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 ПІДБОРУ НОУТБУКІВ З ВИКОРИСТАННЯМ РЕКОМЕНДАЦІЙНОЇ СИСТЕМИ AI З ВИКОРИСТАННЯМ REACT, NODE.JS ТА MYSQL

DEVELOPMENT OF A WEB APPLICATION FOR LAPTOP SELECTION USING AN AI RECOMMENDATION SYSTEM WITH REACT, NODE.JS AND MYSQL

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

Виконав: здобувач вищої освіти
групи ІПЗ-41
Кондратюк Вадим Олександрович

Керівник:
Вознюк Анастасія Вадимівна

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 2024р.

ЗАВДАННЯ

НА КВАЛІФІКАЦІЙНУ РОБОТУ ЗДОБУВАЧУ ВИЩОЇ ОСВІТИ

Кондратюку Вадиму Олександровичу
(прізвище, ім'я, по батькові)

1. Тема кваліфікаційної роботи *«Розробка вебдодатку для підбору ноутбуків з використанням рекомендаційної системи AI з використанням React, node.js та MySQL»*
Керівник роботи: *асистент Вознюк А. В.*
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, MySQL, Visual Studio code, методичні вказівки до виконання кваліфікаційної роботи бакалавра*

4. Зміст розрахунково-пояснювальної записки (перелік питань, що потрібно розробити): *вступ, аналіз платформ для підбору ноутбуків, постановка завдань, специфікація вимог, проектування системи, вибір технологій і алгоритмів, реалізація вебдодатку, тестування та налагодження, розробка бази даних, захист системи, документація для користувача, висновки*

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	*Вознюк А. В.*		
Специфікація вимог до розробленої системи	*Вознюк А. В.*		
Розробка об'єкта проектування	*Вознюк А. В.*		
Нормоконтроль	*Вознюк А. В.*		
Гарант ОП	*Ліщина Н. М.*		
Показник запозичень тексту	3.42 %		
Академічна доброчесність	*Вознюк А. В.*		

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.	Виконано
2	Аналіз проблеми розробки та впровадження об'єкту проектування	до 01.03.2025 р.	Виконано
3	Обґрунтування вибору шляхів, технологій і засобів вирішення поставленого завдання	до 15.03.2025 р.	Виконано
4	Розробка функціонально-структурної схеми роботи об'єкта проектування та проектування бази даних	до 29.03.2025 р.	Виконано
5	Практична реалізація об'єкта проектування та розробка бази даних	до 26.04.2025 р.	Виконано
6	Тестування та налагодження об'єкта проектування	до 03.05.2025 р.	Виконано
7	Здача чистового варіанту кваліфікаційної роботи бакалавра на кафедру	до 10.06.2025 р.	Виконано

Здобувач вищої освіти _____ Вадим КОНДРАТЮК

Керівник кваліфікаційної роботи _____ Анастасія ВОЗНЮК

АНОТАЦІЯ

Кондратюк В. О. Розробка вебдодатку для підбору ноутбуків з використанням рекомендаційної системи AI з використанням React, Node.js та MySQL. 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 та сформульовано завдання. У другому розділі описано вимоги до системи, обґрунтовано вибір технологій і алгоритмів. У третьому розділі реалізовано функціонал додатку, здійснено тестування та налагодження. У висновках підсумовано результати роботи, наведено рекомендації щодо вдосконалення.

Ключові слова: вебдодаток, ноутбук, рекомендаційна система, React, Node.js, MySQL.

ABSTRACT

Kondratiuk V. Development of a web application for laptop selection using an AI recommendation system with React, Node.js and MySQL. Manuscript. Bachelor's qualification work of the study program "Software Engineering", specialty "Software Engineering". Lutsk National Technical University. Lutsk, 2025.

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

The first chapter presents an analysis of the current state of the problem and defines the research objectives. The second chapter describes the system requirements and substantiates the choice of technologies and algorithms. The third chapter presents the implementation of the application, its testing, and debugging. The conclusions summarize the work results and provide suggestions for further improvement.

Keywords: web application, laptop, recommendation system, React, Node.js, MySQL.

ЗМІСТ

ВСТУП 7
РОЗДІЛ 1 АНАЛІЗ ПЛАТФОРМ ДЛЯ ПІДБОРУ НОУТБУКІВ І ПОСТАНОВКА ЗАВДАНЬ НА КВАЛІФІКАЦІЙНУ РОБОТУ 9
1.1 Аналіз сучасного стану проблеми 9
1.2 Постановка завдання на кваліфікаційну роботу бакалавра 11
Висновки до розділу 1 12
РОЗДІЛ 2 СПЕЦИФІКАЦІЯ ВИМОГ ДО РОЗРОБЛЮВАНОЇ СИСТЕМИ 14
2.1 Аналіз, визначення вимог до розроблюваного програмного забезпечення та проектування програмного забезпечення 14
2.2 Вибір засобів, методів і алгоритмів вирішення поставленого завдання 19
Висновки до розділу 2 22
РОЗДІЛ 3 РОЗРОБКА ВЕБДОДАТКУ ДЛЯ ПІДБОРУ НОУТБУКІВ З ВИКОРИСТАННЯМ РЕКОМЕНДАЦІЙНОЇ СИСТЕМИ 24
3.1 Практична реалізація об’єкта проєктування 24
3.2 Тестування та налагодження інформаційно-комп’ютерної системи 28
3.3 Розробка бази даних 30
3.4 Захист інформаційно-комп’ютерної системи 33
3.5 Розробка документації для програмного продукту 35
Висновки до розділу 3 37
ВИСНОВКИ 39
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ 42

ВСТУП

Актуальність теми. У сучасних умовах розвитку інформаційних технологій і зростання ринку комп'ютерної техніки користувачі часто стикаються з проблемою вибору оптимального ноутбука для власних потреб. Різноманіття моделей, технічних характеристик, виробників та цінових категорій ускладнює процес прийняття рішення, особливо для користувачів без глибоких технічних знань. Це створює об'єктивну потребу у використанні інтелектуальних програмних засобів, які б автоматизували процес аналізу характеристик ноутбуків і формували персоналізовані рекомендації на основі запитів користувача.

Сьогодні у світі активно розвиваються рекомендаційні системи – алгоритмічні модулі, що дозволяють адаптувати вибір товарів або послуг до індивідуальних вподобань користувача. Вони ефективно застосовуються у сфері електронної комерції, онлайн-навчання, стрімінгових сервісів тощо. У сегменті підбору комп'ютерної техніки рекомендаційні системи використовуються недостатньо широко, хоча потенціал їх впровадження є значним. Існуючі вебплатформи здебільшого надають лише фільтрацію за базовими параметрами, без врахування особистих потреб або специфіки використання, що не завжди забезпечує релевантний результат.

З огляду на це, актуальність кваліфікаційної роботи полягає у створенні інноваційного вебдодатку для підбору ноутбуків, який використовує алгоритми рекомендаційної системи з елементами штучного інтелекту. Такий підхід дозволить значно підвищити якість прийняття рішень користувачами, забезпечити гнучкий і точний підбір техніки, зменшити витрати часу на пошук необхідного пристрою.

Метою кваліфікаційної роботи є розробка вебзастосунку для інтелектуального підбору ноутбуків, реалізованого на базі рекомендаційної системи. Галузь застосування – онлайн-платформи продажу техніки, магазини електроніки, інформаційні сервіси з підбору електронних пристроїв.

Об’єктом кваліфікаційної роботи виступає інформаційно-комп’ютерна система підбору ноутбуків.

Предмет кваліфікаційної роботи – програмна реалізація вебдодатку із рекомендаційною підсистемою. Програмний продукт буде створено з використанням сучасного вебстеку технологій: React для клієнтської частини, Node.js для серверної логіки та MySQL як засобу збереження і обробки даних.

Завдання дослідження:

1) провести аналіз сучасних онлайн-платформ для підбору ноутбуків;
2) визначити функціональні та нефункціональні вимоги до системи, побудувати UML-діаграми;
3) обґрунтувати вибір технологій та описати алгоритм рекомендаційної системи;
4) реалізувати інтерфейс, логіку фільтрації, обробку замовлень і взаємодію з API;
5) провести тестування і налагодження всіх функціональних модулів;
6) створити структуру бази даних, реалізувати SQL-запити;
7) реалізувати авторизацію, захист маршрутів та хешування паролів;
8) описати мінімальні вимоги, інтегровану довідку та інструкції для встановлення.

Кваліфікаційна робота узгоджується з сучасними тенденціями цифровізації споживчого вибору, є логічним продовженням попередніх розробок у сфері розумних електронних сервісів і спрямована на підвищення зручності користування IT-продуктами широким колом споживачів.

РОЗДІЛ 1
АНАЛІЗ ПЛАТФОРМ ДЛЯ ПІДБОРУ НОУТБУКІВ І ПОСТАНОВКА ЗАВДАНЬ НА КВАЛІФІКАЦІЙНУ РОБОТУ

1.1 Аналіз сучасного стану проблеми

У сучасному інформаційному середовищі існує значна кількість онлайн-платформ, які пропонують функціональність підбору ноутбуків, зокрема такі відомі ресурси, як Hotline.ua, Rozetka.ua, Stylus.ua, MOYO.ua та інші. Більшість із них забезпечують користувачам лише можливість ручного фільтрування товарів за окремими технічними параметрами (процесор, RAM, тип накопичувача, розмір екрана, тощо). Проте така модель підбору є обмеженою, оскільки не враховує цілі використання пристрою (наприклад, офісна робота, геймінг, навчання чи подорожі), рівень технічної обізнаності користувача, а також його індивідуальні уподобання.

Таблиця 1.1 демонструє, що усі аналізовані платформи реалізують лише базову фільтрацію параметрів, не маючи повноцінних алгоритмів персоналізованої рекомендації. Наявність зручного інтерфейсу та опції «обране» підвищує зручність взаємодії, однак жоден із сервісів не враховує комплексно потреби користувача – наприклад, ціль використання ноутбука, пріоритети за продуктивністю чи автономністю. Це підкреслює доцільність створення нового вебзастосунку, який поєднує AI-рекомендації з індивідуальними вподобаннями користувача.

Таблиця 1.1 – Порівняльна характеристика існуючих вебплатформ для підбору ноутбуків

Платформа	Наявність фільтрів	Рекомендаційна система	Персоналізація запиту	Інтерфейс користувача	Можливість збереження обраного
Hotline.ua [1]	Так	Ні	Ні	Середній	Так
Rozetka.ua [2]	Так	Частково	Ні	Високий	Так
Stylus.ua [3]	Так	Ні	Ні	Середній	Так
MOYO.ua [4]	Так	Ні	Ні	Високий	Так

Проведений аналіз літературних джерел із тематики персоналізованих рекомендаційних систем в електронній комерції дозволяє окреслити основні напрями розвитку цієї галузі та підходи, які можуть бути ефективно застосовані для побудови інтелектуального вебдодатку з підбору ноутбуків. У роботі A. G. AG та співавторів [5] наголошується на ролі машинного навчання у формуванні персоналізованого досвіду користування на платформах e-commerce, що сприяє зростанню залученості клієнтів. Узагальнене бачення сучасних методів рекомендацій подано в оглядовій роботі H. Ko та співавторів [6], де класифікуються алгоритми фільтрації, гібридизації, колаборації, а також розглядаються галузі їх практичного застосування. Дослідження L. Liu [7] підтверджує ефективність таких підходів через точне прогнозування потреб користувача на основі його поведінки.

Водночас K. Singh та співавтори [8] пропонують гібридну модель, яка поєднує ансамблеве навчання з аналізом настроїв, що значно підвищує точність рекомендацій та персоналізацію. У дослідженні M. Tahir та співаторів [9] розглянуто створення e-commerce платформи з вбудованою рекомендаційною системою на основі машинного навчання, де особлива увага приділяється автоматизованій обробці великих обсягів даних та динамічній адаптації рекомендацій до змін у поведінці користувачів, що є досить важливим у реальному комерційному середовищі.

Серед українських джерел І. Р. Лошенюк та співавтори [10] вивчають ефективність застосування штучного інтелекту у персоналізованому маркетингу, що безпосередньо стосується й рекомендаційних механізмів. П. Е. Ситнікова та М. О. Гребенюк [11] презентують гібридну модель користувача, орієнтовану на компактне зберігання профілів та ефективну обробку запитів, що є перспективним підходом для задач із обмеженими обчислювальними ресурсами. Загалом, аналіз джерел підтверджує доцільність і актуальність створення вебсистеми, яка поєднує технології машинного навчання, гібридні підходи до формування рекомендацій та адаптивний

інтерфейс користувача для досягнення максимальної відповідності результатів очікуванням користувачів.

Патентний пошук не виявив систем, що інтегрують алгоритмічну рекомендацію ноутбуків з урахуванням індивідуального профілю користувача у контексті саме українських онлайн-магазинів, що додатково підтверджує інноваційність підходу. Окрім того, у більшості вітчизняних рішень немає реалізованої логіки персоналізації, рекомендаційної аналітики або адаптації на основі зворотного зв’язку від користувача.

З урахуванням зазначеного, створення інтелектуального вебзастосунку з використанням рекомендаційної системи на основі AI-технологій є доцільним та актуальним. Така система зможе автоматично підбирати ноутбуки, аналізуючи запити користувачів, їхні пріоритети, а також технічні характеристики пристроїв, що дозволить значно підвищити якість та релевантність результатів пошуку, спростити процес вибору для кінцевого користувача та покращити користувацький досвід.

1.2 Постановка завдання на кваліфікаційну роботу бакалавра

У межах виконання кваліфікаційної роботи бакалавра ставляться такі основні цілі, досягнення яких забезпечить реалізацію повнофункціонального вебдодатку для підбору ноутбуків з використанням рекомендаційної системи:

1) провести аналіз сучасних платформ для підбору ноутбуків, виявити їх переваги та недоліки, визначити потребу у вдосконаленому рішенні на основі рекомендаційних технологій;

2) сформулювати функціональні та нефункціональні вимоги до майбутньої системи, враховуючи специфіку предметної області та очікування кінцевого користувача, провести проєктування системи, використовуючи UML-діаграми.

3) обрати та обгрунтувати засоби, методи та алгоритми для майбутньої розробки;

4) реалізувати функціонал вебдодатоку із використанням сучасних технологій: React – для створення інтерфейсу користувача, Node.js – для серверної логіки;

5) здійснити тестування працездатності системи, перевірку її функціоналу та оптимізацію швидкодії;

6) розробити базу даних MySQL для зберігання інформації про товари, користувачів та результати підбору;

7) реалізувати базові засоби захисту даних системи, з урахуванням вимог до безпеки;

8) підготувати супровідну документацію та надати інструкції щодо використання програмного забезпечення.

В цілому, успішне виконання вказаних завдань дозволить створити повноцінну інформаційно-комп'ютерну систему, що полегшить вибір ноутбука та підвищить задоволеність користувачів завдяки інтелектуальним рекомендаціям.

Висновки до розділу 1

У першому розділі було проаналізовано сучасний стан ринку вебплатформ для підбору ноутбуків, а також розглянуто існуючі програмні рішення, що використовуються в цій сфері. У результаті дослідження встановлено, що більшість актуальних сервісів (Hotline.ua, Rozetka.ua, Stylus.ua, MOYO.ua, DNS.ua) пропонують лише базові інструменти фільтрації товарів за технічними параметрами, без урахування особистих вподобань користувача, цілей використання пристрою або його індивідуального профілю. Персоналізовані рекомендації та інтелектуальний підхід до вибору відсутні або реалізовані на мінімальному рівні.

Проведений огляд літературних джерел і тенденцій у сфері створення рекомендаційних систем засвідчив, що контентна та гібридна фільтрація, а також алгоритми машинного навчання є ефективними підходами до

формування релевантних результатів для користувача. Разом із цим, патентний пошук вказує на відсутність подібних рішень, орієнтованих на ринок ноутбуків саме в українському сегменті.

На основі аналізу було сформульовано чітке технічне завдання на кваліфікаційну роботу бакалавра, яке включає розробку вебдодатку з рекомендаційною системою на базі технологій React, Node.js та MySQL. Така система має забезпечити автоматизований, персоналізований підбір ноутбуків відповідно до заданих користувачем критеріїв, підвищуючи ефективність і зручність процесу вибору.

РОЗДІЛ 2
СПЕЦИФІКАЦІЯ ВИМОГ ДО РОЗРОБЛЮВАНОЇ СИСТЕМИ

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

Проєктування вебдодатку для підбору ноутбуків з використанням рекомендаційної системи потребує формального підходу до визначення його архітектури, функціональних характеристик та вимог до інтерфейсу. На початковому етапі було обґрунтовано доцільність застосування каскадної (waterfall) моделі життєвого циклу ПЗ, яка забезпечує поетапне планування, реалізацію, тестування та підтримку програмного продукту. Для опису та візуалізації структури системи було обрано UML-нотацію, яка дозволяє ефективно формалізувати функціональні залежності та компоненти ПЗ [12].

Аналіз вимог здійснювався на основі функціонального аналізу користувацьких сценаріїв, опитування цільової аудиторії щодо очікувань від сервісу, а також порівняння із існуючими аналогами.

Порівняння з існуючими аналогами, такими як Hotline, Rozetka, Stylus або MOYO, показує, що більшість сучасних платформ для підбору ноутбуків реалізують лише базову фільтрацію за технічними параметрами – обсяг пам’яті, тип процесора, ціновий діапазон тощо, – без врахування комплексного профілю користувача, його індивідуальних потреб чи сценаріїв використання. На відміну від них, запропонований у цій роботі вебдодаток реалізує персоналізовану рекомендаційну систему, яка аналізує вподобання, цілі покупки (навчання, робота, ігри, портативність тощо), а також здатна адаптувати рекомендації за допомогою алгоритмів штучного інтелекту. Такий підхід забезпечує точніший, релевантніший і зручніший для користувача підбір ноутбуків, особливо для недосвідчених покупців.

Мета системи – забезпечити швидкий, персоналізований підбір ноутбуків відповідно до параметрів, що вводить користувач (тип використання, бюджет,

вагові пріоритети – продуктивність, автономність, портативність тощо), з використанням інтелектуальної моделі рекомендації.

Функціональні вимоги:

1) авторизація та реєстрація користувача з валідацією введених даних;
2) застосування алгоритму рекомендації для побудови рейтингу відповідних моделей;
3) перегляд картки товару з деталями;
4) збереження обраного до кошика;
5) пошук ноутбуків за ключовими словами;
6) додавання відгуку на товар;
7) адміністрування бази даних товарів (створення, редагування, видалення);
8) забезпечення захисту даних користувача і цілісності транзакцій.

Нефункціональні вимоги:

1) вебдоступність за протоколом HTTPS;
2) сумісність з сучасними браузерами (Chrome, Firefox, Safari);
3) відгук інтерфейсу не більше 1 секунди;
4) масштабованість до 10 000 активних користувачів;
5) захист від SQL-ін’єкцій та XSS-атак;
6) розмежування прав користувачів (звичайний/адміністратор).

Інтерфейс вебдодатку реалізується з урахуванням принципів доступності (accessibility), мінімалізму та інтуїтивної навігації. Для цього використовується React.js, що дозволяє створити односторінкову динамічну структуру (SPA). Основними екранами системи є:

1) головна сторінка з каталогом товарів;
2) результати пошуку;
3) кошик з вибраними товарами;
4) сторінка товару, де також розміщено відгуки та рекомендації.

Моделювання інтерфейсу та функцій супроводжується UML-діаграмами. Use Case (рис. 2.1) – демонструє дії користувачів і систему.

Рисунок 2.1 – Діаграма випадків використання

Ця діаграма ілюструє взаємодію трьох основних ролей із системою: гостя, зареєстрованого користувача та адміністратора. Гість має можливість переглядати товари, шукати ноутбуки за назвою або характеристиками, а також реєструватися для отримання розширеного доступу. Після реєстрації користувач отримує доступ до персоналізованих рекомендацій, може додавати товари до кошика, оформлювати замовлення та переглядати їх історію. Адміністратор керує товарами в системі, має змогу редагувати інформацію про

ноутбуки, додавати нові позиції або видаляти існуючі, а також контролює всі замовлення користувачів. Така діаграма демонструє чіткий поділ функціоналу між типами користувачів, що забезпечує належний рівень доступу й безпеки в системі.

Class Diagram (рис. 2.2) – відображає структуру класів, що взаємодіють з базою даних. Ця діаграма описує логічну структуру системи, що лежить в основі реалізації вебдодатку.

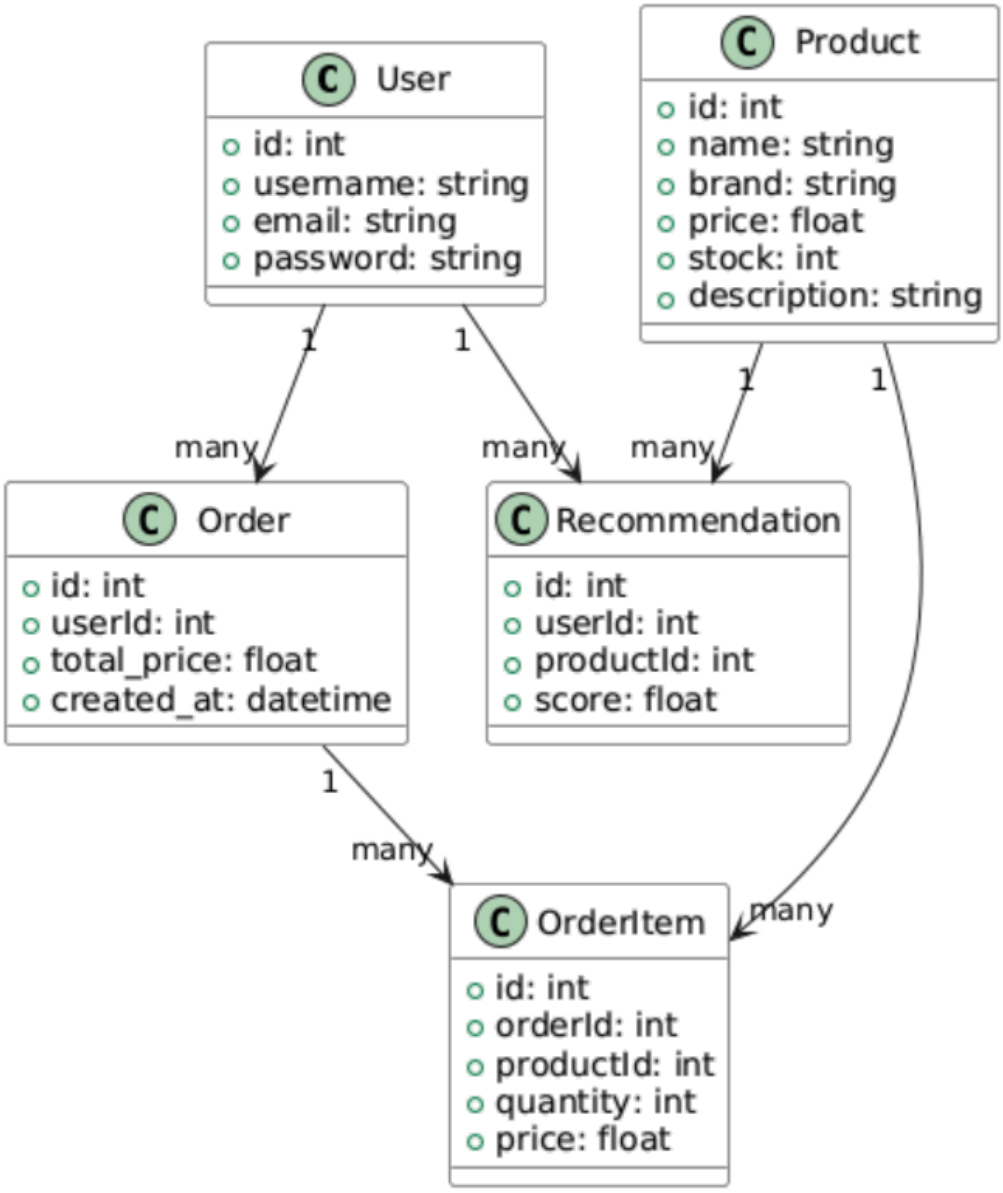

Рисунок 2.2 – Діаграма класів

Основною сутністю є клас користувача, який містить ідентифікаційні та автентифікаційні атрибути. Користувач пов’язаний із замовленнями, кожне з яких включає список товарів через проміжну структуру з кількістю та ціною. Клас товару описує всі характеристики ноутбука – назву, бренд, опис, ціну та залишки на складі. Між користувачем і товаром існує додатковий клас рекомендації, що зберігає результат обчислення релевантності конкретного товару для конкретного користувача. Усі ці класи зв’язані між собою в межах реляційної моделі, що дозволяє організувати зручне зберігання, пошук та обробку даних. Така структура забезпечує гнучкість системи, її масштабованість та можливість розширення за рахунок нових функцій.

Sequence Diagram (рис. 2.3) – описує взаємодію користувача з логікою рекомендацій. Ця діаграма демонструє послідовність дій при формуванні рекомендацій. Користувач вводить параметри через інтерфейс, дані передаються на сервер, де відбувається звернення до бази даних і виклик модуля рекомендації. Результат повертається користувачу у вигляді списку відповідних ноутбуків.

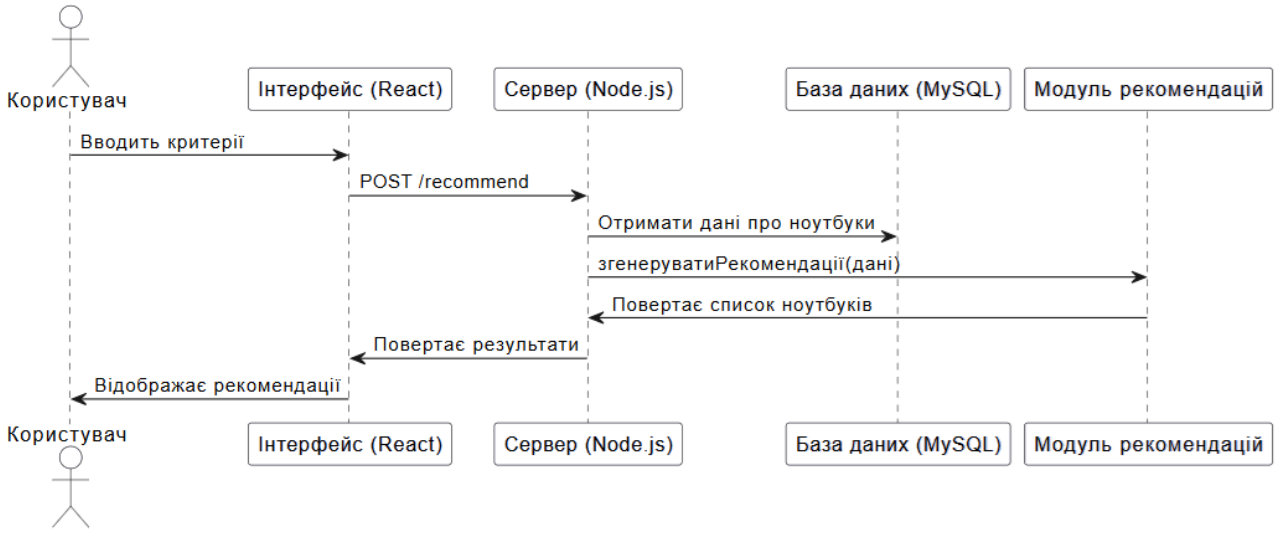

Рисунок 2.3 – Діаграма послідовностей

База даних створюється в MySQL, із таблицями: users, laptops, criteria, favorites, logs. Передбачена нормалізація даних, забезпечення цілісності

зовнішніми ключами, реалізація індексів для оптимізації запитів. Система обробляє дані у форматі JSON, обмін відбувається через REST API на Node.js. Запити до бази захищені через параметризовані SQL-запити.

Аналіз моделей виявив коректність визначених зв’язків між сутностями системи, що забезпечує розширюваність архітектури та надійність бізнес-логіки. Використання рекомендаційного алгоритму (зокрема, спрощеної гібридної моделі контентного та фільтраційного типу) дозволяє забезпечити релевантність добору, виходячи з уподобань користувачів.

2.2 Вибір засобів, методів і алгоритмів вирішення поставленого завдання

Для реалізації вебдодатку з функціональністю рекомендаційної системи було проведено аналіз сучасних технологій, інструментів та алгоритмів, що застосовуються при створенні подібних інформаційних систем. Вибір конкретного стеку технологій обумовлений вимогами до продуктивності, гнучкості, масштабованості та зручності розробки.

Для реалізації клієнтської частини обрано фреймворк React.js, який забезпечує побудову швидких односторінкових застосунків (SPA), дозволяє реалізовувати компонентно-орієнтовану архітектуру інтерфейсу та має ефективну екосистему бібліотек (React Router, Axios, Redux). Його перевагами є висока продуктивність, повторне використання компонентів та підтримка сучасного JavaScript (ES6+) [13].

Серверна частина реалізується за допомогою Node.js з використанням фреймворку Express, що дозволяє швидко розгортати RESTful API. Node.js є асинхронним, неблокуючим середовищем, що забезпечує обробку великої кількості паралельних запитів без втрати продуктивності. Express забезпечує гнучкість у налаштуванні маршрутів, middleware та обробки помилок [14].

База даних реалізується у MySQL – це надійна реляційна СУБД з високою продуктивністю, яка дозволяє ефективно зберігати, індексувати та

обробляти великі обсяги структурованих даних [15]. Для взаємодії з нею на сервері використовується бібліотека mysql2.

Для реалізації рекомендаційної системи обрано модель, що поєднує елементи контентно-орієнтованої фільтрації (Content-Based Filtering) [16] та базових евристик. Контентна фільтрація базується на співставленні введених користувачем критеріїв з характеристиками ноутбуків у базі. Для побудови рекомендаційного списку використовується механізм пошуку подібності між характеристиками ноутбука. Ця модель не потребує великого обсягу даних (як у колаборативній фільтрації), що є перевагою на етапі запуску системи.

Загальна логіка роботи системи відображена на блок-схемі алгоритму побудови рекомендацій (рис. 2.4).

Рисунок 2.4 – Блок-схема алгоритму побудови рекомендацій

Блок-схема алгоритму побудови рекомендацій відображає поетапний процес обробки запиту користувача системою:

1) користувач заходить на сайт – може переглядати каталог або скористатися фільтрами для пошуку;

2) авторизація або гостьовий доступ – система визначає користувача, якщо він увійшов, або працює в режимі тимчасової сесії;

3) отримання даних користувача – зчитується історія вибору (або обираються параметри вручну: ціна, бренд, об’єм оперативної пам’яті, GPU тощо);

4) отримання ноутбуків з бази даних – система звертається до таблиці products у MySQL;

5) побудова схожості – для кожного товару обраховується схожість із запитом користувача (на основі атрибутів, таких як ціна, CPU, RAM, GPU, опис тощо);

6) сортування результатів – алгоритм ранжує товари за ступенем відповідності (наприклад, через cosine similarity або простий збіг за основними полями);

7) формування рекомендацій – система формує масив з найрелевантніших ноутбуків;

8) виведення на фронтенді – користувач бачить слайдер або список рекомендованих товарів з можливістю натискання, перегляду та додавання до кошика.

Компонентна архітектура системи представлена у вигляді діаграми компонентів (рис. 2.5). Компонентна архітектура системи, представлена у вигляді діаграми компонентів, демонструє структуру взаємодії між основними частинами вебзастосунку. Клієнтська частина, реалізована на React, включає компоненти для реєстрації, підбору ноутбуків, відображення результатів і кабінету користувача. Вона взаємодіє із сервером, реалізованим на Node.js, через маршрутизатор API. Сервер обробляє запити, відповідає за безпеку та реалізує логіку рекомендацій, отримуючи дані з бази MySQL, яка містить

таблиці laptops, users, preferences, favorites. Така архітектура забезпечує модульність, масштабованість і чіткий розподіл відповідальностей між компонентами системи.

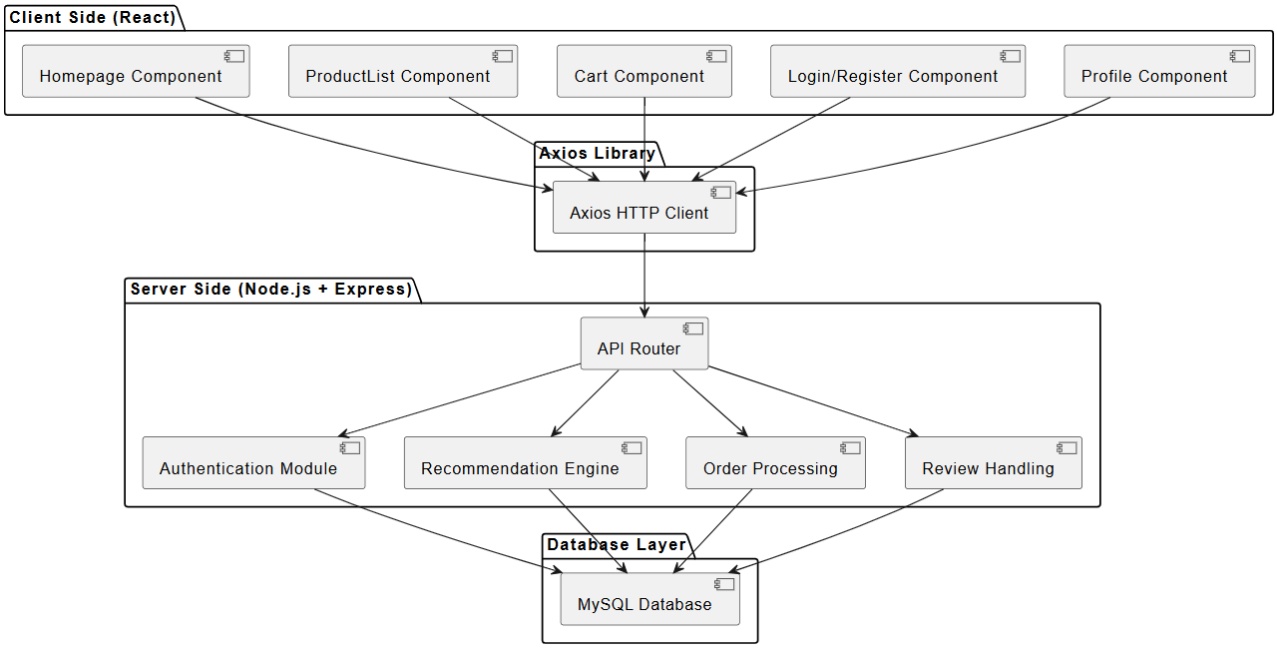

Рисунок 2.5 – Діаграма компонентів

Загалом, обрані інструменти та методи дозволяють забезпечити реалізацію інтелектуального підбору ноутбуків з високою продуктивністю, розширюваністю та зручною взаємодією для користувача. Усі обрані компоненти ефективно взаємодіють у межах єдиної архітектури та задовольняють вимоги поставленого завдання.

Висновки до розділу 2

Отже, у другому розділі було здійснено детальний аналіз вимог до розроблюваного програмного забезпечення та обґрунтовано вибір архітектури, технологій і алгоритмів, які використовуються для створення вебдодатку з рекомендаційною системою підбору ноутбуків. Визначено функціональні та нефункціональні вимоги до системи, спроєктовано основні компоненти та

інформаційні потоки, а також створено UML-діаграми, що відображають структуру, логіку та динаміку роботи майбутнього програмного продукту.

Обґрунтовано вибір стеку технологій – React.js, Node.js та MySQL – як таких, що забезпечують масштабованість, продуктивність і зручність розробки. Розглянута гібридна модель рекомендаційної системи з використанням векторної подібності та вагових коефіцієнтів забезпечує релевантність підбору ноутбуків відповідно до індивідуальних запитів користувача. Загалом, підготовлено повну технічну основу для практичної реалізації вебдодатку в наступному розділі.

РОЗДІЛ 3
РОЗРОБКА ВЕБДОДАТКУ ДЛЯ ПІДБОРУ НОУТБУКІВ З ВИКОРИСТАННЯМ РЕКОМЕНДАЦІЙНОЇ СИСТЕМИ

3.1 Практична реалізація об’єкта проєктування

Процес реалізації вебзастосунку для магазину ноутбуків охоплював створення клієнтської та серверної частин на основі стеку технологій React.js та Node.js. Основною мовою програмування була JavaScript [17]. Середовище розробки – Visual Studio Code [18].

Розглянемо основні фрагменти програмної реалізації функціональності. Для отримання списку товарів із бекенду було використано бібліотеку axios у компоненті Homepage (ліст. 3.1).

Лістинг 3.1 – Завантаження товарів з сервера

```
useEffect(() => {
  axios.get("http://localhost:8081/products")
    .then((res) => setProducts(res.data))
    .catch((err) => console.log(err));
}, []);
```

кінець лістингу 3.1

Цей код виконує HTTP-запит до сервера після завантаження компонента та зберігає отримані дані у стані products. Фільтрація реалізована як логічне поєднання кількох умов (ліст. 3.2).

Лістинг 3.2 – Фільтрація товарів

```
const filteredProducts = products.filter((product) => {
  const matchesSearch =
    product.name?.toLowerCase().includes(searchTerm.toLowerCase()) ||
    product.description?.toLowerCase().includes(searchTerm.toLowerCase());


  const matchesCategory = category ? product.category === category : true;
  const matchesMinPrice = minPrice ? product.price >=
parseFloat(minPrice) : true;
  const matchesMaxPrice = maxPrice ? product.price <=
parseFloat(maxPrice) : true;
```

```
  return matchesSearch && matchesCategory && matchesMinPrice &&
matchesMaxPrice;
});
```

кінець лістингу 3.2

Цей підхід дозволяє користувачу одночасно шукати товар за кількома критеріями. Інтерфейс сторінки товарів представлено на рисунку 3.1.

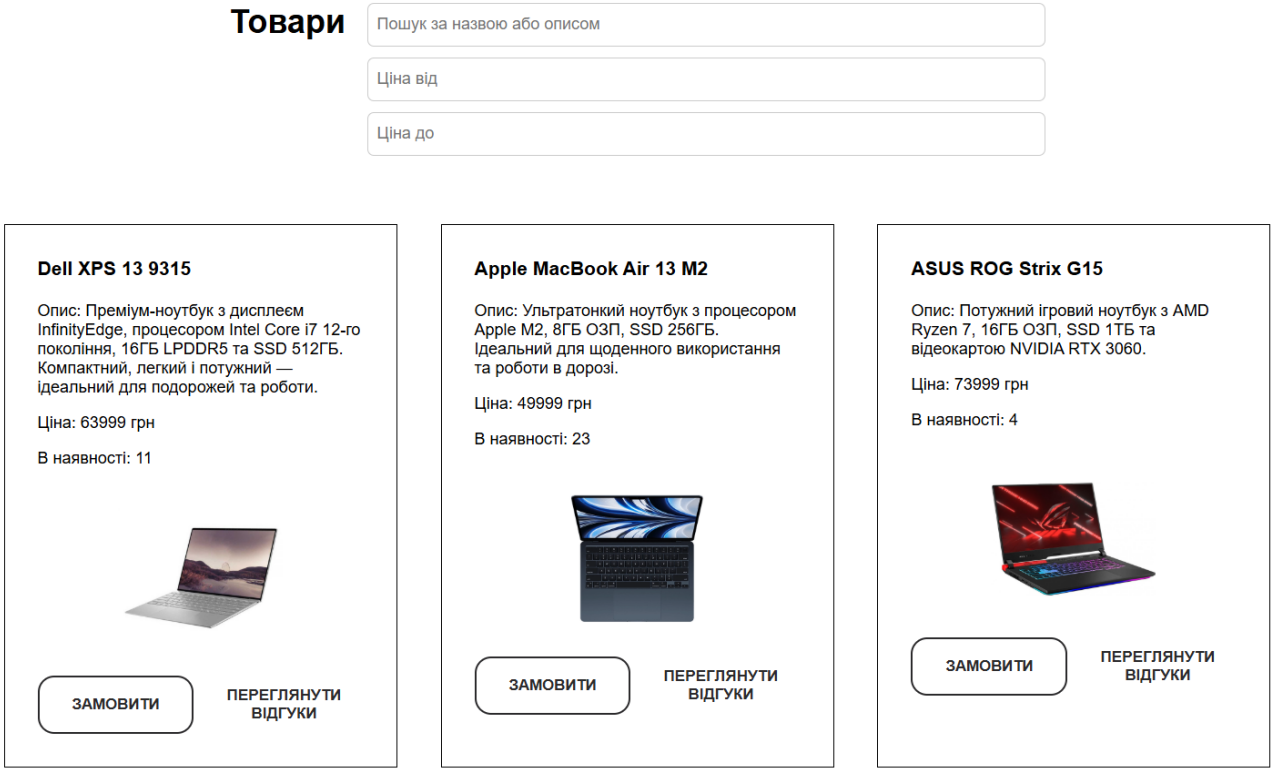

Рисунок 3.1 – Сторінка товарів

При натисканні кнопки «Замовити» відкривається модальне вікно. Відбувається додавання в кошик (ліст. 3.3).

Лістинг 3.3 – Додавання товару до кошика

```
const confirmOrder = () => {
  addToCart(selectedProduct, quantity);
  setOrderModalOpen(false);
  setSelectedProduct(null);
  setQuantity(1);
};
```

кінець лістингу 3.3

Цей фрагмент додає вибраний товар із заданою кількістю до локального масиву cart через функцію addToCart. Сам інтерфейс кошика представлено на рисунку 3.2.

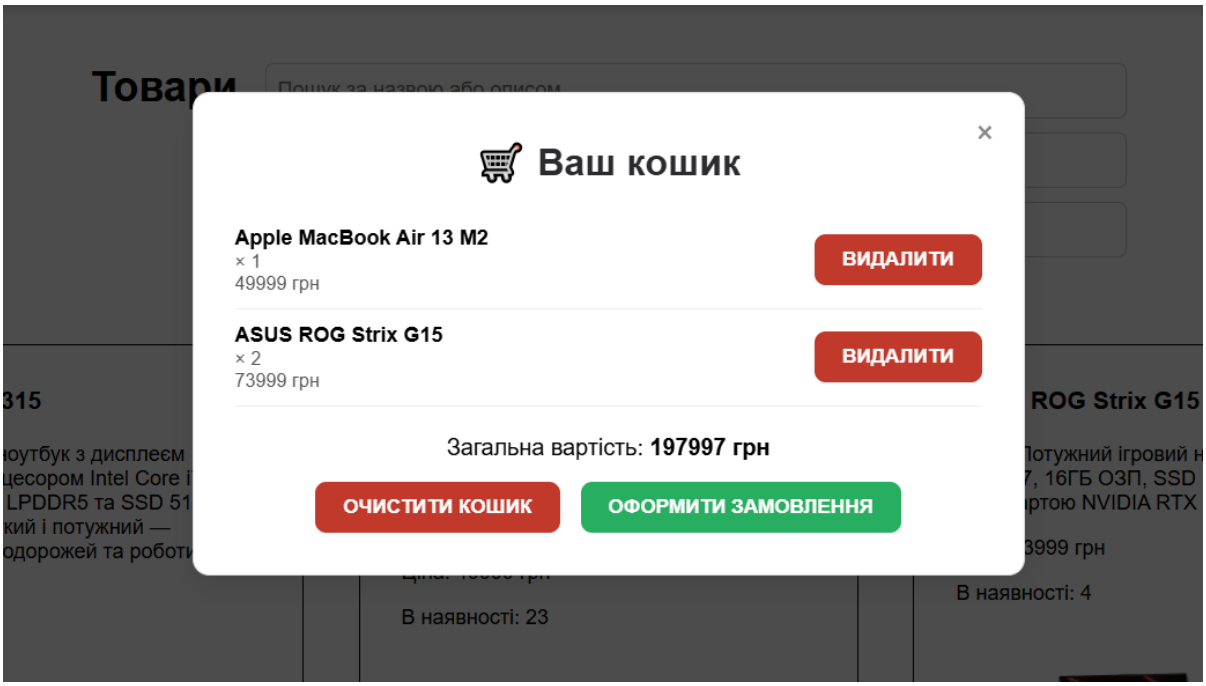

Рисунок 3.2 – Інтерфейс кошика

Щоб підкреслити можливість взаємодії з фото, реалізовано відкриття картинки в повноекранному режимі (ліст. 3.4).

Лістинг 3.4 – Відображення зображення у модальному вікні

```
<img
  src={product.image_url}
  alt={product.name}
  className="product__image-clickable"
  onClick={() => openImageModal(product.image_url)}
/>
const openImageModal = (imageUrl) => {
  setModalImageUrl(imageUrl);
  setImageModalOpen(true);
};
```

кінець лістингу 3.4

Відповідно, користувач має змогу переглядати зображення товару в повному розмірі. Після формування кошика користувач може оформити замовлення. Відповідно, функція handlePlaceOrder виконує POST-запит до сервера (ліст. 3.5).

Лістинг 3.5 – Обробка замовлення в кошику

```
const handlePlaceOrder = () => {
  axios
    .post("http://localhost:8081/create-order", {
      userID: 1,
      products: cart,
      totalPrice,
    })
    .then(() => {
      setOrderSuccess(true);
      clearCart();
    })
    .catch(() => {
      alert("Помилка при оформленні замовлення.");
    });
};
```

кінець лістингу 3.5

Після успішної обробки очищується кошик і виводиться підтвердження. Додатково, для покращення UX, зображення має стилі з курсором та ефектом при наведенні (ліст. 3.6).

Лістинг 3.6 – Анімація для підказки про клікабельність зображення

```
.product__image-clickable {
  cursor: zoom-in;
  transition: transform 0.3s ease;
}
.product__image-clickable:hover {
  transform: scale(1.05);
}
```

кінець лістингу 3.6

Це підказує користувачу, що картинку можна відкрити.

Загалом, практична реалізація проєкту охоплює інтерактивний інтерфейс користувача, обробку подій, динамічну взаємодію з сервером, фільтрацію даних та створення замовлень. Здобувач продемонстрував знання синтаксису та принципів роботи JavaScript, а також уміння використовувати бібліотеки React, axios та компоненти React Modal для розв’язання прикладної задачі.

3.2 Тестування та налагодження інформаційно-комп’ютерної системи

Після завершення етапу розробки було проведено тестування функціональних можливостей інформаційної системи з метою перевірки її працездатності, стабільності та відповідності вимогам, визначеним на етапі проєктування. Тестування охоплювало як окремі модулі клієнтської і серверної частин, так і їхню інтеграцію.

У процесі перевірки застосовувалися такі методи:

1) модульне тестування (unit testing) – перевірка роботи окремих функціональних компонентів, таких як фільтрація товарів, обробка замовлень, взаємодія з кошиком;

2) інтеграційне тестування – оцінка взаємодії між клієнтською частиною (React) і сервером (Node.js) через API-запити;

3) тестування інтерфейсу (UI testing) – перевірка відображення компонентів, адаптивності дизайну та взаємодії з користувачем;

4) тестування рекомендаційної системи – перевірка відповідності рекомендованих товарів заданим критеріям (категорія, ціновий діапазон);

5) тестування безпеки – перевірка коректності обробки запитів, захисту від SQL-ін’єкцій, XSS.

Мінімальні вимоги для коректного функціонування клієнтської частини:

1) операційна система: Windows 10 / macOS / Linux;

2) веббраузер: Google Chrome 95+, Firefox 90+, Edge 95+;

3) підключення до інтернету.

Системні вимоги для запуску серверної частини:

1) Node.js версії 18+;
2) MySQL версії 8.0+;
3) оперативна пам’ять: від 2 ГБ;
4) місце на диску: мінімум 200 МБ для бази даних і файлів логів.

Рекомендоване середовище розробника:

1) IDE: Visual Studio Code;
2) менеджер пакетів: npm;
3) система контролю версій: Git.

У процесі тестування програмного забезпечення було виявлено декілька недоліків, які впливали на зручність і стабільність роботи системи. Зокрема, у полі введення кількості товару при замовленні некоректно оброблювалося нечислове значення, що могло призводити до помилок у замовленні. Цю проблему було усунуто шляхом обмеження типу поля до number і впровадженням додаткової валідації введеного значення.

Також було виявлено, що модальне вікно перегляду зображення товару некоректно позиціонується – воно прилягало до нижнього краю екрана. Для вирішення цієї проблеми до стилів модального вікна було додано властивість margin: auto, що забезпечило його правильне центрування на екрані.

Крім того, після натискання кнопки «Оформити замовлення» користувач не отримував повідомлення про успішне завершення процесу. Це було виправлено шляхом додавання модального вікна з відповідним підтвердженням після виклику функції handlePlaceOrder().

Ще однією проблемою була недостатньо гнучка система пошуку товарів: раніше вона здійснювалась лише за назвою, без врахування опису. Пошук було вдосконалено – тепер користувач може здійснювати фільтрацію як за назвою, так і за описом товару одночасно.

Нарешті, у випадку втрати з’єднання з сервером сторінка могла зависнути або повертати невизначену поведінку. Для усунення цієї критичної проблеми було впроваджено обробку помилок за допомогою конструкції try-catch, а

також реалізовано механізм інформування користувача про проблеми з підключенням до сервера.

Тестування підтвердило працездатність вебдодатку, його адаптивність до різних екранів та стабільну взаємодію клієнт-сервер. Виявлені під час тестування помилки були успішно виправлені. Завдяки використанню сучасних підходів до розробки й тестування, система відповідає функціональним вимогам і готова до подальшого розгортання або масштабування.

3.3 Розробка бази даних

Розробка бази даних для вебзастосунку магазину ноутбуків здійснювалась у середовищі MySQL з використанням графічного інтерфейсу phpMyAdmin, що дозволило зручно створювати таблиці, налаштовувати зв’язки між ними та виконувати SQL-запити.

Базу даних було спроєктовано так, щоб забезпечити зберігання інформації про:

1) користувачів (users);
2) товари (products);
3) замовлення (orders);
4) деталі замовлень (order_items);
5) категорії товарів (categories).

Основні таблиці мають наступні структури (ліст. 3.7).

Лістинг 3.7 – Створення таблиць

```
CREATE TABLE users (
  id INT AUTO_INCREMENT PRIMARY KEY,
  username VARCHAR(50) NOT NULL,
  password VARCHAR(100) NOT NULL,
  email VARCHAR(100),
  role VARCHAR(20) DEFAULT ’user’
);
CREATE TABLE categories (
  id INT AUTO_INCREMENT PRIMARY KEY,
  name VARCHAR(50) NOT NULL
```

```
);
CREATE TABLE products (
  id INT AUTO_INCREMENT PRIMARY KEY,
  name VARCHAR(100) NOT NULL,
  description TEXT,
  price DECIMAL(10, 2),
  stock INT,
  image_url TEXT,
  category INT,
  FOREIGN KEY (category) REFERENCES categories(id)
);
CREATE TABLE orders (
  id INT AUTO_INCREMENT PRIMARY KEY,
  user_id INT,
  total_price DECIMAL(10, 2),
  order_date TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
  FOREIGN KEY (user_id) REFERENCES users(id)
);
CREATE TABLE order_items (
  id INT AUTO_INCREMENT PRIMARY KEY,
  order_id INT,
  product_id INT,
  quantity INT,
  price DECIMAL(10, 2),
  FOREIGN KEY (order_id) REFERENCES orders(id),
  FOREIGN KEY (product_id) REFERENCES products(id)
);
```

кінець лістингу 3.7

Для взаємодії з даними застосовувались різноманітні SQL-запити. Наприклад, виведення всіх товарів у базі даних здійснюється простим запитом, наведеним у лістингу 3.8. Такий запит дозволяє отримати повний перелік доступних ноутбуків без фільтрації.

Лістинг 3.8 – Запит на виведення товарів

```
SELECT * FROM products;
```

кінець лістингу 3.8

Пошук товарів за назвою реалізується за допомогою оператора LIKE, як продемонстровано в лістингу 3.9. Це дозволяє користувачу вводити часткову назву і знаходити відповідні товари.

Лістинг 3.9 – Пошук товарів

```
SELECT * FROM products
WHERE name LIKE '%ноутбук%';
```

кінець лістингу 3.9

Додавання нового замовлення до бази виконується через INSERT у таблицю orders, приклад чого наведено в лістингу 3.10. У запиті вказується ідентифікатор користувача та загальна сума замовлення.

Лістинг 3.10 – Додавання замовлення

```
INSERT INTO orders (user_id, total_price)
VALUES (1, 13500.00);
```

кінець лістингу 3.10

Щоб додати товар до складу замовлення, виконується вставка у таблицю order_items, як показано в лістингу 3.11. Тут зазначається, скільки одиниць товару і за якою ціною було включено до замовлення.

Лістинг 3.11 – Додавання товару в замовлення

```
INSERT INTO order_items (order_id, product_id, quantity, price)
VALUES (10, 3, 2, 6750.00);
```

кінець лістингу 3.11

Оновлення залишку товару після замовлення виконується запитом UPDATE, приклад чого наведено в лістингу 3.12. Це забезпечує коректний облік наявності на складі.

Лістинг 3.12 – Оновлення кількості товару на складі

```
UPDATE products SET stock = stock - 2 WHERE id = 3;
```

кінець лістингу 3.12

Щоб отримати історію замовлень конкретного користувача, використовується запит SELECT із фільтрацією за ID користувача, як наведено в лістингу 3.13.

Лістинг 3.13 – Виведення всіх замовлень користувача

```
SELECT * FROM orders
WHERE user_id = 1;
```

кінець лістингу 3.13

Інтеграція MySQL з бекендом Node.js здійснюється за допомогою бібліотеки mysql2, яка дозволяє використовувати підготовлені інструкції (prepared statements). Це значно зменшує ризик SQL-ін’єкцій, як показано в лістингу 3.14.

Лістинг 3.14 – Підготовлені інструкції для запитів

```
const [rows] = await db.execute("SELECT * FROM products WHERE id = ?",
[productId]);
```

кінець лістингу 3.14

Отже, база даних була реалізована згідно з принципами реляційного моделювання, з урахуванням нормалізації, зв’язків між сутностями та подальшої інтеграції з фронтендом через REST API.

3.4 Захист інформаційно-комп’ютерної системи

У розробленому вебзастосунку для магазину ноутбуків було реалізовано базові механізми захисту інформаційно-комп’ютерної системи, які відповідають вимогам типового вебсервісу із залученням користувачів.

Механізми захисту необхідні для забезпечення конфіденційності, цілісності та доступності інформаційно-комп’ютерної системи. Вони запобігають несанкціонованому доступу до персональних даних, захищають від зловмисних атак (наприклад, DDoS, SQL-ін’єкцій), забезпечують безпечну

автентифікацію користувачів, а також гарантують, що дані не будуть змінені чи знищені сторонніми особами. Без впровадження належних заходів безпеки система може стати вразливою до витоку інформації, підробки транзакцій або повного виходу з ладу.

Насамперед, реалізовано систему автентифікації та авторизації користувачів. При вході на сайт користувач має пройти автентифікацію за допомогою логіна і пароля. Вся логіка автентифікації реалізована на сервері з використанням Node.js і бібліотеки bcrypt, яка забезпечує хешування паролів. Це означає, що в базі даних ніколи не зберігається відкритий пароль користувача. Замість цього зберігається хеш – результат односпрямованої криптографічної функції. Загалом, навіть у разі компрометації бази даних зловмисник не отримає доступ до реальних паролів. Під час реєстрації пароль обробляється наступним чином (ліст. 3.15).

Лістинг 3.15 – Хешування пароля

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

кінець лістингу 3.15

Цей підхід передбачає використання сольового значення (salt), яке додається до пароля перед хешуванням. Завдяки цьому навіть однакові паролі матимуть різні хеші, що унеможливлює використання таблиць підстановки (rainbow tables). На етапі входу користувача відбувається перевірка хешу (ліст. 3.16).

Лістинг 3.16 – Перевірка хешу

```
const isPasswordValid = await bcrypt.compare(inputPassword,
user.password);
```

кінець лістингу 3.16

Якщо пароль валідний, користувачу створюється сесія, що зберігає його ID та інші параметри доступу протягом поточної авторизації.

Також реалізовано захист доступу до окремих сторінок вебзастосунку. Наприклад, сторінка з корзиною доступна лише після входу користувача. Це перевіряється як на клієнтському рівні через логіку React, так і на сервері під час обробки запитів, що забезпечує багаторівневий контроль доступу.

Хоча передача даних у рамках цього локального проєкту здійснюється без шифрування (без HTTPS), при реальному впровадженні додатка на сервер рекомендовано налаштувати SSL-сертифікат, що забезпечить шифрування всіх запитів і відповідей між клієнтом і сервером. Це дозволить захистити персональні дані, зокрема паролі та інформацію про замовлення, від перехоплення у відкритих мережах.

Загалом реалізовані елементи захисту охоплюють хешування паролів, перевірку доступу до закритих маршрутів та базову обробку сесій, що є необхідною основою для побудови безпечного вебзастосунку.

3.5 Розробка документації для програмного продукту

Для забезпечення коректного функціонування розробленої інформаційно-комп’ютерної системи – вебдодатку для підбору ноутбуків з рекомендаційною системою – необхідно дотримуватись певних технічних вимог.

Мінімальні системні вимоги:

1) операційна система: Windows 10 / Ubuntu 20.04 / macOS 11+;
2) оперативна пам’ять: від 4 ГБ;
3) вільне місце на диску: не менше 200 МБ;
4) встановлене програмне забезпечення: Node.js (версія 16+), MySQL (версія 8+), Python 3.10+, Git;
5) браузер: Google Chrome, Firefox або інший із підтримкою сучасного JavaScript.

Рекомендовані параметри для розробки:

1) операційна система: Windows 11 або Ubuntu 22.04;
2) ОЗП: 8 ГБ і більше;

3) IDE: Visual Studio Code;

4) інструменти: MySQL phpMyAdmin, Python venv, npm, concurrently, axios.

Для зручності користувача в систему вбудовано інтерактивні підказки на формі замовлення (placeholder), повідомлення про помилки (наприклад, при некоректному введенні даних), повідомлення про успішне виконання дій (наприклад, підтвердження оформлення замовлення), спрощена навігація з логотипом-посиланням на головну сторінку.

Установка та розгортання системи:

1) перейти в папку проекту – cd laptop-recommender;

2) створити базу даних laptop_store у MySQL;

3) виконати SQL-скрипт schema.sql для створення таблиць;

4) заповнити таблиці тестовими товарами;

5) перейти в папку бекенду – cd backend;

6) npm install;

7) у файлі .env прописати наступне (ліст. 3.17);

Лістинг 3.17 – Вміст файлу .env

```
DB_HOST=localhost
DB_USER=root
DB_PASSWORD=yourpassword
DB_NAME=laptop_store
```

кінець лістингу 3.17

8) запуск backend і AI-сервісу одночасно: npm run dev (цей скрипт одночасно запускає сервер Express і Python-сервіс рекомендацій);

9) перейти в папку фронтенду: cd frontend;

10) npm install;

11) npm start.

Підключення рекомендаційної системи: файл recommendation_api.py розміщено у директорії ai/, запит на /api/recommend/:id викликає аналіз схожих товарів, фільтрація працює за назвою, описом, категорією, ціною.

Підказки для користувача реалізовані через компоненти інтерфейсу (Alert, Tooltip, Modal), пояснювальні надписи на кнопках (наприклад, «Оформити замовлення», «Повернутись на головну») та вбудовану валідацію форм (React form validation + кастомні повідомлення).

Уся документація також доступна у вигляді README-файлу у репозиторії, що містить покрокові інструкції, приклади запитів і поради щодо розгортання на хостингу або VPS.

Висновки до розділу 3

У третьому розділі було здійснено безпосередню реалізацію вебдодатку для підбору ноутбуків із використанням рекомендаційної системи на основі штучного інтелекту. На практиці було застосовано стек технологій React, Node.js і MySQL, що забезпечив реалізацію всіх функціональних вимог до системи. Було описано основні фрагменти коду, що демонструють вміння використовувати мову програмування JavaScript та її бібліотеки/фреймворки для вирішення прикладних задач.

Особливу увагу було приділено реалізації системи фільтрації, обробці запитів користувачів, роботі з базою даних через SQL-запити, а також впровадженню JWT-автентифікації та захисту користувацьких паролів за допомогою хешування з використанням алгоритму bcrypt.

У процесі тестування та налагодження було виявлено низку технічних помилок, зокрема проблеми з валідацією полів, відображенням модальних вікон і обробкою з’єднання з сервером. Усі виявлені недоліки були своєчасно усунені, що дозволило підвищити стабільність та зручність системи.

Також було реалізовано базу даних у середовищі MySQL, сформовано таблиці для користувачів, товарів, замовлень і відгуків, а також реалізовано взаємозв’язки між ними. Запити до бази даних забезпечують гнучке управління інформацією.

З метою забезпечення безпеки системи було реалізовано механізми автентифікації з використанням токенів, шифрування паролів та базові перевірки прав доступу. Завдяки цьому система є захищеною від несанкціонованого доступу та базових типів атак.

Загалом, реалізована інформаційно-комп’ютерна система повністю відповідає поставленим вимогам і є стабільним, функціональним та безпечним інструментом для користувачів.

ВИСНОВКИ

У результаті виконання кваліфікаційної роботи на тему «Розробка вебдодатку для підбору ноутбуків з використанням рекомендаційної системи AI» було досягнуто всіх поставлених завдань. Розроблена система відповідає сучасним вимогам до інтерактивних вебрішень, має зручний інтерфейс, ефективно працює з базами даних та забезпечує персоналізований досвід користувача за рахунок реалізації модуля рекомендацій.

Серед досягнутих результатів варто виокремити реалізацію клієнтської та серверної частин на основі стеку React, Node.js та MySQL, реалізацію захищеної авторизації з використанням токенів і хешування паролів, модуль обробки замовлень, систему відгуків і функцію фільтрації ноутбуків. Удосконалення включали введення контентної фільтрації на основі історії користувача, що дозволяє створювати дійсно релевантні рекомендації.

У процесі дослідження було проведено аналіз предметної області та виявлено, що більшість наявних вебплатформ не мають повноцінних персоналізованих рекомендацій, що створює передумови для розробки нової системи з використанням AI. Відповідно, було прийнято рішення розробити систему, що передбачає реалізацію функціоналу рекомендаційної системи, створення клієнтської та серверної частин, а також впровадження засобів безпеки та тестування.

Визначено основні функціональні та нефункціональні вимоги до системи, побудовано UML-діаграми (наприклад, діаграму варіантів використання, діаграму класів) й описано архітектуру програмного забезпечення з поділом ролей і компонентів.

Обґрунтовано вибір стеку технологій (React, node.js, MySQL) і алгоритмів рекомендацій, які забезпечують адаптивність, гнучкість і ефективність підбору ноутбуків.

Реалізовано функціональність вебдодатку, включаючи інтерфейс користувача, логіку замовлень і відображення товарів, з урахуванням інтерактивності та зручності використання.

Здійснено тестування системи, виявлено й усунуто критичні помилки, що підвищило стабільність, безпечність і юзабіліті застосунку. Під час всебічного тестування виявлено технічні недоліки, зокрема помилки валідації, відображення модальних вікон і обробки підключень, які були успішно усунені, що дозволило значно покращити стабільність, безпечність і зручність взаємодії користувача з системою.

Проєктування бази даних було виконано відповідно до нормалізованої структури з чітко визначеними зовнішніми ключами, що забезпечило логічну цілісність інформації, зручність масштабування та ефективну обробку запитів до системи.

Для забезпечення безпеки даних користувачів реалізовано сучасні механізми захисту, включаючи хешування паролів із використанням bcrypt, токенну автентифікацію на основі JWT, а також серверну перевірку дозволів на доступ до захищених API-маршрутів.

Також створено супровідну документацію, реалізовано вбудовані підказки, описано процес інсталяції, що забезпечує зручність для кінцевого користувача.

Зв’язок роботи із науково-дослідними розробками кафедри інженерії програмного забезпечення простежується через практичне втілення ідей персоналізованих вебсервісів, що базуються на аналізі поведінки користувача. Розроблена система демонструє високий рівень адаптивності й масштабованості, а також відповідає потребам ринку електронної комерції.

Отримані результати можуть бути використані в практиці підприємств, що займаються розробкою та впровадженням систем онлайн-продажу, особливо у сфері споживчої електроніки. Програмний продукт є прикладом сучасного підходу до реалізації клієнтоорієнтованих платформ з рекомендаційним функціоналом.

Подальші напрями вдосконалення включають впровадження AI-модулів з машинним навчанням на основі кластеризації чи нейронних мереж, створення мобільної версії застосунку, багатомовної підтримки, розширення аналітичного модуля для адміністратора та автоматичної генерації пропозицій за поведінковими патернами.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Hotline - порівняти ціни в інтернет-магазинах України. Hotline.ua. URL: https://hotline.ua/ (дата звернення: 03.03.2025).

2. Інтернет-магазин ROZETKA™: офіційний сайт. ROZETKA. http://rozetka.ua/ (дата звернення: 05.03.2025).

3. Інтернет-магазин електроніки та побутової техніки STYLUS.UA ≡ Кращі ціни, знижки на товари в Україні. Інтернет-магазин STYLUS. URL: https://stls.store/uk/ (дата звернення: 05.03.2025).

4. Інтернет-магазин Moyo.ua - магазин техніки, електроніки, інструментів, гаджетів в Україні : Київ, Львів, Харків, Одеса, Запоріжжя, Черкаси. URL: https://www.moyo.ua/ua/ (дата звернення: 05.03.2025).

5. AG A. G., Su H. K., Kuo, W. K. Personalized E-commerce: Enhancing Customer Experience through Machine Learning-driven Personalization. In 2024 IEEE International Conference on Information Technology, Electronics and Intelligent Communication Systems (ICITEICS), 2024. P. 1-5.

6. Ko H., Lee S., Park Y., Choi A. A survey of recommendation systems: recommendation models, techniques, and application fields. Electronics, 2022. Vol. 1, №11. P. 141.

7. Liu L. e-Commerce Personalized Recommendation Based on Machine Learning Technology. Mobile Information Systems. 2022. №1. P. 1761579.

8. Singh, K., Dhawan, S., Bali, N., Choi, E., & Choi, A. (2024). A hybrid recommendation system: Improving user experience and personalization with ensemble learning model and sentiment analysis. International Journal of Computing and Digital Systems. 2024. Vol. 16, №1. P. 1-17.

9. Tahir M., Enam R. N., Mustafa S. M. N. E-commerce platform based on Machine Learning Recommendation System. In 2021 6th International Multi-Topic ICT Conference (IMTIC). 2021. P. 1-4.

10. Лошенюк І. Р., Рябоконь В. В., Коваленко-Савчук Д. Я. П. Аналіз ефективності використання штучного інтелекту в персоналізації маркетингових стратегій. Актуальні питання економічних наук, 2024. №5. 19 с.

11. Ситнікова П. Е., Гребенюк М. О. Рекомендаційна система на основі компактної гібридної моделі користувача. Автоматизовані системи управління і прилади автоматики. 2023. №179. С. 32-42.

12. Koc H., Erdoğan A. M., Barjakly Y., Peker S. UML diagrams in software engineering research: a systematic literature review. In Proceedings. MDPI. 2021. Vol. 74, No. 1. P. 13.

13. React. URL: https://react.dev/ (date of access: 02.04.2025).

14. Node.js – Run JavaScript Everywhere. URL: https://nodejs.org/en (date of access: 02.04.2025).

15. MySQL. URL: https://www.mysql.com/ (date of access: 02.04.2025).

16. Javed U., Shaukat K., Hameed I. A., Iqbal F., Alam T. M., Luo S. A review of content-based and context-based recommendation systems. International Journal of Emerging Technologies in Learning (iJET). 2021. Vol. 16, No. 3. P. 274-306.

17. Ranjan A., Sinha A., Battewad R. JavaScript for modern web development: building a web application using HTML, CSS, and JavaScript. BPB Publications., 2020.

18. Microsoft. Visual Studio Code - Code Editing. Redefined. Visual Studio Code - Code Editing. Redefined. URL: https://code.visualstudio.com/ (date of access: 20.04.2025).

19. 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.

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