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 ІНТЕРНЕТ-МАГАЗИНУ «КНИГОКОД»

DEVELOPMENT OF "KNYGOKOD" INTERNET STORE

спеціальність 121 Інженерія програмного забезпечення
(шифр і назва спеціальності)

освітня програма «Інженерія програмного забезпечення»
(назва освітньої програми)

Виконав: здобувач вищої освіти
групи ІПЗс-31
Назаровець Денис Павлович
_____
(підпис)

Керівник:
д.т.н., професор
Брежнєв Євген Віталійович
_____
(підпис)

Кваліфікаційну роботу
допущено до захисту
«8» 08 2024 р.
к.т.н., доцент
Гарант освітньої програми:
Ліщина Наталія Миколаївна
_____
(підпис)

Луцьк – 2024 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
_____ Н. Лицун
«2» 02 2024 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ ЗДОБУВАЧУ ВИЩОЇ ОСВІТИ

Назаровцю Денису Павловичу
(прізвище, ім'я, по батькові)

1. Тема кваліфікаційної роботи: *Розробка інтернет магазину «Книгокод»*

Керівник роботи: *д.т.н., професор, Брежнєв Євген Віталійович*
затверджені наказом закладу вищої освіти від «*30» грудня 2023 р.* № *477/01-02*
2. Строк подання здобувачем вищої освіти кваліфікаційної роботи *«5» червня 2024 р.*
3. Вихідні дані до роботи: *технічне та програмне забезпечення ЕОМ, вимоги до розробки програмного забезпечення, ергономічні вимоги до функціонування програмного засобу.*

4. Зміст розрахунково-пояснювальної записки (перелік питань, що потрібно розробити):
Аналіз сучасного стану проблеми, існуючих методів і засобів її розв'язання, аналіз, визначення вимог до розроблюваного програмного забезпечення та проектування програмного забезпечення, опис функціонального наповнення об'єкта проектування, практична реалізація об'єкта проектування.

5. Перелік графічного матеріалу: *18 рисунків; 4 лістинга; діаграми.*

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис	
		завдання видав	завдання прийняв
Аналіз предметної області	*Брежнєв Є. В.*		
Специфікації вимог до розробленої системи	*Брежнєв Є. В.*		
Розробка об'єкта проєктування	*Брежнєв Є. В.*		
Нормоконтроль	*Повстяна Ю. С.*		
Гарант ОП	*Ліщина Н. М.*		
Показник запозичень тексту		*12,1 %*	
Академічна доброчесність	*Ящук А. А.*		

7. Дата видачі завдання *«2» лютого 2024 р.*

КАЛЕНДАРНИЙ ПЛАН

№ З/п	Назва етапів кваліфікаційної роботи бакалавра	Строк виконання етапів роботи	*Примітка*
1	*Огляд літературних джерел по темі кваліфікаційної роботи бакалавра*	*07.02.2024 р.*	вик.
2	*Аналіз проблеми розробки та впровадження об'єкту проектування*	*27.02.2024 р.*	вик.
3	*Обґрунтування вибору шляхів, технологій (алгоритмів) і засобів вирішення поставленого завдання*	*12.03.2024 р.*	вик.
4	*Розробка функціонально-структурної схеми роботи об'єкта проектування та проектування бази даних*	*17.04.2024 р.*	вик.
5	*Практична реалізація об'єкта проектування та розробка бази даних*	*18.05.2024 р.*	вик.
6	*Тестування та налагодження об'єкта проектування*	*01.06.2024 р.*	вик.
7	*Здача чистового варіанту кваліфікаційної роботи бакалавра на кафедру*	*05.06.2024 р.*	вик.

Здобувач вищої освіти _____ (підпис) *Назаровець Д. П.* (прізвище, ініціали)

Керівник кваліфікаційної роботи _____ (підпис) *Брежнєв Є. В.* (прізвище, ініціали)

Міністерство освіти і науки України
Луцький національний технічний університет

Рецензія
на кваліфікаційну роботу

Здобувач вищої освіти: Назаровець Денис Павлович

Тема: «Розробка інтернет магазину Книгокод»

Коротка характеристика кваліфікаційної роботи: Робота бакалавра присвячена розробці інтернет магазину з використанням Laravel.

Самостійні розробки і пропозиції автора: Дослідження демонструє систематизацію методів створення онлайн-магазину на основі фреймворку Laravel. Крім того, проведено аналіз, який підтверджує результативність застосованих технологічних рішень та алгоритмічних підходів.

Практичне значення роботи: Методологія, розроблена в рамках цього дослідження, знайшла ефективне застосування в операційній діяльності онлайн-крамниці. Кваліфікаційна робота має практичне значення, апробована на міжнародній науково-практичній конференції.

Недоліки: Роботу можна було б покращити, якби в ній детальніше розглянули, як зробити дизайн великих онлайн-магазинів кращим та швидшим. Було б добре показати кілька прикладів, де це вдало зробили.

Загальний висновок: Робота виконана на високому науковому рівні, має логічну та послідовну структуру, відповідає встановленим вимогам, може бути представлена до захисту на державній екзаменаційній комісії та заслуговує позитивної оцінки.

Рецензент: Ященчишин Олександр Михайлович, к.т.н., доцент кафедри автоматизації та комп'ютерно-інтегрованих технологій
(прізвище, ім'я, по-батькові, посада)

Рецензент кваліфікаційної роботи _____
(підпис)

«1» 06 2024 р.

Міністерство освіти і науки України
Луцький національний технічний університет

Факультет комп'ютерних та інформаційних технологій
Кафедра інженерії програмного забезпечення

ВІДГУК
КЕРІВНИКА НА КВАЛІФІКАЦІЙНУ РОБОТУ БАКАЛАВРА
Здобувач вищої освіти *Назаровець Денис Павлович*
група *ІПЗс-31*

Тема кваліфікаційної роботи: *Розробка інтернет магазину «Книгокод»*
Керівник *Брежнєв Євген Віталійович*

Актуальність теми: Створення інтернет магазинів з використанням Laravel залишається важливим напр'ямком у галузі програмування. Хоча наявні розробки вже вирішили деякі завдання в цій сфері, все ще є місце для вдосконалення. Зокрема, потребують уваги такі аспекти, як поліпшення взаємодії з користувачами, розширення можливостей інтеграції та адаптація до різноманітних сфер послуг.

Об'єкт дослідження – це комплекс заходів з проектування та кінцевий продукт створення інтернет магазину.

Характеристика теоретичного рівня, наявності самостійних розробок і практичної значимості роботи: у рамках цієї роботи було розроблено ефективний інтернет-магазин на базі фреймворку Laravel, що демонструє високий теоретичний рівень та практичну значущість. Система відзначається зручним інтерфейсом управління контентом та каталогом товарів, що спрощує адміністрування платформи.

Зауваження та недоліки: істотних недоліків не виявлено.

Загальний висновок робота виконана на хорошому рівні та заслуговує оцінки «добре» за умови успішного захисту.

Керівник _____ (Брежнєв Є.В.)
(підпис) (прізвище, ім'я, по батькові)

«7» 06 2024 р.

АНОТАЦІЯ

Назаровець Д. П. Розробка інтернет магазину «Книгокод». Рукопис.

Кваліфікаційна робота бакалавра ОП «Інженерія програмного забезпечення». Луцький національний технічний університет. Луцьк, 2024.

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

У першому розділі здійснено аналіз сучасного стану проблеми розробки інтернет магазинів і сформульовано завдання. В другому розділі визначено вимоги до розроблюваної системи, а також засоби, методи і алгоритми розробки. У третьому розділі розроблено повністю функціонуючий інтернет магазин. У висновках узагальнено інформацію, відображену у попередніх частинах. Результати розробки демонструють можливості роботи інтернет магазину

Ключові слова: розробка, Laravel, PHP, діаграма, вимоги.

ABSTRACT

Nazarovets D. P. Development of the "Knygokod" internet store. Manuscript.

Bachelor's qualifying thesis of the educational program "Software Engineering". Lutsk National Technical University. Lutsk, 2024.

The bachelor's qualifying thesis consists of an introduction, three sections, conclusions and proposals, a list of used sources and annexes.

In the first chapter, an analysis of the current state of the problem of developing computer games was carried out and the task was formulated. The second section defines the requirements for the developed system, as well as means, methods and development algorithms. In the third chapter, internet store was developed and tested. The conclusions summarize the information presented in the previous parts. The development results demonstrate the possibilities of internet store

Keywords: development, Laravel, PHP, diagram, requirements.

ЗМІСТ

ВСТУП

Актуальність теми. Інтернет магазин книг, побудований на основі фреймворка Laravel. Цей фреймворк забезпечує високу продуктивність і масштабованість, що дозволить інтернет магазину легко адаптуватися до зростання кількості користувачів і розширення асортименту. Laravel також пропонує широкий набір інструментів і бібліотек, що спрощують процес розробки і забезпечують безпеку вебдодатку.

Метою кваліфікаційної роботи є створення вебсайту для надання послуг у сфері продажу книг. Проект передбачає розробку зручного зрозумілого інтерфейсу для користувачів, а також впровадження ефективної системи управління замовленнями.

Розробка вебсайту для продажу книг, що включає користувацький інтерфейс, систему управління контентом, функціонал для обробки замовлень.

Об’єкт дослідження – інтерфейс інтернет магазину, реалізований за допомогою фреймворку Laravel.

Предмет дослідження – структури даних, логіка обробки запитів, інструменти для роботи з базою даних, складові частини системи, контролери, міграції.

Завдання**:**

– аналіз вимог до інтернет магазину книг;

– проектування архітектури вебсайту та бази даних;

– розробка користувацького інтерфейсу з використанням сучасних вебтехнологій;

– реалізація функціоналу для обробки замовлень та управління кошиком покупок;

– тестування і оптимізація вебсайту для забезпечення високої продуктивності та зручності використання;

– розгортання проекту на хостингу.

РОЗДІЛ 1
АНАЛІЗ ІНТЕРНЕТ МАГАЗИНІВ КНИГ НА LARAVEL ПОСТАНОВКА ЗАВДАНЬ НА КВАЛІФІКАЦІЙНУ РОБОТУ

1.1 Аналіз сучасного стану проблеми

Сучасні компанії все частіше вирішують створювати вебдодатки, бо вони розуміють, наскільки важливо бути присутніми в інтернеті. Завдяки цьому, вони мають можливість використовувати новітні технології та залучати клієнтів через рекламу в мережі. У таких умовах, для конкурентоспроможності та успішного бізнесу, магазинам необхідно будувати інформаційні системи, які відповідають сучасним вимогам. Швидкі темпи розвитку Інтернету вимагають від компаній створювати вебсайти для надання різноманітних послуг.

Інтернет магазини заслуговують на особливу увагу. Незалежно від їх розміру, вони сприяють нашому комфорту, оскільки кожен із нас хоча б раз користувався послугами таких ресурсів.

Інтернет ресурси досить різноманітні. У компанії є сайти, які містять коротку інформацію про компанію та її послуги, а також великі онлайн-каталоги, які містять детальну інформацію про товари, зображення та ціни. Такі онлайн-каталоги, як правило, створюються для того, щоб користувачі могли знайти детальні описи та фотографії товарів.

Інтернет магазини використовують прямі маркетингові стратегії разом із традиційними магазинами. Інтернет магазини можуть пропонувати широкий асортимент товарів і послуг, а також надавати клієнтам широку інформацію, щоб допомогти їм прийняти рішення щодо покупки. Це одна з переваг інтернет магазинів порівняно зі звичайними магазинами.

Поєднання інтернет технологій і традиційної торгівлі є основною проблемою в створенні інтернет магазину. У звичайних магазинах покупці можуть візуально оцінити товари, щоб дізнатися про їхні характеристики та якість. Клієнт в онлайн-магазині не має такої можливості.

Попри це, візуальна інформація зазвичай достатня для правильного прийняття рішення. У сучасному світі онлайн-аудиторія стрімко зростає, а онлайн-продажі в містах зростають. Експерти прогнозують, що онлайн-продажі скоротяться. Інтернет магазини стають все більш поширеними завдяки зручності, яку вони пропонують клієнтам і дозволяють їм економити час і гроші. Інтернет магазини працюють цілодобово, а деякі товари можуть продаватися автоматично без участі продавця. Вам потрібно буде просто домовитися з постачальником і придбати необхідний продукт. У порівнянні зі звичайним магазином, інтернет магазин не обмежений кількістю товарів, які можна придбати. Це тому, що продукти можуть бути доставлені будь-яким способом, включаючи експрес-доставку.

Розгляньмо декілька інтернет магазинів, які спеціалізуються на схожих товарах і мають подібні цілі.Одним із таких вебсайтів є вебсайт магазину книг BOOKOVKA (рис. 1.1).

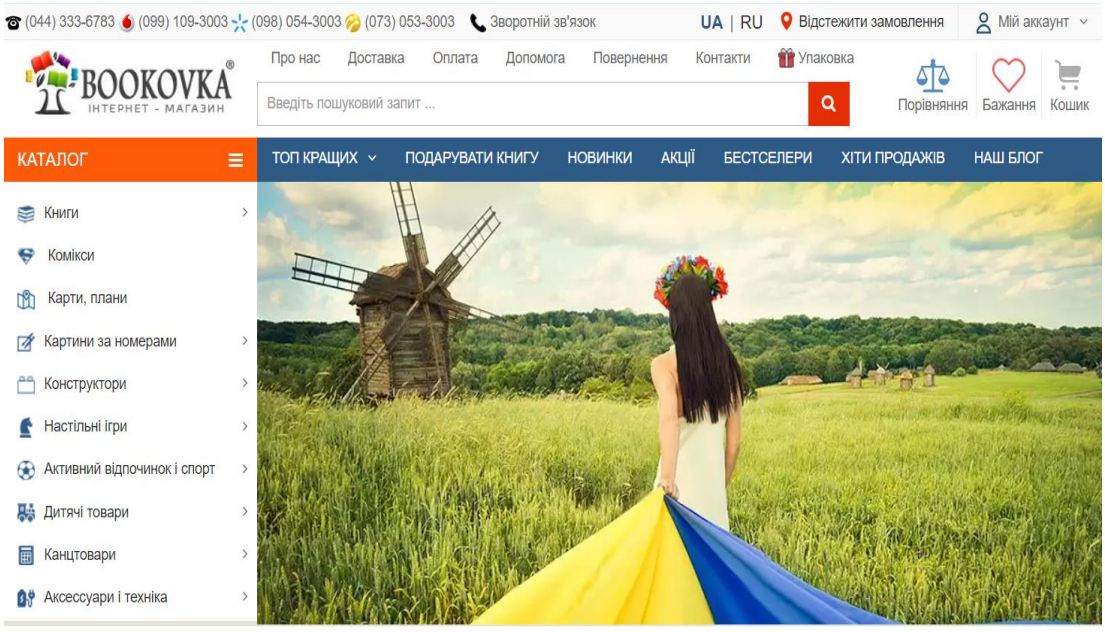

Рисунок 1.1 – Головна сторінка інтернет магазину BOOKOVKA [1]

Дизайн цього вебсайту потребує оновлення. Він виглядає застарілою і не відповідає сучасним стандартам. Можливо, зміна дизайну допоможе покращити користувацький досвід і зробить сайт більш привабливим для відвідувачів.

Головний недолік цього вебсайту полягає в тому, що при переході на його головну сторінку миттєво виникають помилки в консолі (рис. 1.2) розробника, яка відкривається за допомогою клавіші F12.

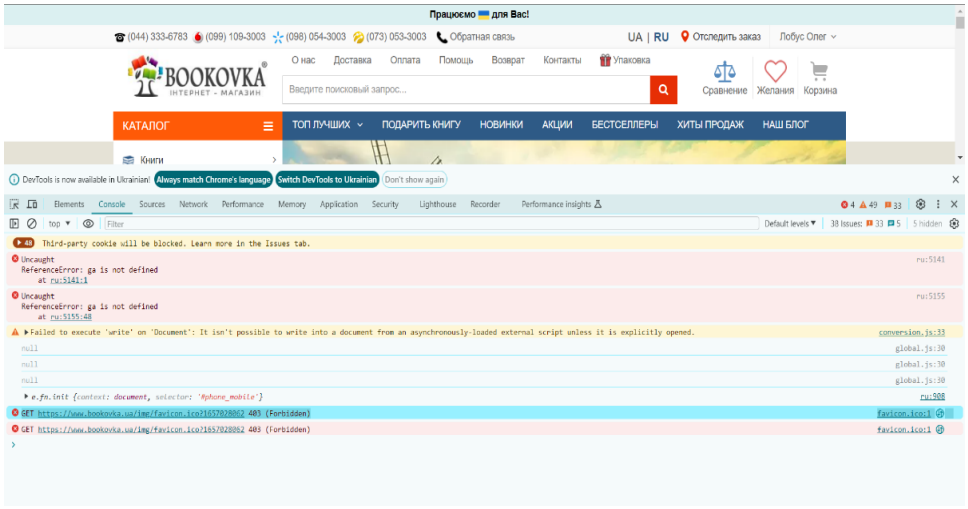

Рисунок 1.2 – Приклад помилок в консолі на головній сторінці BOOKOVKA

Це означає, що браузер намагається отримати файл «favicon.ico», який зазвичай видно поруч із URL-адресою або у вкладці браузера. Однак сервер отримує HTTP-код помилки 403, що означає, що доступ до цього ресурсу заборонений сервером. Це може бути результатом неправильної конфігурації сервера або відсутності дозволів, необхідних для отримання цього файлу.

Якщо виникає подібна проблема, перше, що потрібно зробити:

– перевірка наявності файлу. Переконайтеся, що файл favicon.ico дійсно існує на сервері;

– перевірка прав доступу. Переконайтеся, що налаштування прав доступу до файлу дозволяють серверу зчитувати файл favicon.ico. Права доступу повинні бути налаштовані таким чином, щоб сервер мав можливість читати файл;

– перезавантаження сервера. Іноді перезавантаження сервера може вирішити проблему, особливо якщо вона виникла після оновлення конфігурації або встановлення нових файлів.

Це основні кроки, які зазвичай допомагають у розв’язанні такої проблеми. Розглянемо наступний випадок, де ми розглянемо інтернет магазин книг РАНОК (рис. 1.3).

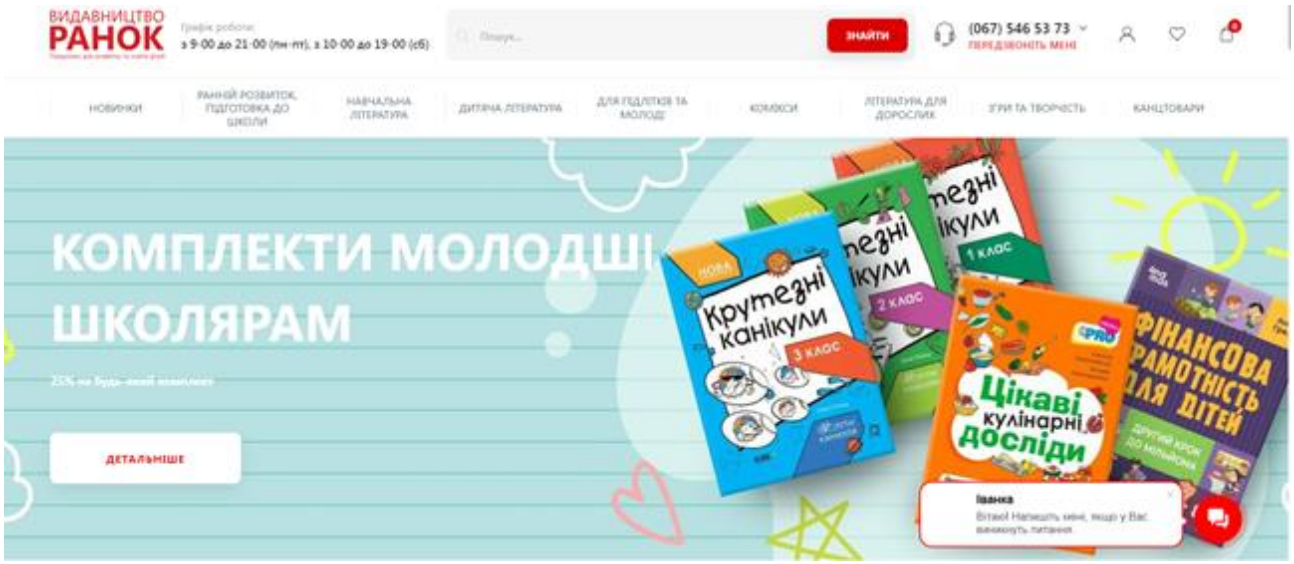

Рисунок 1.3 – Головна сторінка інтернет магазину РАНОК [2]

Зрозумілий, простий і простий інтерфейс сайту робить його зручним для користувачів будь-якого рівня. Крім того, чітка структура сторінок полегшує швидкий і простий пошук. Однак на цьому сайті є і недоліки, такі як проблеми з DNS: Проблеми з DNS можуть перешкодити вашому браузеру змінити доменне ім’я на IP-адресу, яку він повинен використовувати для завантаження ресурсу (рис. 1.4).

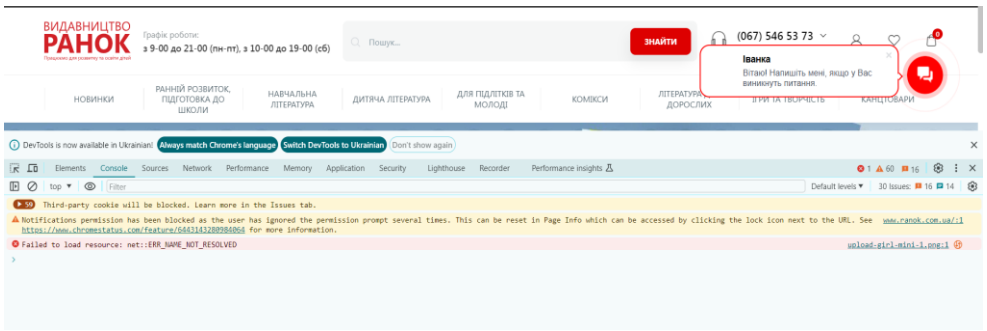

Рисунок 1.4 – Приклад помилки в консолі

Розглянемо ще один приклад інтернет магазину, на цей раз проінспектуємо сайт магазину книг КСД представлений (рис. 1.5).

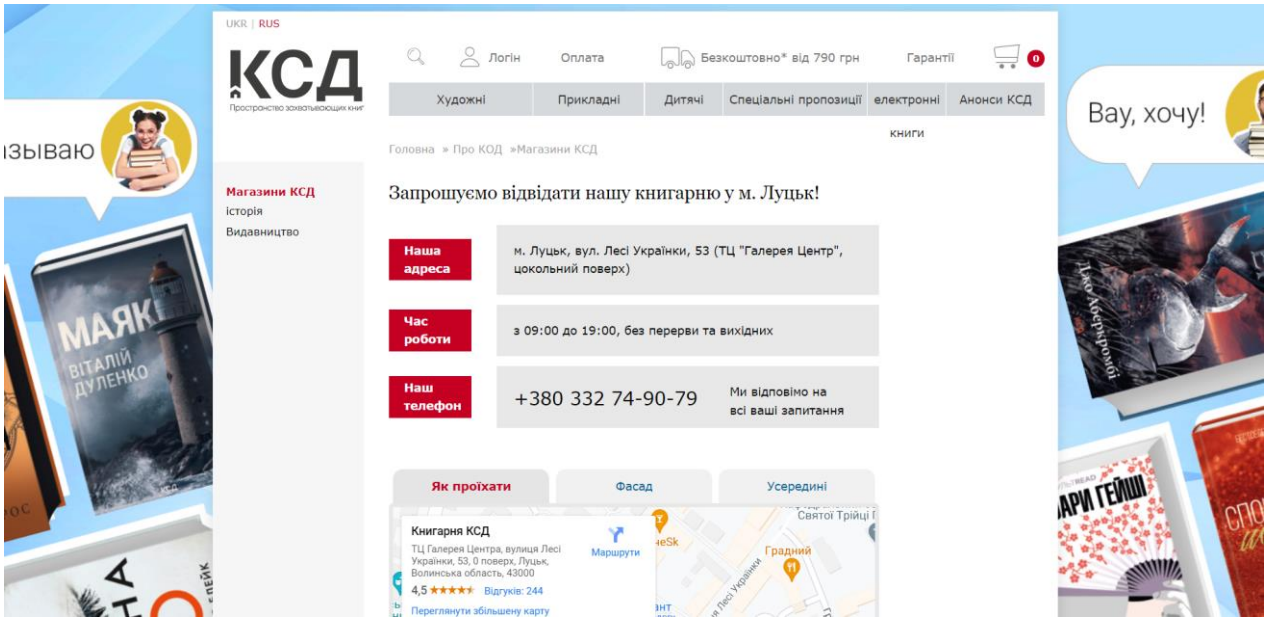

Рисунок 1.5 – Головна сторінка інтернет магазину КСД [3]

Необхідно відзначити, що сайт має застарілий дизайн, а також неефективне оформлення та його шапку, яка має обмежену інформативність. Відвідувачі можуть бути менш зацікавлені в послугах або товарах компанії через брак важливих даних і погану презентацію. Такі елементи можуть значно вплинути на досвід користувачів і загальний успіх вебсайту.

1.2 Постановка завдання на кваліфікаційну роботу бакалавра

Завдання**:**

– аналіз вимог до інтернет магазину книг;

– проектування архітектури вебсайту та бази даних;

– розробка користувацького інтерфейсу з використанням сучасних вебтехнологій;

– реалізація функціоналу для обробки замовлень та управління кошиком покупок;

– тестування і оптимізація вебсайту для забезпечення високої продуктивності та зручності використання;

– розгортання проекту на хостингу.

Магазин повинен мати зручну навігацію, щоб користувачі могли швидко знаходити потрібні книги. Важливо забезпечити можливість фільтрації та сортування книг за різними критеріями, такими як жанр, автор чи ціна. Процес покупки має бути максимально простим, включаючи швидке оформлення замовлення та зручні способи оплати.

Сайт повинен бути адаптований для різних пристроїв, включаючи комп’ютери, планшети та смартфони, щоб забезпечити зручність користування у будь-який час і з будь-якого місця.

Сайт повинен використовувати сучасні технології верстки, такі як HTML5, CSS3 та JavaScript, для забезпечення швидкого завантаження та плавної роботи [4]. Важливо також використовувати фреймворки, наприклад, Bootstrap або Foundation, для створення адаптивного та кроссбраузерного дизайну [5]. Сучасна верстка дозволяє легко додавати нові функції та оновлення, що забезпечить довготривалу актуальність сайту.

Висновки до розділу 1

Основною метою проєкту є створення інтернет магазину з продажу книг, який буде зручним і функціональним для користувачів. Важливо врахувати помилки, допущені сайтами конкурентів, щоб забезпечити швидку та якісну роботу сайту. Для цього слід підібрати відповідний хостинг, який забезпечить необхідні параметри. Проєкт також передбачає створення системи реєстрації та авторизації користувачів, можливості перегляду та додавання книг до кошика, оформлення замовлень та пошуку необхідних книжок. Це забезпечить зручний та швидкий процес покупок, задовольняючи потреби клієнтів.

РОЗДІЛ 2
СПЕЦИФІКАЦІЯ ВИМОГ ДО РОЗРОБЛЮВАНОЇ СИСТЕМИ

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

Для своєї роботи я обрав тип моделі Scram [6]. Скрам є фреймворком управління проєктами, який базується на тому, що команди, які працюють разом, постачають готові продукти у встановлені терміни, також відомі як спринти. Щоб успішно використовувати скрам, вам потрібно знати про його структуру.

Ось кілька причин чому було обрано саме цю модель для реалізації свого проекту:

– гнучкість та адаптивність. Ітераційний підхід Scrum дозволяє швидко реагувати на зміни вимог та пристосовуватись до мінливого середовища;

– тісна співпраця з замовником. Регулярні огляди спринтів забезпечують постійний зворотний зв’язок від замовника і можливість своєчасного коригування;

– підвищена продуктивність команди. Scrum заохочує самоорганізацію, крос-функціональність та щоденну комунікацію в команді;

– занній випуск робочого програмного забезпечення. Scrum зосереджений на регулярній поставці працюючої функціональності замовнику;

– зниження ризиків. Завдяки ітераціям та частим релізам ризики виявляються та мінімізуються на ранніх етапах;

– гнучке масштабування команди за рахунок розбиття на підгрупи;

– підвищення прозорості та контролю над прогресом через скрам-артефакти.

Беручи до уваги всі перелічені переваги та причини, прийнято обґрунтоване рішення застосувати методологію Scrum для реалізації роботи.

Після вибору методології Scrum для реалізації проекту, наступним важливим кроком є визначення та опис функціональних вимог для розроблюваного застосунку інтернет магазину книг, орієнтованих на клієнтів:

– перегляд каталогу книг з можливістю фільтрації, сортування та пошуку за різними параметрами;

– детальний перегляд інформації про окрему книгу, включаючи опис, відгуки та рейтинги;

– можливість додавання книг до «Списку бажань» або «Кошика» для подальшого оформлення замовлення;

– зручний процес оформлення замовлення з вибором способу оплати та доставки;

– інтеграція з платіжними системами для безпечної оплати замовлень;

– особистий кабінет користувача для перегляду історії замовлень та управління даними;

– можливість залишати відгуки та рейтинги для прочитаних книг;

– зручна система пошуку та навігації на сайті забезпечує легкий і швидкий доступ до потрібної інформації;

– адаптивний та зручний для користувача інтерфейс, який чудово працює на різних пристроях, забезпечуючи комфортне використання незалежно від розміру екрану;

Нефункціональні вимоги для клієнтської частини сайту:

– всі паролі користувачів повинні зберігатися в захешованому вигляді в базі даних для захисту від розголошення;

– сайт повинен швидко завантажуватися та забезпечувати високу продуктивність для забезпечення позитивного досвіду користувачів;

– запити до бази даних мають бути оптимізовані для мінімізації затримок та підвищення швидкодії системи;

– архітектура веб-сайту повинна бути спроектована з урахуванням можливості масштабування для забезпечення стабільної роботи при зростанні кількості користувачів;

– користувацький інтерфейс повинен бути адаптивним та зручним для використання на різних пристроях;

– сайт має відповідати стандартам доступності веб-контенту для забезпечення рівного доступу людей з інвалідністю;

– необхідно передбачити можливість інтеграції з різними платіжними системами та службами доставки для зручності клієнтів;

– система повинна забезпечувати конфіденційність та захист персональних даних користувачів відповідно до законодавства та політики конфіденційності.

Після опису функціональних вимог, для полегшення роботи при розробці слід відобразити описане на UML-діаграмі. У моєму випадку буде представлено UML-діаграму зв’язків класів, яка відображає функціонал додавання відгуків до товарів (рис. 2.1).

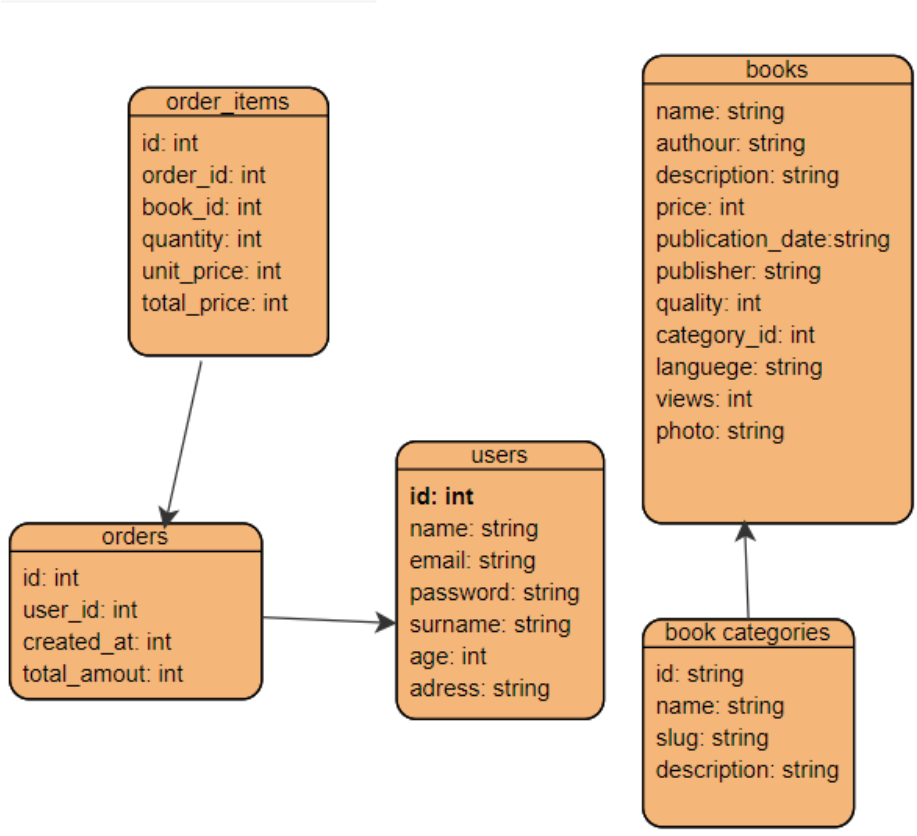

Рисунок 2.1 – Діаграма зв’язків у відображенні структури даних та взаємозв’язку між різними сутностями в системі

Діаграми поведінки описують кілька потоків, які формують дії в одній операції, щоб також можна було використовувати для інших цілей.

Діаграма діяльності містить стани діяльності, які показують, що робить оператор під час процедури або виконання дії під час робочого процесу. Замість того, щоб очікувати події, як це робить звичайний стан очікування, стан активності чекає, поки його обчислення завершиться.

Тепер переходим до проєктування дизайну інтерфейсу.

У цьому прикладі визначено необхідні функціональні елементи та компоненти, які мають бути присутніми на головній сторінці вебсайта:

– меню навігації повинні містити категорії товарів, посилання на головну сторінку, авторизація, персональний кабінет, кошик, категорії, номер телефону, фактичну адресу, пошук і графік роботи;

– головні банери;

– блок покупками за категоріями;

– блок з найкращими товарами;

– блок з інформаційними банерами;

– футер.

Під час проектування системи інтернет магазину книг була створена діаграма варіантів використання (рис. 2.2). Ця діаграма виявилась надзвичайно корисною на етапі розробки, оскільки надала чітке та зрозуміле загальне уявлення про те, яким чином веб-сайт повинен функціонувати та яка взаємодія відбуватиметься між клієнтами та основною бізнес-логікою системи. Загалом, існує величезна кількість різноманітних UML-діаграм, які можуть значно полегшити процес розробки програмного забезпечення. Це діаграми для опису станів об’єктів, діаграми для візуалізації функціонування бази даних, діаграми визначення прав доступу користувачів тощо. Перелік таких діаграм є досить великим. Кожна з них допомагає візуалізувати та краще зрозуміти певні аспекти проектованої системи, що спрощує процеси проектування, розробки та супроводу програмного забезпечення.

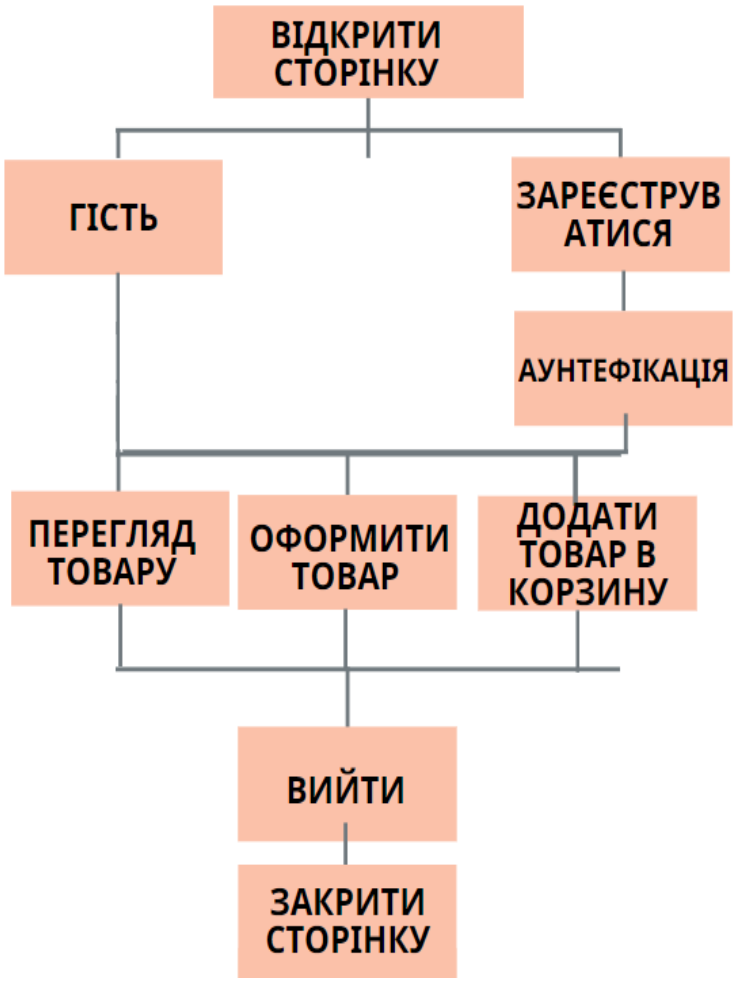

Рисунок 2.2 – Діаграма діяльності інтернет магазину

2.2 Вибір засобів, методів і алгоритмів вирішення поставленого завдання

Сучасна веброзробка являє собою широкий спектр завдань різного рівня складності. Від створення простих сайтів-візиток до реалізації масштабних бізнес-проектів з великою кількістю функціонала. Однак для кожного завдання існує безліч можливих способів та підходів до його вирішення. На сьогоднішній день розробникам доступна величезна кількість інструментів, фреймворків, бібліотек та технологій, які значно полегшують процеси веб-розробки та написання коду, підвищуючи продуктивність та ефективність роботи.

Завдяки стрімкому розвитку веб-технологій, активним дослідженням у цій сфері та величезним зусиллям спільноти розробників, з’являються нові передові рішення, покликані спростити та оптимізувати процеси веброзробки,

впроваджуючи найкращі практики та сучасні підходи до створення високоякісних, безпечних та масштабованих веб-додатків.

Будь-який сучасний вебзастосунок залежить від бази даних. Вона повинна гарантувати безпеку зберігання та доступу до інформації. У сучасному світі реляційні бази даних є однією з найпоширеніших типів баз даних. Для зберігання даних реляційні бази даних використовують таблиці, де кожен запис представлений рядком, а кожне поле даних представлено стовпцем. Вони базуються на теорії реляційних відношень, яка дозволяє використовувати мову структурованих запитів SQL для ефективного доступу та маніпулювання даними.

Багато переваг використання реляційних баз даних включають можливість використовувати складні запити для отримання необхідної інформації. Гарантувати, що дані залишаються цілісними за допомогою обмежень, зв’язків і транзакцій, і підтримувати нормалізацію даних, щоб запобігти дублюванню та неоднозначності.

Незважаючи на це, реляційні бази даних мають деякі недоліки. Наприклад, їх масштабованість обмежена через великий обсяг даних і їм важко моделювати складні зв’язки між даними. Таким чином, іноді використовують альтернативні типи баз даних, такі як NoSQL, оскільки вони пропонують більшу гнучкість і швидкодію для певних потреб застосування.

Було обрано базу даних MySQL [7]. Вона однією з найвідоміших відкритих реляційних баз даних. Ось кілька причин для вибору MySQL:

– MySQL вже багато років використовується мільйонами компаній та розробників. Вона довірена для зберігання великого обсягу даних у високонавантажених системах;

– MySQL є дуже швидкою базою даних, здатною обробляти великі обсяги запитів та транзакцій швидко та ефективно;

– MySQL підтримує багато різних типів даних, індексів, функцій та схем, що дозволяє вам створювати різноманітні та потужні бази даних для різних вимог;

– MySQL є відкритою програмою та має широку спільноту розробників. Це означає, що ви можете легко знайти допомогу, підтримку та розвинені інструменти для роботи з нею;

– MySQL легко інтегрується з іншими популярними технологіями, такими як PHP, Python, Java та багатьма іншими, що робить її привабливим вибором для веб-розробників;

– MySQL може бути легко масштабована, щоб ви змогли збільшувати її продуктивність зростанням вашого бізнесу та обсягу даних.

Загалом, MySQL є потужним та надійним рішенням для багатьох типів проектів, від малих веб-сайтів до великих корпоративних систем [8].

Перед початком розробки важливо чітко уявляти, як система повинна працювати. Для цього добре використовувати діаграми. Діаграми полегшують подальший процес розробки, оскільки програміст отримує чітке уявлення про елементи, які потрібно виконати.

Серед фреймфорків було обрано використовувати фреймворк Laravel. Laravel – це один з найпопулярніших PHP-фреймворків. На його основі створено багато українських сайтів, таких як:

– Rozetka один з найбільших онлайн-магазинів в Україні;

– forbes.ua досить популярне видання;

– Prom.ua платформа для онлайн-торгівлі, яка об’єднує безліч продавців;

– OLX популярний сайт оголошень;

– LUN.ua сервіс з пошуку нерухомості;

– Work.ua один з провідних сайтів з пошуку роботи.

Усі ці веб-ресурси працюють добре. Кожен день тисячі, якщо не десятки тисяч, користувачів з України та інших країн світу відвідують їх.

Laravel має багато переваг, ось деякі з них:

– ефективна система маршрутизації, яка спрощує роботу з URL-адресами та контролює доступ до різних сторінок додатку;

– зручна система шаблонів Blade, яка дозволяє розбивати вигляд додатку на компоненти та вбудовувати PHP-код безпосередньо в HTML;

– міграції баз даних, що дозволяють створювати, змінювати та видаляти таблиці та поля за допомогою коду, що спрощує розвиток проекту та спільну роботу над ним;

– велика та активна спільнота розробників, яка забезпечує багатий екосистему розширень та пакетів для Laravel;

– широкий набір вбудованих інструментів для роботи з сесіями, аутентифікацією, авторизацією та іншими аспектами безпеки;

– підтримка різних систем кешування, що допомагає підвищити продуктивність додатку;

– вбудована система тестування, яка спрощує процес написання та виконання автоматизованих тестів для перевірки роботи додатку;

– чудова документація, яка дозволяє швидко засвоїти фреймворк та знайти відповіді на багато питань під час розробки.

Використання фреймворка Laravel для створення backend-частини є розумним рішенням, оскільки він може значно скоротити час, необхідний для розробки вебсайтів з нуля. Laravel ідеально підходить для проектів будь-якої складності завдяки своїй простоті використання та гнучкості. Від простих вебсайтів до складних вебдодатків можна виконувати за допомогою його розширених функцій. Наявність великої кількості готових компонентів і бібліотек Laravel робить процес розробки простішим. Крім того, активна спільнота розробників забезпечує постійну підтримку та швидке реагування на проблеми. Враховуючи ці переваги, використання Laravel є правильним вибором для завершення проекту.

HTML, або «Мова розмітки гіпертекста», – це код, який використовується для створення вебсторінок та вебдодатків, доступних через інтернет. Ця мова дозволяє веброзробникам вказувати браузеру, як відображати елементи, такі як зображення, текст, форми та інтерактивні функції.

Веброзробники часто використовують HTML разом із каскадними таблицями стилів CSS та JavaScript – двома іншими мовами програмування, щоб

створювати вебсайти та додатки, доступні через інтернет браузери, такі як Chrome, Firefox, Safari та Edge [9].

У 1980 році Тім Бернерс-Лі, працюючи в Європейській організації ядерних досліджень, почав розробляти прототип HTML. До кінця десятиліття він створив універсальну мову розмітки, а також веббраузер і серверне програмне забезпечення.

CSS, або каскадні таблиці стилів – це мова, яка використовується для опису стилів, що повторно використовуються, для подання документів, написаних мовою розмітки [10]. Концепція CSS була створена 1994 році. У 1996 році консорціум розробив специфікацію для CSS, що дозволило веброзробникам змінювати макет і зовнішній вигляд своїх вебсторінок. Наприклад, CSS може використовуватися для зміни шрифту, розміру та кольору певного HTML-елемента. Один CSS-файл може бути пов’язаний з декількома сторінками, що дозволяє розробнику одночасно змінювати зовнішній вигляд усіх сторінок.

PHP – це скриптова мова програмування, створена для генерації HTML-сторінок на серверній стороні та роботи з базами даних. В даний час вона підтримується більшістю хостинг-провайдерів.

Препроцесор – це програма, яка здійснює попередню обробку даних для подальшого використання іншими програмами, такими як компілятори. Вихідні дані препроцесора називаються препроцесованими і призначені для подальшої обробки іншими програмами. Деякі з них можуть виконувати просту текстову підстановку, а інші – функціонувати на рівні скриптових мов програмування. Зазвичай препроцесори використовуються для обробки вихідного коду перед його компіляцією. Препроцесори використовуються в мовах програмування C/C++ і комп’ютерній верстці, значно розширюючи їхні можливості. Назва широко використовуваної скриптової мови програмування PHP є рекурсивним акронімом «PHP: Hypertext Preprocessor».

В області програмування для інтернету, PHP є однією з найпопулярніших скриптових мов завдяки своїй простоті, швидкості виконання, багатій функціональності та поширенню початкового коду на основі ліцензії PHP.

PHP вирізняється наявністю ядра та підключених модулів, які дозволяють працювати з базами даних, сокетами, графікою, криптографією, документами формату PDF тощо. Будь-хто може створити власне розширення та підключити його до системи. Існують сотні розширень, проте стандартна поставка містить лише кілька десятків тих, що добре зарекомендували себе.

Інтерпретатор PHP підключається до вебсерверу або через спеціальний модуль, або як CGI-додаток. Окрім цього, PHP може використовуватися для виконання адміністративних завдань в операційних системах UNIX, GNU/Linux, Microsoft Windows, Mac OS X і AmigaOS. Проте в цій ролі він менш популярний, поступаючись Perl, Python і VBScript.

Синтаксис PHP схожий на синтаксис мови C, а деякі елементи, такі як асоціативні масиви і цикл foreach, були запозичені з Perl. Сьогодні PHP використовується тисячами програмістів. Мільйони вебсайтів повідомляють про використання PHP, що становить понад п’яту частину доменів інтернету.

Команда розробників PHP складається з багатьох людей, які добровільно працюють над ядром і розширеннями PHP, а також над суміжними проектами, такими як PEAR або документація до мови.

Назва PHP є рекурсивною абревіатурою, яка означає «PHP: Hypertext Preprocessor». Спочатку PHP був створений як надбудова над Perl для полегшення розробки вебсторінок.

PHP зазвичай використовується як серверна мова, на відміну від мов, таких як JavaScript, які зазвичай виконуються на стороні клієнта. Програмування на стороні клієнта означає діяльність на вебсайті, яка відбувається локально на комп’ютері користувача через його веббраузер. Мови на стороні клієнта, такі як HTML, CSS і JavaScript, дозволяють веббраузерам інтерпретувати та змінювати контент на екрані комп’ютера. JavaScript є мовою сценаріїв, як і PHP, проте процеси, скриптовані на JavaScript, відбуваються на стороні клієнта, де браузер виконує інструкції. Сторона клієнта – це те, що користувач бачить під час взаємодії з інтернетом.

Laravel добре працює з архітектурою MVC, що дозволяє розділити бізнес-логіку та уявлення. Шаблон MVC Laravel містить багато вбудованих функцій, які забезпечують високий рівень захисту вебдодатків і підвищують продуктивність і зручність використання. Щоб захистити паролі користувачів, фреймворк Laravel використовує сильні методи шифрування. Використання структурованих операторів SQL допомагає запобігти атакам SQL-ін'єкцій [11].

Система міграції баз даних Laravel зменшує ймовірність втрати даних, дозволяючи змінювати структуру бази даних без її повного перебудовування. Крім того, ця функція дозволяє використовувати PHP-код для зміни структури бази даних, що є зручним і надійним методом [12].

Висновки до розділу 2

Створення backend – частини проекту за допомогою фреймворка Laravel є вигідним рішенням. Завдяки своїй простоті використання, гнучкості та великій кількості готових компонентів і бібліотек Laravel значно скорочує час розробки. Laravel пропонує постійну підтримку та швидке вирішення проблем завдяки активній спільноті розробників.

Причина популярності фреймворка Laravel полягає в тому, що він має вбудовані шаблони, які дозволяють створювати привабливі та інтерактивні макети. Створення сучасних і функціональних вебдодатків можливо за допомогою використання HTML, CSS та JavaScript разом із PHP. Laravel пропонує високу продуктивність, безпеку та зручність у розробці, добре інтегруючись з архітектурою MVC.

Система міграції баз даних Laravel полегшує зміну структури бази даних і запобігає втраті даних. Обидва MySQL і PostgreSQL мають переваги, але MySQL зазвичай обирають через його простоту та високу продуктивність.

Scrum – це хороший інструмент для управління проектами, особливо в поєднанні з Laravel. Scrum використовує спринти, артефакти та ролі для організації робочого процесу. Команди можуть ефективно досягати цілей і

гнучко реагувати на зміни завдяки використанню беклогу продукту та беклогу спринту. Інкременти гарантують постійний прогрес і дозволяють оцінювати результати роботи на кожному етапі.

Загалом, поєднання Laravel і Scrum забезпечує структурований підхід, високу якість і ефективне управління процесом розробки, що дозволяє створювати міцну основу для успішного управління проектами.

РОЗДІЛ 3
СТВОРЕННЯ ІНТЕРНЕТ МАГАЗИНУ НА LARAVEL

3.1 Налаштування середовища розробки

Для розробки нашого інтернет магазину необхідно завантажити та встановити PHP (рис. 3.1) і Composer (рис. 3.2). Composer є менеджером пакетів для PHP, який допомагає керувати залежностями та бібліотеками, необхідними для проєкту. PHP, як основна мова програмування для Laravel, забезпечить роботу всіх серверних функцій. Установивши ці інструменти, ми зможемо швидко і легко налаштувати середовище розробки та почати створювати наш інтернет магазин [13].

```
C:\Users\lenovo>php -v
PHP 8.0.30 (cli) (built: Sep  1 2023 14:15:38) ( ZTS Visual C++ 2019 x64 )
Copyright (c) The PHP Group
Zend Engine v4.0.30, Copyright (c) Zend Technologies
```

Рисунок 3.1 – Встановлений PHP

```
   ______
  / ____/___  ____ ___  ____  ____  ________  _____
 / /   / __ \/ __ `__ \/ __ \/ __ \/ ___/ _ \/ ___/
/ /___/ /_/ / / / / / / /_/ / /_/ (__  )  __/ /
\____/\____/_/ /_/ /_/ .___/\____/____/\___/_/
                    /_/
Composer version 2.7.4 2024-04-22 21:17:03

Usage:
  command [options] [arguments]

Options:
  -h, --help                     Display help for the given command. When no command is given display help for the list command
  -q, --quiet                    Do not output any message
  -V, --version                  Display this application version
      --ansi|--no-ansi           Force (or disable --no-ansi) ANSI output
  -n, --no-interaction           Do not ask any interactive question
      --profile                  Display timing and memory usage information
      --no-plugins               Whether to disable plugins.
      --no-scripts               Skips the execution of all scripts defined in composer.json file.
  -d, --working-dir=WORKING-DIR  If specified, use the given directory as working directory.
      --no-cache                 Prevent use of the cache
  -v|vv|vvv, --verbose           Increase the verbosity of messages: 1 for normal output, 2 for more verbose output and 3 for debug

Available commands:
  about                Shows a short information about Composer
  archive              Creates an archive of this composer package
  audit                Checks for security vulnerability advisories for installed packages
  browse               [home] Opens the package's repository URL or homepage in your browser
  bump                 Increases the lower limit of your composer.json requirements to the currently installed versions
  check-platform-reqs  Check that platform requirements are satisfied
  clear-cache          [clearcache|cc] Clears composer's internal package cache
  completion           Dump the shell completion script
  config               Sets config options
  create-project       Creates new project from a package into given directory
  depends              [why] Shows which packages cause the given package to be installed
  diagnose             Diagnoses the system to identify common errors
  dump-autoload        [dumpautoload] Dumps the autoloader
  exec                 Executes a vendored binary/script
  fund                 Discover how to help fund the maintenance of your dependencies
  global               Allows running commands in the global composer dir ($COMPOSER_HOME)
  help                 Display help for a command
  init                 Creates a basic composer.json file in current directory
  install              [i] Installs the project dependencies from the composer.lock file if present, or falls back on the composer.json
  licenses             Shows information about licenses of dependencies
  list                 List commands
  outdated             Shows a list of installed packages that have updates available, including their latest version
  prohibits            [why-not] Shows which packages prevent the given package from being installed
```

Рисунок 3.2 – Встановлений composer

Далі необхідно створити новий проект за допомогою фреймворку Laravel. Для цього ми використаємо команду (рис. 3.3) яка створює новий додаток Laravel.

```
composer create-project laravel/laravel example-app
```

Рисунок 3.3 – Команда для створення проєкта на Laravel

Ця команда працює над створенням нового проекту Laravel, який містить усі необхідні файли та структуру директорій. Тепер ми готові використовувати інструменти Laravel для створення та налаштування вашого інтернет магазину.

Після створення проекту Laravel необхідно відкрити середовище розробки та вибрати створений проект для подальшої роботи. Потім запуск проєкта через консоль, виконавши команду php artisan serve (рис. 3.4).

```
C:\xampp\htdocs\example-app
php artisan serve

  INFO  Server running on [http://127.0.0.1:8000].

  Press Ctrl+C to stop the server
```

Рисунок 3.4 – Запуск проєкта

3.2 Розробка основної функціональності

Початок написання коду передбачає розробку основної функціональності, а в моєму випадку – це наповнення товарів. Спершу необхідно описати міграцію, яка створить відповідну таблицю для продуктів у базі даних (лістинг 3.1). Щоб створити міграцію, я скористаюся командою php artisan make:model Book -m, яка автоматично створить модель та міграцію для неї. Після цього, для заповнення

таблиці бази даних даними, я буду використовувати клас Seeder. Це дозволить згенерувати початкові дані для таблиці продуктів.

Лістинг 3.1 – Демонстрація коду міграції таблиці

```
<?php

use Illuminate\Database\Migrations\Migration;
use Illuminate\Database\Schema\Blueprint;
use Illuminate\Support\Facades\Schema;

class CreateBooksTable extends Migration
{
    /**
     * Run the migrations.
     *
     * @return void
     */


    public function up()
    {
        Schema::create('books', function (Blueprint $table) {
            $table->id();
            $table->string('name');
            $table->string('author'); // Перевірте наявність цього рядка
            $table->text('description');
            $table->float('price');
            $table->date('publication_date');
            $table->string('publisher');
            $table->integer('quantity');
            $table->unsignedBigInteger('category_id');
            $table->foreign('category_id')->references('id')-
>on('book_categories_knigo_kod')->onDelete('cascade');
            $table->string('language');
            $table->integer('views');
            $table->string('photo')->nullable();
            $table->timestamps();
        });

    }

    /**
     * Reverse the migrations.
     *

     * @return void

     */
```

```
    public function down()
    {
        Schema::dropIfExists('books');
    }
}
```

Кінець лістингу 3.1

Після запуску міграцій ми заходимо до нашої бази даних, щоб переконатися, що таблиця була створена. Якщо таблиця відображається правильно, можна переходити до наступного етапу розробки. Перевірим міграції, щоб виправити потенційні проблеми, якщо виникнуть помилки (рис. 3.5).

Рисунок 3.5 – Створена таблиця книг

Після успішного створення таблиці в базі даних переходимо до створення контролера. Контролер в Laravel – це клас, який має методи для обробки різних HTTP-запитів та виконання бізнес-логіки додатку. За допомогою вбудованої команди Artisan php artisan make:controller BookController ми створюємо його. Потім можна визначити методи контролера для обробки різних запитів (GET, POST, PUT, DELETE) та виконання відповідних дій, таких як збереження, оновлення або видалення даних з бази.

Після розробки контролера ми переходимо до розробки моделі, яка відображає дані в базі даних і дозволяє виконувати різні операції з ними. Модель Laravel зазвичай відображає одну таблицю бази даних, але вона також може містити логіку для взаємодії з іншими таблицями. Після розробки моделі можна встановити процедури для зчитування, оновлення, видалення та створення даних відповідно до вимог додатку (лістинг 3.2).

Лістинг 3.2 – Код моделі Book

```
namespace App\Models;
use Illuminate\Database\Eloquent\Model;
use App\Models\Category; // Додайте імпорт класу Category

class Book extends Model
{
    protected $table = ‘books’;

    public function category()
    {
        return $this->belongsTo(BookCategory::class, ‘category_id’);
    }

    protected $fillable = [
        ‘name’,
        ‘author’,
        ‘description’,
        ‘price’,
        ‘publication_date’,
        ‘publisher’,
        ‘quantity’,
        ‘category_id’,
        ‘cover_image’,
        ‘language’,
        ‘views’,
        ‘photo’,
    ];
}
```

Кінець лістингу 3.2

3.3 Створення представлень і маршрутизації

Тепер створюєм файл Blade.php – це файл, який відповідає за відображення вебсторінок. Blade є шаблонним двигуном для Laravel, який дозволяє вбудовувати PHP код безпосередньо в HTML розмітку. Це полегшує процес розробки фронтенду, а також дозволяє створювати більш динамічні сторінки. Після того, як ми створимо файл.blade.php, ми можемо використовувати його для створення різних вебсторінок, які будуть відображатися користувачам нашого додатку (рис. 3.6).

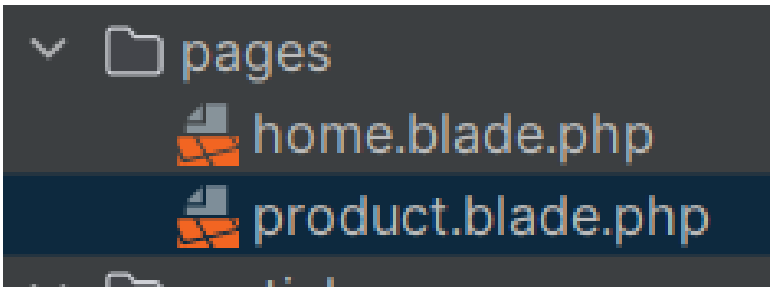

Рисунок 3.6 – Створення файла з продуктом інтернет магазина

Після розробки шаблону. Потрібно налаштувати маршрути в blade.php, щоб відобразити його контент на вебсторінці. У Laravel маршрути знаходяться в папці routes у файлі web.php. Можна використовувати функцію Route::get() для відображення сторінки за допомогою маршруту. У цьому сценарії ми вказуємо URL-шлях і встановлюємо контролер або функцію зворотного виклику, яка поверне вміст сторінки (рис. 3.7).

```
Route::get( uri: '/product', [ProductController::class, 'show']);
```

Рисунок 3.7 – Створений роута для сторінки з продуктами

При переході на сайт, система відображає головну сторінку (рис. 3.8). З головної сторінки можна взаємодіяти з різними категоріями, кошиком, обновками, тощо.

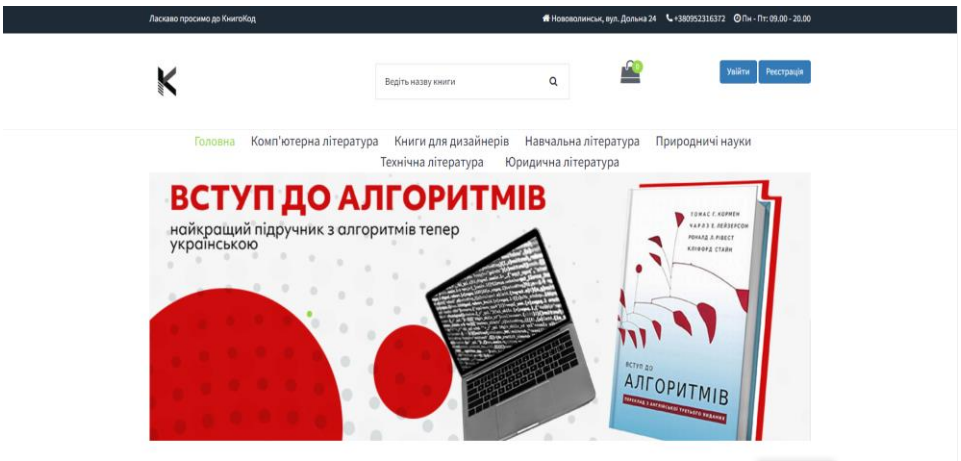

Рисунок 3.8 – Головна сторінка

Виберіть один із пунктів у меню навігаційної панелі, щоб переглянути товари відповідно до обраної категорії. Це дозволить вам швидко вивчити асортимент товарів і вибрати те, що вас цікавить. Зручна навігація покращить ваш досвід тадозволить вам краще використовувати час, щоб знайти те, що вам потрібно (рис. 3.9).

Рисунок 3.9 – Категорія комп’ютерна література

Ви можете переглянути картку товару, щоб дізнатися про його важливі характеристики, такі як ціна, автор, мова тощо, щоб допомогти вам зробити правильний вибір при покупці.

На картці товару також можуть бути фотографії товару, щоб ви могли побачити його з різних кутів і розглянути кожну деталь (рис. 3.10).

Ця функція дозволить вам отримати повну інформацію про товар перед покупкою, що робить ваше рішення більш обґрунтованим і впевненим.

Рисунок 3.10 – Картка товару

Кнопка «Перейти до оформлення замовлення» полегшує процес покупки для користувача, надаючи їм простий і швидкий доступ до оформлення замовлення. Після натискання цієї кнопки він буде перенаправлений на сторінку оформлення замовлення. Там він зможе заповнити всі необхідні дані щодо оплати та доставки. Зручно коригувати замовлення в реальному часі, змінюючи кількість товару, не виходячи з кошика (рис. 3.11). Автоматичний перерахунок загальної суми замовлення знижує ймовірність помилок і забезпечує більш точний процес покупки. Користувач може отримати повну картину витрат і зробити розумний вибір перед покупкою, отримавши інформацію про загальну суму покупки.

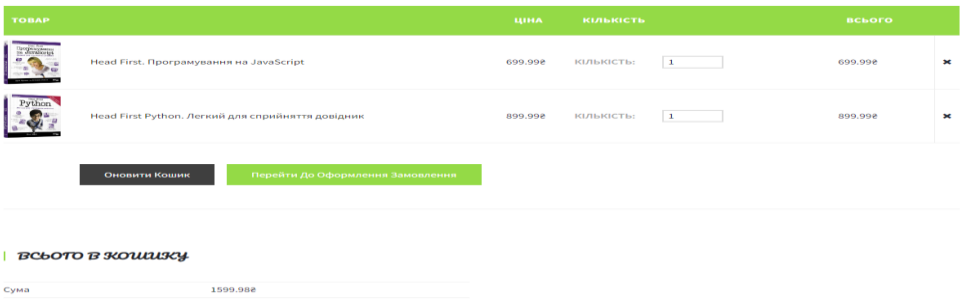

Рисунок 3.11 – Кошик

Напишемо базовий тест для успішної реєстрації. Реєстрація – це процес створення облікового запису користувача в системі або на вебсайті. Під час реєстрації користувачі зазвичай надають особисті дані, такі як ім’я, електронна адреса та пароль. Ці дані зазвичай використовуються для аутентифікації користувача при подальших входах в систему.

Процес реєстрації може включати валідацію введених даних, щоб забезпечити їхню коректність та безпеку. Після успішної реєстрації користувачі часто отримують підтвердження електронною поштою або повідомленням про успішну реєстрацію.

Реєстрація дає користувачам можливість отримати доступ до особистого кабінету, зберігати персоналізовані налаштування та виконувати інші дії, які можуть бути доступні лише для автентифікованих користувачів.

Деякі вебсайти або системи можуть також надавати можливість реєстрації через соціальні мережі, що спрощує процес для користувачів і дозволяє швидше отримати доступ до системи.

Зазвичай реєстрація є першим кроком для користувачів, які бажають отримати повний доступ до функціоналу вебсайту або системи.

Процеси автентифікації та реєстрації користувачів у системі обробляються контролером, який належить до простору імен додатку Http Controllers Auth. Для реєстрації нових користувачів і створення облікових записів він використовує вбудовані можливості Laravel.

Цей контролер може валідувати дані реєстрації та створювати нового користувача в системі на основі переданих даних. Крім того, він використовує мідлвар «гостя» для того, щоб переконатися, що користувач, який вже зареєструвався в системі, не може знову зареєструватися.

Крім того, цей контролер має метод показу форми реєстрації. Цей метод передає інформацію про категорії книг у вигляді змінної до відповідного представлення для відображення на сторінці реєстрації та відображає форму реєстрації користувача (лістинг 3.3).

Лістинг 3.3 – Код контролера Register

```
<?php
namespace App\Http\Controllers\Auth;
use App\Http\Controllers\Controller;
use App\Models\User;
use App\Models\BookCategory; // додайте це
use Illuminate\Foundation\Auth\RegistersUsers;
use Illuminate\Support\Facades\Hash;
use Illuminate\Support\Facades\Validator;
use Illuminate\Http\Request;

class RegisterController extends Controller
{
    use RegistersUsers;

    protected $redirectTo = ‘/home’;

    public function __construct()
    {
        $this->middleware(‘guest’);
    }

    protected function validator(array $data)
    {
        return Validator::make($data, [
            ‘name’ => [‘required’, ‘string’, ‘max:255’],
            ‘email’ => [‘required’, ‘string’, ‘email’, ‘max:255’,
‘unique:users’],
            ‘password’ => [‘required’, ‘string’, ‘min:8’, ‘confirmed’],
        ]);
    }

    protected function create(array $data)
    {
        return User::create([
            ‘name’ => $data[‘name’],
            ‘email’ => $data[‘email’],
            ‘password’ => Hash::make($data[‘password’]),
        ]);
    }

    public function showRegistrationForm()
    {
        $categories = BookCategory::all(); // отримати всі категорії
        return view(‘auth.register’, [‘categories’ => $categories]);
    }
}
```

Кінець лістингу 3.3

Завдяки взаємодії з моделлю BookCategory у цьому контролері можна отримати доступ до даних про категорії книг з бази даних.

При отриманні запиту на головну сторінку відповідь містить всі категорії книг, які доступні для відображення. Це викликає метод індекс().

Моделі BookCategory використовується для отримання даних про категорії книг з бази даних. Ця модель взаємодіє з таблицею категорій у базі даних.

Цей контролер виконує концепцію Model-View-Controller (MVC), де модель відповідає за роботу з даними, а представлення відповідає за відображення. Контролер також відповідає за обробку запитів і передачу даних між моделлю та представленням (лістинг 3.4).

Лістинг 3.4 – Код контролера Home

```
<?php

namespace App\Http\Controllers;
use App\Models\BookCategory;

use Illuminate\Http\Request;

class HomeController extends Controller
{
    public function index()
    {
        $categories = BookCategory::all();
        return view(‘pages.home’, [‘categories’ => $categories]);
```

Кінець лістингу 3.4

Шаблони можна використовувати для створення статичного фітера та хедера на вебсайті. У першу чергу можна створити файли хедера та футера, наприклад header.blade.php і footer.blade.php. Елементи футера та хедера, такі як навігаційні посилання, логотип і контактна інформація, розташовані в цих файлах відповідно.

Файли футера та хедера можна включити в кожну сторінку вебсайту, використовуючи директиву @include у файлі вигляду після створення.

Наприклад, для основного макету сторінок у файлі layout.blade.php можна додати рядки коду @include(‘header’) @include(‘footer’).

Це автоматично створить сторінку, включивши вміст файлів header.blade.php і footer.blade.php. Таким чином код футера та хедера можна зберігати в окремих файлах, що полегшує управління та редагування (рис. 3.11).

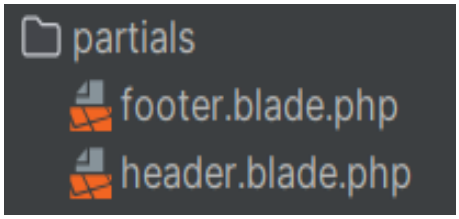

Рисунок 3.11 – Header, footer в окремій папці

Висновки до розділу 3

Успішно налаштовано середовище розробки з PHP та Composer.

Створено проект на базі фреймворку Laravel.

Реалізовано основну функціональність інтернет магазину, включаючи базу даних товарів, моделі та контролери.

Розроблено динамічні представлення за допомогою Blade.

Впроваджено систему маршрутизації для правильного відображення сторінок.

Створено зручний інтерфейс користувача з головною сторінкою, категоріями товарів та кошиком.

Реалізовано систему реєстрації та автентифікації користувачів.

Застосовано архітектуру MVC для чіткого розділення обов'язків.

Використано шаблони для статичних елементів сайту.

Проведено базове тестування функціональності.

ВИСНОВКИ

Розробка вебсайту книг з використанням фреймворка Laravel виявилася ефективним і доцільним підходом для створення масштабованого і продуктивного інтернет магазину. Виконані завдання дозволили досягти всіх поставлених цілей проекту, забезпечивши користувачам зручний і функціональний сервіс для купівлі книг.

Проведений аналіз дозволив чітко визначити потреби потенційних користувачів і ключові функціональні вимоги до системи. Це включало необхідність у зручному інтерфейсі для пошуку та покупки книг, можливість керування кошиком покупок, а також безпечної і ефективної системи обробки замовлень та платежів.

На основі вимог була розроблена архітектура вебсайту, яка забезпечує високу продуктивність і масштабованість. Проектування бази даних зосереджувалося на ефективному зберіганні та швидкому доступі до даних про книги, замовлення та користувачів.

Створений інтерфейс забезпечує інтуїтивно зрозумілу навігацію і приємний користувацький досвід. Використання таких технологій, як HTML, CSS, JavaScript та бібліотек типу Bootstrap, сприяло швидкій розробці адаптивного і сучасного дизайну.

Система управління замовленнями була успішно інтегрована з кошиком покупок, забезпечуючи користувачам можливість легко додавати книги до кошика, оформляти замовлення. Використання Laravel спростило реалізацію складної логіки для обробки замовлень, забезпечивши надійність і безпеку даних.

Тестування включало перевірку функціональності, безпеки та продуктивності вебсайту. Були виявлені і виправлені помилки, що сприяло підвищенню стабільності і швидкості роботи. Оптимізація коду та бази даних дозволила досягти високої продуктивності навіть при значних навантаженнях.

Проект був успішно розгорнутий на обраному хостингу. Було налаштовано резервне копіювання та моніторинг системи для забезпечення безперебійної роботи. Використання Laravel також сприяло швидкому і безпечному розгортанню вебсайту.

Загалом, створення інтернет магазину книг на основі фреймворка Laravel дозволило досягти всіх запланованих результатів, забезпечивши користувачам зручний та безпечний сервіс для покупки книг. Проект демонструє високу продуктивність і гнучкість, що дозволяє йому адаптуватися до зростання кількості користувачів і розширення асортименту книг.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. BOOKOVKA. URL: https://www.bookovka.ua/ (дата звернення: 9.04.2024).

2. РАНОК. URL: https://www.ranok.com.ua/ (дата звернення: 9.04.2024).

3. КСД. URL: https://bookclub.ua/ (дата звернення: 9.04.2024).

4. JavaScript. URL: https://developer.mozilla.org/ JavaScript_basics (дата звернення: 16.04.2024).

5. Bootstrap. URL: https://getbootstrap.com/docs/3.4/css/ (дата звернення: 19.04.2024).

6. Методика Scrum. URL: https://worksection.com/blog/scrum.html (дата звернення: 22.04.2024).

7. MySQL документація. URL: https://www.mysql.com (дата звернення: 01.05.2024).

8. MSQL основи. URL https://www.guru99.com/ru/mysql-tutorial.html (дата звернення: 24.05.2024).

9. Довідник HTML теги. URL: https://css.in.ua/html/tags (дата звернення: 13.05.2024).

10. CSS. URL: https://www.php.net/manual/ru/intro-whatis.php (дата звернення: 22.05.2024).

11. Laravel документація. URL: https://laravel.com/docs (дата звернення: 01.05.2024).

12. Авторизація з використанням Laravel. URL: https://laravel.com/docs/ 8.x/authorization (дата звернення: 24.05.2024).

13. Composer. URL: https://getcomposer.org/ (дата звернення: 01.06.2024).

ДОДАТКИ

Додаток А

Код Cart контролера який відповідає за логіку корзини

```
<?php

namespace App\Http\Controllers\API\Product;

use App\Http\Controllers\Controller;
use App\Http\Filters\ProductFilter;
use App\Http\Requests\API\Product\IndexRequest;
use App\Http\Resources\Product\ProductMinResource;
use App\Http\Resources\Product\ProductResource;
use App\Models\Product;
use Illuminate\Http\Request;
use Illuminate\Support\Facades\Log;
use Illuminate\Support\Facades\DB;

class ProductController extends Controller
{
    public function index(IndexRequest $request)
    {
        $data = $request->validated();

        $data['sortBy'] = $data['sortBy'] ?? 'price';
        $data['sortType'] = $data['sortType'] ?? 'asc';

        $filter = app()->make(ProductFilter::class, ['queryParams' =>
array_filter($data)]);
        $products = Product::filter($filter)->where('is_published',
true)->orderBy($data['sortBy'], $data['sortType'])->paginate(12, ['*'],
'page', $data['page']);

        return ProductMinResource::collection($products);
    }

    public function show(Product $product)
    {
        return new ProductResource($product);
    }

    public function getBestsellers()
    {
        $bestsellers = Product::where('is_published', true)
      ->where('is_bestseller', true)
            ->paginate(20);

        return ProductMinResource::collection($bestsellers);
    }

    public function getLatest()
```

```
    {
        $latest = Product::where('is_published', true)
            ->orderBy('created_at', 'desc')
            ->paginate(10);

        return ProductMinResource::collection($latest);
    }

    public function getMoreRated()
    {
        $topRatedProducts = Product::withCount('reviews')
            ->where('is_published', true)
            ->orderByDesc('reviews_count')
            ->take(5)
            ->get();



        return ProductMinResource::collection($topRatedProducts);
    }
}
```

Додаток Б
Kernel.php відповідає за централізоване керування HTTP-запитами

```
<?php

namespace App\Http;

use Illuminate\Foundation\Http\Kernel as HttpKernel;

class Kernel extends HttpKernel
{
    /**
     * The application’s global HTTP middleware stack.
     *
     * These middleware are run during every request to your application.
     *
     * @var array<int, class-string|string>
     */
    protected $middleware = [
        // \App\Http\Middleware\TrustHosts::class,
        \App\Http\Middleware\TrustProxies::class,
        \Illuminate\Http\Middleware\HandleCors::class,
        \App\Http\Middleware\PreventRequestsDuringMaintenance::class,
        \Illuminate\Foundation\Http\Middleware\ValidatePostSize::class,
        \App\Http\Middleware\TrimStrings::class,

\Illuminate\Foundation\Http\Middleware\ConvertEmptyStringsToNull::class,
    ];

    /**
     * The application’s route middleware groups.
     *
     * @var array<string, array<int, class-string|string>>
     */
    protected $middlewareGroups = [
        ‘web’ => [
            \App\Http\Middleware\EncryptCookies::class,

\Illuminate\Cookie\Middleware\AddQueuedCookiesToResponse::class,
            \Illuminate\Session\Middleware\StartSession::class,
            \Illuminate\View\Middleware\ShareErrorsFromSession::class,
            \App\Http\Middleware\VerifyCsrfToken::class,
            \Illuminate\Routing\Middleware\SubstituteBindings::class,
        ],

        ‘api’ => [
            //
\Laravel\Sanctum\Http\Middleware\EnsureFrontendRequestsAreStateful::class
,
            ‘throttle:api’,
```

```
            \Illuminate\Routing\Middleware\SubstituteBindings::class,
        ],
    ];

    /**
     * The application's route middleware.
     *
     * These middleware may be assigned to groups or used individually.
     *
     * @var array<string, class-string|string>
     */
    protected $routeMiddleware = [
        'auth' => \App\Http\Middleware\Authenticate::class,
        'auth.basic' =>
\Illuminate\Auth\Middleware\AuthenticateWithBasicAuth::class,
        'auth.session' =>
\Illuminate\Session\Middleware\AuthenticateSession::class,
        'cache.headers' =>
\Illuminate\Http\Middleware\SetCacheHeaders::class,
        'can' => \Illuminate\Auth\Middleware\Authorize::class,
        'guest' => \App\Http\Middleware\RedirectIfAuthenticated::class,
        'password.confirm' =>
\Illuminate\Auth\Middleware\RequirePassword::class,
        'signed' => \App\Http\Middleware\ValidateSignature::class,
        'throttle' =>
\Illuminate\Routing\Middleware\ThrottleRequests::class,
        'verified' =>
\Illuminate\Auth\Middleware\EnsureEmailIsVerified::class,
    ];
}
```