та спрямування. Далі розглянуто ключові особливості цих програмних продуктів, їх сильні сторони та слабкі місця.

Mindbody

Mindbody – це одна з найбільш популярних платформ для управління фітнес-бізнесом, що пропонує комплексне рішення для фітнес-клубів, спа-салонів та оздоровчих центрів. Система забезпечує широкий спектр функцій: від онлайн-бронювання та управління розкладом до обробки платежів та маркетингових інструментів [9]. Mindbody (рис. 1.2) особливо відомий своїми можливостями інтеграції з мобільними застосунками, такими як Fitbit та MyFitnessPal, що дозволяє клієнтам відстежувати свій прогрес та маркетплейсом, який дозволяє користувачам

знаходити фітнес-послуги поблизу. Однак, за відгуками користувачів на Capterra, система має досить складний інтерфейс, що вимагає тривалого навчання персоналу, а також відносно високу вартість, особливо для малих фітнес-клубів.

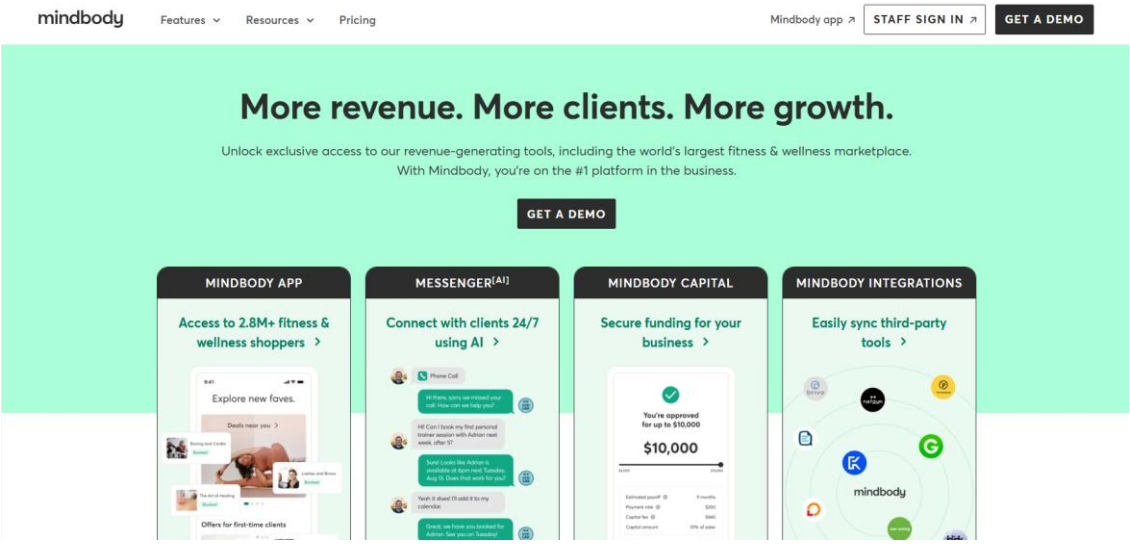

Рисунок 1.2 – Вигляд інтерфейсу Mindbody

Таблиця 1.1 – Опис Mindbody

Назва	Mindbody
Розробник	Mindbody Inc
Архітектура	Cloud-based (3-tier web application), mobile application
Мова реалізації	.NET, JavaScript, React
Функції	– онлайн-бронювання та управління розкладом; – обробка платежів та управління клієнтами; – мобільний застосунок та інтеграція з маркетплейсом; – маркетингові інструменти та автоматизація; – звітність та аналітика; – інтеграція з POS-системами; – інтеграція з іншими бізнес-інструментами (Fitbit, MyFitnessPal).
Переваги	– гнучка система налаштувань; – потужна маркетплейс-екосистема; – широкі можливості інтеграцій; – регулярні оновлення та розвиток; – масштабованість для великих мереж.

Кінець таблиці 1.1

Недоліки	– висока вартість; – складний інтерфейс; – тривале навчання персоналу; – обмежені можливості персоналізації тренувань; – недостатня гнучкість для специфічних бізнес-моделей.
Джерело інформації	https://www.mindbodyonline.com

Glofox

Glofox – це сучасна хмарна платформа для управління фітнес-бізнесом, яка акцентує увагу на автоматизації адміністративних процесів та покращенні взаємодії з клієнтами. Система пропонує інструменти для управління членством, розкладом, платежами, маркетингом та аналітикою (рис. 1.3). Особливістю Glofox є можливість створення персоналізованих маршрутів для клієнтів на основі їхньої активності та вподобань.

Платформа також підтримує інтеграцію з соціальними мережами та надає гнучкі рішення для бізнесів різного масштабу. Крім того, Glofox має потужні маркетингові інструменти та брендований мобільний застосунок, який клуби можуть пропонувати своїм клієнтам під власним брендом [10].

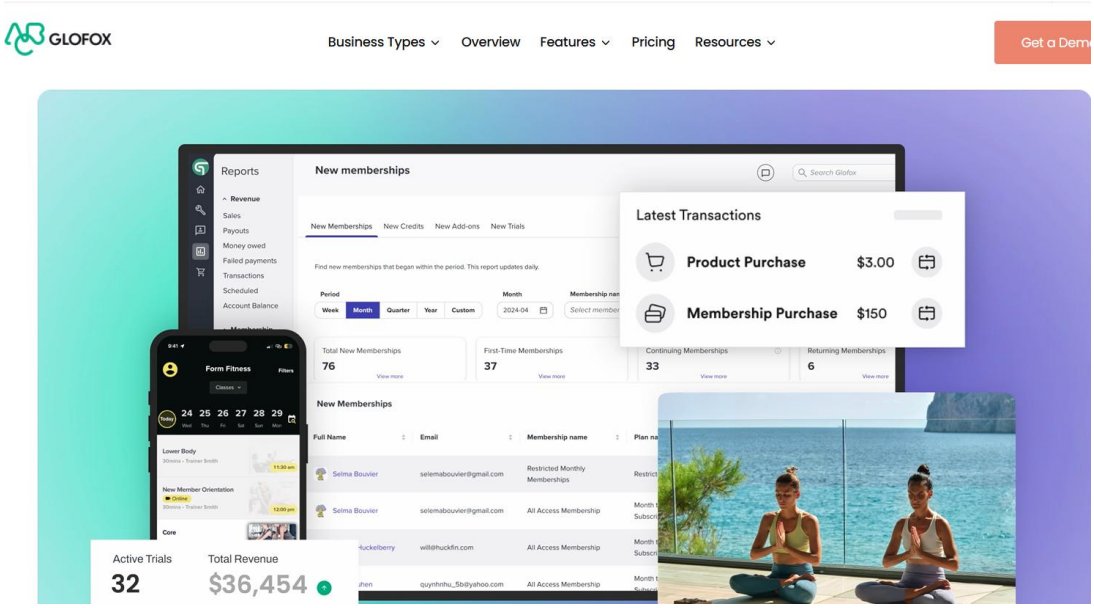

Рисунок 1.3 – Вигляд інтерфейсу Glofox

Таблиця 1.2 – Опис Glofox

Назва	Glofox
Розробник	ABC Glofox
Архітектура	Cloud-based (3-tier web application)
Мова реалізації	JavaScript, Node.js
Функції	– брендований мобільний застосунок; – управління членством та абонементами; – онлайн-бронювання та розклад занять; – автоматизовані маркетингові кампанії; – обробка платежів та інтеграція з платіжними системами; – інструменти комунікації з клієнтами.
Переваги	– інтуїтивний інтерфейс; – сильні мобільні рішення; – гнучкість брендингу; – орієнтація на малий та середній бізнес; – якісна підтримка клієнтів.
Недоліки	– обмежена аналітика; – недостатньо розвинені функції фінансового обліку; – обмежені можливості офлайн-роботи; – менше інтеграцій з іншими сервісами.
Джерело інформації	https://www.glofox.com/

GymMaster

GymMaster – це ПЗ для управління фітнес-клубами, яке пропонує комплексне рішення для контролю доступу, управління членством, розкладом та фінансового обліку. Система підтримує автоматизовані повідомлення, онлайн-бронювання та мобільний застосунок для клієнтів. Однією з ключових особливостей GymMaster є інтеграція з системами контролю доступу та інтеграцією з різними типами турнікетів та біометричними системами, що робить його цінним для середніх та великих фітнес-центрів з розвиненою інфраструктурою.

GymMaster (рис. 1.4) пропонує одні з найкращих функцій для управління відносинами з клієнтами (CRM) у своєму сегменті, проте поступається конкурентам у можливостях для онлайн-продажів та маркетингу. Користувачі

відзначають високу надійність системи та гнучкість налаштувань, але вказують на певні обмеження в інтеграції з деякими сторонніми сервісами [11].

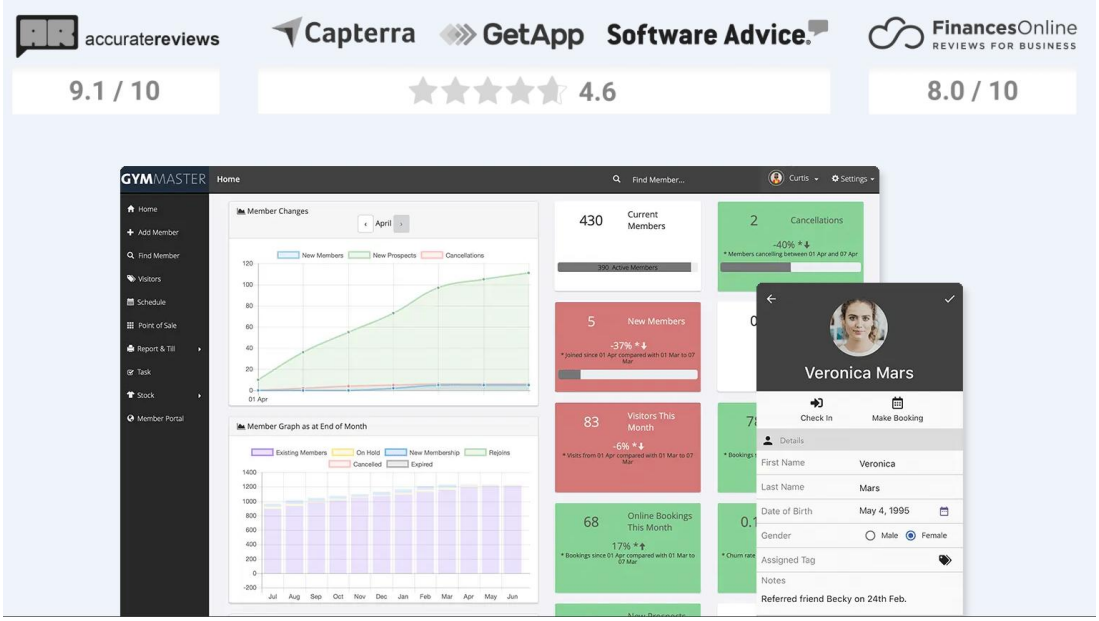

Рисунок 1.4 – Вигляд інтерфейсу GymMaster

Таблиця 1.3 – Опис GymMaster

Назва	GymMaster
Розробник	Treshna Enterprises Ltd
Архітектура	Cloud-based (3-tier web application)
Мова реалізації	C#, .NET, JavaScript
Функції	– контроль доступу та інтеграція з турнікетами; – управління членством та абонементами; – CRM та комунікація з клієнтами; – фінансовий облік та звітність; – управління розкладом та ресурсами; – мобільний застосунок для клієнтів; – використання штучного інтелекту для оптимізації маркетингу і утримання тренажерного залу.
Переваги	– гнучкі варіанти розгортання; – потужний контроль доступу; – детальна фінансова звітність; – надійність роботи; – великий вибір інтеграцій з обладнанням.

Кінець таблиці 1.3

Недоліки	– застарілий дизайн інтерфейсу; – обмежені маркетингові інструменти; – складність інтеграції з деякими сервісами; – недостатній розвиток онлайн-продажів; – менш розвинуті мобільні рішення.
Джерело інформації	https://www.gymmaster.com/

Аналіз сучасного програмного забезпечення для управління фітнес-клубом показує, що кожна система має свої переваги та недоліки. Вибір відповідного програмного забезпечення залежить від конкретних вимог фітнес-клубу, його розмірів, фінансових можливостей та організаційних цілей. Детальна оцінка функціональності систем, а також їхніх переваг і недоліків дозволяє вибрати найбільш підходяще рішення для ефективного управління фітнес-бізнесом.

Висновки до розділу 1

У першому розділі кваліфікаційної роботи проведено всебічний аналіз предметної області управління фітнес-клубами, що дозволив виявити ключові тенденції розвитку фітнес-індустрії та обґрунтувати необхідність розробки спеціалізованого вебзастосунку. Дослідження показало, що сучасний ринок фітнес-послуг характеризується стрімким зростанням, цифровою трансформацією та підвищенням попиту на персоналізовані рішення. Структурно-функціональний аналіз процесів управління фітнес-клубом виявив низку проблем у традиційних підходах. На основі виявлених недоліків сформульовано ключові вимоги до розроблюваного вебзастосунку.

Проведено огляд та аналіз сучасних програмних рішень у сфері управління фітнес-клубами, зокрема таких, як Mindbody, Glofox та GymMaster. Визначено їхні функціональні можливості, архітектурні особливості, переваги та недоліки. Отже, результати першого розділу формують надійну основу для подальшої розробки та втілення дієвої інформаційної системи управління фітнес-клубом, яка матиме можливості персоналізованого планування тренувань.

2 ВИЗНАЧЕННЯ ВИМОГ ДО ВЕБЗАСТОСУНКУ

2.1 Аналіз сучасного інструментарію, моделей та методів

Для успішної реалізації вебзастосунку управління фітнес-клубом з функціями персоналізованого планування тренувань надзвичайно важливим є вибір сучасних, надійних та ефективних інструментальних засобів розробки. Правильний стек має забезпечити надійність серверної логіки, гнучкість клієнтського інтерфейсу, швидку розробку, зрозумілу підтримку коду і комфортну роботу користувачів.

Laravel обрано як основний інструмент для розробки серверної частини вебзастосунку завдяки його потужності, гнучкості та елегантному синтаксису. Цей PHP-фреймворк з відкритим кодом пропонує комплексне рішення для створення сучасних вебзастосунків. Laravel використовує архітектурний шаблон MVC (Model-View-Controller), що забезпечує чітке розділення бізнес-логіки, представлення даних та керування взаємодією з користувачем.

Вбудована ORM-система Eloquent значно спрощує роботу з БД, дозволяючи працювати з таблицями як з об'єктами, що особливо зручно при моделюванні таких сутностей, як користувачі, тренери, плани тренувань, розклади тощо. Eloquent підтримує складні зв'язки між моделями (один-до-одного, один-до-багатьох, багато-до-багатьох), що дозволяє ефективно відображати різноманітні відносини між об'єктами предметної області фітнес-клубу.

Laravel пропонує потужну систему міграцій, яка забезпечує версіонування схеми бази даних та полегшує командну розробку. Це дозволяє розробникам синхронізувати зміни структури бази даних та уникати конфліктів під час розробки. Система сідерів (seeders) та фабрик (factories) спрощує наповнення БД тестовими даними, що важливо для тестування функціональності застосунку.

MySQL обрано як систему керування БД для проєкту завдяки її надійності, продуктивності та широкій підтримці. Це реляційна БД з відкритим кодом, яка забезпечує високу швидкість читання даних, що особливо важливо для

вебзастосунку фітнес-клубу, де операції читання (перегляд розкладу, доступність тренерів, історія платежів) значно переважають операції запису.

MySQL підтримує механізм InnoDB, який забезпечує транзакційність та цілісність даних за рахунок підтримки зовнішніх ключів та ACID-властивостей. Це критично важливо для таких операцій, як бронювання тренувань чи обробка оплат, де потрібно забезпечити атомарність змін у кількох пов'язаних таблицях. Вибір MySQL також обумовлений його чудовою інтеграцією з Laravel, що дозволяє використовувати всі переваги Eloquent без додаткових налаштувань.

З метою розширення функціональності серверної частини та впровадження можливостей ШІ, у проєкті також реалізовано окремий сервіс на мові Python. Вибір Python обумовлений завдяки простоті, читабельності та ефективності, що робить її придатною як для початківців, так і для досвідчених програмістів. Програми на Python залишаються легкими для розуміння та підтримки завдяки природному та чітко визначеному стилю програмування. Крім того, Python порівняно з іншими мовами програмування вимагає відносно менше зусиль для написання коду, тому є природним вибором для застосунків ШІ. Ще одна велика перевага Python полягає в тому, що вона має багатий набір вбудованих бібліотек і фреймворків, які спрощують розробку ШІ [12]. Особливістю цього сервісу є те, що він покладається на бібліотеку g4f, передовий інструмент ШІ, для пошуку інформації забезпечуючи генерацію тренувальних планів на основі обробки даних про фізичні показники клієнтів [13]. Архітектурно він працює як мікросервіс: узаємодія з основним PHP-сервером відбувається через RESTful API, що гарантує гнучкість розгортання та незалежність зміни логіки ШІ-модуля без впливу на основну частину застосунку. Такий підхід дозволяє поєднувати стабільність та зрілість Laravel-стека з широкими можливостями Python-середовища для аналітики й машинного навчання.

Для розробки клієнтської частини вебзастосунку обрано фреймворк Next.js – це легкий фреймворк React, який використовується для розробки статичних та серверних застосунків. Next.js використовує каталог папок як метод маршрутизації для вебсторінок. Сторінкою за замовчуванням є сам застосунок. Використовуючи

каталог сторінок, Next.js забезпечує автоматичну маршрутизацію сторінки, в той час як серверна частина рендерить результати та дані для кожного запиту [14]. Використання TypeScript як мови програмування забезпечує статичну типізацію, що значно підвищує якість коду та продуктивність розробників.

Next.js пропонує інноваційну систему маршрутизації, що значно спрощує організацію коду та навігацію між різними екранами застосунку, такими як особистий кабінет, розклад тренувань чи каталог товарів.

Одна з ключових переваг Next.js – підтримка різних стратегій рендерингу сторінок. Серверний рендеринг забезпечує швидке відображення сторінок при першому завантаженні та кращу індексацію пошуковими системами, що важливо для залучення нових клієнтів. Статична генерація дозволяє попередньо згенерувати сторінки під час збірки, що забезпечує максимально швидке завантаження для таких розділів, як лендінг, опис послуг чи контакти.

TypeScript забезпечує строгу типізацію даних, що особливо важливо при роботі зі складними об'єктами. Це дозволяє виявляти помилки на етапі компіляції, а не під час виконання, що значно підвищує надійність застосунку та спрощує рефакторинг.

Для управління станом застосунку обрано вбудований в React механізм Context API, який забезпечує зручний спосіб передачі даних через дерево компонентів без необхідності пробкидання пропсів через проміжні рівні. Цей підхід є оптимальним для середніх за розміром застосунків, таких як вебзастосунок управління фітнес-клубом.

Context API використовується для централізованого зберігання та управління станом різних аспектів застосунку, таких як авторизація користувача, кошик для оплати послуг, налаштування інтерфейсу та тимчасові дані форм. Кожен такий контекст інкапсулює власну логіку та надає компонентам доступ лише до тих даних та функцій, які їм необхідні. На відміну від більш складних рішень для управління станом, таких як Redux чи MobX, Context API не вимагає встановлення додаткових залежностей та має нижчий поріг входження для розробників. Це прискорює процес розробки та спрощує підтримку коду в майбутньому.

Для прискорення розробки інтерфейсу користувача обрано бібліотеку компонентів shadcn/ui, яка надає набір високоякісних, доступних та настроюваних React-компонентів. Компоненти shadcn/ui побудовані з використанням Radix UI – низькорівневої бібліотеки примітивів, яка забезпечує доступність (a11y) без шкоди для функціональності чи дизайну.

Бібліотека пропонує широкий спектр компонентів, необхідних для вебзастосунку управління фітнес-клубом, включаючи форми для реєстрації та авторизації, календарі для вибору дати тренування, таблиці для відображення розкладів, картки для представлення товарів, модальні вікна для підтвердження дій та багато іншого.

Компоненти shadcn/ui повністю сумісні з Tailwind CSS, що забезпечує єдиний підхід до стилізації всього застосунку. Бібліотека також пропонує підтримку тем, що дозволяє легко реалізувати світлий та темний режими для покращення досвіду користувача.

В свою чергу, для стилізації вебзастосунку обрано фреймворк Tailwind CSS, який представляє підхід «utility-first» до CSS. Замість написання власних CSS-стилів, розробники використовують готові службові класи безпосередньо в HTML-розмітці, що значно прискорює процес розробки та забезпечує узгодженість дизайну.

Tailwind CSS пропонує комплексну систему дизайну з чітко визначеними масштабами для відступів, розмірів шрифтів, кольорів та інших властивостей. Це забезпечує візуальну гармонію інтерфейсу та дотримання єдиного стилю в усіх частинах застосунку.

Щоб інтерфейс відчувався «живим», використовується Framer Motion. Він дає змогу точно налаштувати вступні, вихідні та проміжні стани компонентів, підтримує drag-and-drop, hover-ефекти та складні переходи. Це покращує UX: користувач бачить зворотний зв’язок і розуміє, що саме відбувається при взаємодії з елементами.

Таким чином, обрані інструментальні засоби для розробки вебзастосунку управління фітнес-клубом формують сучасний, надійний та масштабований

технологічний стек, який дозволяє реалізувати всі необхідні функціональні вимоги та забезпечити високу якість користувацького досвіду. Кожна технологія обрана з урахуванням її переваг та оптимальної сумісності з іншими компонентами системи, що забезпечує ефективну розробку та подальшу підтримку застосунку.

2.2 Специфікація вимог до програмного забезпечення

1. Призначення та межі проєкту

1.1 Призначення системи

Вебзастосунок призначений для управління фітнес-клубом з функціями персоналізованого планування тренувань, що включає базове управління клієнтами та тренерами, управління асортиментом спортивного харчування та інтеграцію Stripe для проведення платежів.

1.2 Погодження, що ухвалені в програмній документації

– специфікація вимог базується на технічному завданні проєкту;

– погоджено використання актуальних вебтехнологій задля гарантування кросплатформної сумісності.

1.3 Межі проєкту ПЗ

– система охоплює лише вебінтерфейс і серверну частину (Laravel, Python та MySQL на боці сервера, Next.js на боці клієнта);

– не включає розробку мобільних застосунків;

– не передбачає інтеграцію зі сторонніми CRM.

2. Загальний опис

2.1 Сфера застосування

Вебзастосунок призначений для внутрішнього використання співробітниками фітнес-клубу та доступу клієнтів, тренерів через вебінтерфейс.

2.2 Характеристики користувачів

Система розрахована на три основні категорії користувачів:

– адміністратор – повний доступ до всіх функцій управління клубом;

– тренер – створення та редагування тренувальних планів вручну або за допомогою ШІ, перегляд прогресу клієнтів, створення розкладу тренувань, придбання товарів;

– клієнт – запис на заняття, перегляд тренувальних планів і відстеження прогресу, придбання товарів.

2.3 Загальна структура і склад системи

– клієнтська частина – вебінтерфейс на основі Next.js та React з адаптивним дизайном;

– серверна частина – RESTful API на Laravel, мікросервіс на Python та БД MySQL;

– інтеграція – підключення Stripe для проведення платежів.

2.4 Загальні обмеження

– сумісність із сучасними браузерами (Chrome, Firefox, Safari, Edge);

– наявність стабільного інтернет-з’єднання;

– обмеження на обробку платежів відповідно до можливостей сервісу.

3. Функції системи

3.1 Реєстрація та авторизація користувачів

3.1.1 Опис функції

Система повинна забезпечувати реєстрацію нових користувачів та авторизацію існуючих з розподілом доступу відповідно до ролей (клієнт, тренер, адміністратор).

3.1.2 Вхідна і вихідна інформація

– вхідна: ім’я, email, пароль, підтвердження паролю та роль під час реєстрації та email, пароль для авторизації;

– вихідна: статус операції (успіх або помилка), дані авторизованого користувача і токен для автентифікації запитів.

3.1.3 Функціональні вимоги

– система повинна забезпечувати реєстрацію користувачів з використанням email;

– система повинна забезпечувати верифікацію email через надсилання підтверджуючого листа;

– система повинна зберігати паролі у зашифрованому вигляді;

– система повинна забезпечувати розмежування доступу відповідно до ролей користувачів.

3.2 Персоналізоване планування тренувань

3.2.1 Опис функції

Тренер має можливість створення персоналізованих тренувальних програм на основі даних клієнта, його цілей та рівня підготовки.

3.2.2 Вхідна і вихідна інформація

– вхідна: дані профілю клієнта, цілі тренувань, доступність обладнання та часові обмеження;

– вихідна: персоналізований план тренувань з детальним описом вправ,підходів та навантаження та рекомендації щодо інтенсивності та частоти тренувань.

3.2.3 Функціональні вимоги

– ручне створення тренувальних програм тренером;

– створення плану за допомогою ШІ;

– коригування планів тренувань;

– фільтрація планів тренувань за назвою, типом;

– зручний інтерфейс для перегляду планів тренувань клієнтами.

3.3 Управління розкладом тренувань

3.3.1 Опис функції

Функція повинна забезпечувати планування та управління розкладом індивідуальних та групових тренувань.

3.3.2 Вхідна і вихідна інформація

– вхідна: дані про вільний час тренерів, запити клієнтів на запис на тренування, дані про групові тренування (час, тривалість, максимальна кількість учасників тощо);

– вихідна: актуальний розклад тренувань, підтвердження запису на тренування, відображення змін в розкладі.

3.3.3 Функціональні вимоги

– система повинна забезпечувати зручний інтерфейс для перегляду розкладу тренувань;

– запис на індивідуальні та групові тренування;

– запобігання конфліктам у розкладі;

– відображення змін в розкладі;

– можливість скасування запису на тренування.

3.4 Проведення аналітики

3.4.1 Опис функції

Функція повинна забезпечувати збір, аналіз та відображення даних про відвідуваність, продажі та прогрес клієнтів.

3.4.2 Вхідна і вихідна інформація

– вхідна: дані про відвідуваність тренувань, зарєстрованих користувачів, прогрес клієнтів та про типи тренувальних планів;

– вихідна: графіки та діаграми для візуалізації даних.

3.4.3 Функціональні вимоги

– збір та агрегація даних про активність користувачів;

– візуалізація аналітичних даних у вигляді графіків та діаграм.

3.5 Управління асортиментом спортивного харчування

3.5.1 Опис функції

Функція повинна забезпечувати управління каталогом спортивного харчування, включаючи додавання, редагування та видалення товарів, а також контроль запасів.

3.5.2 Вхідна і вихідна інформація

– вхідна: дані про товари (назва, опис, ціна, категорія), дані про кількість товарів на складі, зображення товарів;

– вихідна: актуальний каталог товарів, відображення низького рівня запасів, статистика продажів.

3.5.3 Функціональні вимоги

– забезпечення зручного інтерфейсу для управління товарами;

– категоризація товарів для зручного пошуку;

– контроль рівнем запасів;

– встановлення знижок та акційних пропозицій.

3.6 Управління профілем клієнта

3.6.1 Опис функції

Функція повинна забезпечувати можливість заповнення та редагування профілю клієнта, включаючи особисті дані та параметри для майбутньої персоналізації тренувань.

3.6.2 Вхідна і вихідна інформація

– вхідна: особисті дані клієнта (вік, стать, зріст, вага), цілі тренувань, медичні обмеження (за наявності), рівень фізичної підготовки;

– вихідна: оновлений профіль клієнта та рекомендації щодо тренувань на основі даних профілю.

3.6.3 Функціональні вимоги

– зберігання історії змін ключових параметрів (вага, основні показники);

– забезпечення валідації введених даних;

– захист конфіденційних даних клієнта.

3.7 Оплата спортивного харчування

3.7.1 Опис функції

Функція повинна забезпечувати можливість оплати спортивного харчування через інтеграцію з платіжною системою Stripe.

3.7.2 Вхідна і вихідна інформація

– вхідна: дані про вибрані товари, платіжні дані клієнта (через захищений інтерфейс Stripe);

– вихідна: підтвердження оплати, електронний чек, оновлення статусу замовлення.

3.7.3 Функціональні вимоги

– безпечна обробка платежів через інтеграцію з Stripe;

– зберігання історії платежів.

3.8 Відстеження прогресу

3.8.1 Опис функції

Клієнти можуть фіксувати та переглядати зміни основних фізичних параметрів (вага, відсоток жиру, об’єми тощо) для відстеження прогресу.

3.8.2 Вхідна і вихідна інформація

– вхідна: дата, вага, відсоток жиру, м’язова маса, об’єми (груди, талія, стегна, біцепс тощо), нотатки;

– вихідна: історія змін параметрів, графіки прогресу.

3.8.3 Функціональні вимоги

– додавання, редагування та видалення записів прогресу;

– перегляд історії змін у вигляді списку та графіків;

– валідація введених даних.

3.9 Кошик та оформлення замовлення

3.9.1 Опис функції

Система дозволяє клієнтам та тренерам додавати товари до кошика, переглядати його вміст, змінювати кількість товарів, видаляти товари з кошика та оформлювати замовлення.

3.9.2 Вхідна і вихідна інформація

– вхідна: ідентифікатор товару, кількість, контактні дані для оформлення замовлення, спосіб оплати;

– вихідна: поточний вміст кошика, підсумкова сума, статус оформлення замовлення, деталі замовлення.

3.9.3 Функціональні вимоги

– додавання, редагування та видалення товарів у кошику;

– оформлення замовлення з вибором способу оплати (готівка/картка);

– перевірка наявності товару на складі під час оформлення;

– очищення кошика після успішного оформлення замовлення.

4. Вимоги до інформаційного забезпечення

4.1 Джерела і зміст вхідної інформації (даних)

– користувачі системи (клієнти, тренери, адміністратори), які надають: персональні дані при реєстрації та заповненні профілів, дані про цілі тренувань та фізичну підготовку, дані для створення тренувальних програм, інформацію для формування розкладу тренувань;

– адміністративний персонал фітнес-клубу, який надає: дані про асортимент спортивного харчування, інформацію про групові тренування та їх розклад;

– зовнішні системи, які надають: дані про проведені платежі (через API Stripe) та дані для верифікації електронної пошти.

4.2 Нормативно-довідкова інформація

Немає вимог до даного пункту.

4.3 Вимоги до способів організації, збереження та ведення інформації

– усі дані системи зберігаються в БД MySQL;

– конфіденційні дані (паролі, платіжна інформація) повинні зберігатися у зашифрованому вигляді.

5. Вимоги до технічного забезпечення

– клієнтська частина: будь-який сучасний веббраузер (Chrome 90+, Firefox 85+, Safari 14+, Edge 90+), на персональних комп'ютерах, ноутбуках, планшетах чи смартфонах;

– серверна частина: процесор (мінімум 4 ядра, рекомендовано 8 ядер), оперативна пам'ять (мінімум 8 ГБ, рекомендовано 16 ГБ), дисковий простір (мінімум 100 ГБ SSD);

– мережеве забезпечення: швидкісне підключення до Інтернету, підтримка HTTP/HTTPS з використанням сертифікатів SSL/TLS.

6. Вимоги до програмного забезпечення

6.1 Архітектура програмної системи

Система повинна бути побудована на основі клієнт-серверної архітектури з використанням RESTful API для взаємодії між frontend та backend компонентами. Архітектура повинна забезпечувати: чітке розділення бізнес-логіки, представлення даних та збереження даних, масштабованість для забезпечення можливості збільшення навантаження та високу доступність та стійкість до збоїв.

6.2 Системне програмне забезпечення

– БД: MySQL 8.0+;

– PHP 8.1+;

– Python 3.11.5+;

– Node.js 18.0+.

6.3 Мережне програмне забезпечення

– Підтримка протоколів HTTP/HTTPS;

– SSL/TLS для шифрування даних, що передаються.

6.4 Програмне забезпечення ведення інформаційної бази

– БД: MySQL 8.0+;

– ORM (Object-Relational Mapping) для Laravel (Eloquent);

– сервіс штучного інтелекту: Python 3.11.5+, бібліотека g4f (версія ≥ 0.1.0) для обробки запитів до GPT-моделей;

– система міграцій баз даних Laravel.

6.5 Мова і технологія розробки ПЗ

– frontend: Next.js, TypeScript, Tailwind CSS 4.0, shadcn/ui, Framer Motion;

– backend: Laravel, Python;

– БД: MySQL.

7. Вимоги до зовнішніх інтерфейсів

7.1 Інтерфейс користувача

– адаптивність для різних пристроїв (десктоп, планшет, мобільний телефон);

– інтуїтивно зрозуміле навігаційне меню з чіткою структурою;

– мінімалістичний дизайн з використанням компонентів shadcn/ui та з акцентом на основні функції.

7.2 Апаратний інтерфейс

Не потребує нічого спеціалізованого обладнання, стандартне підключення до Інтернету, працює на стандартних комп'ютерах і мобільних пристроях.

7.3 Програмний інтерфейс

API для інтеграції з платіжним сервісом Stripe.

7.4 Комунікаційний протокол

Застосунок використовує HTTP/HTTPS протоколи.

8. Властивості програмного забезпечення

8.1 Доступність

– час безперебійної роботи системи повинен становити не менше 99% часу;

– система повинна бути доступна 24/7 з плановими технічними перервами не більше 4 годин на місяць;

– час відгуку системи на запити користувачів не повинен перевищувати 2 секунди для 95% запитів;

– система повинна підтримувати одночасну роботу до 500 активних користувачів без погіршення якості обслуговування.

8.2 Супроводжуваність

– модульна архітектура, що дозволяє легко замінювати або розширювати окремі компоненти;

– застосування автоматизованих тестів для перевірки функціональності;

– ведення журналу змін (changelog) для відстеження оновлень системи.

8.3 Переносимість

– незалежність від операційної системи;

– кросбраузерність;

– адаптивність до різних пристроїв.

8.4 Продуктивність

– відповіді API приблизно 200 мс під навантаженням 100 запитів/с;

– час завантаження початкової сторінки не повинен перевищувати 3 секунди;

– підтримка одночасної роботи до 500 активних користувачів.

8.5 Надійність

– регулярне автоматичне резервне копіювання бази даних з можливістю відновлення;

– журналювання помилок з детальною інформацією для розробників.

8.6 Безпека

– шифрування даних при передачі з використанням HTTP/HTTPS;

– зберігання паролів у зашифрованому вигляді з використанням сучасних алгоритмів хешування;

– захист від поширених вразливостей (SQL-ін'єкції, XSS, CSRF);

– контроль доступу на основі ролей користувачів.

9. Інші вимоги

– можливість підключення додаткових платіжних систем крім Stripe;

– підготовка до інтеграції з фітнес-трекерами та іншими пристроями;

– можливість розширення системи для впровадження нових можливостей.

Висновки до розділу 2

У другому розділі проведено детальний аналіз сучасного інструментарію, моделей та методів для моделювання вебзастосунку управління фітнес-клубом. Обґрунтовано вибір інструментів для розробки вебзастосунку.

Розроблена специфікація вимог, чітко окреслює призначення проєкту, межі, загальний опис, функціональні і нефункціональні вимоги до ПЗ, а також вимоги до інформаційного й технічного забезпечення та властивостей якості. Отримані результати створюють міцну основу для подальшої розробки архітектури та реалізації вебзастосунку, забезпечуючи чітке розуміння функціональності та взаємозв'язків між компонентами системи.

3 ПРОЄКТУВАННЯ ВЕБЗАСТОСУНКУ

3.1 Діаграма варіантів використання

Діаграма варіантів використання в UML – це візуальне представлення, яке ілюструє взаємодію між користувачами (акторами) і системою. Вона відображає функціональні вимоги системи, показуючи, як різні користувачі взаємодіють з різними варіантами використання або конкретними функціональними можливостями в системі.

Діаграми варіантів використання надають високорівневий огляд поведінки системи, що робить їх корисними для зацікавлених сторін, розробників та аналітиків, щоб зрозуміти, як система має працювати з точки зору користувача, і як різні процеси пов'язані один з одним.

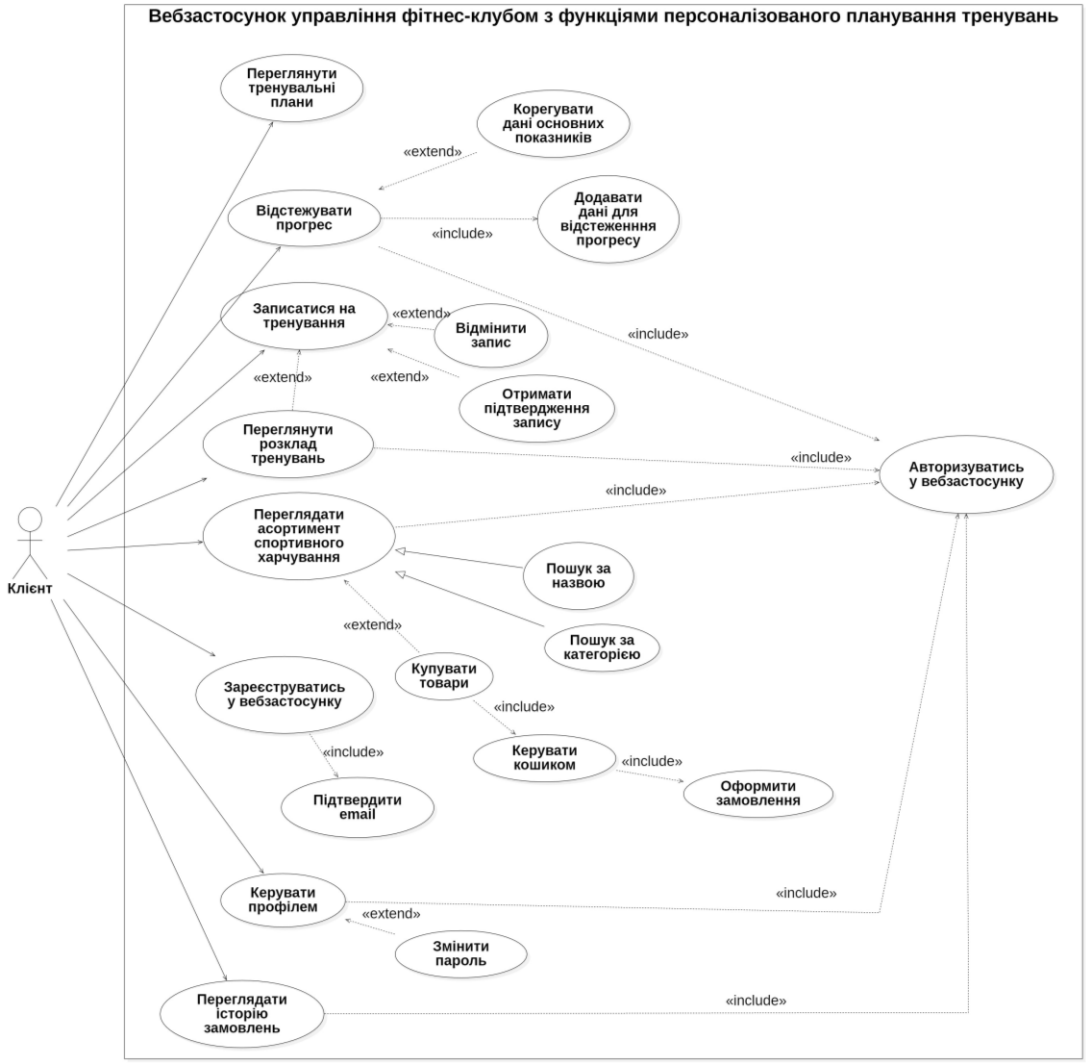

Рисунок 3.1 – Діаграма варіантів використання для клієнта

На діаграмі (рис. 3.1) показано взаємодію клієнта з системою. Діаграма відображає споживацькоорієнтовану функціональність, зосереджену на тренувальному процесі з можливістю перегляду планів, відстеження прогресу через коригування даних основних показників і додавання нових даних, а також запису на тренування з опціями скасування та отримання підтвердження. Для зручності придбання товарів передбачено перегляд асортименту спортивного харчування з функціями пошуку за різними параметрами, керування кошиком та оформлення замовлення, при цьому система забезпечує реєстрацію нових користувачів із подальшим підтвердженням електронної пошти, автентифікацію у вебзастосунку, керування профілем із можливістю зміни паролю та перегляду історії замовлень, що в комплексі створює персоналізований досвід використання системи та сприяє задоволенню різноманітних потреб клієнтів фітнес-клубу.

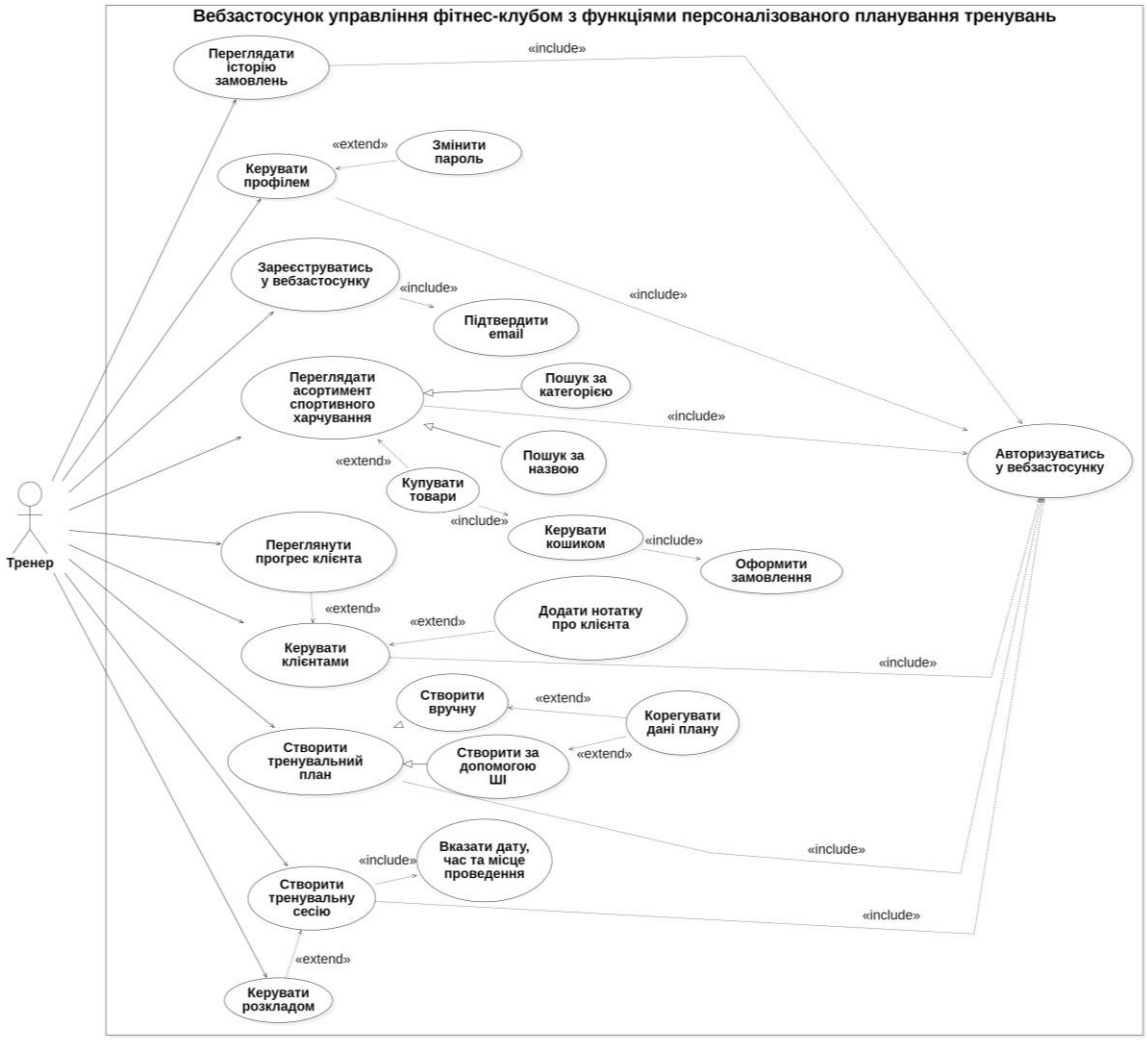

Рисунок 3.2 – Діаграма варіантів використання для тренера

В свою чергу, на діаграмі (рис. 3.2) використання представлено функціонал для тренера, який охоплює функціональність, сфокусовану на професійній діяльності з можливістю створення тренувальних планів вручну чи за допомогою ШІ, коригування даних планів, керування клієнтами з додаванням нотаток і моніторингом їхнього прогресу, організації тренувальних сесій із зазначенням дати, часу та місця проведення, а також керування розкладом. Додатково відображено можливість перегляду та купівлі спортивного харчування з доступними функціями пошуку за назвою чи категорією, керуванням кошиком та оформленням замовлення, при цьому всі варіанти використання потребують автентифікації у вебзастосунку, а функціональність керування профілем передбачає зміну паролю та перегляд історії замовлень, що в комплексі забезпечує ефективне виконання професійних обов'язків тренера та оптимізацію його робочих процесів.

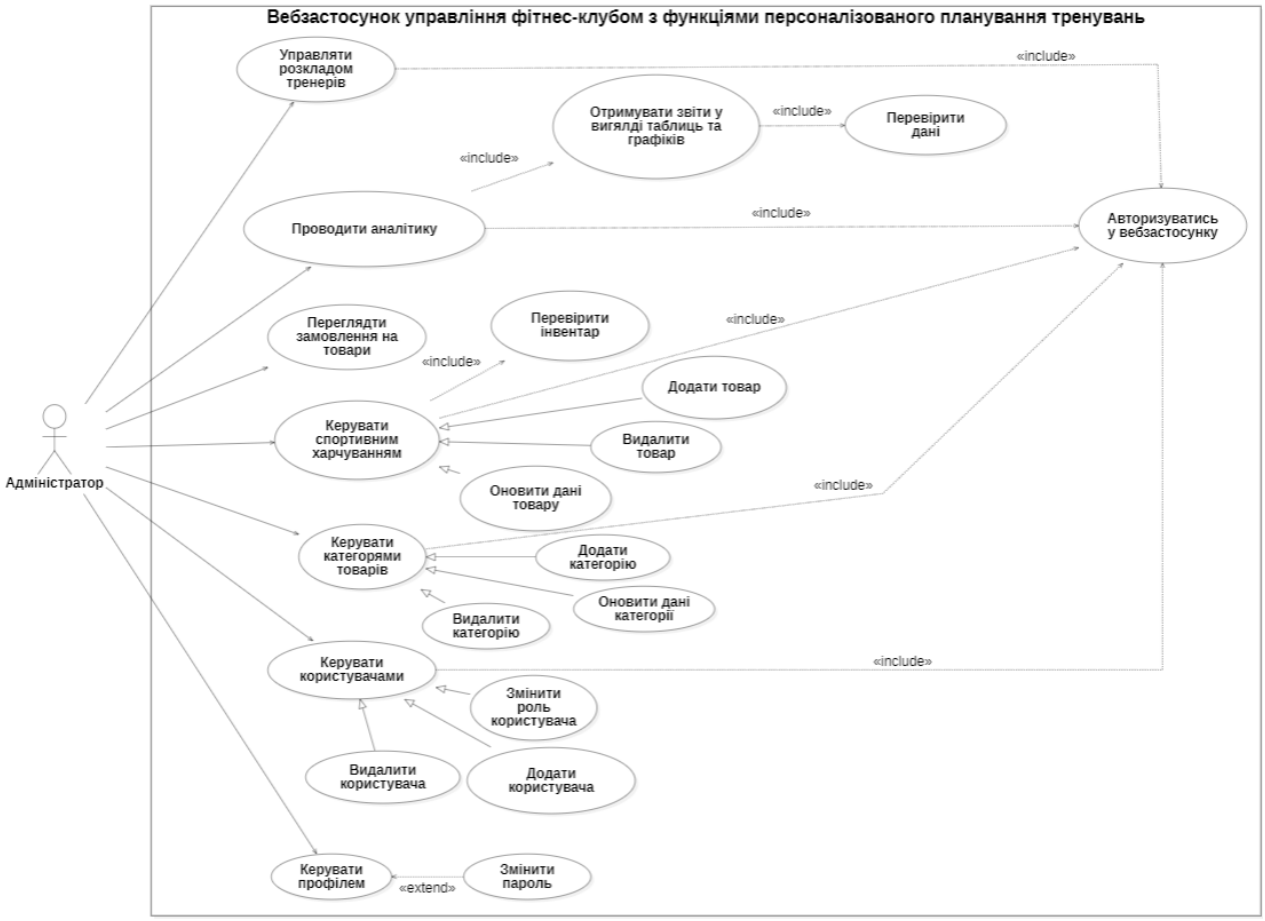

Рисунок 3.3 – Діаграма варіантів використання для адміністратора

Представлена діаграма варіантів використання (рис. 3.3) для адміністратора системи управління фітнес-клубом демонструє багатогранність функціональності, що охоплює керування розкладом тренерів, проведення аналітики, адміністрування товарної складової через додавання, видалення й оновлення даних товарів та їхніх категорій, а також комплексне управління користувачами з можливістю додавання нових, видалення існуючих та зміни ролей. Важливим аспектом функціональності виступає моніторинг замовлень і товарів та отримання аналітичних звітів у формі таблиць і графіків, що уможливлює контроль фінансових та операційних показників діяльності фітнес-клубу.

Крім того, сценарії використання можливо представити у трьох виглядах:

Use case №1 (коротка форма): Зареєструватись у вебзастосунку як Клієнт

Користувач відкриває сторінку авторизації вебзастосунку управління фітнес-клубом. Він вводить своє ім’я, пошту, пише пароль та підтвержує його, обирає роль «Клієнт» та натискає кнопку «Реєстрація». Система перевіряє введені дані та, у разі успіху, надсилає лист з підтвердженням на електронну пошту. Він підтверджує свою електронну адресу, натиснувши на посилання, що було надіслано. Після цього, Клієнт успішно зареєстрований.

Use case №2 (поверхнева форма): Управління асортиментом спортивного харчування

Головний сценарій (успішний):

Адміністратор авторизується в системі та переходить до розділу управління асортиментом. Він вибирає опцію додавання нового товару, заповнює всі необхідні поля інформацією про продукт, а саме: назва, опис продукту, обирає категорію, вказує ціну, за потреби акційну ціну, вводить кількість товару та його складський номер, також завантажує фотографію та зберігає зміни. Система оновлює БД та відображає новий товар в каталозі для всіх користувачів.

Альтернативні сценарії:

1) адміністратор намагається додати товар з назвою, яка вже існує в системі. Після заповнення всіх полів і натискання «Зберегти» система перевіряє унікальність назви і виявляє збіг із вже наявним товаром. Вона відображає

повідомлення про помилку «Товар з такою назвою вже існує» й пропонує або змінити назву;

2) під час завантаження фото товару виникає помилка через невідповідний формат файлу. Якщо адміністратор обирає зображення у форматі GIF або TIFF, система зупиняє процес завантаження та виводить підказку «Підтримуються лише формати JPEG, JPG, PNG або WebP»;

3) адміністратор вводить некоректний формат складського номеру. Якщо під час заповнення поля «Складський номер» адміністратор вводить артикул, що не відповідає встановленому шаблону (наприклад, замість «SN-12345» – «12345SN»), система виявляє невідповідність формату;

4) сесія адміністратора спливає в процесі редагування. Якщо адміністратор забарився із заповненням форми і його робоча сесія завершилася через бездіяльність, система перенаправляє його на сторінку входу;

5) адміністратор намагається видалити товар, який вже є в активних замовленнях клієнтів. При натисканні «Видалити» система перевіряє наявність незавершених замовлень, до складу яких входить цей товар. Якщо такі замовлення є, з’являється повідомлення «Неможливо видалити: товар використовується в активних замовленнях».

Use case №3 (повна форма): Створити персоналізований план тренувань (табл. 3.1)

Таблиця 3.1 – Створити персоналізований план тренувань

Usecase section	Comment
Use Case Name	Створити персоналізований план тренувань
Scope	Вебзастосунок управління фітнес-клубом з функціями персоналізованого планування тренувань
Level	Успішно створити персоналізований план тренувань
Primary Actor	Тренер
Stakeholders and interests	1) клієнт – отримання ефективного плану тренувань; 2) тренер – створення оптимального плану для клієнта; 3) адміністратор – забезпечення якості послуг фітнес-клубу.
Preconditions	Тренер авторизований в системі. Профіль клієнта створено та містить необхідну інформацію (вік, вага, цілі тренувань тощо).

Продовження таблиці 3.1

Success guarantee	1) система зберігає всі внесені тренером дані; 2) тренер підтверджує план і він стає доступним клієнту; 3) план відповідає цілям та можливостям клієнта.
Main Success Scenario	1) тренер відкриває в меню розділ «Мої клієнти» та обирає потрібний профіль; 2) система відображає всю доступну інформацію про клієнта; 3) тренер аналізує дані клієнта та його цілі; 4) тренер переходить в розділ «Плани тренувань»; 5) тренер натискає «Створити план тренувань»; 6) система показує шаблон форми: поля для вправ, кількості повторень, підходів, днів тижня; 7) тренер вручну заповнює всі поля: обирає вправи з довідника, задає параметри кожної вправи, розподіляє їх по днях; 8) тренер переглядає згенерований план; 9) тренер зберігає фінальну версію плану; 10) система оновлює профіль клієнта з новим планом тренувань.
Extensions	Недостатньо даних для створення плану: 1) система виявляє, що у профілі клієнта відсутні обов’язкові поля (наприклад, рівень підготовки); 2) система повідомляє тренера про необхідність додаткової інформації; 3) тренер запитує додаткову інформацію у клієнта. Виявлено медичні протипоказання: 1) тренер виявляє медичні протипоказання в профілі клієнта; 2) система попереджає тренера про необхідність адаптації плану; 3) тренер модифікує план з урахуванням медичних обмежень; Перевищення рекомендованих навантажень: 1) система виявляє, що план перевищує рекомендовані навантаження; 2) система пропонує альтернативи з меншою інтенсивністю; 3) тренер розглядає пропозиції та корегує план; 4) система повторно перевіряє план на відповідність рекомендаціям. Відсутність потрібного шаблону: 1) тренер намагається використати специфічний шаблон плану; 2) система повідомляє, що шаблон відсутній; 3) тренер створює новий шаблон; 4) система зберігає новий шаблон для майбутнього використання.
Special Requirements	Інтерфейс створення плану має бути максимально зручним для ручного редагування: підтримувати масове копіювання/переміщення днів, швидкий пошук вправ за назвою чи м’язовою групою.

Кінець таблиці 3.1

Technology and Data Variations List	1) вся інформація про вправи та шаблони зберігаються локально в БД застосунку; 2) дані про фізичні параметри клієнта можуть вводитися вручну або імпортуватися з фітнес-трекерів та смарт-годинників; 3) візуалізація плану тренувань може бути реалізована у вигляді календаря, графіка або інтерактивної діаграми; 4) опції сортування та фільтрації вправ (за інтенсивністю, тривалістю, м’язовою групою); 5) комунікація між тренером та клієнтом щодо плану може відбуватися через вбудований чат, відеозв'язок або систему коментарів.
Frequency of Occurrence	25 % від усіх запитів створення/редагування планів.
Miscellaneous	Розглянути можливість інтеграції з пристроями для отримання додаткових даних про фізичну активність клієнта.

На представлених діаграмах прецедентів та написаних сценаріях використання відображено ключовий функціонал вебзастосунку управління фітнес-клубом з персоналізованим плануванням тренувань для трьох основних категорій користувачів: клієнта, тренера та адміністратора. Крім того, використано всі види зв’язків, а саме: directional association, generalization, extend relationship та include relationship.

3.2 Побудова діаграм послідовності

Діаграми послідовності в UML – це спеціальний вид діаграм взаємодії, який відтворює об’єкти чи класи у вигляді паралельних «ліній життя» (lifelines) та ряд горизонтальних стрілок, що показують відправлені і отримані повідомлення в порядку їх виконання. Такі діаграми широко застосовують для моделювання динамічного поведінкового аспекту системи, оскільки вони чітко і наочно ілюструють, які саме повідомлення й у якій послідовності обмінюються між елементами системи під час виконання конкретного сценарію використання.

У контексті вебзастосунку управління фітнес-клубом діаграми послідовності дозволяють деталізувати кожен крок взаємодії з користувачем (адміністратором, тренером, клієнтом), а також внутрішні виклики сервісів і бази даних. Це допомагає

виявити вузькі місця в логіці обробки запитів, уточнити обов’язкові та альтернативні потоки, а також спланувати майбутню імплементацію й тести системи.

Діаграма послідовності «Записатися на тренування» (рис. 3.4) відображає процес запису на тренування, який розпочинається з автентифікації клієнта через вебінтерфейс системи та переходу до головної сторінки користувача, звідки здійснюється запит на перегляд розкладу тренувань, що обробляється контролером розкладу через звернення до відповідного сервісу для пошуку та отримання інформації про доступні тренувальні сесії, які потім відображаються у зручному форматі на вебінтерфейсі.

Центральними елементами процесу є вибір клієнтом конкретного тренування для запису, що ініціює ланцюжок взаємодій: від передачі запиту на бронювання до контролера розкладу, перевірки доступності та резервування місця відповідним сервісом, надсилання підтвердження через сервіс сповіщень, до фінального відображення статусу бронювання на вебінтерфейсі.

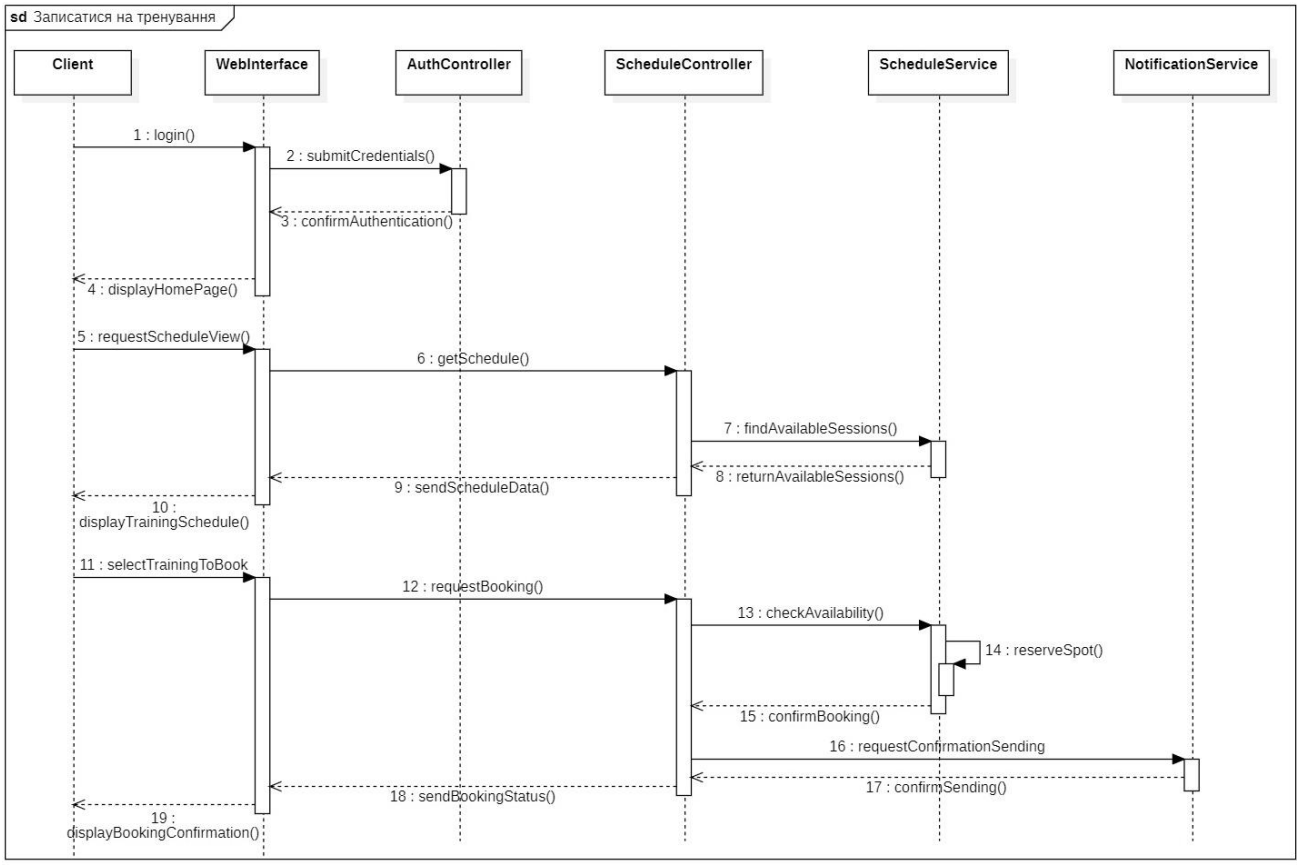

Рисунок 3.4 – Діаграма послідовності запису на тренування

Діаграма послідовності «Створити тренувальний план» (рис. 3.5) демонструє складний процес створення персоналізованого тренувального плану для клієнта, який розпочинається з автентифікації тренера через вебінтерфейс та переходу до панелі тренера, де здійснюється вибір клієнта та ініціювання запиту на формування плану, що далі передається контролеру тренувальних планів та відповідному сервісу, який звертається до інноваційного компонента системи – сервісу ШІ для аналізу даних клієнта та генерації рекомендацій на основі його фізичних параметрів, історії тренувань та визначених цілей.

Особливої уваги заслуговує взаємодія між сервісом тренувальних планів та ШІ-сервісом, де відбувається обмін ключовими даними з наступним формуванням плану, його збереженням у БД та надсиланням повідомлення клієнту через сервіс сповіщень про створення нового плану. Комплексність процесу підкреслюється додатковим етапом внесення коректив тренером у автоматично згенерований план, що включає передачу змін через вебінтерфейс до контролера, сервісу та таблиці БД тренувальних планів з отриманням підтверджень на кожному рівні та фінальним відображенням оновленого плану.

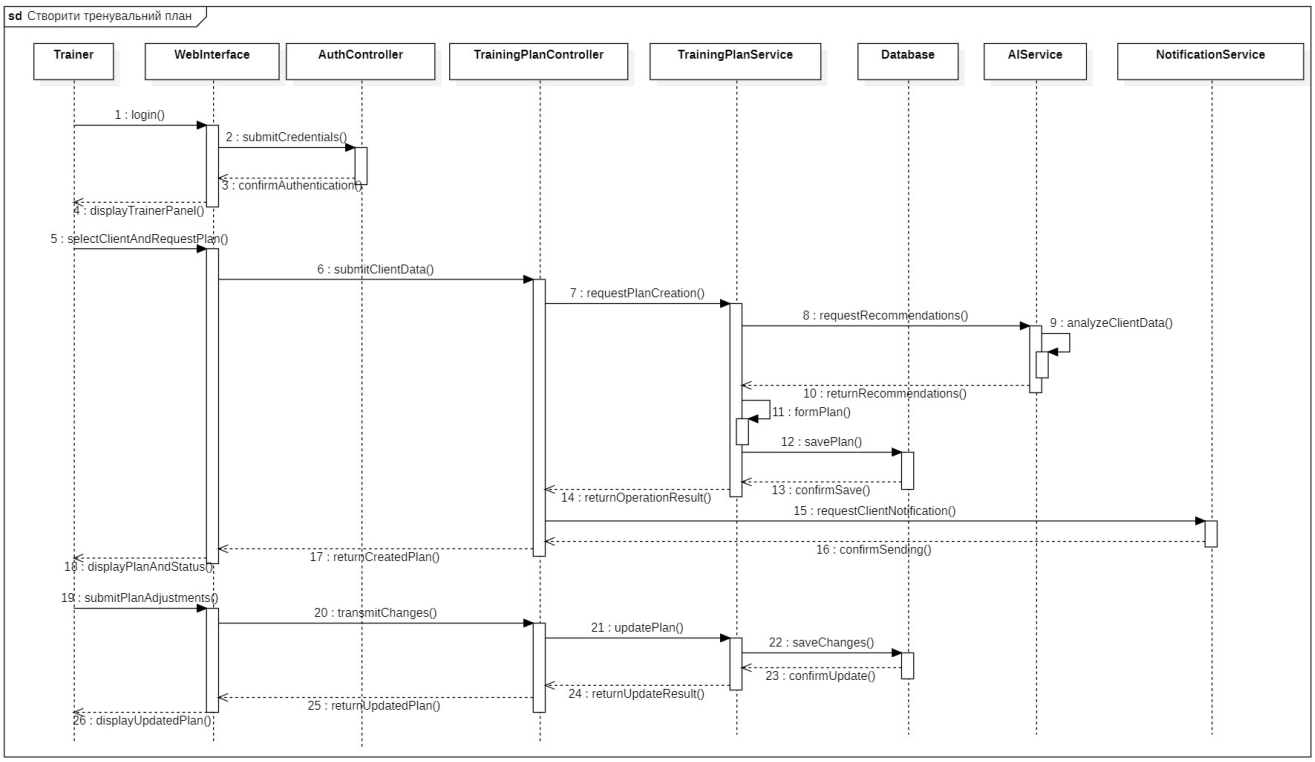

Рисунок 3.5 – Діаграма послідовності створення тренувального плану

Діаграма послідовності «Керувати спортивним харчуванням» (рис. 3.6) відображає процес керування товарами та категоріями, що починається з автентифікації адміністратора через вебінтерфейс системи, передачу облікових даних контролеру автентифікації, після чого здійснюється надання доступу до панелі адміністратора, де відбувається основний функціональний процес – додавання нового товару, що включає передачу даних від вебінтерфейсу до контролера товарів, створення та валідацію товару сервісом товарів, збереження його у БД з отриманням підтвердження, а також оновлення кешу за допомогою сервісу сповіщень для синхронізації даних в системі.

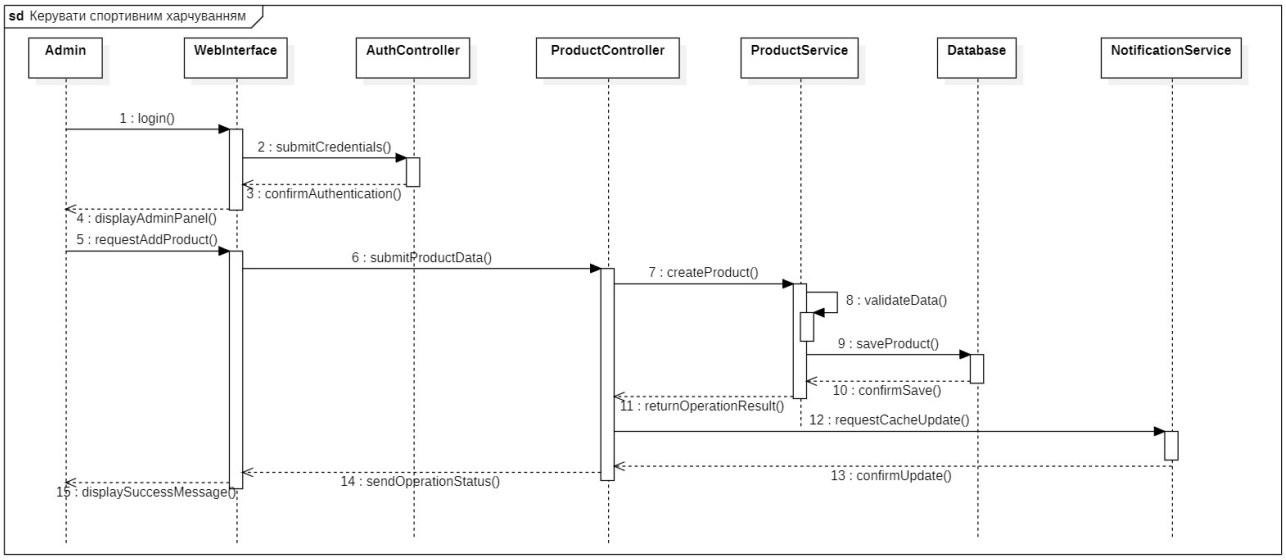

Рисунок 3.6 – Діаграма послідовності керування товарами

Розроблені діаграми послідовності дозволяють наочно відтворити порядок викликів методів та передачі даних між об’єктами всередині системи управління фітнес-клубом. Чітке розмежування логічних шарів і різноманіття типів повідомлень закладають фундамент для наступної етапу програмної реалізації, сприяючи впорядкованій оптимізації бізнес-процесів. Детальне описання поведінкових механізмів на стадії проєктування суттєво зменшує невизначеність під час кодування та надає повне уявлення як про структурні взаємозв’язки компонентів, так і про їхню послідовність у часі.

3.3 Діаграма класів вебзастосунку

Діаграма класів є одним із ключових структурних елементів UML, що надає візуальне представлення статичної структури системи через відображення її класів, атрибутів, методів та взаємозв'язків між ними. У контексті вебзастосунку управління фітнес-клубом діаграма класів набуває особливого значення, оскільки сприяє чіткому розмежуванню відповідальності між компонентами системи та формуванню ефективної архітектури. Завдяки цьому інструменту забезпечується уніфіковане уявлення про архітектуру ПЗ, що сприяє підвищенню ефективності процесів проєктування, реалізації та супроводу.

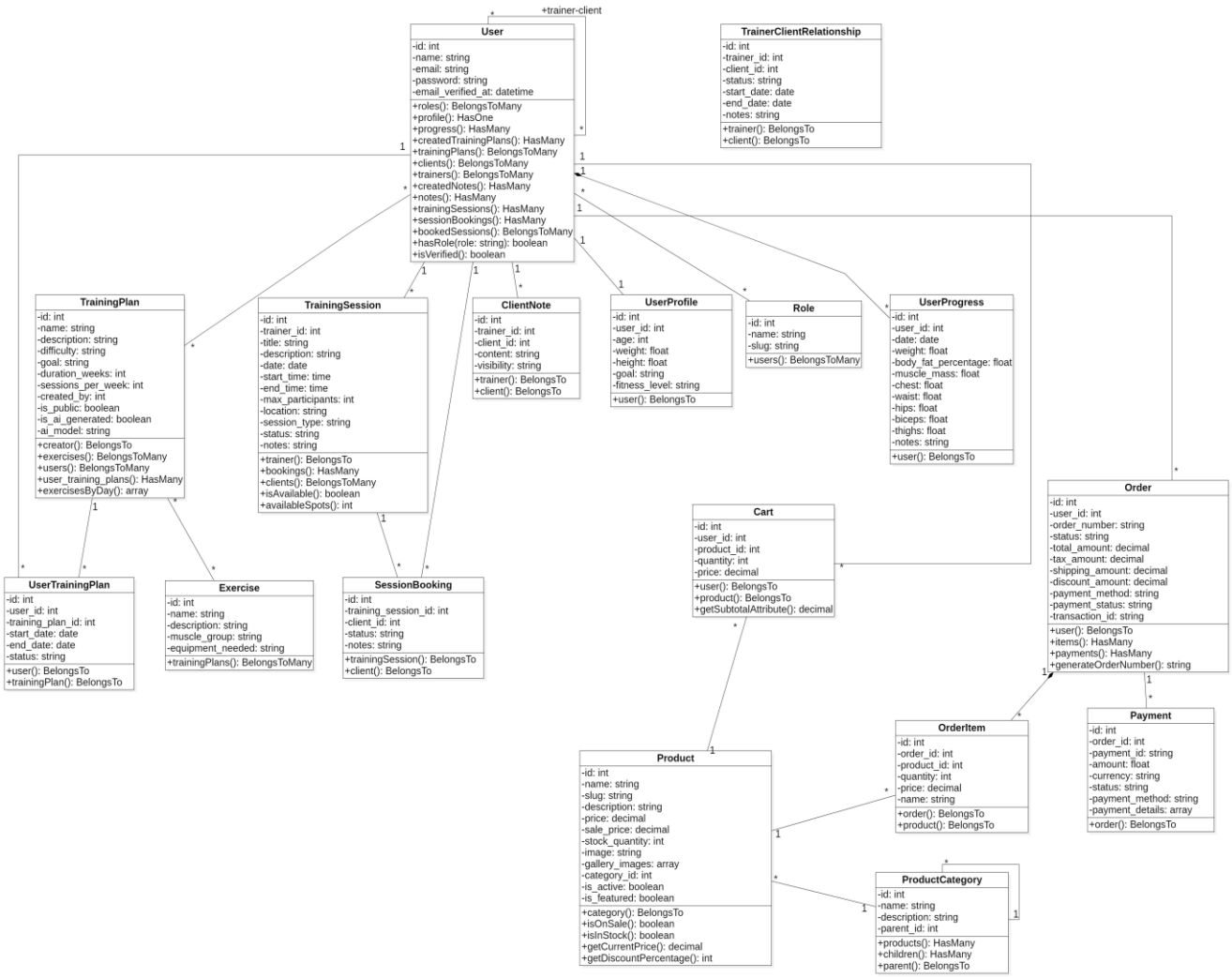

Рисунок 3.7 – Діаграма класів вебзастосунку

Розроблена діаграма класів (рис. 3.7) відображає структурну організацію вебзастосунку управління фітнес-клубом, демонструючи ключові доменні сутності

системи та їхні взаємозв'язки. Центральне місце в архітектурі посідає клас User, що характеризується атрибутами ідентифікації та аутентифікації (id, name, email, password, email_verified_at) та методами доступу до пов'язаних сутностей. Особливістю цього класу є різноманітність його відношень з іншими класами системи: композиційний зв'язок із UserProgress, що вказує на концептуальну нероздільність користувача та його прогресу, асоціативні зв'язки з Role через проміжну таблицю, що забезпечує розмежування повноважень у системі, рефлексивна асоціація з іншими екземплярами User через TrainerClientRelationship, яка моделює відношення «тренер-клієнт» із зазначенням статусу, початкової та кінцевої дати взаємодії.

Тренувальний процес у системі репрезентовано класами TrainingPlan та Exercise, пов'язаними через багато-до-багатьох відношення, що дозволяє гнучко формувати тренувальні плани різної складності та специфікації. Клас TrainingPlan містить атрибути для зберігання назви, опису, складності, мети тренування, а також інформації про тривалість плану у тижнях та кількість тренувань на тиждень. Важливим аспектом є атрибут is_ai_generated, що вказує на можливість автоматичного формування плану за допомогою ШІ. Зв'язок TrainingPlan із User є двостороннім: з одного боку, кожен план має творця (created_by), з іншого – план може бути призначений багатьом користувачам через проміжну сутність UserTrainingPlan, що дозволяє зберігати інформацію про початок та кінець тренувального циклу, а також його статус.

Система управління розкладом тренувань реалізована класами TrainingSession та SessionBooking. TrainingSession характеризується атрибутами, що описують час, місце, максимальну кількість учасників та тип тренування, а також методами для перевірки доступності (isAvailable) та розрахунку кількості вільних місць (availableSpots). SessionBooking виступає проміжною сутністю, що пов'язує тренувальну сесію з клієнтом та містить інформацію про статус бронювання та додаткові нотатки. Важливим елементом системи є клас ClientNote, що забезпечує можливість ведення тренером нотаток про клієнтів із зазначенням видимості цих нотаток у системі.

Електронна комерція представлена комплексом класів Product, ProductCategory, Cart, Order, OrderItem та Payment. Клас Product містить інформацію про товар, включаючи ціну, знижку, кількість на складі, а також методи для визначення наявності знижки (isOnSale), наявності товару на складі (isInStock), отримання актуальної ціни (getCurrentPrice) та розрахунку відсотка знижки (getDiscountPercentage). ProductCategory моделює ієрархічну структуру категорій товарів через рефлексивну асоціацію, де кожна категорія може мати батьківську категорію та дочірні категорії. Клас Cart відображає кошик користувача з методом для розрахунку проміжної суми (getSubtotalAttribute). Оформлення замовлення реалізовано через клас Order, що містить інформацію про загальну суму, податки, доставку та знижки, а також статус замовлення та платежу. OrderItem виступає деталізацією замовлення, пов'язуючи його з конкретними товарами, а Payment забезпечує зберігання інформації про платіж, включаючи його статус та деталі.

Розроблена діаграма класів утворює комплексну архітектурну модель вебзастосунку управління фітнес-клубом, забезпечуючи системний підхід до реалізації функціональних вимог та оптимізацію структурної організації ПЗ. Чітке розмежування відповідальності між доменними сутностями сприяє підвищенню якості коду, спрощенню подальшої підтримки та розширення функціональності, а також покращенню загальної масштабованості системи в умовах зростання кількості користувачів. Використання об'єктно-реляційного відображення в представлених класах, що є характерним для фреймворка Laravel, дозволяє ефективно взаємодіяти з БД, зберігаючи при цьому об'єктно-орієнтовану парадигму розробки.

3.4 Схема бази даних вебзастосунку

Фундаментальною основою будь-якого вебзастосунку, яким, безперечно, є система управління фітнес-клубом, виступає ретельно спроєктована схема бази даних, що визначає логічну структуру сховища даних, типи збережених сутностей та взаємозв’язки між ними. Для досягнення цієї мети застосовується модель

«сутність–зв’язок» (ER-модель), яка дозволяє концептуалізувати предметну область шляхом визначення сутностей, їхніх атрибутів та взаємозв’язків між ними. Цей підхід сприяє створенню логічної структури бази даних, що відповідає вимогам функціональності та масштабованості системи. Візуалізація цієї моделі у вигляді ER-діаграми слугує основою для подальшої реалізації фізичної структури бази даних. Представлена нижче схема бази даних (рис. 3.8) розроблена з урахуванням ключових функціональних вимог вебзастосунку, охоплюючи аспекти управління користувачами, тренувальними планами, розкладом занять, прогресом клієнтів та електронною комерцією.

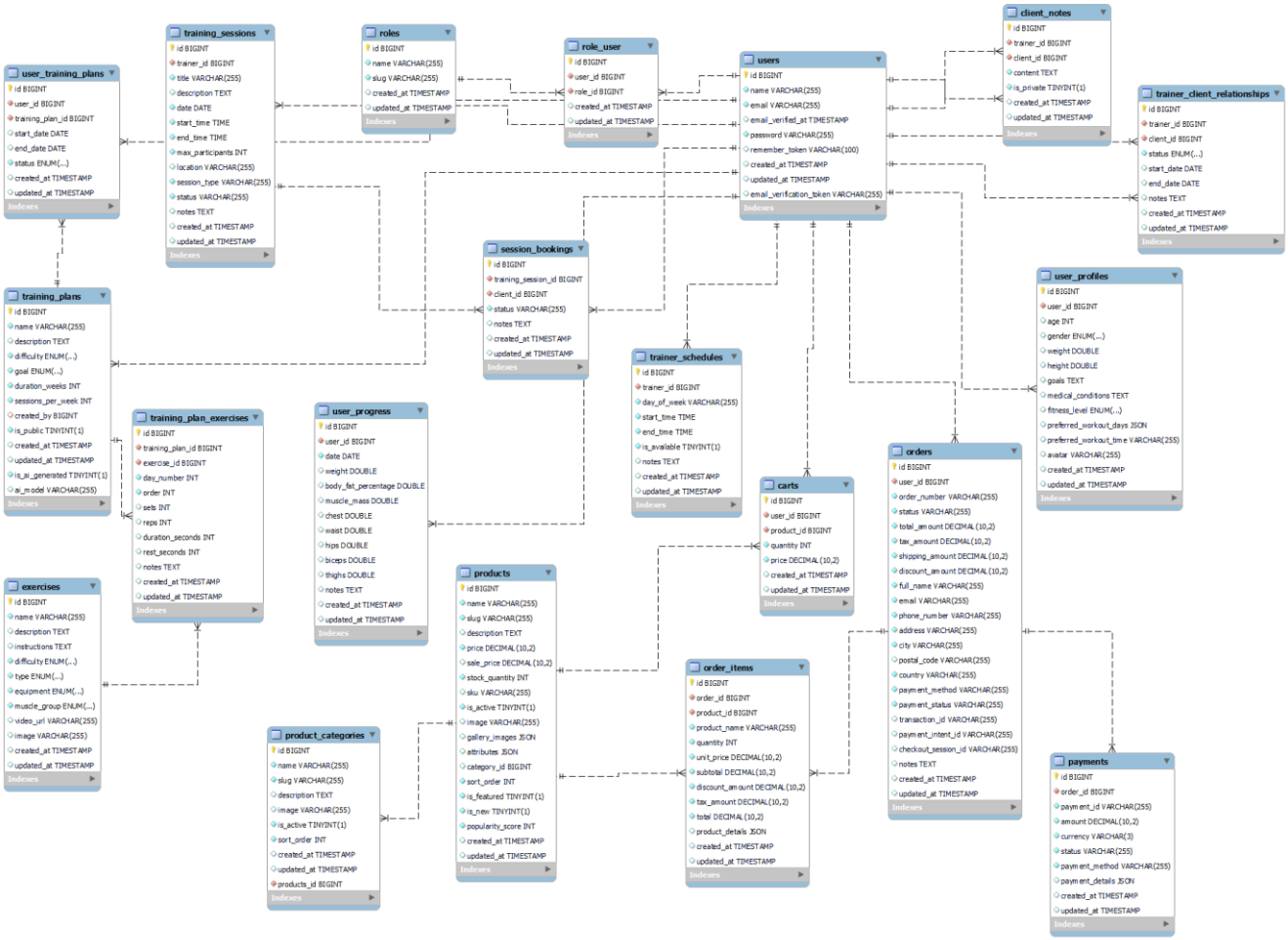

Рисунок 3.8 – Схема бази даних вебзастосунку

Представлена схема БД організована навколо ключових сутностей, що відображають основні аспекти діяльності сучасного фітнес-клубу, та реалізує їх взаємозв'язки через механізми первинних (PK) та зовнішніх (FK) ключів, забезпечуючи реляційну цілісність даних.

1) Управління користувачами та їхніми ролями:

– таблиця users є центральною, зберігаючи базову інформацію про всіх користувачів системи (ідентифікатор id BIGINT PK, ім'я name VARCHAR(255), електронна пошта email VARCHAR(255) UNIQUE, хешований пароль password VARCHAR(255), токени та мітки часу);

– таблиця user_profiles розширює дані користувача, зберігаючи деталізовану профільну інформацію (вік age INT, стать gender ENUM, вага weight DOUBLE, зріст height DOUBLE, цілі goals TEXT, медичні особливості medical_conditions TEXT тощо) та пов'язана з таблицею users відношенням «один-до-одного» через user_id BIGINT FK;

– система ролей реалізована через таблиці roles (що містить назву ролі name VARCHAR(255) та її системний ідентифікатор slug VARCHAR(255)) та role_user (проміжна таблиця для реалізації відношення «багато-до-багатьох» між users та roles, використовуючи user_id BIGINT FK та role_id BIGINT FK).

2) Управління тренувальним процесом:

– таблиця training_plans містить інформацію про тренувальні програми (назва name VARCHAR(255), опис description TEXT, рівень складності difficulty ENUM, тривалість duration_weeks INT), включаючи посилання на автора плану created_by BIGINT FK (з таблиці users);

– таблиця exercises слугує каталогом вправ, деталізуючи кожну вправу (назва name VARCHAR(255), опис description TEXT, інструкції instructions TEXT, задіяна група м'язів muscle_group ENUM);

– зв'язок між тренувальними планами та вправами встановлюється через проміжну таблицю training_plan_exercises, яка дозволяє не лише реалізувати відношення «багато-до-багатьох», але й зберігати специфічні атрибути для кожної вправи в межах конкретного плану (кількість підходів sets INT, повторень reps INT, тривалість duration_seconds INT);

– призначення тренувальних планів користувачам фіксується в таблиці user_training_plans, що пов'язує users та training_plans через user_id BIGINT FK та

training_plan_id BIGINT FK, а також містить дати початку/завершення плану та його статус;

– таблиця user_progress призначена для відстеження фізичного прогресу користувачів (вага weight DOUBLE, відсоток жиру body_fat_percentage DOUBLE, антропометричні виміри chest DOUBLE, waist DOUBLE тощо) та пов'язана з таблицею users відношенням «один-до-багатьох» через user_id BIGINT FK.

3) Управління розкладом та взаємодією тренер-клієнт:

– таблиця training_sessions зберігає інформацію про заплановані тренувальні сесії (ідентифікатор тренера trainer_id BIGINT FK, назва title VARCHAR(255), дата date DATE, час початку/закінчення start_time TIME, end_time TIME, максимальна кількість учасників max_participants INT);

– бронювання сесій клієнтами реалізовано через таблицю session_bookings, що пов'язує training_sessions та users (клієнтів) через training_session_id BIGINT FK та client_id BIGINT FK, фіксуючи статус бронювання;

– таблиця trainer_schedules дозволяє тренерам визначати свою доступність (день тижня day_of_week VARCHAR(255), час початку/закінчення start_time TIME, end_time TIME, статус доступності is_available TINYINT(1));

– для персоналізованої взаємодії передбачено таблицю client_notes, де тренер (trainer_id BIGINT FK) може залишати нотатки щодо клієнта (client_id BIGINT FK);

– формалізація відносин між тренером та клієнтом здійснюється через таблицю trainer_client_relationships, що містить ідентифікатори тренера та клієнта, статус відносин, дати початку/закінчення співпраці.

4) Електронна комерція:

– таблиця products містить каталог товарів (назва name VARCHAR(255), ціна price DECIMAL(10,2), кількість на складі stock_quantity INT, зображення image VARCHAR(255), галерея зображень gallery_images JSON);

– категоризація товарів реалізована через таблицю product_categories (назва категорії name VARCHAR(255)), яка підтримує ієрархічну структуру через

поле parent_id BIGINT FK, що посилається на ту ж таблицю. Зв'язок з товарами встановлюється через category_id BIGINT FK у таблиці products;

– таблиця carts слугує для зберігання вмісту кошиків користувачів, де кожен запис представляє окремий товар (product_id BIGINT FK) у кошику конкретного користувача (user_id BIGINT FK) із зазначенням кількості quantity INT та ціни price DECIMAL(10,2);

– інформація про замовлення агрегується в таблиці orders (ідентифікатор користувача user_id BIGINT FK, загальна сума total_amount DECIMAL(10,2), статус замовлення status VARCHAR(255), адреса доставки);

– деталізація кожного замовлення здійснюється через таблицю order_items, що пов'язує orders та products через відповідні зовнішні ключі та зберігає кількість, ціну за одиницю та інші деталі для кожного товару в замовленні;

– таблиця payments фіксує інформацію про платежі, пов'язані із замовленнями (order_id BIGINT FK), включаючи суму платежу amount DECIMAL(10,2), валюту currency VARCHAR(3), статус status VARCHAR(255) та деталі платіжної транзакції payment_details JSON.

3.5 Розробка діаграми компонентів та розгортання

Діаграма компонентів є важливим інструментом для відображення організаційної та структурної композиції системи на вищому рівні абстракції, ніж окремі класи, демонструючи логічні групи функціональності та взаємозв'язки між ними. У контексті розроблюваного вебзастосунку управління фітнес-клубом така діаграма дозволяє чітко окреслити компонентну архітектуру системи, виділяючи чіткі функціональні блоки та інтерфейси їхньої взаємодії. Ця візуалізація сприяє розумінню не лише статичної структури системи, але й потоків даних між компонентами, що особливо важливо для комплексних програмних рішень із багатошаровою архітектурою.

Представлена діаграма компонентів (рис. 3.9) вебзастосунку управління фітнес-клубом демонструє структурну організацію системи на рівні архітектурних елементів високого рівня абстракції. Центральною віссю архітектури виступає

компонент MainComponent, що інкапсулює сутність GymApplication – ядро вебзастосунку, яке координує взаємодію між іншими архітектурними елементами системи. Цей компонент забезпечує узгодженість роботи різних шарів, утворюючи єдину точку входу для запуску та конфігурації системи. Особливістю компонента є його безпосередній зв'язок із презентаційною логікою та бізнес-логікою.

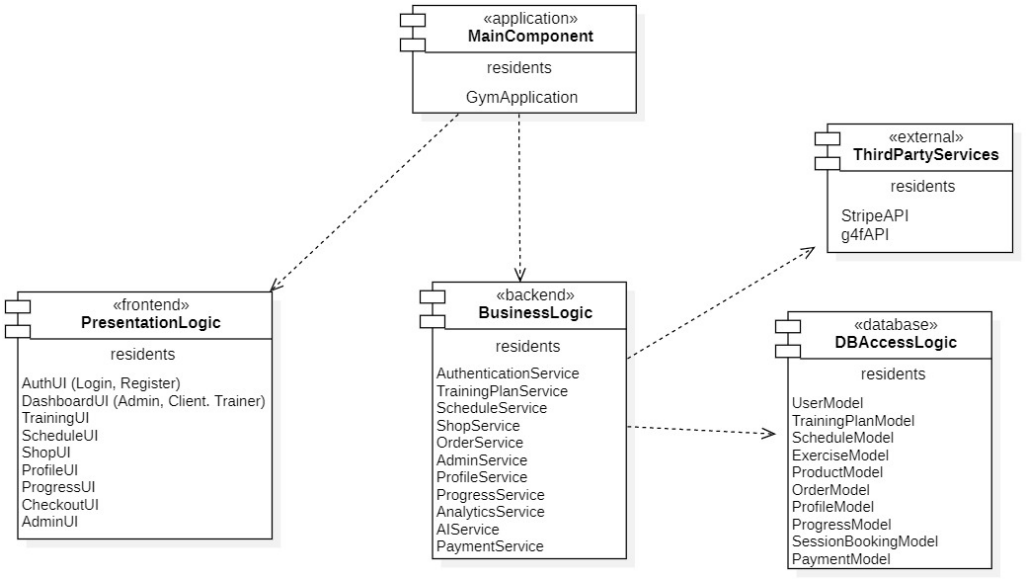

Рисунок 3.9 – Діаграма компонентів

Презентаційний шар представлено компонентом PresentationLogic з позначкою, що охоплює численні інтерфейсні елементи, орієнтовані на взаємодію з різними типами користувачів системи. У межах компонента виокремлено спеціалізовані модулі: AuthUI для автентифікації (реєстрація та вхід), DashboardUI з диференціацією інтерфейсів за користувацькими ролями (адміністратор, клієнт, тренер), а також окремі частини інтерфейсу для роботи з тренуваннями, розкладом, магазином, профілем, прогресом і оформленням платежів. Така структура забезпечує чітке відділення візуальної і логічної складової та спрощує підтримку й розширення фронтенда.

Ядро бізнес-логіки системи втілено в компоненті BusinessLogic. Ключовими складовими цього шару є спеціалізовані сервіси, що інкапсулюють окремі домени застосунку: AuthenticationService для керування автентифікацією та авторизацією, TrainingPlanService та ScheduleService для організації тренувального процесу, ShopService та OrderService для забезпечення електронної комерції, ProfileService

та ProgressService для роботи з користувацькими даними, AnalyticsService для аналізу та звітності, AIService для інтеграції алгоритмів ШІ та PaymentService для обробки платежів.

У DBAccessLogic зібрані моделі даних, які відображають усі сутності предметної області: користувачі, тренувальні плани, розклади, вправи, продукти, замовлення, інформація про прогрес тощо. Нарешті, компонент ThirdPartyServices групує зовнішні інтеграції, що забезпечує взаємодію з зовнішніми сервісами, зокрема StripeAPI для обробки платежів та g4fAPI для генерації персоналізованих тренувальних рекомендацій із використанням машинного навчання.

В свою чергу, діаграма розгортання – це тип UML діаграми, що визначає фізичне обладнання, на якому працюватиме програмна система. Він також визначає, як ПЗ розгортається на базовому обладнанні. Він прив’язує частини програмного забезпечення системи до пристрою, який збирається її виконувати. Діаграма розгортання відображає архітектуру ПЗ, створену при проєктуванні, на архітектуру фізичної системи, яка її виконує. Для вебзастосунку управління фітнес-клубом, що характеризується комплексною багатошаровою структурою, така діаграма набуває особливого значення, створюючи чітке уявлення про інфраструктурні рішення, необхідні для розгортання та функціонування системи в продуктивному середовищі.

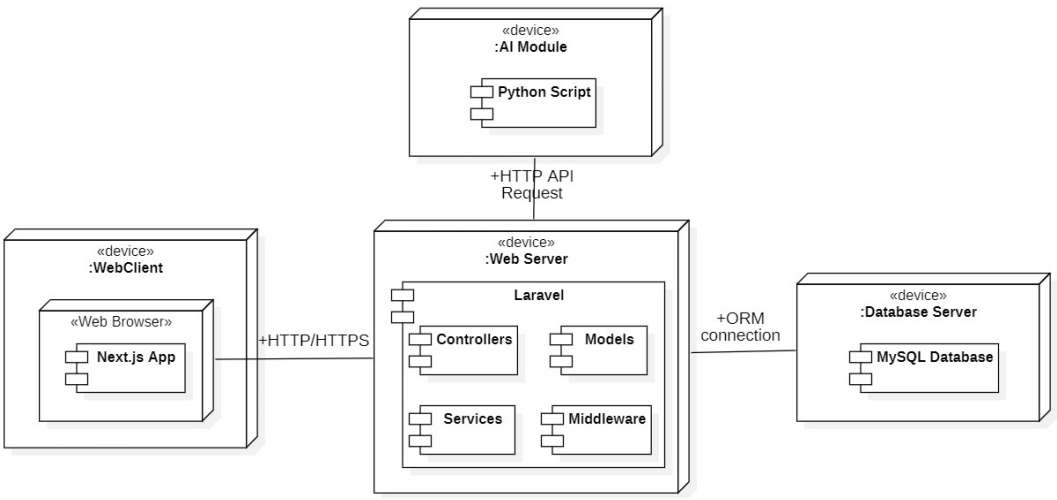

Рисунок 3.10 – Діаграма розгортання

Діаграма розгортання (рис. 3.10) вебзастосунку демонструє чітко структуровану фізичну архітектуру системи, візуалізуючи розподіл програмних компонентів між обчислювальними вузлами та комунікаційні механізми їхньої взаємодії. Ключовими вузлами інфраструктури виступають: клієнтський пристрій (WebClient) із компонентом Next.js App, інкапсульованим у середовищі браузера, що забезпечує інтерактивний інтерфейс користувача, серверна платформа (Web Server) з архітектурно структурованим Laravel-застосунком, декомпонованим на функціональні модулі (Controllers, Models, Services, Middleware), що реалізують бізнес-логіку системи, сервер управління даними (Database Server) та спеціалізований модуль штучного інтелекту (AI Module), реалізований із використанням Python для генерації персоналізованих тренувальних програм.

Висновки до розділу 3

В третьому розділі кваліфікаційної бакалаврської роботи розроблено архітектуру вебзастосунку для управління фітнес-клубом за допомогою різних типів UML-діаграм. Діаграми варіантів використання показали, які дії можуть виконувати різні користувачі системи: клієнти, тренери та адміністратори, що допомогло чітко визначити їхні можливості в програмі.

Діаграми послідовності показали, як різні частини системи взаємодіють між собою під час виконання основних завдань, а діаграма класів визначила основні об'єкти системи та зв'язки між ними. Також було створено схему бази даних та діаграми компонентів і розгортання, що допомогли краще зрозуміти, як система буде працювати на фізичному рівні та як будуть організовані її складові частини.

4 ПРОГРАМНА РЕАЛІЗАЦІЯ ВЕБЗАСТОСУНКУ

4.1 Структура вебзастосунку

Програмна реалізація вебзастосунку управління фітнес-клубом ґрунтується на сучасних принципах веброзробки та дотриманні парадигми розділення відповідальності, що забезпечує високу модульність та масштабованість системи. Розроблена система спирається на клієнт-серверну архітектуру з виразним розмежуванням клієнтської та серверної логіки, що дозволяє оптимізувати розробку, тестування та подальший супровід окремих компонентів.

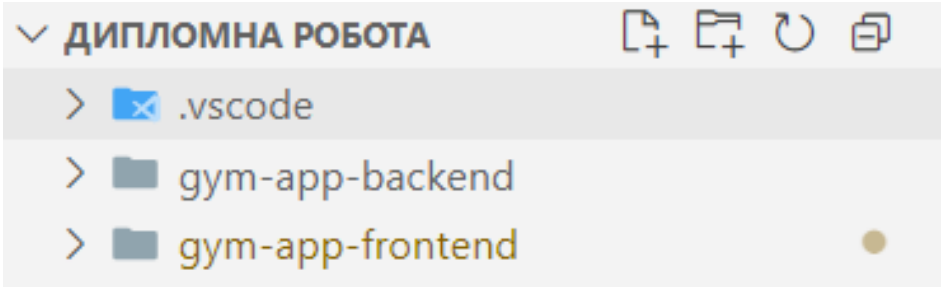

Рисунок 4.1 – Загальна структура застосунку

Загальна структурна організація вебзастосунку (рис. 4.1) складається з двох основних частин: клієнтської (фронтенд) та серверної (бекенд) підсистем, взаємодія між якими здійснюється через RESTful API. Клієнтська частина реалізована з використанням фреймворку Next.js, що забезпечує оптимальну продуктивність і взаємодію з користувачем через рендеринг на стороні сервера та статичну генерацію сторінок. Серверна частина побудована на базі фреймворку Laravel, який надає потужні інструменти для розробки бізнес-логіки, взаємодії з базою даних та обробки запитів.

Структура бекенду (рис. 4.2) дотримується архітектурних принципів Laravel та організована з урахуванням концепції розділення відповідальності, що сприяє підтримуваності та розширюваності системи.

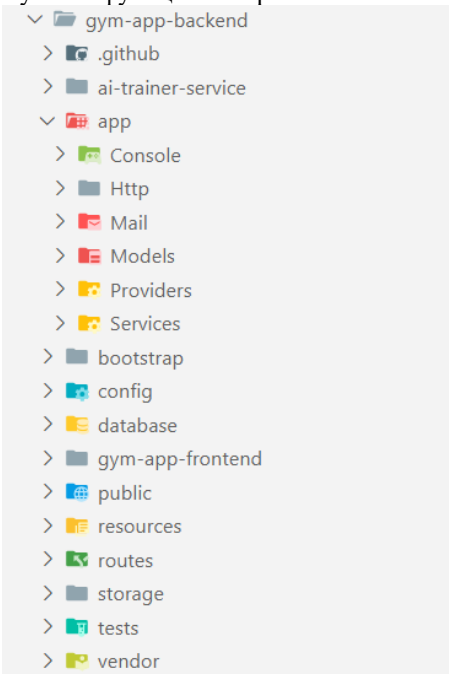

Рисунок 4.2 – Структура бекенду

Серверна частина включає такі ключові компоненти:

– директорія app/Models містить набір класів, що відповідають за взаємодію з базою даних через ORM Eloquent. Кожна модель представляє окрему сутність предметної області (User, TrainingPlan, Product, Order тощо) та інкапсулює логіку роботи з відповідними даними;

– директорія app/Http/Controllers зосереджує контролери, що опрацьовують HTTP-запити, організовані за функціональними модулями: API/AuthController (автентифікація), API/TrainingPlanController (управління тренувальними планами), API/ScheduleController (розклад тренувань) та інші;

– директорія app/Services виокремлює бізнес-логіку застосунку в спеціалізовані сервісні класи, забезпечуючи додатковий рівень абстракції між контролерами та моделями;

– директорія routes визначає маршрутизацію API-запитів до відповідних контролерів, структуруючи доступні ендпоінти системи за функціональними групами;

– директорія database містить міграції та сідери для створення й наповнення бази даних;

– директорія ai-trainer-service представляє окремий мікросервіс для генерації тренувальних програм із використанням алгоритмів машинного навчання, реалізований на Python та інтегрований з основним застосунком через API.

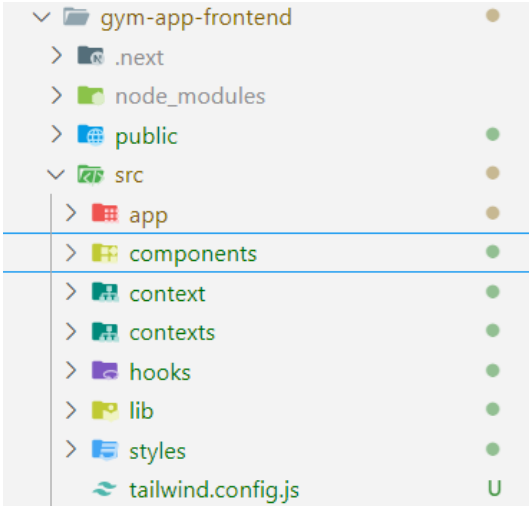

Рисунок 4.3 – Структура фронтенду

Клієнтська частина вебзастосунку (рис. 4.3) реалізована з використанням фреймворку Next.js та організована відповідно до принципів компонентної архітектури, що забезпечує повторне використання коду та спрощує тестування:

– директорія src/app містить основні сторінки застосунку, організовані за маршрутами відповідно до концепції файлової маршрутизації Next.js. Виокремлено директорії (auth) для сторінок автентифікації та (dashboard) для основного функціоналу після авторизації;

– директорія src/components зосереджує багаторазові компоненти інтерфейсу, структуровані за функціональним призначенням: ui (базові елементи інтерфейсу), forms (компоненти форм), dashboard (компоненти для панелі керування), layout (структурні компоненти) тощо;

– директорія src/lib містить допоміжні класи та хуки для взаємодії з API, управління станом, форматування даних та інших службових операцій;

– директорія src/hooks включає спеціалізовані React-хуки для інкапсуляції логіки, пов'язаної з отриманням та обробкою даних, взаємодією з користувачем та управлінням життєвим циклом компонентів.

4.2 Реалізація ключових функціональних модулів

У межах розробленого вебзастосунку управління фітнес-клубом неможливо представити всю кодову базу через значний обсяг програмних компонентів та модулів. Втім, доцільно розглянути ключові фрагменти коду, що відображають архітектурні рішення та підходи до реалізації основних функціональних вимог системи.

4.2.1 Модуль створення та відстеження тренувальних планів

Одним із інноваційних аспектів розробленого вебзастосунку є генерація персоналізованих тренувальних планів з використанням технологій ШІ. Цей функціонал реалізовано через спеціалізований Python-мікросервіс та сервіс G4FTrainerService для комунікації з зовнішніми API.

```python
@app.post("/generate-plan")
async def generate_training_plan(request: TrainingPlanRequest):
    """...
    try:
        logger.info(f"Received request for plan: {request.name}")

        prompt = format_prompt(request.dict())

        response = None

        try:
            logger.info("Trying automatic provider selection")
            response = g4f.ChatCompletion.create(...

            if response:
                logger.info("Got response from automatic provider")
        except Exception as e:
            logger.warning(f"Automatic selection failed: {str(e)}")

        # Спроба використання доступних провайдерів
        if not response:
            for provider in AVAILABLE_PROVIDERS:...

        if not response:
            logger.error("All providers failed, using default plan")
            return {"success": False, "data": create_default_plan()}

        logger.info(f"Raw response: {response[:200]}...")
        plan_data = parse_json_from_response(response)

        return {
            "success": True,
            "data": plan_data
        }

    except Exception as e:
        logger.error(f"Error generating training plan: {str(e)}")
        return {
            "success": False,
            "message": f"Помилка при генерації плану тренувань: {str(e)}",
            "data": create_default_plan()
        }
```

Рисунок 4.4 – Створення тренувального плану за допомогою ШІ

Представлений фрагмент коду (рис. 4.4) демонструє реалізацію головного API-ендпоінту /generate-plan, що відповідає за створення персоналізованих тренувальних планів. Алгоритм функціонування базується на отриманні запиту з

параметрами користувача, формуванні спеціалізованого промпту та його подальшій обробці через g4f бібліотеку. Система передбачає багаторівневий механізм відмовостійкості: спочатку виконується спроба автоматичного вибору провайдера моделі, при невдачі – послідовний перебір доступних провайдерів з особливими налаштуваннями для окремих сервісів (You, Yqcloud), а у випадку повної недоступності зовнішніх сервісів – повернення стандартного плану тренувань. Такий підхід забезпечує стабільність функціонування системи навіть при тимчасових збоях зовнішніх залежностей, що критично важливо для користувацького досвіду.

Окрім Python-мікросервісу, в архітектурі системи розроблено PHP-сервіс G4fTrainerService, який забезпечує інтеграцію між основним Laravel-застосунком та Python-мікросервісом:

```php
public function generateTrainingPlan(array $prompt): array
{
    try {
        Log::info(message: 'Sending request to G4F Trainer Service', context: ['prompt' => $prompt]);

        $response = Http::timeout(seconds: 120)->post(url: $this->apiUrl . '/generate-plan', data: $prompt);

        if ($response->successful()) {
            $result = $response->json();

            if (isset($result['success']) && $result['success'] && isset($result['data'])) {
                Log::info(message: 'Successfully received plan from G4F Trainer Service');
                return [
                    'success' => true,
                    'data' => $result['data']
                ];
            } else {…
            }
        } else {…
        }
    } catch (\Exception $e) {…
    }
}
```

Рисунок 4.5 – Функція підключення до Python-мікросервісу з Laravel-застосунку

Наведений фрагмент коду G4fTrainerService (рис. 4.5) демонструє міст між основним Laravel-застосунком та Python-мікросервісом. Сервіс встановлює HTTP-з'єднання з мікросервісом, передає параметри запиту та обробляє відповідь. Завдяки реалізації багаторівневої обробки помилок та логування, система зберігає стабільність навіть при збоях зовнішнього сервісу. Збільшений таймаут до 120 секунд забезпечує достатньо часу для генерації плану, враховуючи складність операцій, які виконуються на стороні ШІ-моделі.

4.2.2 Модуль управління розкладом тренувань

Модуль управління розкладом є одним із критичних компонентів вебзастосунку фітнес-клубу, що забезпечує синхронізацію роботи тренерів та клієнтів. У реалізації цього модуля особливу увагу приділено диференціації функціональності залежно від ролі користувача.

```php
public function index(Request $request): JsonResponse|mixed
{
    $user = Auth::user();
    $startDate = $request->input(key: 'start_date', default: Carbon::now()->startOfWeek()->toDateString());
    $endDate = $request->input(key: 'end_date', default: Carbon::now()->endOfWeek()->toDateString());
    $isSummary = $request->input(key: 'summary', default: false);

    if ($user->hasRole(role: 'trainer')) {
        if ($isSummary) {
            $scheduledCount = TrainingSession::where(column: 'trainer_id', operator: $user->id)
                ->where(column: 'status', operator: 'scheduled')
                ->where(column: 'date', operator: '>=', value: Carbon::today()->toDateString())
                ->count();

            return response()->json(data: [
                'success' => true,
                'scheduled_sessions_count' => $scheduledCount
            ]);
        }

        $sessions = TrainingSession::where(column: 'trainer_id', operator: $user->id)
            ->whereBetween(column: 'date', values: [$startDate, $endDate])
            ->with(relations: [
                'bookings.client' => function ($query): void {
                    $query->select('id', 'name', 'email');
                }
            ])
            ->orderBy(column: 'date')
            ->orderBy(column: 'start_time')
            ->get();

        return response()->json(data: [...
        ]);
    }

    if ($user->hasRole(role: 'client')) {...
    }

    return response()->json(data: [
        'success' => false,
        'message' => 'Unauthorized'
    ], status: 403);
}
```

Рисунок 4.6 – Фрагмент методу index() в ScheduleController

Наведений фрагмент методу index() в ScheduleController (рис. 4.6) демонструє адаптивність системи до типу користувача. Для тренерів надається розширена інформація, включаючи дані про клієнтів, записаних на їхні сесії, тоді як клієнти отримують лише доступні для запису тренування. Важливою архітектурною особливістю є використання вкладених зв'язків у запитах на вибірку даних, що забезпечує оптимізацію кількості звернень до бази даних.

```php
public function book($id): JsonResponse|mixed
{
    $user = Auth::user();

    if (!$user->hasRole(role: 'client')) {
        return response()->json(data: [ ···
        ], status: 403);
    }

    $session = TrainingSession::findOrFail(id: $id);

    if (!$session->isAvailable()) {
        return response()->json(data: [ ···
        ], status: 400);
    }

    $isClientTrainer = $user->trainers()
        ->where(column: 'trainer_id', operator: $session->trainer_id)
        ->where(column: 'status', operator: 'active')
        ->exists();

    if (!$isClientTrainer) {
        return response()->json(data: [ ···
        ], status: 403);
    }

    $existingBooking = SessionBooking::where(column: 'training_session_id', operator: $id)
        ->where(column: 'client_id', operator: $user->id)
        ->first();

    if ($existingBooking) { ···
    }

    $booking = SessionBooking::create(attributes: [ ···
    ]);

    return response()->json(data: [
        'success' => true,
        'message' => 'Session booked successfully',
        'data' => $booking
    ], status: 201);
}
```

Рисунок 4.7 – Фрагмент методу book() в ScheduleController

Метод book() (рис. 4.7) демонструє критично важливу бізнес-логіку запису на тренування. Система перевіряє не лише наявність у користувача ролі клієнта та доступність тренування, але й наявність активного зв'язку між клієнтом і тренером. Такий підхід гарантує цілісність даних і захищає систему від несанкціонованих записів, що підвищує загальну надійність програмного рішення.

4.2.3 Модуль аналітики та відстеження прогресу

Ключовою інноваційною складовою вебзастосунку фітнес-клубу є розвинений модуль аналітики і відстеження прогресу, що надає користувачам інструменти для кількісного аналізу своїх фізичних змін та візуалізації тренувальних результатів.

```php
public function index(Request $request): JsonResponse|mixed
{
    $user = $request->user();

    $startDate = $request->input(key: 'start_date');
    $endDate = $request->input(key: 'end_date');
    $metric = $request->input(key: 'metric', default: 'weight');

    $query = UserProgress::where(column: 'user_id', operator: $user->id)->orderBy(column: 'date', direction: 'asc');

    if ($startDate) {
        $query->where(column: 'date', operator: '>=', value: $startDate);
    }

    if ($endDate) {
        $query->where(column: 'date', operator: '<=', value: $endDate);
    }

    $progressRecords = $query->get();

    if ($metric && $metric !== 'all') {
        $chartData = $progressRecords->map(callback: function (UserProgress $record) use ($metric): array {
            return [
                'date' => $record->date->format('Y-m-d'),
                'value' => $record->{$metric},
            ];
        })->filter(callback: function (array{date: mixed, value: mixed} $item)… {
            return $item['value'] !== null;
        })->values();

        return response()->json(data: [
            'progress' => $progressRecords,
            'chart_data' => $chartData,
            'metric' => $metric,
        ]);
    }

    return response()->json(data: [
        'progress' => $progressRecords,
    ]);
}
```

Рисунок 4.8 – Функція для відображення прогресу користувача

Метод index() (рис. 4.8) демонструє гнучкість системи у відображенні прогресу користувача. Програмне рішення підтримує параметризований пошук за часовими рамками та окремими метриками, що дозволяє користувачам концентруватись на конкретних показниках свого фізичного розвитку. Ключовою особливістю даного методу є автоматична підготовка двох форматів даних – повного набору записів та спеціально сформованих даних для візуалізації через графіки. Використання функціональних можливостей колекцій Laravel (методи map(), filter(), values()) демонструє ефективне застосування сучасних програмних парадигм для трансформації даних.

```php
public function summary(Request $request): JsonResponse|mixed
{
    $user = $request->user();

    $firstRecord = UserProgress::where(column: 'user_id', operator: $user->id)
        ->orderBy(column: 'date', direction: 'asc')
        ->first();

    $latestRecord = UserProgress::where(column: 'user_id', operator: $user->id)
        ->orderBy(column: 'date', direction: 'desc')
        ->first();

    if (!$firstRecord || !$latestRecord) {
        return response()->json(data: [
            'message' => 'Недостатньо даних для аналізу',
            'summary' => null,
        ]);
    }

    // Розрахунок інших змін показників...
    $daysDifference = $firstRecord->date->diffInDays($latestRecord->date);

    $weeksCount = $daysDifference / 7;

    // Розрахунок тижневих змін...
    $summary = [
        'first_date' => $firstRecord->date->format('Y-m-d'),
        'latest_date' => $latestRecord->date->format('Y-m-d'),
        'days_tracked' => $daysDifference,
        'total_records' => UserProgress::where(column: 'user_id', operator: $user->id)->count(),
        'weight' => [
        ],
        'body_fat' => [
        ],
        'muscle_mass' => [
        ],
        'measurements' => [
        ],
    ];

    return response()->json(data: [
        'summary' => $summary,
    ]);
```

Рисунок 4.9 – Функція, що виконує комплексний аналіз прогресу користувача

Метод summary() (рис. 4.9) представляє інтелектуальне ядро аналітичної підсистеми, що виконує комплексний аналіз прогресу користувача. Метод розраховує як абсолютні зміни показників між першим та останнім записами, так і нормалізовані тижневі зміни, що дозволяє оцінити темп прогресу незалежно від тривалості тренувального періоду. Ключовою особливістю алгоритму є врахування можливості відсутності даних для певних метрик, що забезпечує стабільність роботи системи навіть з неповними наборами даних. У результаті формується багаторівнева структура даних, що охоплює всі аспекти фізичного розвитку користувача та забезпечує основу для подальшої візуалізації прогресу на клієнтській частині застосунку.

4.3 Інтерфейс користувача

Інтерфейс користувача становить ключову ланку взаємодії між обчислювальними можливостями вебзастосунку та користувачем, безпосередньо впливаючи на його суб'єктивне сприйняття функціональності системи незалежно від технічної складності реалізованих алгоритмів. При розробці інтерфейсу вебзастосунку управління фітнес-клубом особливу увагу приділено адаптивності відображення компонентів на різних пристроях, інтуїтивності навігаційних елементів та доступності інтерфейсу для користувачів різного рівня технічної підготовки. Принципи мінімалізму та функціональності, закладені в дизайн системи, дозволили створити середовище, що відповідає сучасним тенденціям проєктування інтерфейсів та забезпечує ефективне досягнення користувацьких цілей.

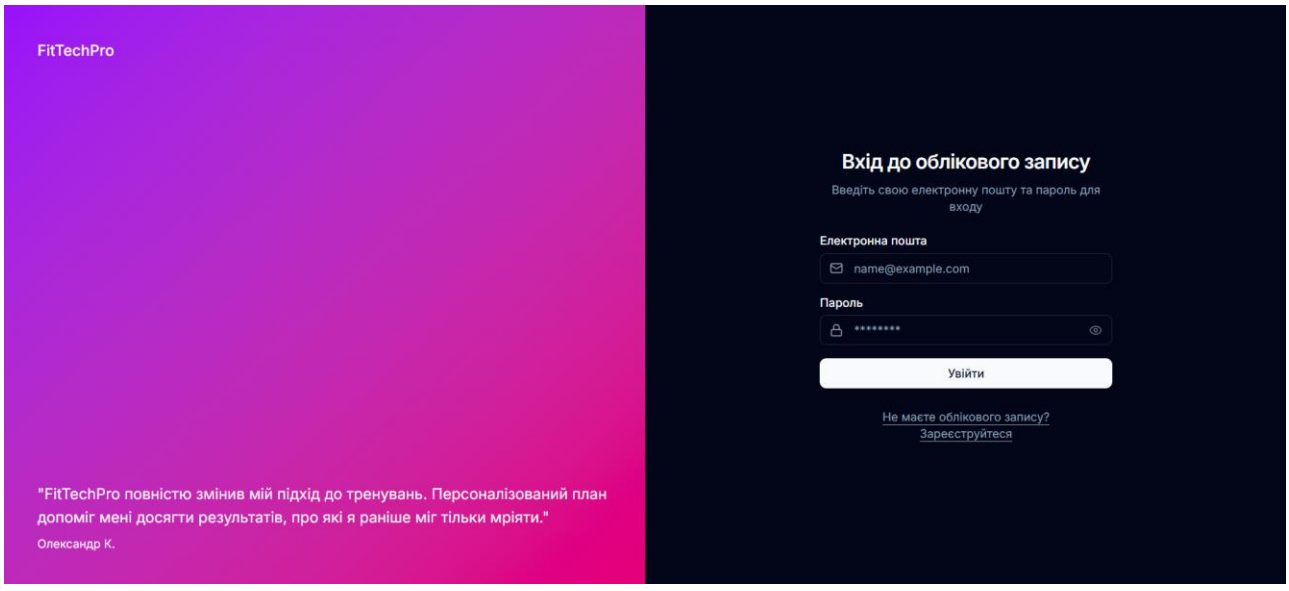

Рисунок 4.10 – Сторінка входу до облікового запису

Сторінка входу (рис. 4.10) містить форму авторизації з полями для вводу електронної пошти та паролю, кнопку «Увійти» при натисканні якої відбувається вхід до облікового запису та посилання на реєстрацію, якщо користувач ще немає акаунту.

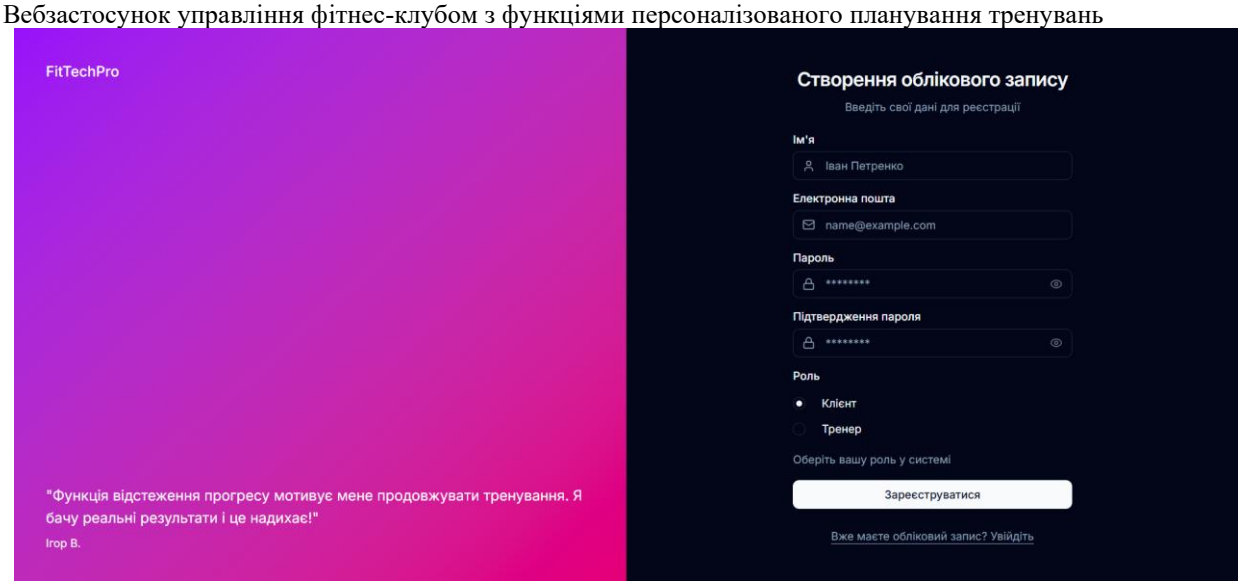

Рисунок 4.11 – Сторінка реєстрації

В свою чергу, сторінка реєстрації (рис. 4.11) має форму створення облікового запису з полями для вводу імені, електронної пошти, паролю та його підтвердження та вибору ролі користувача (клієнт або тренер). При натисканні кнопки «Зареєструватись» користувач перенаправляється на сторінку з інформацією, що необхідно підтвердити електронну пошту.

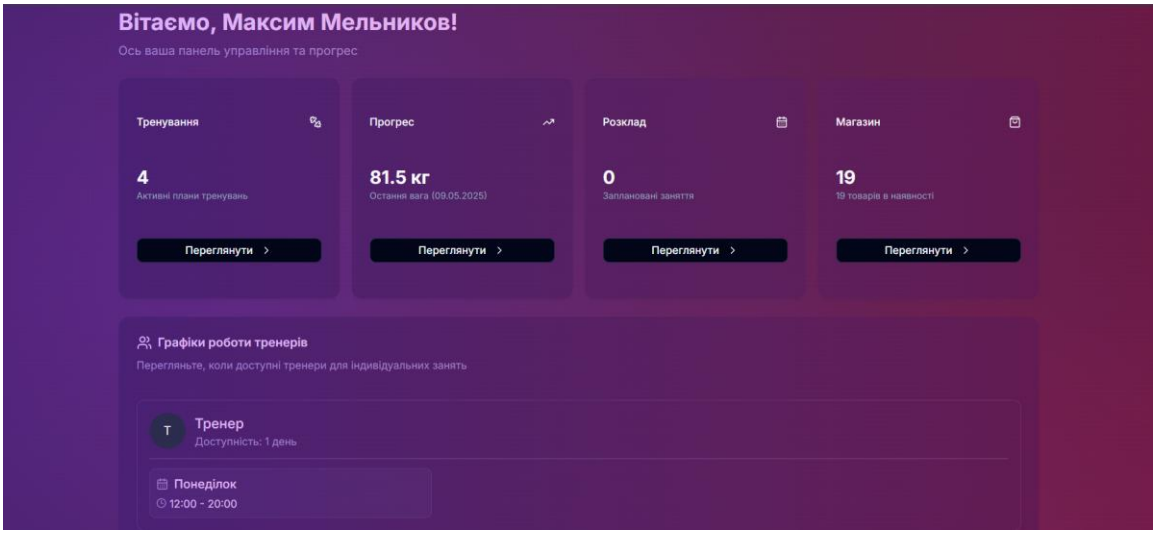

Рисунок 4.12 – Головна сторінка клієнта

Головна панель управління для клієнта фітнес-клуюу або іншими словами «дашборд» (рис. 4.12) показує вітання користувача, його ключові метрики (активні плани тренувань, останню записану вагу, заплановані заняття та товари в магазині) і графік роботи тренерів.

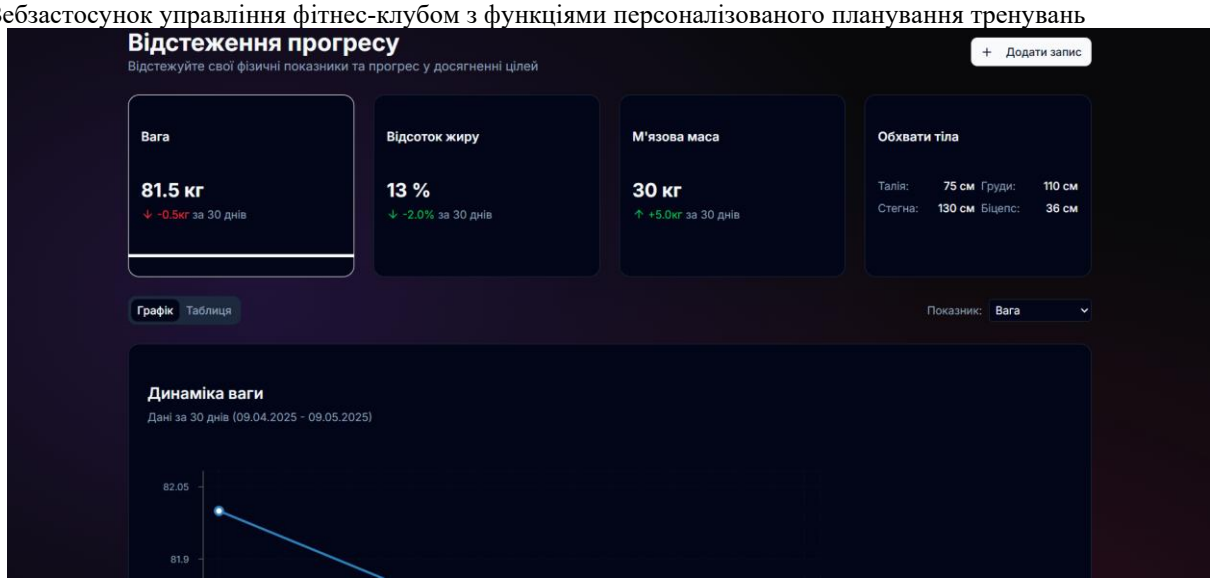

Рисунок 4.13 – Сторінка відстеження прогресу клієнту

Сторінка відстеження прогресу клієнту (рис. 4.13) візуалізує фізичні показники клієнта, а саме: вага, відсоток жиру, м'язова маса та обхвати тіла з динамікою змін за 30 днів та графіками прогресу або у вигляді табличних значень.

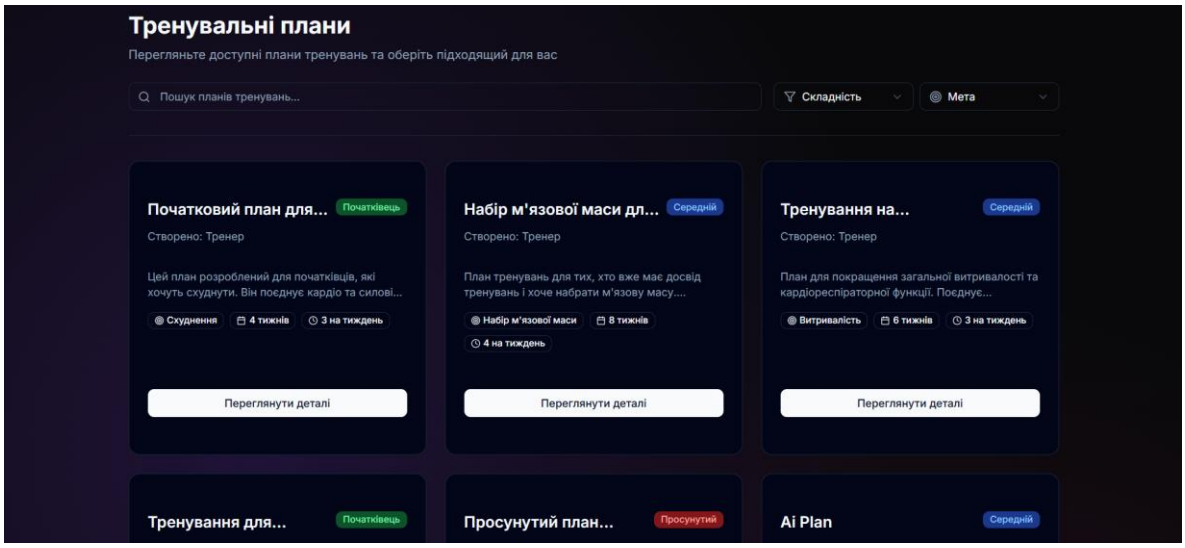

Рисунок 4.14 – Сторінка для перегляду тренувальних планів

Каталог тренувальних планів (рис. 4.14) дозволяє клієнту переглядати та вибирати плани тренувань різної складності з можливістю фільтрації та пошуку. Натиснувши на кнопку «Переглянути деталі», клієнт побачить детальну інформацію про обраний тренувальний план з можливістю перегляду вправ у дні тренувань, також є можливість розпочати план, зупинити або завершити його.

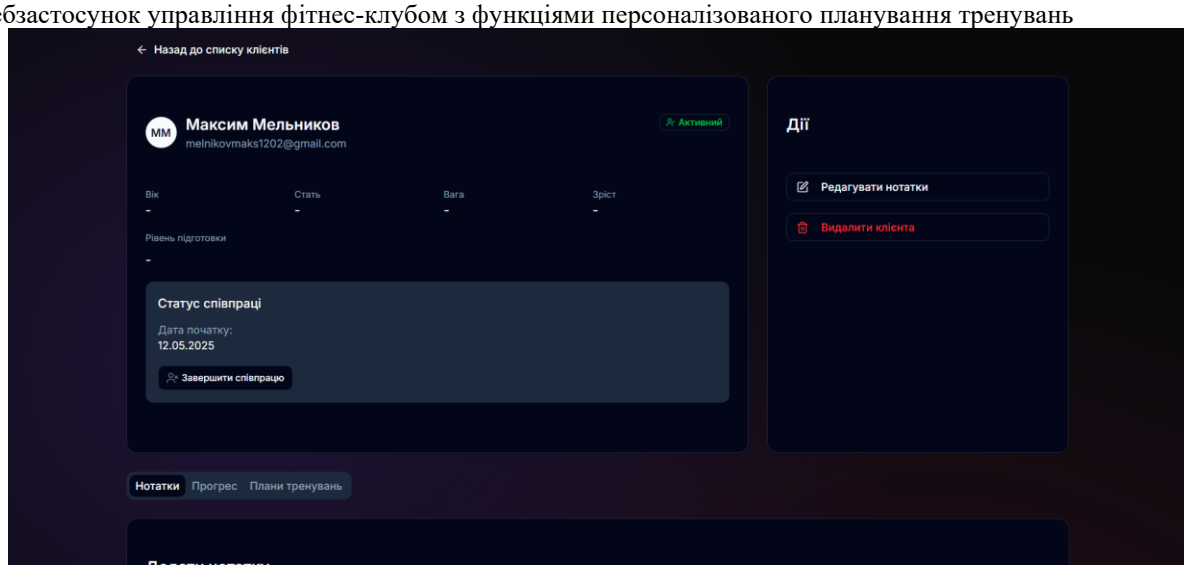

Рисунок 4.15 – Сторінка з детальною інформацією про клієнта

Сторінка з детальною інформацією про клієнта (рис. 4.15) доступна для користувача з роллю тренер. Дана сторінка містить детальну інформацію про користувача, статус співпраці та інструменти для керування (редагування нотаток, видалення клієнта). Також тренер має можливість подивитись прогрес свого клієнта та його активні плани тренувань, крім того, тренер може призначити самостійно певний план тренувань.

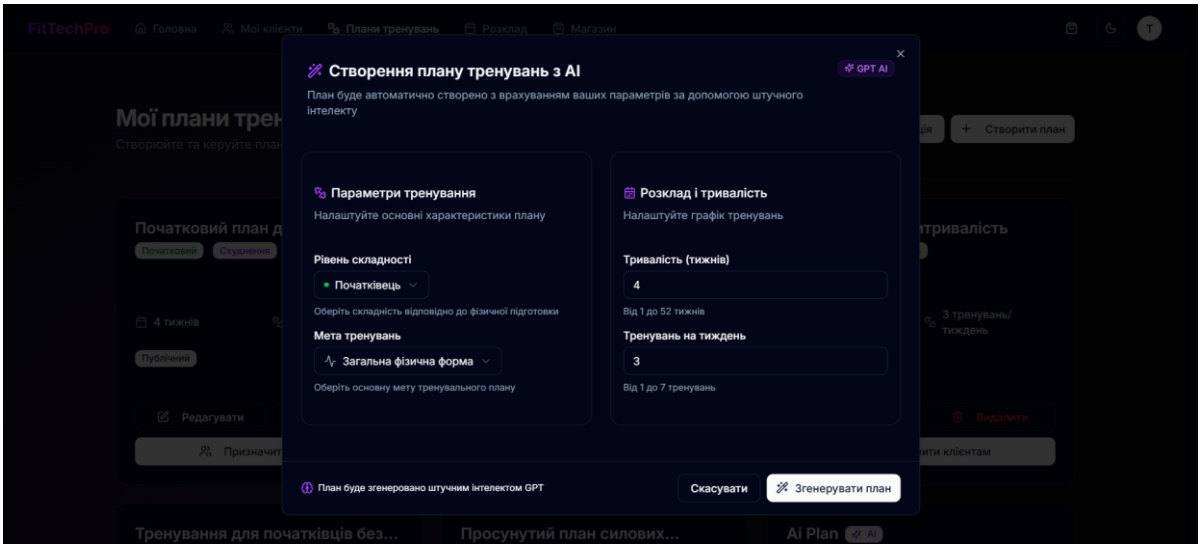

Рисунок 4.16 – Спливаюче вікно для створення тренувального плану з ШІ

Створення тренувального плану з ШІ (рис. 4.16) – це спливаюче вікно для генерації персоналізованих планів тренувань з налаштуванням параметрів тренування, розкладу і тривалості, а також можливістю вибору клієнта для якого

буде створений цей план, на основі його основних показників. При натисканні кнопки «Згенерувати план», відбувається запит до мікросервісу, який в свою чергу звертається до вільної моделі ШІ для генерації тренувального плану.

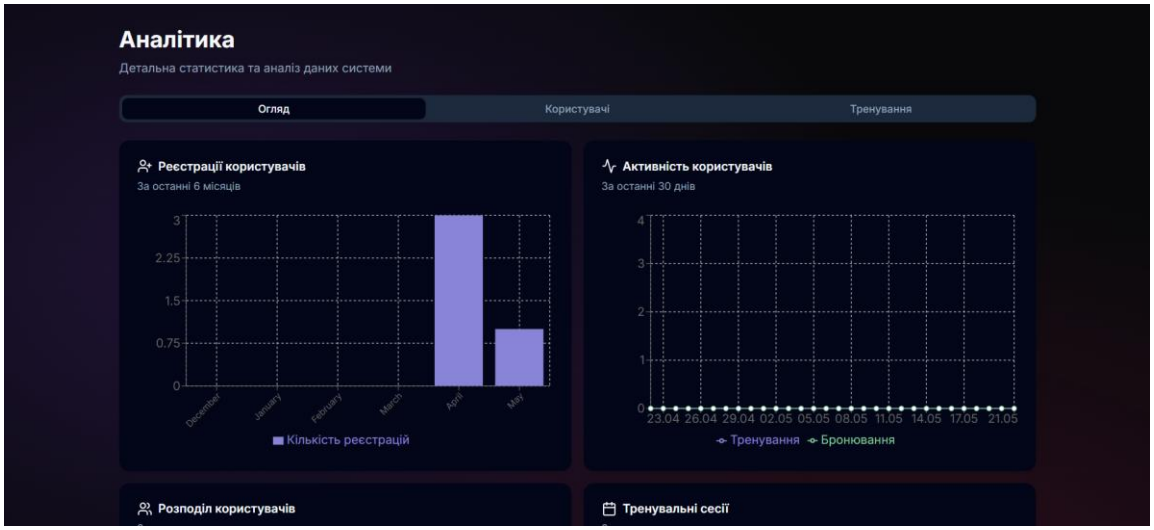

Рисунок 4.17 – Сторінка аналітики

Аналітична панель (рис. 4.17) доступна для користувача з роллю адміністратор та відображає статистику системи з графіками реєстрацій та активності користувачів. Також присутні графіки розподілу користувачів за ролями. Над графіками є перемикачі сторінок, на кожній з яких містяться аналітичні дані для користувачів чи тренувань з відповідними графіками.

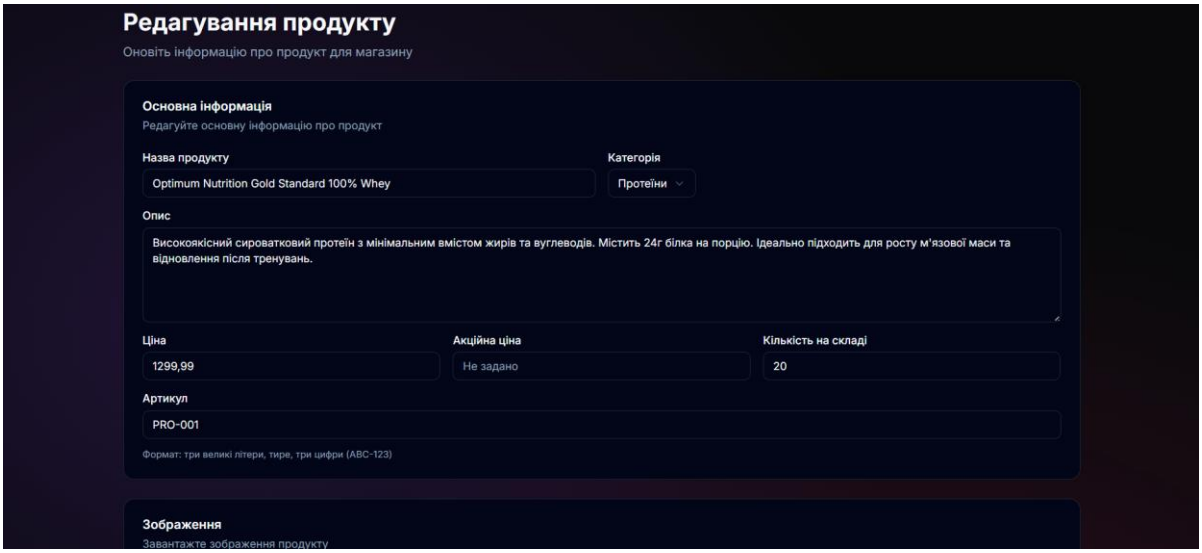

Рисунок 4.18 – Сторінка редагування продукту

Редагування продукту (рис. 4.18) являє собою форму для управління товаром в магазині з полями для зміи назви, категорії, опису, ціни, кількості та артикулу. Крім того, адміністратор має можливість змінити зображення продукту, яке буде відображатись на картці товару, а також може вказати додаткові характеристики, такі як порядок відображення, чи рекомендовний товар, чи новинка.

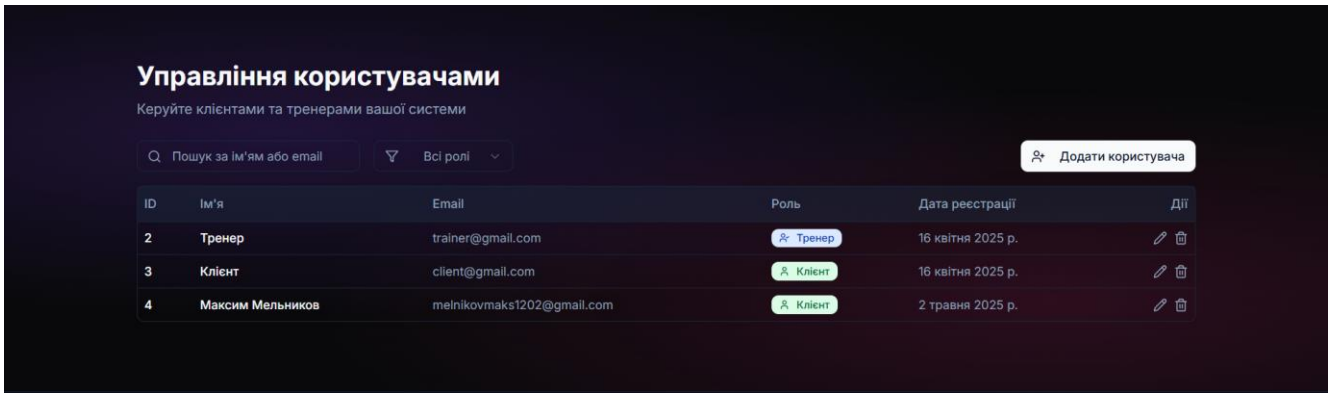

Рисунок 4.19 – Сторінка управління користувачами

Управління користувачами (рис. 4.19) – це сторінка на якій розміщена таблиця зареєстрованих користувачів. Адміністратор має можливість фільтрувати та шукати користувачів, такоє редагувати роль користувача чи взагалі видаляти його акаунт.

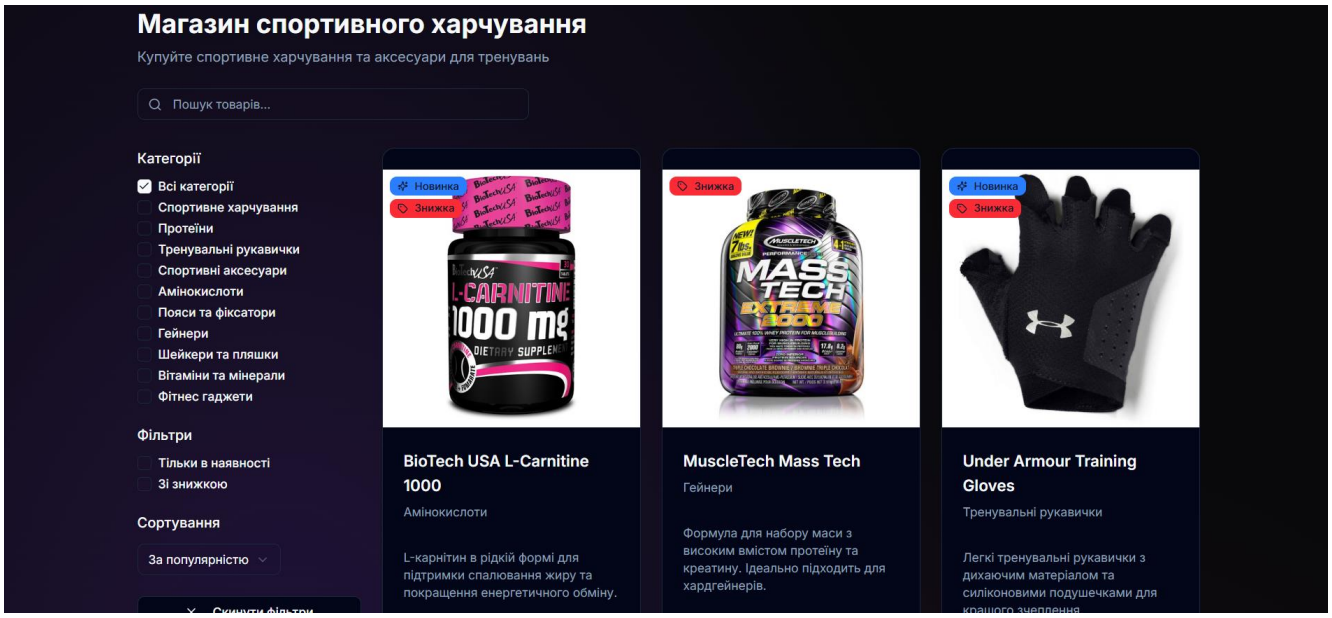

Рисунок 4.20 – Сторінка магазину продуктів

Магазин спортивного харчування (рис. 4.20) відображає каталог доступних спортивних товарів з можливістю пошуку товару та фільтрацією за певними характеристиками, які розміщені в панелі зліва від карток товарів. Клієнт або тренер має можливість переглянути опис товару, наявність товару, ціну та додати товар до кошику.

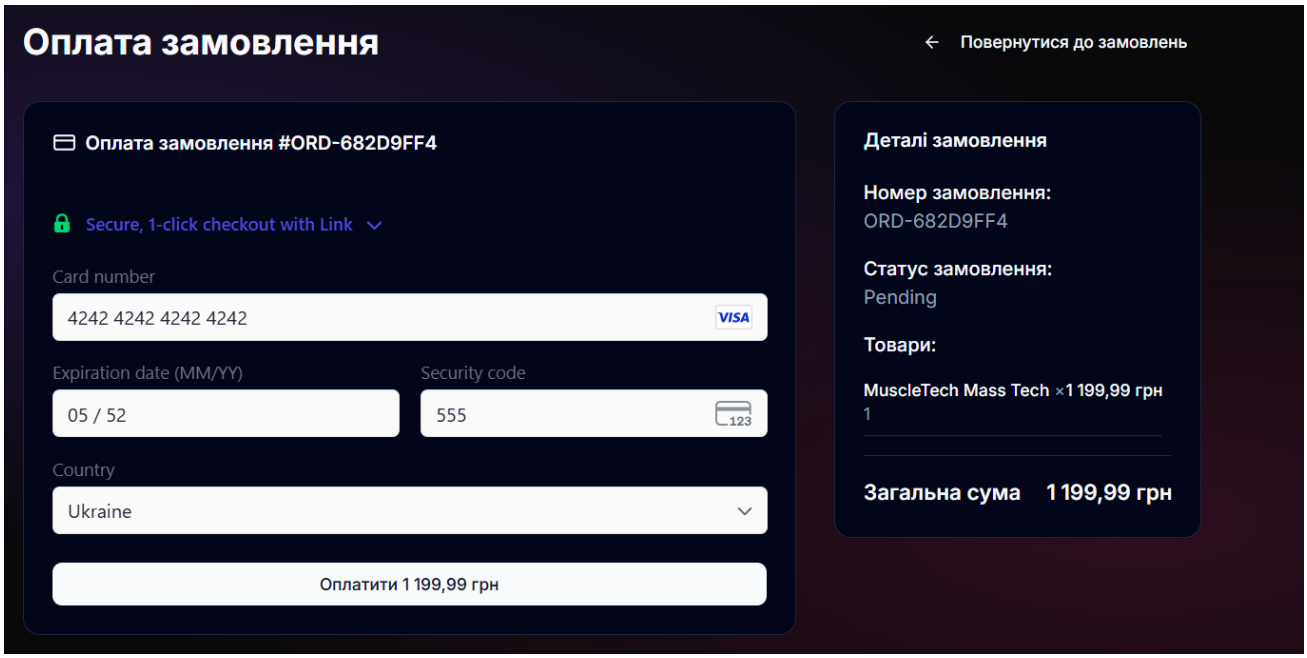

Рисунок 4.21 – Сторінка оплати замовлення

Сторінка оплати замовлення (рис. 4.21) має інтерфейс для введення даних платіжної картки та перегляду деталей замовлення. При натисканні кнопки «Оплатити», автоматично відбувається звернення до API платіжного сервісу Stripe, який працює в тестовому режимі та якщо введені дані коректні, то статус замовлення змінюється на «Оплачено».

4.4 Тестування вебзастосунку

Тестування є невід’ємною складовою процесу розробки сучасних вебзастосунків, оскільки дозволяє своєчасно виявляти та усувати помилки, забезпечуючи стабільність, надійність і відповідність функціональних можливостей заявленим вимогам.

У межах даного застосунку вебінтерфейс тестується вручну, що передбачає перевірку основних функцій, навігації, форм введення та відображення

повідомлень про помилки. А, в свою чергу, серверна частина вебзастосунку підлягає автоматизованому тестуванню за допомогою юніт-тестів, що дозволяє системно перевіряти коректність реалізації ключових бізнес-процесів. Для цього у проєкті використовуються засоби тестування, притаманні фреймворку Laravel, зокрема PHPUnit, що забезпечує написання, запуск і аналіз результатів тестів у структурованому вигляді.

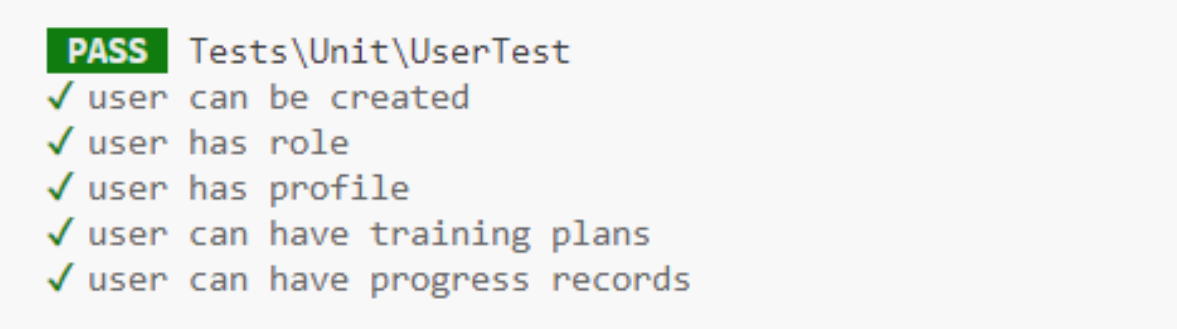

Рисунок 4.22 – Результат виконання юніт-тестів для UserTest

Юніт-тести охоплюють ключові аспекти функціонування моделі користувача (рис. 4.22). Основний акцент зроблено на тестуванні процесу створення користувача та його асоціації з різними ролями в системі, що є критично важливим для управління доступом. Також перевіряється наявність та коректність профілю користувача, що містить додаткову інформацію. Важливим елементом є тестування можливості користувача мати власні плани тренувань та записи прогресу, що відображає персоналізований підхід до користувачів.

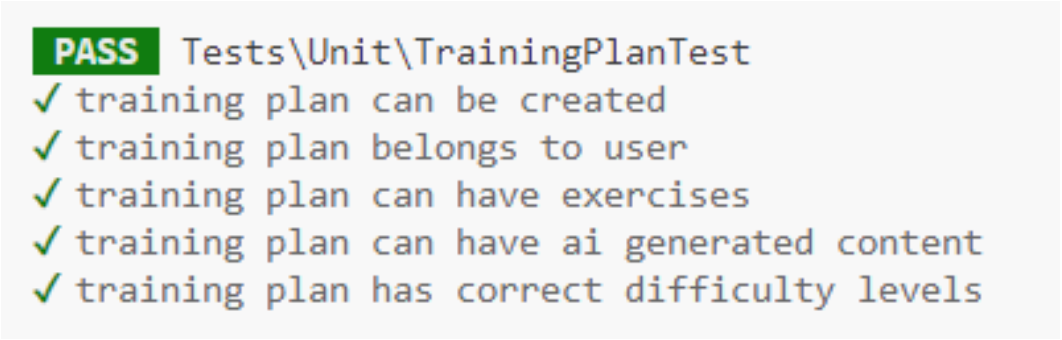

Рисунок 4.23 – Результат виконання юніт-тестів для TrainingPlanTest

Тести для сутності TrainingPlan (рис. 4.23) перевіяють функціонал, пов'язаний з планами тренувань. Проводиться тестування створення нового плану

тренувань, встановлення зв'язку між планом та користувачем, який його створив, а також додавання окремих вправ до плану. Особлива увага приділяється перевірці обробки контенту, що може бути згенерований ШІ, та коректності визначення і обробки різних рівнів складності тренувань, що підкреслює гнучкість системи.

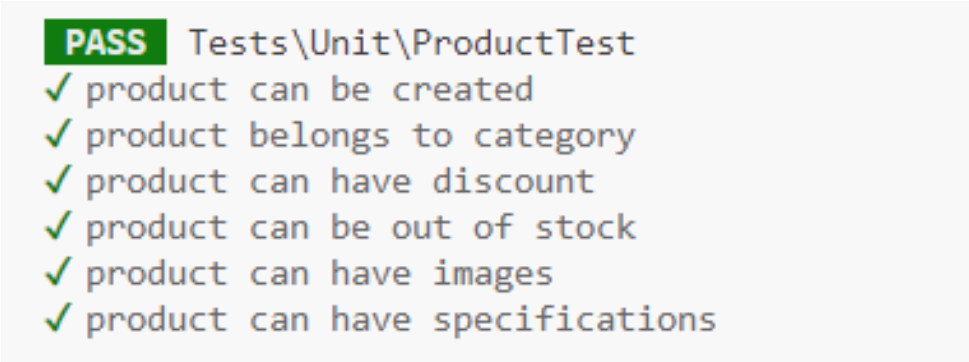

Рисунок 4.24 – Результат виконання юніт-тестів для ProductTest

Тести для сутності Product (рис. 4.24) – юніт-тести, спрямовані на оцінку функціональних можливостей продуктової моделі. Тести включають перевірку створення продукту та його асоціації з категорією, що забезпечує структурну цілісність даних. Також детально аналізуються аспекти, пов'язані з ціноутворенням, зокрема, застосування знижок, та управління запасами, що відображається у тестах на стан «немає в наявності». Додатково верифікується коректність збереження та доступу до зображень та специфікацій продукту.

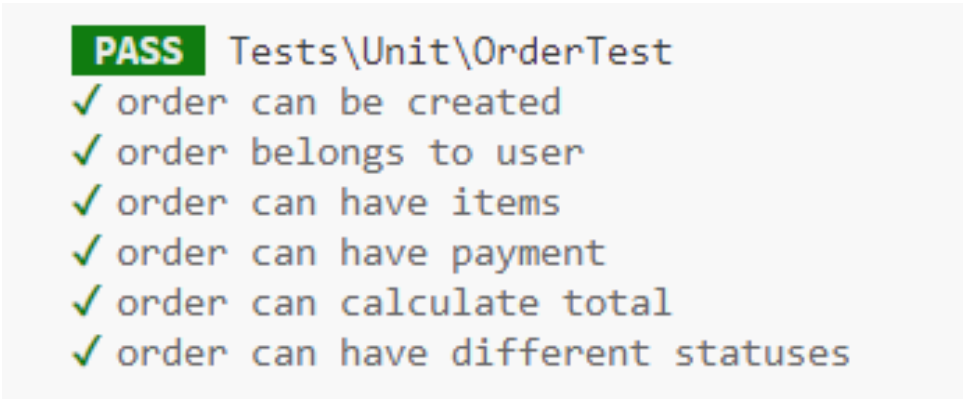

Рисунок 4.25 – Результат виконання юніт-тестів для OrderTest

Набір тестів (рис. 4.25) ретельно перевіряє функціональність, пов'язану із замовленнями у системі. Зокрема, здійснюється верифікація процесу створення замовлення, коректності зв'язку між замовленням та користувачем, а також

механізму додавання товарних позицій до замовлення. Крім того, тести охоплюють перевірку інтеграції платіжних даних та здатності системи точно розраховувати загальну вартість замовлення, включаючи перевірку різних можливих статусів замовлення.

Висновки до розділу 4

У четвертому розділі кваліфікаційної бакалаврської роботи було здійснено комплексну програмну реалізацію архітектури вебзастосунку, що охопила як серверний (gym-app-backend), так і клієнтський (gym-app-frontend) компоненти, забезпечивши функціональну повноту системи. В межах цього розділу розглянуто реалізацію ключових функціональних модулів – створення персоналізованих тренувальних планів, керування розкладом занять, а також реалізацію підсистеми аналітики, що дозволяє візуалізувати та аналізувати дані фітнес-клубу в цілому, а також виконано огляд інтерфейсу користувача з його описом.

Особливу увагу було приділено верифікації коректності реалізованого функціоналу шляхом застосування методології юніт-тестування. Розроблено та налагоджено серію юніт-тестів для критично важливих моделей даних системи, таких як User, TrainingPlan, Product, Order, Role, UserProfile, OrderItem, та ProductCategory. Успішне проходження розроблених тестів підтверджує функціональну цілісність та відповідність програмної реалізації поставленим вимогам.

ВИСНОВКИ

У межах кваліфікаційної бакалаврської роботи здійснено програмну реалізацію вебзастосунку управління фітнес-клубом з функціями персоналізованого планування тренувань. Проведено глибокий аналіз процесів функціонування фітнес-клубів, виявлено ключові компоненти систем управління та досліджено наявні ринкові аналоги з метою визначення оптимальних підходів до цифровізації. Це дозволило сформувати теоретичну базу та окреслити можливі проблеми для подальшої розробки.

Наступним кроком стало моделювання майбутньої системи, що включало вибір відповідного інструментарію та методів, деталізацію сценаріїв використання, які відображають взаємодію користувачів із застосунком, а також розробку специфікації вимог до програмного забезпечення. Цей етап заклав основу для архітектурних та функціональних рішень, забезпечивши чітке розуміння цілей та завдань розробки.

Фаза проєктування передбачала візуалізацію структури та поведінки системи за допомогою стандартних діаграм UML. Були побудовані діаграми варіантів використання для ілюстрації функціональних можливостей, діаграми послідовності для демонстрації взаємодії об'єктів у часі, діаграма класів, що відображає статичну структуру даних, а також схема бази даних, що деталізує організацію інформаційного сховища. Проєктування діаграм компонентів та розгортання окреслило фізичну архітектуру та середовище функціонування вебзастосунку.

Завершальним етапом роботи стала безпосередня програмна реалізація, описана у четвертому розділі, де було імплементовано як серверний, так і клієнтський компоненти, реалізовано ключові модулі, включаючи підсистеми створення та відстеження тренувальних планів, управління розкладом, а також аналітики та моніторингу прогресу користувачів. Окрім того, представлено користувацький інтерфейс, що забезпечує інтуїтивну взаємодію з системою. Важливим етапом реалізації стало модульне тестування, спрямоване на

верифікацію коректності функціонування окремих компонентів та їхньої відповідності структурі даних, визначеній у міграціях. Проведення тестів та внесення відповідних коригувань до фабрик даних та тестових сценаріїв підтвердило функціональну цілісність та готовність вебзастосунку до виконання поставлених завдань, що свідчить про успішне досягнення мети роботи.

У результаті досягнуто поставленої мети – створено стабільно працюючий застосунок, здатний покращити цифрове забезпечення фітнес-клубу й надати користувачам ефективні інструменти для управління особистим спортивним процесом.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Fitness Industry Statistics By Market Size, Revenue, Demographics, Personal Training and Facts. URL: https://www.coolest-gadgets.com/fitness-industry-statistics/ (Last accessed: 22.04.2025).

2. 30+ Interesting Fitness Industry Statistics. URL: https://upmetrics.co/blog/fitness-industry-statistics (Last accessed: 22.04.2025).

3. Health & Fitness App Market 2024: Size, Growth & Trends. URL: https://www.wellnesscreatives.com/fitness-app-market/ ((Last accessed: 23.04.2025).

4. Fitness Tracker Market Size To Hit USD 362.96 Bn By 2034. URL: https://www.precedenceresearch.com/fitness-tracker-market (Last accessed: 23.04.2025).

5. Home Fitness Industry Statistics and Trends for 2025. URL: https://www.ptpioneer.com/statistics/home-fitness-industry-statistics/ (Last accessed: 23.04.2025).

6. Фітнес як стиль життя. Особливості розвитку в Україні (на прикладі мегаполісів). URL: https://buki.com.ua/blogs/fitnes-iak-stil-zittia-osoblivosti-rozvitku-v-ukrayini-na-prikladi-megapolisiv/ (дата звернення: 23.04.2025).

7. In-Person Fitness Rebounds Post-Pandemic, IHRSA Report Finds. URL: https://athletechnews.com/in-person-fitness-rebounds-post-pandemic-ihrsa-global-report/ (Last accessed: 23.04.2025).

8. Balance Training Under Fatigue: A Randomized Controlled Trial on the Effect of Fatigue on Adaptations to Balance Training / M. Keller et al. Journal of Strength and Conditioning Research. 2023. Vol. 38, no. 2.

9. Best Fitness & Wellness Management Software. URL: https://www.mindbodyonline.com/ (Last accessed: 25.04.2025).

10. Welcome to Glofox: The #1 fitness management software. URL: https://www.glofox.com (Last accessed: 25.04.2025).

11. Gym Management Software - GymMaster Member Management. URL: https://www.gymmaster.com/ (Last accessed: 25.04.2025).

12. Dhruvitkumar V. Talati. Python: The alchemist behind AI’s intelligent evolution. International Journal of Science and Research Archive. 2021. Vol. 3, no. 1. P. 235–248.

13. Voice AI-Intelligence Based Voice Assistant / P. Garai et al. 2025 International Conference on Intelligent and Innovative Technologies in Computing, Electrical and Electronics (IITCEE), Bangalore, India, 16–17 January 2025. 2025. P. 1–6.

14. React Apps with Server-Side Rendering: Next.js / H. A Jartarghar et al. Journal of Telecommunication, Electronic and Computer Engineering (JTEC). 2022. Vol. 14, no. 4. P. 25–29.