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

КВАЛІФІКАЦІЙНА РОБОТА

ЗА СТУПЕНЕМ ВИЩОЇ ОСВІТИ «БАКАЛАВР»

РОЗРОБКА ВЕБДОДАТКУ ДЛЯ ГЕНЕРАЦІЇ ТРЕНУВАНЬ ТА РОЗРАХУНКУ ДОБОВОЇ КАЛОРІЙНОСТІ ПРОДУКТІВ ЗА ХАРАКТЕРИСТИКАМИ КОРИСТУВАЧА З ВИКОРИСТАННЯМ REACT

DEVELOPMENT OF A WEB APPLICATION FOR WORKOUT GENERATION AND DAILY CALORIE CALCULATION BASED ON USER CHARACTERISTICS USING REACT

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

Виконав: здобувач вищої освіти
групи ІПЗ-41
Павлик Владислав Михайлович

Керівник:
Гульчук Юрій Миколайович

Кваліфікаційну роботу
допущено до захисту
«10» 06 2025 р.
Гарант освітньої програми:
к.т.н., доцент
Ліщина Наталія Миколаївна

_____

Луцьк – 2025 року

ЛУЦЬКИЙ НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ

Факультет *комп'ютерних та інформаційних технологій*
Кафедра *інженерії програмного забезпечення*
Ступінь вищої освіти *бакалавр*
Галузь знань: *12 «Інформаційні технології»*
Спеціальність: *121 «Інженерія програмного забезпечення»*
Освітня програма: «*Інженерія програмного забезпечення»*

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____
«*20*» *12* 20*24*р.

ЗАВДАННЯ

НА КВАЛІФІКАЦІЙНУ РОБОТУ ЗДОБУВАЧУ ВИЩОЇ ОСВІТИ

Павлику Владиславу Михайловичу
(прізвище, ім'я, по батькові)

1. Тема кваліфікаційної роботи *«Розробка вебдодатку для генерації тренувань та розрахунку добової калорійності продуктів за характеристиками користувача з використанням React»*
Керівник роботи: *асистент Гульчук Ю. М.*
затверджені наказом закладу вищої освіти від *«19» грудня 2024 р.* № *474/01-02*
2. Строк подання здобувачем вищої освіти кваліфікаційної роботи бакалавра *«10» червня 2025 р.*
3. Вихідні дані до роботи: *React, Node.js, Express, MySQL, JWT, Cloudinary, Vite, PHPMyAdmin, методичні вказівки до виконання кваліфікаційної роботи бакалавра.*

4. Зміст розрахунково-пояснювальної записки (перелік питань, що потрібно розробити): *аналіз предметної області, опис функціональних та нефункціональних вимог, вибір інструментів, створення та тестування REST API, реалізація автентифікації користувачів, тестування і налагодження додатку, розробка моделей бази даних, захист даних.*

5. Перелік графічного матеріалу: *7 рисунків, 1 таблиця.*

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис	
		завдання видав	завдання прийняв
Аналіз предметної області	*Гульчук Ю. М.*		
Специфікація вимог до розробленої системи	*Гульчук Ю. М.*		
Розробка об'єкта проектування	*Гульчук Ю. М.*		
Нормоконтроль	*Вознюк А. В.*		
Гарант ОП	*Ліщина Н. М.*		
Показник запозичень тексту	3,02 %		
Академічна доброчесність	*Гульчук Ю. М.*		

7. Дата видачі завдання *«20» грудня 2024 р.*

№	Назва етапів кваліфікаційної роботи бакалавра	Строк виконання етапів роботи	Примітка
1	Огляд літературних джерел по темі кваліфікаційної роботи бакалавра	до 04.02.2025 р.	вик.
2	Аналіз проблеми розробки та впровадження об'єкту проектування	до 01.03.2025 р.	вик.
3	Обґрунтування вибору шляхів, технологій і засобів вирішення поставленого завдання	до 15.03.2025 р.	вик.
4	Розробка функціонально-структурної схеми роботи об'єкта проектування та проектування бази даних	до 29.03.2025 р.	вик.
5	Практична реалізація об'єкта проектування та розробка бази даних	до 26.04.2025 р.	вик.
6	Тестування та налагодження об'єкта проектування	до 03.05.2025 р.	вик.
7	Здача чистового варіанту кваліфікаційної роботи бакалавра на кафедру	до 10.06.2025 р.	вик.

Здобувач вищої освіти _____ Владислав ПАВЛИК

Керівник кваліфікаційної роботи _____ Юрій ГУЛЬЧУК

АНОТАЦІЯ

Павлик В. М. Розробка вебдодатку для генерації тренувань та розрахунку добової калорійності продуктів за характеристиками користувача з використанням React. Рукопис. Кваліфікаційна робота бакалавра ОП «Інженерія програмного забезпечення» спеціальності «Інженерія програмного забезпечення». Луцький національний технічний університет. Луцьк, 2025.

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

У першому розділі здійснено аналіз сучасного стану проблеми створення інформаційних систем у галузі фітнесу, а також сформульовано завдання на кваліфікаційну роботу. У другому розділі визначено вимоги до розроблюваної системи, описано вибір технологій та інструментів для реалізації вебдодатку. У третьому розділі розглянуто процес розробки вебдодатку, реалізацію основних функцій, тестування та налагодження, а також захист інформаційної системи. У висновках підсумовано результати роботи, її практичну значущість та рекомендації щодо подальшого розвитку.

Ключові слова: вебдодаток, фітнес, калорійність, тренування, React, інформаційна система, безпека.

ABSTRACT

Pavlyk V. Development of a Web Application for Workout Generation and Daily Calorie Calculation Based on User Characteristics Using React. Manuscript. Qualification work of the bachelor's program "Software engineering" in the specialty "Software engineering". Lutsk National Technical University. Lutsk, 2025.

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

The first chapter presents the current state of the problem in the field of fitness applications and formulates the objectives. The second chapter determines the requirements for the developed system, the choice of technologies and tools for creating the web application. The third chapter describes the development process of the web application, its main functions, testing and debugging, as well as the system's security. The conclusions summarize the results of the work, its practical significance, and recommendations for further development.

Keywords: web application, fitness, calorie calculation, workout, React, information system, security.

ЗМІСТ

ВСТУП

Актуальність теми. У сучасному світі спостерігається стрімке зростання інтересу до здорового способу життя, фізичної активності та раціонального харчування. Паралельно з цим, розвиток інформаційних технологій створює умови для ефективного застосування цифрових інструментів у сфері фітнесу та нутриціології. Вебдодатки, що дозволяють користувачам формувати індивідуальні тренувальні програми та розраховувати добову калорійність харчування, стають дедалі популярнішими завдяки зручності, доступності та персоналізації. Проте значна частина існуючих сервісів є або надмірно складною для звичайного користувача, або орієнтованою лише на одну функцію – без комплексного підходу. Саме тому актуальним є створення зручного, інтуїтивно зрозумілого вебдодатку, який би об'єднував у собі можливість генерації тренувань та розрахунку калорійності продуктів на основі персональних даних користувача.

Відповідно, метою цієї кваліфікаційної роботи є розробка функціонального вебдодатку для генерації індивідуальних тренувальних планів та автоматичного розрахунку добової калорійності продуктів відповідно до характеристик користувача, з використанням технологій React, REST API та бази даних.

Об'єктом кваліфікаційної роботи є процес автоматизованого формування персоналізованих фітнес-програм та розрахунку калорійності у вебсередовищі.

Предметом кваліфікаційної роботи є програмна архітектура, функціональність та інтерфейс вебдодатку, що забезпечує взаємодію користувача з системою для досягнення цілей у сфері здорового способу життя.

Для досягнення поставленої мети були поставлені наступні завдання:

1) аналіз існуючих рішень у сфері фітнес-трекінгу та нутриціології;

2) визначення вимог до функціоналу, інтерфейсу та структури системи, розробка моделі програмного забезпечення, включаючи сценарії використання;

3) вибір відповідних технологій для реалізації (React, MySQL, API, HTML/CSS);

4) розробка та реалізація вебдодатку. створення алгоритмів генерації тренувань та розрахунку калорійності;

5) тестування, налагодження та оптимізація роботи системи;

6) проєктування та розробка бази даних;

7) розробка механізмів захисту.

Галузь застосування охоплює сферу інформаційних технологій у фітнесі, спортивній медицині, нутриціології, персональному коучингу, а також онлайн-освіті щодо ЗСЖ.

Робота має взаємозв’язок з іншими дослідженнями, що стосуються веброзробки, формування персоналізованих рекомендацій, створення user-friendly інтерфейсів, впровадження сучасних інструментів для контролю життєдіяльності людини. Вона інтегрує підходи з інженерії програмного забезпечення, UI/UX-дизайну, систем підтримки прийняття рішень та баз даних

РОЗДІЛ 1
АНАЛІЗ СФЕРИ ФІТНЕС-ДОДАТКІВ І ПОСТАНОВКА ЗАВДАНЬ НА КВАЛІФІКАЦІЙНУ РОБОТУ

1.1 Аналіз сучасного стану проблеми

У зв’язку зі зростаючим інтересом до здорового способу життя, фізичної активності та правильного харчування, значну популярність здобули мобільні та вебзастосунки, які дозволяють користувачам контролювати свою вагу, слідкувати за кількістю спожитих калорій, формувати індивідуальні плани тренувань та вести щоденник харчування. За останні роки розроблено велику кількість подібних програмних рішень, проте більшість із них зосереджені на одному функціональному аспекті – або тренуваннях, або харчуванні, або просто фіксації параметрів тіла. Комплексні рішення, особливо у вигляді доступних вебдодатків, зустрічаються рідше, хоча попит на них зростає.

Упродовж останніх років спостерігається суттєве зростання інтересу до досліджень, присвячених впровадженню цифрових рішень у сферу фізичної культури та здорового способу життя. Наукові праці аналізують не лише технічні аспекти побудови фітнес-додатків, а й поведінкові, мотиваційні, соціальні та економічні чинники їхнього використання.

Згідно з систематичним оглядом S. Angosto та співавторів [1], основними факторами, що впливають на намір користувачів застосовувати фітнес- і фізкультурні додатки, є зручність, інтуїтивність інтерфейсу, можливість персоналізації контенту та соціальна взаємодія. Автори зазначають, що застосування моделей TAM та UTAUT є найбільш поширеним у контексті оцінювання користувацької поведінки щодо подібного програмного забезпечення.

H. Ferreira Barbosa та співавтори [2], використовуючи модель UTAUT2, дослідили вплив мобільних додатків фітнес-центрів на задоволеність клієнтів. Дослідження виявило, що такі характеристики, як інформативність, доступ до персоналізованих тренувань і простота використання, мають вирішальне

значення для формування лояльності користувача. Це підтверджує важливість гнучкого UX/UI-дизайну у розробці відповідного ПЗ.

У роботі E. Whelan та T. Clohessy [3] представлено модель подвійної дії соціального впливу, яка демонструє, що соціальний компонент фітнес-додатків може як підсилювати мотивацію, так і спричиняти залежність або погіршення психологічного стану. Отже, інтеграція соціальних функцій повинна здійснюватися обережно, з урахуванням індивідуальних особливостей користувачів.

Дослідження С. В. Бондаренка і М. Є. Бондаренка [4] акцентує увагу на можливості застосування штучного інтелекту для формування індивідуальних тренувальних планів. У статті описано структуру мобільного додатку, що використовує алгоритми машинного навчання для адаптації навантаження під змінні фізіологічні характеристики користувача.

Кваліфікаційна робота А. С. Кореневої [5] містить практичний опис побудови мобільного додатку супроводження фітнес-тренувань. У роботі наведено програмну реалізацію інтерфейсів, алгоритмів обліку фізичних показників та обґрунтовано архітектурні рішення з урахуванням безпеки персональних даних.

У публікації Т. Мошенської та співавторів [6] розглядається широке застосування онлайн-платформ та фітнес-додатків у процесі формування здорового способу життя. Авторки зазначають, що в умовах постпандемічного періоду цифрові фітнес-рішення стали основним елементом не лише індивідуальних, а й освітньо-оздоровчих програм.

Відповідно, аналіз наукових джерел засвідчує високий інтерес до інноваційних підходів у сфері фітнес-додатків, що поєднують технології штучного інтелекту, зручні мобільні й вебінтерфейси та адаптивні механізми персоналізації. Це підтверджує актуальність теми даної кваліфікаційної роботи та обґрунтовує вибір напрямів розробки.

Під час патентного пошуку у відкритих джерелах (зокрема Google Patents) було виявлено кілька технічних рішень, що стосуються автоматичного добору

фізичних навантажень або обчислення калорійності. Проте ці системи, як правило, реалізовані у вигляді патентів для вузькоспеціалізованих приладів, фітнес-браслетів або закритих мобільних рішень, доступ до яких обмежено або комерціалізовано. Це створює нішу для розробки відкритого вебдодатку з базовою функціональністю, орієнтованого на широке коло користувачів.

До популярних аналогів, що реалізують подібну функціональність, можна віднести такі рішення:

1) MyFitnessPal – потужний сервіс для контролю харчування з великою базою продуктів, однак має складний інтерфейс для новачків та обмежену підтримку персональних тренувань [7];

2) Freeletics – додаток, що спеціалізується на адаптивних тренуваннях, проте без детального підрахунку калорійності харчування [8];

3) Lifesum – комбінований сервіс з акцентом на дієти, але з високим рівнем платної підписки та залежністю від мобільної версії [9].

Основні недоліки існуючих систем:

1) обмеження безкоштовного функціоналу (часто основні можливості доступні лише в преміум-версіях);

2) складність інтерфейсу або надмірне інформаційне навантаження;

3) відсутність вебверсій або погана адаптивність;

4) низька точність при підборі тренувань без урахування мети (набір маси, зниження ваги тощо).

Доцільність створення нової системи полягає в об’єднанні переваг розглянутих рішень і подоланні зазначених недоліків. Запропонований вебдодаток буде реалізований із використанням сучасного інструментарію (React, REST API, MySQL), орієнтований на персоналізацію, мінімалізм у дизайні, відкритий доступ та розширювану архітектуру. Він забезпечить комплексний підхід – об’єднуючи функції формування тренувальних планів, розрахунку калорій та обліку фізичних параметрів користувача.

1.2 Постановка завдання на кваліфікаційну роботу бакалавра

Метою кваліфікаційної роботи є створення функціонального вебдодатку для генерації персоналізованих тренувальних програм та розрахунку добової калорійності продуктів відповідно до індивідуальних характеристик користувача з використанням сучасних вебтехнологій, зокрема React.

Для досягнення цієї мети були сформульовані такі завдання, що відповідають логічній структурі роботи:

1) виконати аналіз сучасного стану проблеми, здійснити огляд літературних джерел, патентів та існуючих веб- і мобільних додатків у сфері фітнесу та нутриціології, виявити їхні переваги та недоліки, обґрунтувати доцільність створення нового веборієнтованого рішення;

2) сформулювати вимоги до програмного забезпечення, включаючи функціональні, нефункціональні, інтерфейсні та інформаційні вимоги, а також розробити UML-моделі системи, сценарії використання та структуру БД;

3) обґрунтувати вибір інструментів розробки, методів реалізації та алгоритмів, необхідних для створення вебдодатку, описати підхід до побудови логіки обробки даних користувача, генерації тренувань і розрахунку калорійності;

4) реалізувати вебдодаток з використанням React, створити клієнтську частину та забезпечити обробку дій користувача, збереження даних та інтеграцію з базою даних;

5) виконати тестування системи, забезпечити виявлення та виправлення помилок, налагодити важливу функціональність, визначити системні вимоги для стабільної роботи додатку;

6) розробити структуру бази даних, налаштувати її для зберігання інформації про користувачів, продукти, фізичні характеристики, історію тренувань, результати розрахунків та запитів;

7) забезпечити захист інформації, реалізувати механізми реєстрації, авторизації, шифрування паролів, обмеження доступу до конфіденційних даних.

Ці завдання охоплюють усі етапи життєвого циклу розробки програмного забезпечення – від аналізу предметної області до реалізації, тестування, захисту даних і супровідної документації, що підтверджує комплексний характер виконаної кваліфікаційної роботи.

Висновки до розділу 1

У першому розділі було проведено ґрунтовний аналіз сучасного стану проблеми створення інформаційних систем у галузі фітнесу та здорового харчування. Розглянуто наукові джерела, патентну інформацію та практичні реалізації у вигляді популярних мобільних і вебдодатків. У результаті встановлено, що більшість існуючих рішень мають обмежений функціонал, не передбачають комплексного підходу до поєднання фізичної активності та нутриціології, або є орієнтованими переважно на мобільні платформи без повноцінної вебпідтримки.

Обґрунтовано доцільність створення нового веборієнтованого програмного продукту, який дозволить користувачеві на основі його індивідуальних параметрів (вік, стать, фізична активність, мета тренування) автоматично формувати тренувальні програми і розраховувати добову норму калорій. Визначено актуальність розробки саме у форматі вебдодатку з використанням сучасного технологічного стеку, зокрема React, що забезпечить масштабованість, доступність і зручність користування.

Сформульовано чіткі завдання, що охоплюють усі етапи життєвого циклу створення програмного забезпечення: від аналізу вимог і проектування до реалізації, тестування, захисту даних та розробки документації. Встановлено міжпредметні зв’язки з галузями UI/UX-дизайну, моделювання програмних систем, безпеки інформації та обчислювальної нутриціології.

Отже, результати першого розділу підтверджують наукову та практичну значущість проєкту, формують теоретичне підґрунтя для подальшої розробки вебдодатку, що буде представлено в наступних розділах роботи.

РОЗДІЛ 2
СПЕЦИФІКАЦІЯ ВИМОГ ДО РОЗРОБЛЮВАНОЇ СИСТЕМИ

2.1 Аналіз, визначення вимог до розроблюваного програмного забезпечення та проектування

Для моделювання програмної системи обрано об’єктно-орієнтований підхід, який є доцільним для побудови масштабованих вебдодатків з розділенням відповідальностей за логіку, інтерфейс та обробку даних. При цьому застосовується каскадна модель життєвого циклу розробки ПЗ (Waterfall Model), що дозволяє послідовно реалізувати етапи аналізу, проєктування, реалізації, тестування та впровадження.

Для виявлення функціональних та нефункціональних вимог до системи було здійснено комплексний підхід, що включав кілька важливих етапів. Насамперед проведено детальний аналіз існуючих рішень-конкурентів, що дозволило визначити типові функції та виявити прогалини, які слід усунути у розроблюваному продукті. Наступним кроком стало формулювання user stories, які відображають реальні сценарії використання системи, зокрема ситуації, коли користувач прагне створити індивідуальне тренування або розрахувати добову норму калорій. Додатково було організовано опитування представників цільової аудиторії під час тестування прототипів інтерфейсу, що дало змогу встановити найбільш затребувані функціональні елементи. Окрему увагу приділено аналізу архітектурних вимог, зокрема питань безпеки, швидкодії, надійності та масштабованості системи в умовах потенційного зростання кількості користувачів.

У процесі проєктування системи були використані UML-нотації. Use Case діаграма (рис. 2.1) – для візуалізації основних сценаріїв взаємодії користувача з системою. Ця діаграма демонструє всі основні дії, які може виконувати користувач системи. Вона відображає загальні сценарії використання: від реєстрації до перегляду результатів.

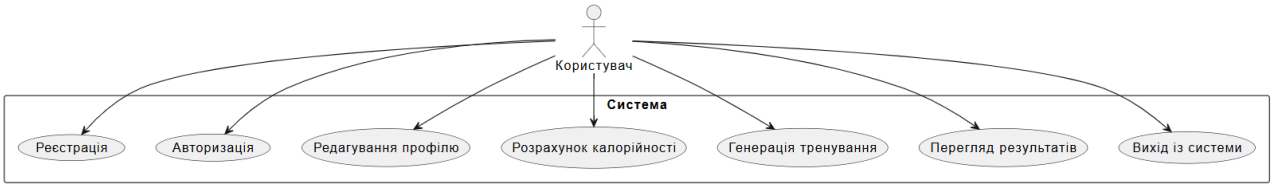

Рисунок 2.1 – Діаграма варіантів використання

Class diagram (рис. 2.2) призначення для моделювання основних сутностей додатку (користувач, тренування, продукт тощо).

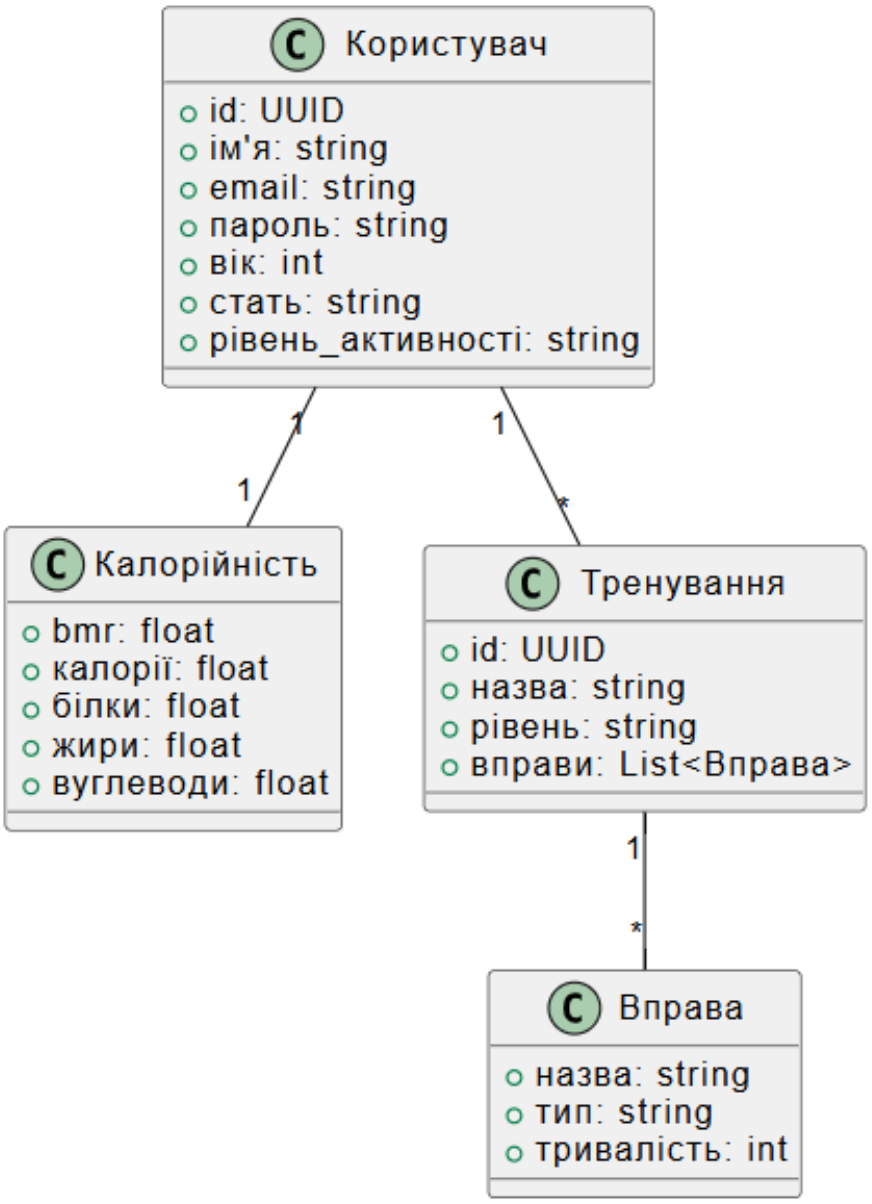

Рисунок 2.2 – Діаграма класів

Ця діаграма описує структуру даних: користувач має персональні показники, які використовуються для розрахунків; кожен користувач має кілька тренувань, які складаються з набору вправ.

Activity diagram (рис. 2.3) відображає покроковий алгоритм, який виконується при розрахунку калорійності на основі введених даних користувача.

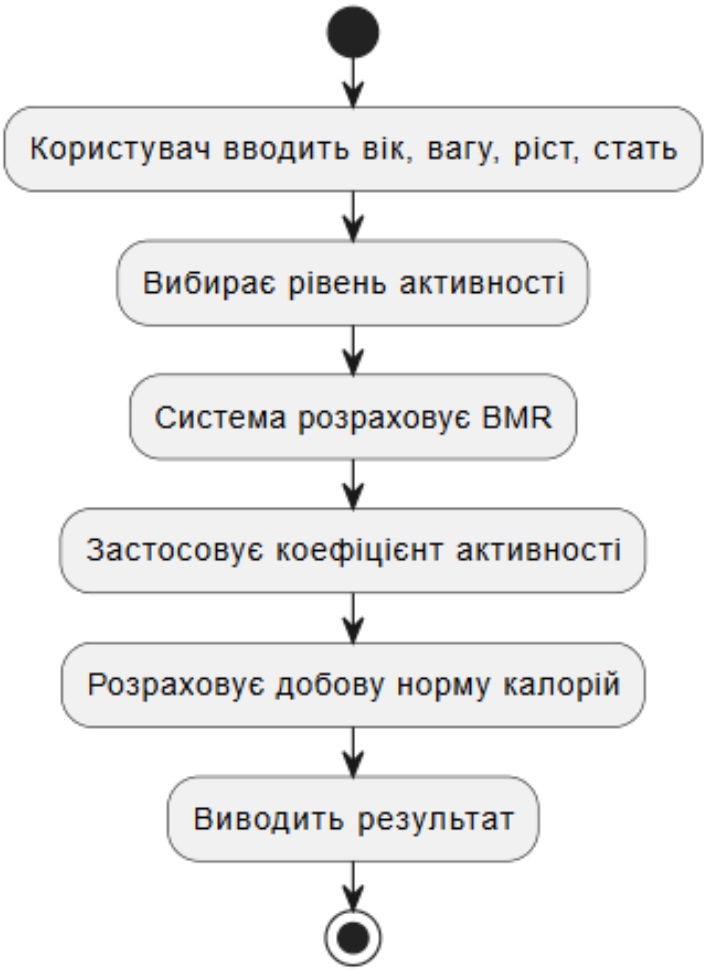

Рисунок 2.3 – Діаграма активностей

Sequence diagram (рис. 2.4) призначена для показу взаємодії клієнта з сервером (REST API). Ця діаграма демонструє послідовність викликів у процесі генерації тренування: від дій користувача до обробки логіки і взаємодії з базою даних.

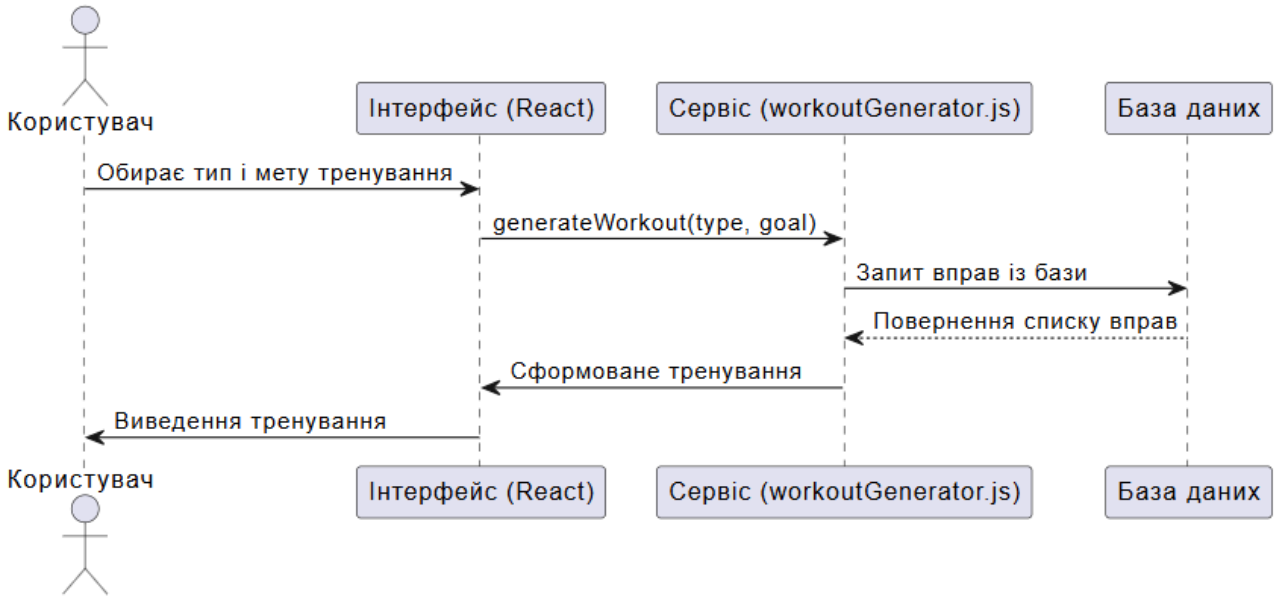

Рисунок 2.4 – Діаграма послідовності

Цілі системи:

1) забезпечення авторизації/реєстрації користувача;

2) збір індивідуальних параметрів користувача;

3) розрахунок добової норми калорій та нутрієнтів;

4) генерація плану тренування відповідно до мети;

5) збереження результатів і параметрів у базі даних;

6) надання інтуїтивного вебінтерфейсу.

Основні функціональні вимоги:

1) створення облікового запису;

2) вхід у систему з перевіркою даних;

3) збір вхідних параметрів (вік, вага, ріст, активність);

4) виведення добової калорійності;

5) формування набору вправ на основі обраної мети;

6) можливість перегляду, редагування профілю;

7) збереження даних на сервері (REST API);

8) обмеження доступу до персональних сторінок (Protected Routes).

Нефункціональні вимоги:

1) підтримка сучасних браузерів;

2) мобільна адаптивність;

3) реакція на дії користувача протягом < 500 мс;

4) захист даних авторизації через JWT;

5) надійне з’єднання з БД через HTTPS.

Взаємодію користувача із системою та відповідні реакції інтерфейсу наведено у таблиці 2.1.

Таблиця 2.1 – Реакції інтерфейсу «Подія – Відгук»

Подія користувача	Відгук системи
Натискання «Реєстрація»	Відкривається форма, перевіряються поля
Натискання «Розрахуват»	Виконується обчислення калорійності
Вибір типу тренування	Генерується відповідний список вправ
Вихід із системи	Сесія закривається, перенаправлення на логін

Інтерфейс користувача побудований відповідно до принципів зручності, адаптивності та доступності:

1) уніфіковане меню навігації;

2) прості форми з підказками та валідацією;

3) відповідність мобільним і десктопним екранам (адаптивний дизайн);

4) форма тренування у вигляді карток з піктограмами;

5) кольорова схема для легшого сприйняття.

Інформаційні потоки в розроблюваному вебдодатку охоплюють чотири основні етапи: вхід, обробку, збереження та виведення даних. На вході система приймає персональні параметри користувача, зокрема вік, стать, вагу та рівень фізичної активності. Ці дані проходять етап обробки, в межах якого виконується розрахунок добової калорійності та підбір відповідних фізичних вправ відповідно до обраної мети. Результати обробки зберігаються у базі даних, що містить профіль користувача, його індивідуальні параметри та історію згенерованих тренувань. На етапі виведення результати роботи системи подаються користувачу через інтерфейс у вигляді структурованих текстових блоків, карток із вправами, числових значень та рекомендацій, що забезпечує зручну взаємодію з додатком.

2.2 Вибір засобів, методів і алгоритмів вирішення поставленого завдання

У процесі розробки програмного забезпечення важливу роль відіграє правильний вибір технологій, інструментів та алгоритмів, які забезпечать стабільну, масштабовану та зручну у використанні інформаційну систему. З огляду на предметну область та функціональні вимоги, було здійснено аналіз доступних засобів розробки, порівняно їхні технічні характеристики, зручність інтеграції, підтримку спільноти, доступність документації, продуктивність та масштабованість.

Для реалізації клієнтської частини вебдодатку було обрано React – одну з найпопулярніших JavaScript-бібліотек для створення динамічних та масштабованих інтерфейсів користувача. React розроблено компанією Meta, і вона широко використовується в комерційній та академічній розробці. Її основною особливістю є компонентно-орієнтований підхід, який дозволяє розділити інтерфейс на незалежні багаторазові елементи [10]. Це значно полегшує підтримку та масштабування проєкту. Крім того, React використовує концепцію віртуального DOM (Virtual DOM), що забезпечує ефективне оновлення сторінки при зміні стану додатку без повного перезавантаження вмісту.

Завдяки хукам (зокрема useState, useEffect, useContext), React надає зручний механізм для керування станом та побудови реактивної логіки додатку [11]. У межах розроблюваного проєкту використання React дозволило реалізувати інтерфейс із високим рівнем інтерактивності, адаптивною поведінкою та чіткою структурою. Серед супутніх рішень, що інтегруються з React, використовуються React Router для маршрутизації між сторінками, Tailwind CSS для стилізації компонентів та Context API для зберігання глобального стану.

Для організації процесу розробки клієнтської частини обрано Vite – сучасний інструмент збирання проєктів, який став популярною альтернативою

Webpack завдяки своїй високій продуктивності. Vite забезпечує надзвичайно швидке завантаження та компіляцію завдяки використанню нативного модуля ES та вбудованого сервера для локальної розробки [12]. Цей інструмент ідеально підходить для проєктів на базі React, оскільки він не лише скорочує час перезавантаження під час зміни коду (завдяки функції hot module replacement), але й забезпечує компактне збирання у production-режимі. Vite має просту конфігурацію, високу сумісність з сучасними браузерами та підтримує TypeScript, JSX, CSS-модулі й PostCSS із коробки.

У межах розробки проєкту Vite значно спростив процес тестування інтерфейсу, дозволив без затримок вносити зміни в компоненти, а також зменшив розмір кінцевого продукту за рахунок ефективного дерева залежностей та оптимізації імпортів.

На серверному боці застосовується зв’язка Node.js + Express, що утворює гнучкий та продуктивний стек для реалізації REST API. Node.js є середовищем виконання JavaScript-проєктів на стороні сервера, побудованим на рушії V8 від Google, що дозволяє виконувати JavaScript-код поза межами браузера. Його асинхронна, неблокуюча модель обробки подій особливо ефективна в контексті високонавантажених вебзастосунків [13].

Express.js, у свою чергу, є мінімалістичним фреймворком, що спрощує створення маршрутів, middleware-функцій, обробку запитів та налаштування серверної логіки. Разом ці технології забезпечують швидку та легку розробку API-інтерфейсів, які обслуговують запити з клієнтської частини додатку, обробляють дані, взаємодіють із базою даних, виконують перевірку автентифікації, авторизацію та інші бекенд-завдання [14]. Для даного вебдодатку реалізовано набір REST-ендпоінтів, які приймають запити з React-додатку, зберігають дані у базі, обробляють розрахунки калорійності, а також повертають згенеровані тренування.

Як система керування базами даних для даного проєкту використовується MySQL – одна з найбільш поширених у світі реляційних СУБД [15]. Вона є відкритим програмним продуктом і пропонує високу продуктивність,

надійність, доступність і підтримку структурованого запиту SQL. MySQL ідеально підходить для застосування у вебдодатках, де важливе значення має цілісність і зв’язаність даних, як у випадку з таблицями користувачів, тренувань, продуктів і результатів розрахунків. MySQL підтримує складні запити, зовнішні ключі, транзакції, індекси, що робить її ефективним інструментом для реалізації бізнес-логіки на рівні даних.

Для зручності адміністрування та візуального управління базами даних у рамках проєкту використано phpMyAdmin – вебінтерфейс, що дозволяє виконувати SQL-запити, створювати таблиці, переглядати та редагувати записи без потреби писати запити вручну. Це значно спрощує етап розробки структури БД, тестування та відлагодження SQL-операцій, а також дає змогу швидко виявляти помилки в даних або конфігурації таблиць.

У якості алгоритму для розрахунку добової норми калорій було обрано формулу Mifflin-St Jeor, яка є однією з найточніших серед формул оцінки базального метаболізму, рекомендованою для використання в клінічній практиці. Цей алгоритм використовує такі параметри, як вага, зріст, вік, стать і рівень фізичної активності, для визначення індивідуальної потреби в калоріях. На основі отриманого значення формується добовий план харчування та перелік фізичних вправ.

Алгоритм генерації тренувань реалізовано на основі категоризації вправ та rule-based логіки. Відповідно до обраної мети (наприклад, «схуднення», «набір маси», «підтримка форми»), система автоматично добирає вправи з відповідної категорії. Генерація списку відбувається або випадковим чином, або за заздалегідь заданими шаблонами, що дозволяє користувачу отримати персоналізований і водночас збалансований план.

Також на рисунку 2.5 представлено алгоритм генерації тренувань у вигляді блок-схеми. Вона демонструє покрокову логіку генерації тренувального плану на основі цілі користувача. Враховується або використання шаблону, або випадковий вибір вправ зі списку за категорією.

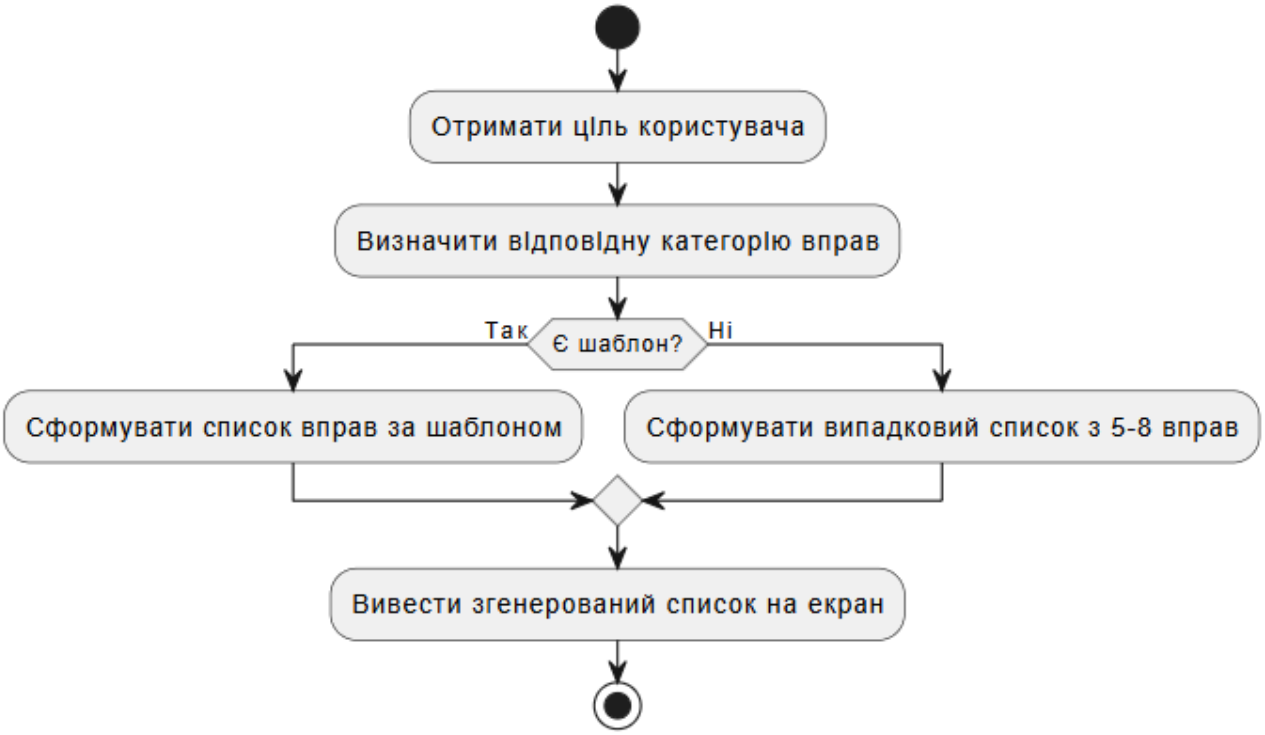

Рисунок 2.5 – Блок-схема алгоритму генерації тренувань

Вибрані інструменти відповідають сучасним вимогам до веброзробки, забезпечують масштабованість, зручність у підтримці та розвиток функціоналу. React у поєднанні з REST API і реляційною БД дозволяє побудувати гнучку, швидкодіючу систему з високим рівнем взаємодії з користувачем. Обрані алгоритми для розрахунку та генерації є ефективними з погляду швидкодії та логічної простоти реалізації, що підтверджує доцільність їх використання у межах даного проєкту.

Висновки до розділу 2

У другому розділі було проведено детальний аналіз вимог до розроблюваної системи, визначено її цілі, функціональні та нефункціональні характеристики, а також побудовано моделі програмного забезпечення за допомогою UML-нотацій. Обґрунтовано вибір об’єктно-орієнтованої моделі проєктування, яка забезпечує гнучкість, масштабованість і підтримуваність

вебдодатку. Виявлення вимог до системи здійснювалося шляхом аналізу аналогів, формулювання user stories, опитування потенційних користувачів та врахування архітектурних обмежень.

На основі сформульованих сценаріїв використання та специфікації вимог побудовано діаграми варіантів використання, класів, активності та послідовності, що візуалізують логіку взаємодії користувача з додатком, а також структуру основних сутностей. Особливу увагу приділено опису інформаційних потоків, способам обробки введених даних, їх зберіганню у базі даних та виведенню результатів у зручному форматі.

В межах вибору інструментів розробки обґрунтовано використання React для клієнтської частини, MySQL для зберігання даних, а також супутніх бібліотек і технологій, що забезпечують безпечну автентифікацію, обмін даними та підтримку сучасного UX/UI. Описано алгоритми, що лежать в основі системи: зокрема, математичну модель для розрахунку добової калорійності (формула Mifflin-St Jeor) та rule-based алгоритм для генерації індивідуальних тренувань.

Загалом результати другого розділу забезпечили комплексне проектне підґрунтя для переходу до етапу безпосередньої реалізації вебдодатку, який розглядається у наступному розділі.

РОЗДІЛ 3
РОЗРОБКА ВЕБДОДАТКУ ДЛЯ ГЕНЕРАЦІЇ ТРЕНУВАНЬ ТА РОЗРАХУНКУ КАЛОРІЙНОСТІ

3.1 Практична реалізація об’єкта проєктування

У рамках дипломного проєкту було розроблено повноцінний вебдодаток, що складається з клієнтської (frontend) та серверної (backend) частин. Основна мета системи – надати користувачам зручний інтерфейс для генерації тренувань на основі обраного рівня фізичної підготовки, відстеження активності, підрахунку витрачених калорій і формування статистики прогресу. Система має масштабовану архітектуру та використовує сучасний стек технологій: React, Node.js, Express, MySQL, Cloudinary, JWT.

Для реалізації інтерфейсу користувача використано React, оскільки ця бібліотека підтримує компонентну структуру, реактивні оновлення за допомогою Virtual DOM, а також надає гнучку систему керування станом через хуки useState, useEffect та useContext. Як середовище розробки обрано Vite за рахунок його швидкого запуску, миттєвого перезавантаження компонентів та оптимізації готової збірки.

Основний компонент Workouts.jsx відповідає за генерацію тренувань, відображення статистики та взаємодію з бекендом. Нижче наведено приклад керування таймером під час виконання тренування (ліст. 3.1).

Лістинг 3.1 – Керування таймером під час тренування

```
useEffect(() => {
    let interval = null;
    if (isTimerRunning) {
        interval = setInterval(() => {
            setWorkoutTimer(prev => prev + 1);
        }, 1000);
    } else if (!isTimerRunning && workoutTimer !== 0) {
        clearInterval(interval);
    }
    return () => clearInterval(interval);
}, [isTimerRunning, workoutTimer]);
```

кінець лістингу 3.1

Цей код демонструє створення таймера з автоматичним оновленням кожну секунду, що дозволяє користувачеві бачити актуальний час виконання вправи в реальному часі.

Для генерації тренувань на основі фітнес-рівня користувача викликається відповідна функція (ліст. 3.2).

Лістинг 3.2 – Функція генерації тренувань

```
const handleGenerateWorkouts = async (type = 'level') => {
    let newWorkouts = type === 'random'
        ? await generateRandomWorkouts(6)
        : await generateWorkoutsForLevel(preferences.fitnessLevel);

    if (newWorkouts.length > 0) {
        toast.success(`Згенеровано ${newWorkouts.length} тренувань`);
    }
};
```

кінець лістингу 3.2

Функція звертається до стора, який реалізовано через Zustand, та викликає відповідні API-запити до серверної частини.

Візуалізація тренувальної картки здійснюється через компонент WorkoutCard, а калькуляція витрачених калорій базується на типі вправи, вазі користувача та інтенсивності (ліст. 3.3)

Лістинг 3.3 – Калькуляція витрачених калорій

```
const getEstimatedCalories = (workout) => {
    const userWeight = getUserWeight() || 70;
    const duration = activeWorkoutIndex ===
currentWorkouts.indexOf(workout)
        ? Math.floor(workoutTimer / 60)
        : workout.duration;
    const base = workout.type.toLowerCase() === 'кардіо' ? 10 :
workout.type === 'hiit' ? 12 : 8;
    const multiplier = workout.intensity.includes('Висока') ? 1.3 :
workout.intensity.includes('Середня') ? 1.0 : 0.8;

    return Math.round(duration * base * multiplier * (userWeight / 70));
};
```

кінець лістингу 3.3

Сервер реалізовано на Node.js з використанням Express.js. Для захисту маршрутів застосовано JWT-токени, які перевіряються у middleware auth.middleware.js. Приклад коду захисту наведено в лістингу 3.4.

Лістинг 3.4 – Код захисту

```
const protectRoute = async (req, res, next) => {
    const token = req.cookies.jwt;
    if (!token) return res.status(401).json({ message: "Токен не надано"
});

    const decoded = jwt.verify(token, process.env.JWT_SECRET);
    const user = await executeQuery("SELECT * FROM users WHERE id = ?",
[decoded.userId]);

    if (!user.length) return res.status(404).json({ message: "Користувача
не знайдено" });

    req.user = user[0];
    next();
};
```

кінець лістингу 3.4

Усі запити до API здійснюються через роутер, приклад якого подано в лістингу 3.5.

Лістинг 3.5 – Приклад роутера

```
router.post("/signup", authController.signup);
router.post("/login", authController.login);
router.get("/check", authMiddleware.protectRoute,
authController.checkAuth);
router.get("/stats", authMiddleware.protectRoute,
authController.getUserStats);
```

кінець лістингу 3.5

Для збереження зображень профілю та інших файлів використано Cloudinary API (ліст. 3.6)

Лістинг 3.6 – Використання Cloudinary

```
cloudinary.config({
```

```
    cloud_name: process.env.CLOUDINARY_CLOUD_NAME,
    api_key: process.env.CLOUDINARY_API_KEY,
    api_secret: process.env.CLOUDINARY_API_SECRET
});
```

кінець лістингу 3.6

Дані зберігаються у реляційній СУБД MySQL, структура якої включає таблиці users, workouts, preferences, stats. Запити виконуються через модуль executeQuery, який інкапсулює підключення до БД.

Стосовно інтерфейсу системи, на рисунку 3.1 представлено сторінку застосунку, яка містить форму входу. Ліворуч представлено короткий опис основних функцій додатку: трекінг харчування, персональні тренування, аналітика прогресу та турбота про здоров'я. Праворуч розміщена форма авторизації, де користувач вводить електронну адресу та пароль для входу до свого облікового запису.

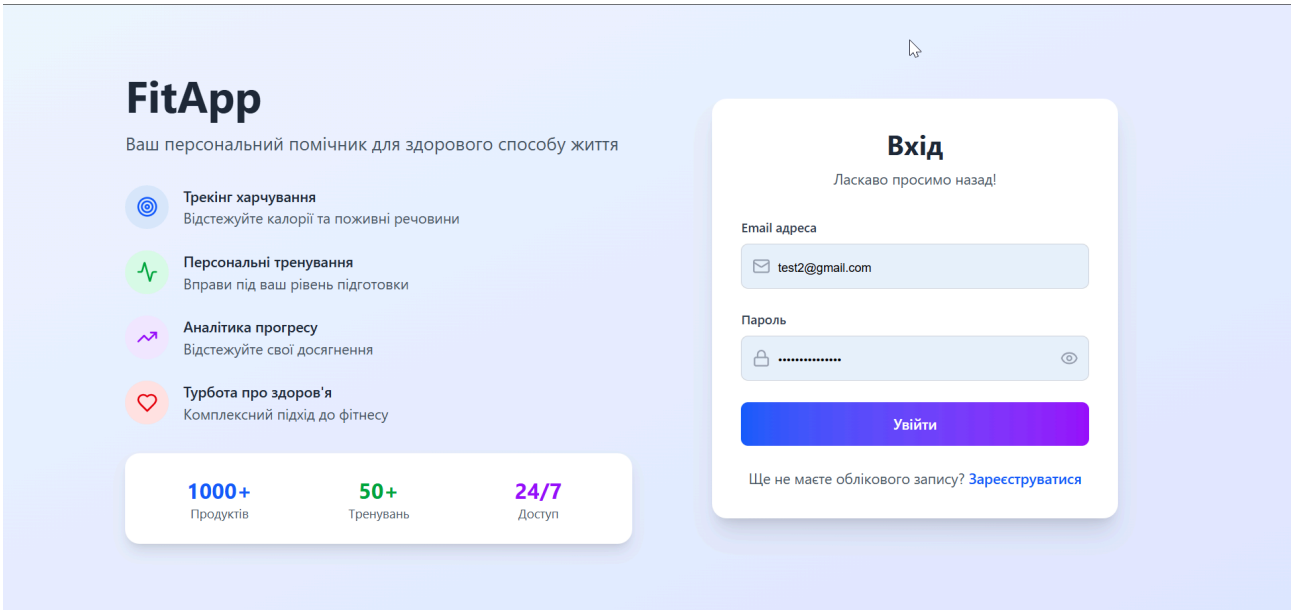

Рисунок 3.1 – Сторінка з формою входу

В свою чергу, на рисунку 3.2 показано головну сторінку після входу користувача до системи. У верхній частині видно привітання з ім'ям користувача. В центрі інтерфейсу – блоки з інформацією про поточний прогрес (спожиті калорії та тривалість тренувань), нагадування про необхідність

заповнення профілю, а також мотиваційне повідомлення, яке підштовхує користувача до дії. Інтерфейс оформлений у стилі, що сприяє зручному навігаційному досвіду.

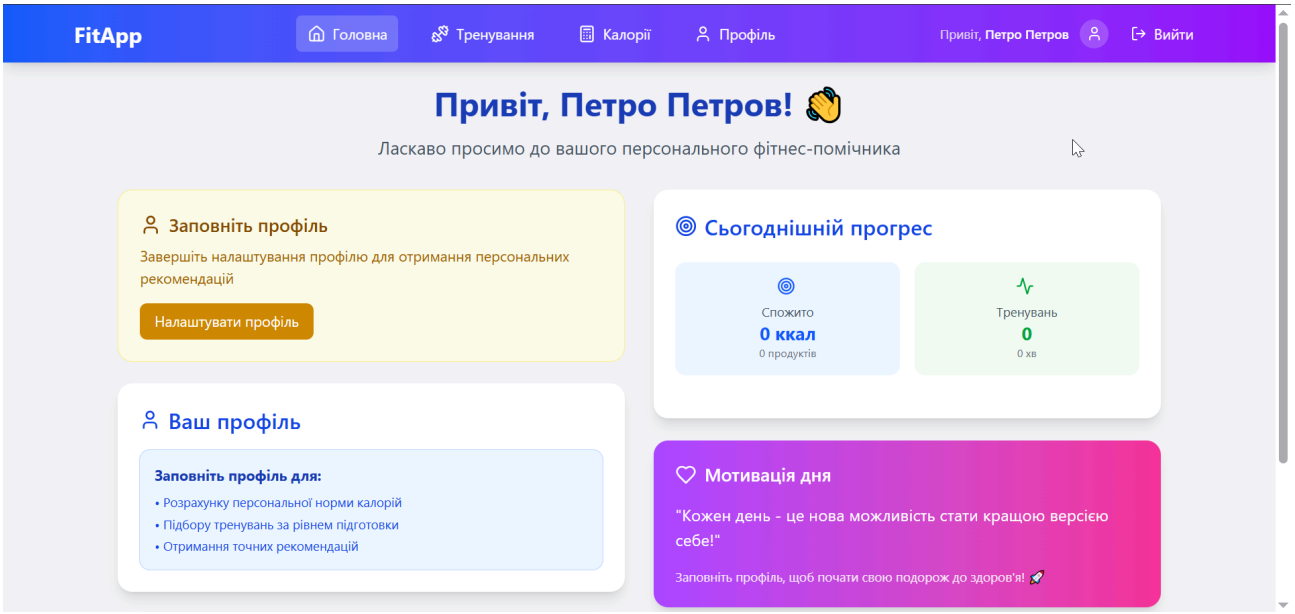

Рисунок 3.2 – Головна сторінка

Загалом, реалізація об’єкта проєктування доводить володіння автором сучасними мовами програмування, здатність працювати з базами даних, вебфреймворками та засобами автентифікації, що дозволило створити ефективний та інтерактивний вебдодаток для моніторингу фітнес-активності.

3.2 Тестування та налагодження інформаційно-комп’ютерної системи

Після завершення основної розробки функціоналу мобільного та веб-додатку було проведено комплексне тестування системи, що охоплювало модульне (unit), інтеграційне, функціональне та ручне тестування клієнтської й серверної частин.

Модульне тестування здійснювалося на рівні окремих функцій – таких як розрахунок калорій, таймер тренування, обробка вводу користувача, валідація

та відправка запитів на сервер. Для цього використовувались вбудовані можливості Node.js та утиліти console.assert() і jest.

Фрагмент тесту для функції розрахунку калорій наведено в лістингу 3.7.

Лістинг 3.7 – Фрагмент тесту для функції розрахунку калорій

```
function calculateCalories(duration, type, weight) {
    const multipliers = {
        cardio: 10,
        strength: 6,
        hiit: 12
    };
    const base = multipliers[type.toLowerCase()] || 8;
    return Math.round(duration * base * (weight / 70));
}

console.assert(calculateCalories(30, 'cardio', 70) === 300, 'Тест не
пройдено');
```

кінець лістингу 3.7

Інтеграційне тестування перевіряло взаємодію між клієнтською частиною (React) та сервером (Node.js + Express). Особлива увага приділялась правильному формуванню запитів до API, обробці помилок, а також збереженню й оновленню даних у базі MySQL.

Фрагмент тестового запиту на оновлення профілю наведено в лістингу 3.8.

Лістинг 3.8 – Фрагмент тестового запиту на оновлення профілю

```
fetch('/api/auth/update-profile', {
  method: 'PUT',
  headers: {
    'Content-Type': 'application/json'
  },
  body: JSON.stringify({ full_name: 'Тест Користувач', age: 25 })
})
  .then(res => res.json())
  .then(data => console.log('Оновлення пройшло успішно:', data))
  .catch(err => console.error('Помилка API:', err));
```

кінець лістингу 3.8

Функціональне тестування передбачало перевірку повного циклу роботи додатку: від реєстрації користувача, налаштування цілей, генерації тренувань, виконання вправ до перегляду статистики. Було створено кілька тестових сценаріїв із реальними даними користувачів, які дозволили виявити помилки відображення інтерфейсу та некоректну обробку деяких типів даних (наприклад, null або undefined у полі типу тренування).

Налагодження проводилось за допомогою браузерної консолі розробника, Node.js debug режиму, логів серверу та використання бібліотеки react-hot-toast для виводу повідомлень про стан програми. Було усунуто такі проблеми:

1) помилка в розрахунку калорій при відсутності інтенсивності;

2) некоректне оновлення статистики після завершення тренування;

3) дублювання запитів при швидкому кліку на кнопки «Почати/Завершити».

Мінімальні системні вимоги:

1) браузер з підтримкою ES6 (Chrome 90+, Firefox 85+), або мобільний додаток на Android 8+;

2) сервер Node.js v18+, Express, MySQL 8, Cloudinary API;

3) операційна система Ubuntu 20.04 або Windows 10+, мінімум 2 ГБ ОЗП.

Рекомендовані параметри для оптимальної роботи:

1) дисплей із роздільною здатністю 360x640 або вище;

2) CPU з 4 ядрами, 8 ГБ оперативної пам’яті, SSD для швидкої обробки запитів, стабільне інтернет-з’єднання.

У результаті тестування було виявлено та усунуто низку технічних недоліків, що стосувались логіки генерації тренувань, обробки часу тренування, а також недопрацьованих сценаріїв у графічному інтерфейсі. Завдяки багаторівневій системі тестування та налагодження, інформаційна система працює стабільно та відповідає вимогам до зручності, продуктивності й точності обчислень.

3.3 Розробка бази даних

У межах реалізації серверної частини інформаційної системи було створено реляційну базу даних MySQL з використанням phpMyAdmin як інтерфейсу адміністрування. Схема бази включає таблиці, що зберігають дані користувача, його фізіологічні параметри, харчування, історію тренувань, шаблони вправ тощо. Таблиці взаємопов'язані через зовнішні ключі, реалізовано індекси для оптимізації пошуку. Наведемо приклади запитів на створення базових таблиць в лістингу 3.9.

Лістинг 3.9 – Створення таблиць користувачів та тренувань

```
CREATE TABLE users (
  id INT AUTO_INCREMENT PRIMARY KEY,
  full_name VARCHAR(100) NOT NULL,
  email VARCHAR(100) UNIQUE NOT NULL,
  password VARCHAR(255) NOT NULL,
  age INT,
  gender ENUM('male', 'female'),
  weight DECIMAL(5,2),
  height INT,
  activity_level ENUM('low', 'moderate', 'high'),
  created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);
CREATE TABLE workout_templates (
  id INT AUTO_INCREMENT PRIMARY KEY,
  name VARCHAR(100) NOT NULL,
  duration_minutes INT NOT NULL,
  intensity ENUM('Легка', 'Середня', 'Висока') NOT NULL,
  fitness_level ENUM('beginner', 'intermediate', 'advanced') NOT NULL,
  created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);
```

кінець лістингу 3.9

Також наведемо приклади SQL-запитів. Запит додавання нового користувача наведено в лістингу 3.10.

Лістинг 3.10 – Запит додавання нового користувача

```
INSERT INTO users (full_name, email, password, age, gender, weight,
height, activity_level)
```

```
VALUES ('Іван Іваненко', 'ivan@example.com', 'hashed_password_123', 28,
'male', 75.5, 180, 'moderate');
```

кінець лістингу 3.10

Пошук тренувань для новачка наведено в лістингу 3.11.

Лістинг 3.11 – Пошук тренувань для новачка

```
SELECT name, duration_minutes, intensity
FROM workout_templates
WHERE fitness_level = 'beginner';
```

кінець лістингу 3.11

В лістингу 3.12 наведено оновлення цілі користувача.

Лістинг 3.12 – Оновлення цілі користувача

```
UPDATE user_preferences
SET daily_calorie_goal = 2200
WHERE user_id = 5;
```

кінець лістингу 3.12

В лістингу 3.13 наведено пошук історії ваги користувача за останній місяць.

Лістинг 3.13 – Пошук історії ваги користувача за місяць

```
SELECT weight, recorded_date
FROM weight_history
WHERE user_id = 2 AND recorded_date >= CURDATE() - INTERVAL 30 DAY
ORDER BY recorded_date DESC;
```

кінець лістингу 3.13

Запропонована структура бази даних дозволяє ефективно зберігати й обробляти велику кількість персоналізованих даних. Вона гнучка, розширювана та забезпечує добру підтримку запитів на агрегацію, фільтрацію, персоналізацію і подальшу аналітику. За рахунок реляційних зв’язків і індексів досягнута як структурна цілісність, так і швидкодія запитів.

3.4 Захист інформаційно-комп’ютерної системи

Під час розробки вебдодатку для моніторингу фізичної активності користувачів особливу увагу було приділено реалізації базових засобів безпеки, зокрема – автентифікації, авторизації, захисту маршрутів, хешування паролів, а також частковій реалізації захисту від несанкціонованого доступу до API.

У системі реалізовано JWT-аутентифікацію (JSON Web Token). Коли користувач проходить реєстрацію або логін, сервер генерує токен (ліст. 3.14).

Лістинг 3.14 – Генерація токена

```
const token = jwt.sign({ userId: user.id }, process.env.JWT_SECRET, {
    expiresIn: '7d'
});
res.cookie('jwt', token, { httpOnly: true, secure: true });
```

кінець лістингу 3.14

JWT містить зашифровану інформацію про користувача (ID), яка передається в cookie. Токен захищено симетричним алгоритмом HMAC SHA-256, ключ до якого зберігається у .env.

На боці сервера запити до захищених маршрутів перевіряються за допомогою protectRoute() (ліст. 3.15). Це забезпечує перевірку автентичності запиту перед виконанням будь-якої логіки користувача.

Лістинг 3.15 – Перевірка запиту

```
const decoded = jwt.verify(token, process.env.JWT_SECRET);
const user = await executeQuery("SELECT * FROM users WHERE id = ?",
[decoded.userId]);
req.user = user;
```

кінець лістингу 3.15

Під час реєстрації паролі користувачів хешуються за допомогою бібліотеки bcrypt, що унеможливлює зберігання паролів у відкритому вигляді (ліст. 3.16).

Лістинг 3.16 – Хешування паролів

```
const salt = await bcrypt.genSalt(10);
const hashedPassword = await bcrypt.hash(req.body.password, salt);
```

кінець лістингу 3.16

Під час входу здійснюється верифікація.

Лістинг 3.17 – Верифікація

```
const isMatch = await bcrypt.compare(req.body.password,
user.password_hash);
```

кінець лістингу 3.17

Алгоритм bcrypt заснований на адаптивному хешуванні з використанням сіль і великої кількості раундів, що значно ускладнює brute-force атаки.

Серверні маршрути захищено проміжним ПЗ (middleware), яке перевіряє наявність та дійсність токена перед наданням доступу. Також реалізовано механізм контролю ролей (ліст. 3.18). Це дозволяє відокремлювати звичайних користувачів від адміністратора (у разі майбутнього розширення).

Лістинг 3.18 – Механізм контролю ролей

```
const requireRole = (roles) => {
    return (req, res, next) => {
        if (!roles.includes(req.user.role)) {
            return res.status(403).json({ message: "Недостатньо прав
доступу" });
        }
        next();
    };
};
```

кінець лістингу 3.18

Всі критичні запити (наприклад, /update-profile, /stats) доступні лише для користувача з валідним JWT. У випадку відсутності токена або недійсного підпису – повертається статус 401 Unauthorized.

Для зберігання профільних зображень використовується сервіс Cloudinary, який використовує HTTPS, має обмеження MIME-типів та підтримує валідацію і форматування зображень (ліст. 3.19). Доступ до API Cloudinary можливий лише через авторизований ключ.

Лістинг 3.19 – Використання Cloudinary

```
cloudinary.config({
    cloud_name: process.env.CLOUDINARY_CLOUD_NAME,
    api_key: process.env.CLOUDINARY_API_KEY,
    api_secret: process.env.CLOUDINARY_API_SECRET
});
```

кінець лістингу 3.19

Отже, у додатку реалізовано такі основні механізми захисту:

1) шифрування автентифікації через JWT;

2) хешування паролів через bcrypt;

3) захист маршрутів middleware;

4) контроль доступу;

5) захищене зберігання зображень через Cloudinary.

Завдяки цим заходам система забезпечує базовий рівень інформаційної безпеки при збереженні та обробці персональних даних користувача. У майбутньому можливе доповнення модулями 2FA, rate-limiting або анти-DDoS.

Висновки до розділу 3

У результаті виконання практичної частини проєкту було повністю реалізовано інформаційно-комп’ютерну систему у вигляді вебдодатку для генерації тренувань та розрахунку добової калорійності. Було обрано і ефективно застосовано стек сучасних технологій: клієнтська частина реалізована на React із використанням бібліотек Tailwind CSS і Zustand, серверна логіка побудована на Node.js із використанням Express, а зберігання даних реалізовано на основі реляційної СУБД MySQL.

У процесі реалізації було створено багатокомпонентну архітектуру, налагоджено механізм взаємодії з REST API, впроваджено засоби аутентифікації користувачів, захисту маршрутів, збереження даних про вправи, харчування, історію тренувань та вагу. Для зберігання медіафайлів інтегровано хмарне сховище Cloudinary.

Здійснено попереднє тестування функціональності, під час якого було виявлено й усунуто низку недоліків, що підтвердило працездатність основних модулів системи. Також було розроблено і протестовано базу даних з підтримкою зв’язків між сутностями, індексуванням та обмеженнями цілісності.

У підсумку система є масштабованою, стабільною, має зручний користувацький інтерфейс і забезпечує базовий рівень інформаційної безпеки. Вона може бути застосована у реальних умовах для підтримки фітнес-режиму користувачів та має перспективи для подальшого розвитку – зокрема інтеграції з носимими пристроями, алгоритмами персоналізованих рекомендацій і модулями аналітики.

ВИСНОВКИ

У межах виконання кваліфікаційної роботи було повністю реалізовано поставлене завдання – створено функціональний вебдодаток для генерації індивідуальних тренувань та розрахунку добової калорійності продуктів з урахуванням фізіологічних характеристик користувача. У процесі розробки були досягнуті як якісні, так і кількісні результати: реалізовано понад 10 взаємодіючих REST API-ендпоінтів, понад 7 пов’язаних сутностей у структурі реляційної бази даних, інтерфейс користувача побудовано відповідно до сучасних принципів UI/UX.

В ході роботи було проведено детальний огляд існуючих фітнес-додатків та аналіз їхніх основних функцій і обмежень. Визначено, що більшість популярних рішень у сфері фітнесу та нутриціології мають обмежений функціонал, об’єднуючи лише одну з можливостей – трекінг харчування або тренувань. Це підкреслює необхідність розробки комплексної системи, яка поєднуватиме функціональність трекінгу калорій, планування тренувань та відстеження прогресу. Огляд наукових праць і патентів підтвердив актуальність даної теми і важливість створення зручних і персоналізованих фітнес-рішень для широкого кола користувачів.

Здійснено всебічний аналіз вимог до розроблюваної системи. Визначено функціональні та нефункціональні вимоги, що є критичними для успішної реалізації вебдодатку. Сформульовано основні цілі системи, які включають авторизацію користувачів, збір персональних даних, розрахунок калорійності та генерацію тренувань. Структурне моделювання за допомогою UML-нотацій дозволило наочно представити основні функції додатку та алгоритми взаємодії компонентів. Це допомогло чітко сформулювати вимоги до бази даних, інтерфейсу та безпеки, що забезпечило ефективне проектування всіх елементів системи.

Розглянуто вибір технологій, інструментів і алгоритмів для реалізації системи. Обрано стійкий стек технологій, який включає React для клієнтської

частини, Node.js + Express для серверної логіки, MySQL для бази даних та JWT для забезпечення безпеки. Основним аспектом було визначення алгоритмів для розрахунку калорійності та генерації тренувань, які відповідають вимогам точності і швидкодії. Вибрані технології забезпечують масштабованість, високу продуктивність і гнучкість у реалізації нових функцій, що забезпечить ефективну та стабільну роботу додатку у реальних умовах.

У процесі реалізації вебдодатку було створено багатокомпонентну систему з інтерактивним інтерфейсом користувача, генератором тренувань та механізмами відстеження прогресу. Завдяки використанню React, REST API та сучасних бібліотек вдалося забезпечити високу швидкодію, адаптивність до різних пристроїв і зручну навігацію. Реалізовано низку функцій: таймер вправ, калькулятор калорій, генератор тренувань за рівнем підготовки, захищені маршрути тощо. В результаті було досягнуто цілісного інтерфейсу з логічно структурованим функціоналом, орієнтованого на потреби користувача.

Під час тестування було перевірено стабільність роботи основних компонентів, виявлено та виправлено ряд логічних і візуальних помилок, що підвищило надійність системи. Було застосовано кілька типів тестування, зокрема модульне, інтеграційне та ручне. Тестові сценарії охопили увесь функціональний цикл: від реєстрації до завершення тренувань. Внаслідок цього система демонструє відповідність вимогам до зручності, точності розрахунків і стабільності роботи.

Створена база даних на MySQL забезпечує ефективне зберігання і обробку персональних даних користувачів, інформації про тренування, харчування та історію змін ваги. Завдяки структурованому проєктуванню з використанням зовнішніх ключів, індексів і цілісних зв’язків досягнуто надійності та масштабованості системи. Реалізовано стандартні SQL-запити, що покривають типові потреби: реєстрація, зчитування статистики, оновлення профілю, агрегація результатів.

Для забезпечення базового рівня безпеки в системі впроваджено механізми авторизації та автентифікації за допомогою JWT, хешування паролів

через bcrypt, захист маршруту middleware-функціями, а також перевірку ролей. Дані користувачів шифруються, критичні дії доступні лише автентифікованим користувачам, а зображення зберігаються в захищеному хмарному середовищі. У результаті реалізовано цілісну систему контролю доступу та збереження конфіденційності.

Практичне використання розробленого програмного продукту можливе в межах індивідуального та групового фітнес-консультування, інтеграції в платформи для тренерів, а також як прототип для стартапів у сфері digital-health. Завдяки масштабованій архітектурі й модульності рішення його можна адаптувати під інші потреби: наприклад, медичні облікові системи, нутрієнтні трекери чи персональні помічники зі здорового способу життя.

Подальше вдосконалення може включати інтеграцію з носимими пристроями (типу Fitbit, Apple Watch), впровадження алгоритмів машинного навчання для персоналізованих рекомендацій, розширення функціоналу аналітики активності користувача, реалізацію мобільної версії на базі Flutter або React Native та впровадження багатомовної підтримки.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. The Intention to Use Fitness and Physical Activity Apps: A Systematic Review / S. Angosto et al. Sustainability. 2020. Vol. 12, no. 16. P. 6641. URL: https://doi.org/10.3390/su12166641 (date of access: 12.04.2025).

2. The use of fitness centre apps and its relation to customer satisfaction: a UTAUT2 perspective / H. Ferreira Barbosa et al. International Journal of Sports Marketing and Sponsorship. 2021. Ahead-of-print, ahead-of-print. URL: https://doi.org/10.1108/ijsms-01-2021-0010 (date of access: 12.04.2025).

3. Whelan E., Clohessy T. How the social dimension of fitness apps can enhance and undermine wellbeing. Information Technology & People. 2020. Ahead-of-print, ahead-of-print. URL: https://doi.org/10.1108/itp-04-2019-0156 (date of access: 12.04.2025).

4. Бондаренко С. В., Бондаренко М. Е. Розробка мобільного фітнес додатку з використанням штучного інтелекту. Проблеми інформатизації : одинадцята міжнародна науково-технічна конференція. 2023. С. 61.

5. Коренева А. С. Мобільний додаток супроводження індивідуальних фітнес–тренувань – робота на здобуття кваліфікаційного рівня магістр – спец. 122 – комп’ютерні науки. Суми : СумДУ, 2020. 64 с.

6. Мошенська Т., Долгополова Н., Сорочинська М. Застосування онлайн-платформ та фітнес-додатків для формування здорового способу життя. Науково-методичні основи використання інформаційних технологій в галузі фізичної культури та спорту. 2023. №7. С. 75-83.

7. Calorie Tracker & BMR Calculator to Reach Your Goals – MyFitnessPal. URL: https://www.myfitnesspal.com/ (date of access: 04.05.2025).

8. Intensive workouts & individual training plans – FREELETICS. URL: https://www.freeletics.com/en/ (date of access: 04.05.2025).

9. Lifesum – Healthy eating. Simplified. URL: https://lifesum.com/ (date of access: 04.05.2025).

10. React. URL: https://react.dev/ (date of access: 07.06.2025).

11. Banks A., Porcello E. Learning React: modern patterns for developing React apps. O'Reilly Media, 2020. 310 p.

12. Vite. vitejs. URL: https://vite.dev/ (date of access: 07.06.2025).

13. Gascón U. Node. js for Beginners. A comprehensive guide to building efficient, full-featured web applications with Node. js. Packt Publishing Ltd, 2024. 382 p.

14. Munirov J. J. NESTJS VS EXPRESS. JS A COMPREHENSIVE COMPARISON. Modern World Education. New Age Problems–New solutions. 2025.Vol. 2, №6. P. 33-39.

15. MySQL. URL: https://www.mysql.com/ (date of access: 08.06.2025).