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 ОДЯГУ НА LARAVEL ТА VUE.JS

DEVELOPMENT OF THE INTERNET CLOTHING STORE ON LARAVEL AND VUE.JS

спеціальність 121 Інженерія програмного забезпечення
(шифр і назва спеціальності)

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

Виконав: здобувач вищої освіти
Групи ІПЗс-31
Кожуховський Михайло Миколайович
_____
(підпис)

Керівник:
к.т.н., доцент
Ліщина Наталія Миколаївна
_____
(підпис)

Кваліфікаційну роботу
допущено до захисту
«8» 06 2024 р.
к.т.н., доцент
Гарант освітньої програми:
Ліщина Наталія Миколаївна
_____
(підпис)

Луцьк – 2024 року

ЛУЦЬКИЙ НАЦІОНАЛЬНИЙ ТЕХГІЧНИЙ 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. Ліщина
«2» 02 2024 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ ЗДОБУВАЧУ ВИЩОЇ ОСВІТИ

Кожуховському Михайлу Миколайовичу
(прізвище, ім'я, по батькові)

1. Тема кваліфікаційної роботи: *«Розробка інтернет магазину одягу на Laravel та Vue.js»*

Керівник роботи: *к.т.н., доцент, Ліщина Наталія Миколаївна*
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: *16 рисунків; 10 лістингів; 3 діаграми.*

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис	
		завдання видав	завдання прийняв
Аналіз предметної області	*Ліщина Н. М.*		
Специфікації вимог до розробленої системи	*Ліщина Н. М.*		
Розробка об'єкта проєктування	*Ліщина Н. М.*		
Нормоконтроль	*Повстяна Ю. С.*		
Гарант ОП	*Ліщина Н. М.*		
Показник запозичень тексту	6,3 %		
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) *Кожуховський М М.* (прізвище, ініціали)

Керівник кваліфікаційної роботи _____ (підпис) *Ліщина Н. М.* (прізвище, ініціали)

Міністерство освіти і науки України
Луцький національний технічний університет

Рецензія
на кваліфікаційну роботу

Здобувач вищої освіти: Кожуховський Михайло Миколайович

Тема: «Розробка інтернет магазину одягу на Laravel та Vue.js»

Коротка характеристика кваліфікаційної роботи: Робота магістра присвячена розробці інтернет магазину з використанням Laravel та Vue, що забезпечує зменшення зусиль розробників та витрат часу на обслуговування проекту.

Самостійні розробки і пропозиції автора: У роботі узагальнено методи розробки інтернет магазину з використанням Laravel та Vue підходу, а також доведено ефективність запропонованих технологій та алгоритмів.

Практичне значення роботи: Запропоновану в роботі методику успішно застосовано для обслуговування роботи інтернет магазину. Кваліфікаційна робота має практичне значення, апробована на міжнародній науково-практичній конференції.

Недоліки: В роботі варто було б детальніше проаналізувати засоби оптимізації стилізації великих інтернет магазинів, навести приклади успішної реалізації.

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: Кабак В.В., завідувач кафедри УОТ
(прізвище, ім'я, по-батькові, посада)

Рецензент кваліфікаційної роботи _____
(підпис)

«07» 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 *Кожуховський Михайло Миколайович*
група *ІПЗс-31*

Тема кваліфікаційної роботи: *Розробка інтернет магазину одягу на Laravel та Vue.js*

Керівник *Ліщина Наталія Миколаївна*

Актуальність теми: Створення інтернет магазинів з використанням Laravel та Vue 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 та Vue, 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 «відмінно» за умови успішного захисту.

Керівник _____ (_____)
(підпис) (прізвище, ім'я, по батькові)

«7» 06 2024 р.

АНОТАЦІЯ

Кожуховський М. М. Розробка інтернет магазину одягу на Laravel та Vue.js. Рукопис.

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

Кваліфікаційна робота бакалавра складається зі вступу, трьох розділів, висновків, 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, Vue, діаграма, вимоги.

ABSTRACT

Kozhukhovskyi M. M. Development of clothing internet store on Laravel and Vue.js. 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 internet stores 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, Vue, diagram, requirements.

ЗМІСТ

ВСТУП

Актуальність теми. Інтернет магазини являються невід'ємною складовою сучасного життя. Кожного дня відбувається безліч успішних покупок користувачами з усього світу. Інтернет магазини це рішення яке полегшує та автоматизує роботу між продавцем та потенційним клієнтом.

Аби інтернет магазин був успішним слід враховувати наступні правила:

- розумно побудована архітектура сайту;
- приємний візуал сайту;
- висока швидкодія;
- тощо.

Виходячи з цих пунктів можна прийти до висновку що для реалізації такого продукту слід докласти чимало зусиль. Саме тому важливим етапом є підбір технологій, так званих інструментів, які і допоможуть реалізувати програмний продукт.

Програміст повинен бути досвідченим, обізнаним та орієнтуватися в обраних ним технологіях, аби швидко, якісно та вчасно закривати потреби клієнта.

Для зручності розробки слід використовувати uml діаграми різних видів. Вони допомагають краще проектувати систему, адже ми бачимо всі сутності прямо перед собою, на діаграмі.

Що стосується клієнта, то він в свою чергу повинен завжди чітко формулювати завдання, проявляти терпимість та бути лояльним. Дотримуючись цих простих правил інтернет магазин просто приречений на успіх.

Мета кваліфікаційної роботи – розробити сучасний та функціональний інтернет магазин.

Об'єкт дослідження – SPA (single age application) інтерфейс інтернет магазину.

Предмет дослідження – моделі даних, контролери, міграції, компоненти, API.

Завдання дослідження:

- провести аналіз предметної області;
- переглянути вже готові сайти конкурентів;
- обрати потрібні інструменти для розробки;
- налаштувати середовище;
- розробити адміністративну частину;
- розробити клієнтську частину;
- провести тестування розробленого функціоналу.

РОЗДІЛ 1
АНАЛІЗ ІНТЕРНЕТ МАГАЗИНІВ ОДЯГУ НА LARAVEL ТА VUE.JS І ПОСТАНОВКА ЗАВДАНЬ НА КВАЛІФІКАЦІЙНУ РОБОТУ

1.1 Аналіз сучасного стану проблеми

Веб-сайт – це комплекс інформації, організованої у вигляді сторінок та доступної через Інтернет. Саме таке визначення веб-сайту є загально прийнятим. Важко уявити наше життя без тих зручностей які нам дарують веб-сайти. Купа інформаційних ресурсів, книг, статей які в рази полегшують та пришвидшують розвиток та роботу людини в тій чи іншій сфері. Сайт це невід'ємна складова нашого повсякдення.

Особливу увагу хочеться приділити саме інтернет магазинам. Великі чи малі, не важливо, усі вони вони полегшують нам існування, адже кожен із нас бодай раз купував щось на одному з таких ресурсів.

Інтернет магазини полегшують життя не тільки звичайному клієнту, а й бізнесу(власнику) котрому він належить. Це повна автоматизація продажів без прямої взаємодії з клієнтом.

Проте що потрібно для успішного інтернет магазину. Чи ціна нижча ніж в конкурентів. Ні, левову частку відіграє:

- розумно побудована архітектура сайту;
- приємний візуал сайту;
- висока швидкодія;
- легкість в шляху від додавання в кошик і до повного оформлення замовлення;
- правильно налаштоване seo сайту;
- красива та зручна мобільна версія(адже більшість клієнтів відвідують сайти саме з телефону).

Про такі речі можна говорити ще довго…але не менш важливо і розуміти які причини можуть відштовхувати потенційних покупців:

- не сучасний або ж застарілий дизайн;

– відсутність кросбраузерності;
– погана оптимізація;
– постійні баги;
– велика навантаженість на сервер.

До усіх цих пунктів потрібно серйозно відноситись та постійно моніторити, аби при їх прояві швидко полагодити проблемні місця.

Розглянемо кілька прикладів інтернет магазинів та розберемо потенційні недоліки.

Сайт інтернет магазину одягу AGER представлений на рисунку 1.1.

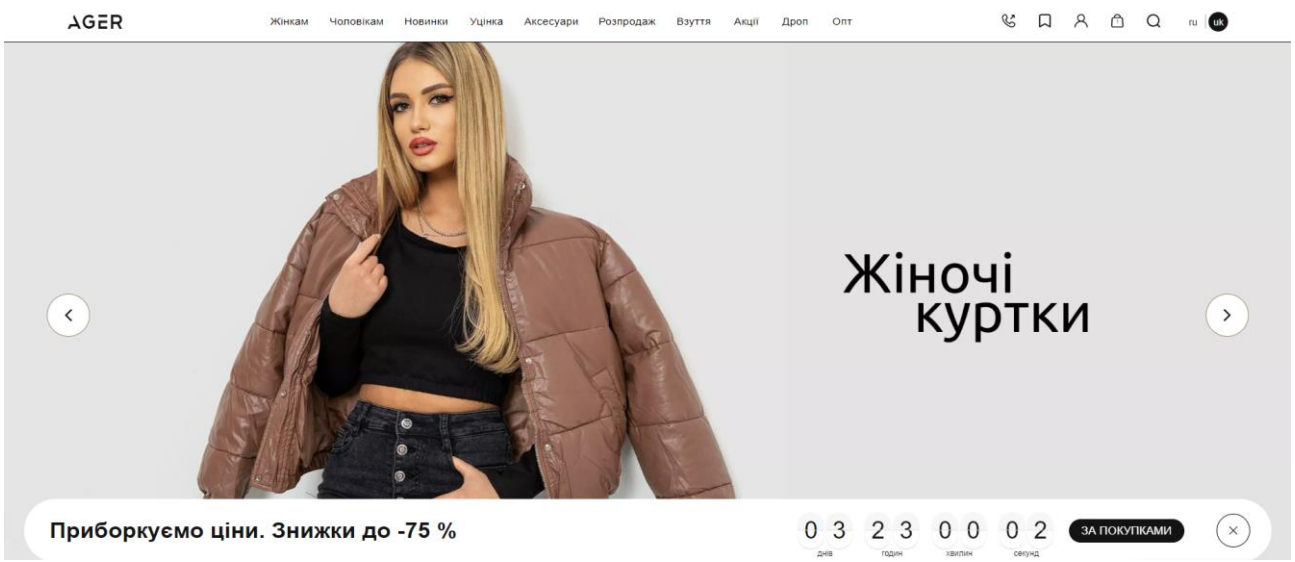

Рисунок 1.1 – Головна сторінка інтернет магазину одягу AGER [1]

Головною проблемою цього сайту є повільна швидкодія на мобільних пристроях, тест виконував за допомогою google speed test приклад цьому представлено на рисунку 1.2.

Швидкодія це важлива складова нормальної роботи сайту, від неї залежить чи залишиться клієнт на сайті продовжуючи пошук потрібних товарів, чи все ж таки піде на сайт конкурентів.

Довге завантаження сторінки може спонукати клієнта покинути цей сайт, незважаючи на всі інші його переваги.

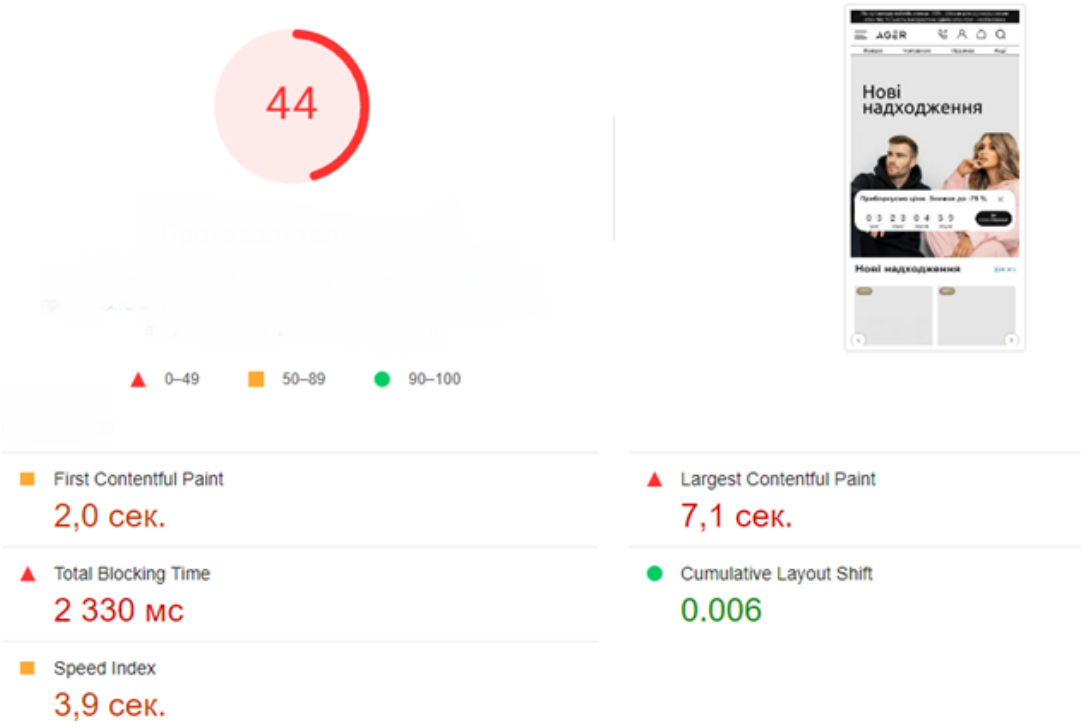

Рисунок 1.2 – Тест швидкодії головної сторінки інтернет магазину одягу AGER [2]

Що в першу чергу слід зробити при подібній проблемі:

– використовувати сучасні формати зображень;
– зменшити розмір javascript;
– використовувати мініфіковані файли;
– видалити css який не використовується;
– зменшити структуру DOM дерева;
– прибрати javascript який не використовується.

Це основні пункти яких зазвичай достатньо для вирішення цієї проблеми. Розглянемо наступний приклад, інтернет магазин одягу LUREX представлений на рисунку 1.3.

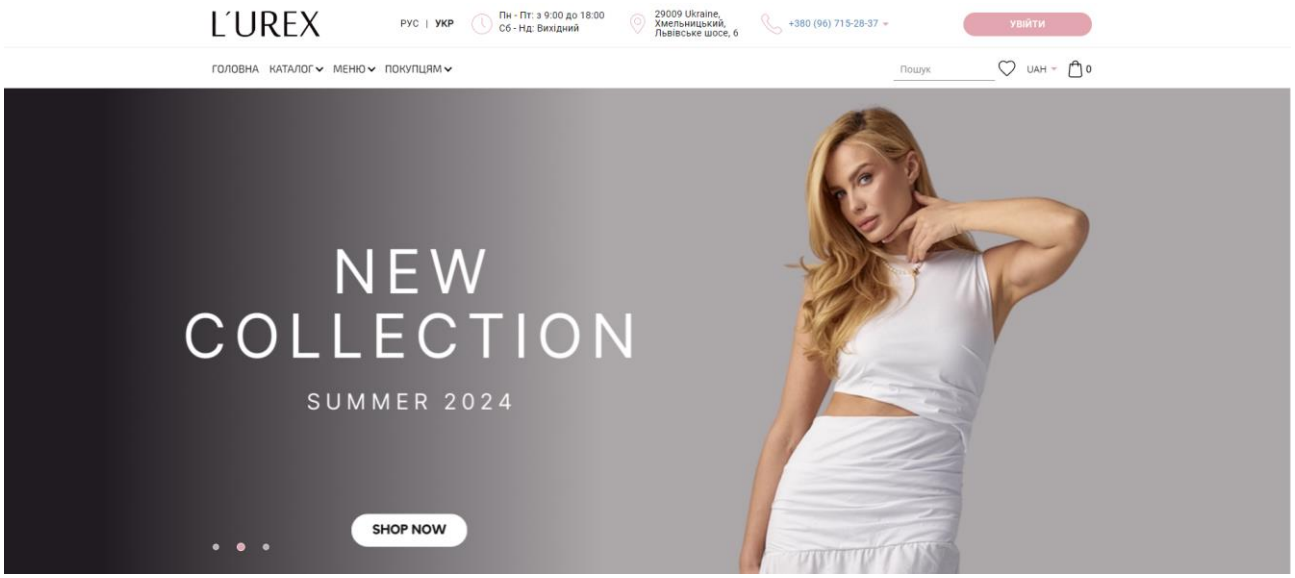

Рисунок 1.3 – Головна сторінка інтернет магазину одягу LUREX [3]

Головна проблема цього сайту полягає в тому, що як тільки ми перейшли на головну сторінку, в консоль розробника (відкривається за допомогою клавіші f12) одразу ж падають помилки, приклад на рисунку 1.4.

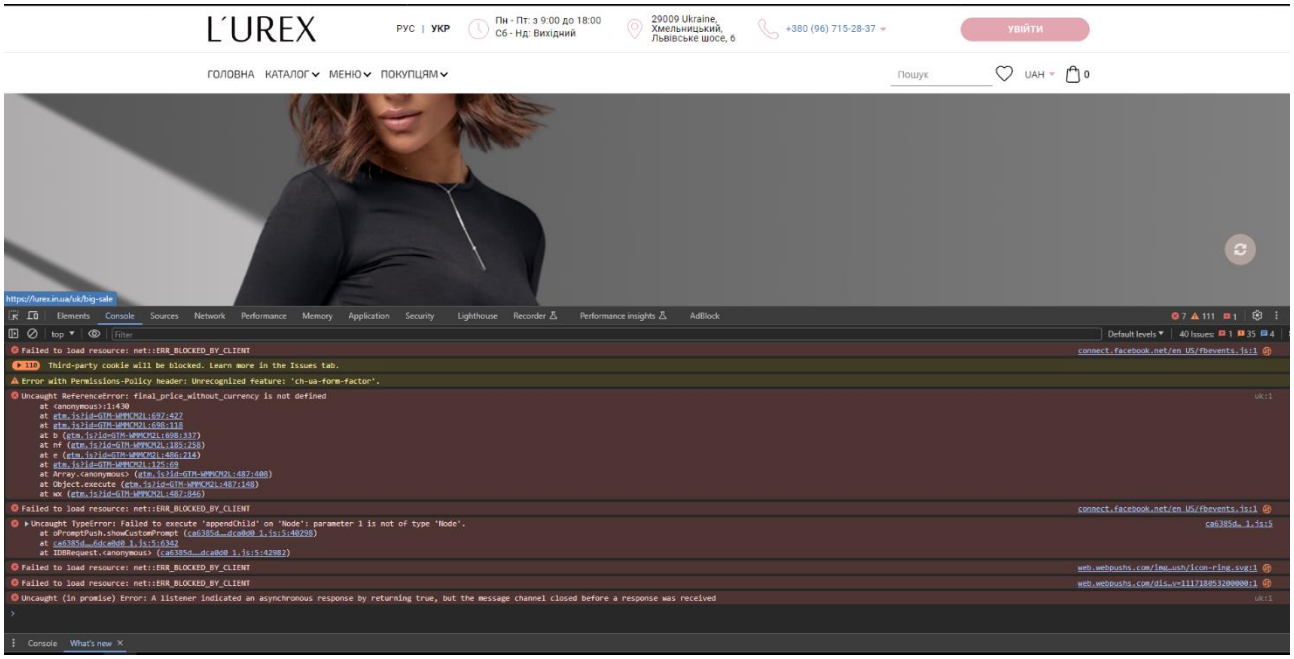

Рисунок 1.4 – Приклад помилок в консолі на головній сторінці магазину LUREX

Це не є добре, js помилки можуть припиняти подальше виконання коду і в найгіршому сценарії сторінка може завантажитись з поламаною верстkою або ж припинить роботу певний функціонал, до прикладу якась форма відправки даних за допомгою ajax. Розробнику завжди слід перевіряти консоль, реагувати і виправляти усі наявні помилки, це забезпечить стабільну та безперебійну роботу сайту.

Розглянемо ще один приклад інтернет магазину, на цей раз проінспектуємо сайт магазину одягу MII STYL представлений на рисунку 1.5.

Рисунок 1.5 – Головна сторінка інтернет магазину MII STYL [4]

Головною проблемо цього сайту є погано реалізована шапка сайту. Дуже багато різних дрібних елементів, написів, тощо. Інтуїтивно не зрозуміло куди натиснути аби відкрити каталог і зробити свою бажану покупку.

Шапки сайту мають бути зручними, без надмірних елементів, аби клієнт інтуїтивно розумів куди саме йому потрібно тиснути.

1.2 Постановка завдання на кваліфікаційну роботу бакалавра

Кваліфікаційна робота бакалавра розробляється для того аби клієнт міг зі зручністю обрати собі потрібний одяг та без проблем замовити його, а менеджери сайту мали змогу без проблем керувати усім в адміністративній панелі сайту.

Інтерфейс користувача – це безпосередньо сам сайт з яким взаємодіє клієнт. Інтерфейс користувача повинен бути красивим але і водночас простим, аби клієнт інтуїтивно розумів куди саме йому потрібно тиснути аби зробити бажану покупку чи підібрати більш потрібний товар.

Інтерфейс користувача повинен містити такий функціонал:

– каталог товарів;

– розумний фільтр який допоможе підібрати потрібний товар з переліку доступних;

– реєстрація та авторизація користувача;

– особистий кабінет користувача зі списком замовлень та інформацією про самого ж користувача;

– кошик;

– відгуки до товарів;

– список бажаних товарів;

– посторінкова навігація.

Адміністративна частина сайту дозволятиме менеджеру управляти контентним наповненням сайту:

– додавати товари, редагувати опис товарів, проставляти товарам відповідні фото, ціни тощо;

– керувати категоріями товарів;

– керувати групами товарів;

– перевіряти відгуки на товар та лишати відповіді;

– переглядати історію замовлень.

Для того аби досягти поставленої мети потрібно:

- провести аналіз предметної області;
- переглянути вже готові сайти конкурентів;
- обрати потрібні інструменти для розробки;
- налаштувати середовище;
- розробити адміністративну частину;
- розробити клієнтську частину;
- провести тестування розробленого функціоналу.

Висновки до розділу 1

Основною метою проекту є розробка інтернет магазину з продажу одягу. Під час розробки повинні бути враховані помилки допущені сайтами конкурентів. Сайт повинен працювати швидко та якісно, для цього слід також підібрати хостинг з відповідними параметрами.

Магазин повинен бути розділеним на дві частини:

- клієнтську;
- адміністративну.

Менеджери повинні мати змогу зручно керувати контентом сайту, а в потенційних клієнтів не повинно виникати труднощів при роботі з сайтом. Весь функціонал клієнтської частини повинен бути інтуїтивно зрозумілим та працювати без помилок. Усе це забезпечить постійний потік клієнтів, збільшить кількість замовлень та масштабує продажі.

РОЗДІЛ 2
СПЕЦИФІКАЦІЯ ВИМОГ ДО РОЗРОБЛЮВАНОЇ СИСТЕМИ

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

Перед початком роботи важливо обрати певний тип моделі для виконання поставленого завдання. Було обрано методологію Agile. Ось кілька причин чому було обрано саме цю модель для реалізації свого проекту:

– впродовж розробки усього проекту, розробник та клієнт повинні взаємодіяти щодня, для більшої результативності;

– над проектом повинні працювати вмотивовані розробники, щоб робота була успішно виконана клієнт повинен підтримувати зв’язок з розробниками та повністю покладатись на них;

– Agile надає можливість корегувати або ж повністю змінювати прописані в технічному завданні вимоги на протязі розробки;

– можливість поетапно додавати нову функціональність до розроблюваного продукту.

Спираючись на усі вищеописані пункти було прийнято рішення використати саме Agil.

Після того як ми визначились з типом моделі не менш важливо описати функціональні вимоги для розроблюваного застосунку. У моєму випадку функціональні вимоги поділяються на дві логічні частини:

– для клієнтської частини сайту;

– для адміністративної частини сайту.

Функціональні вимоги для клієнтської частини сайту:

– клієнт повинен мати можливість створити власний обліковий запис вказавши такі дані як ім’я, прізвище, електронна пошта, пароль та підтвердження правил по роботі з сайтом;

– клієнт повинен мати можливість увійти в систему за допомогою електронної пошти та паролю;

– клієнт повинен мати можливість додавати товари в кошик, змінювати їм кількість прямо в корзині. При зміні кількості товарів повинен працювати і перерахунок загальної вартості;

– клієнт повинен мати можливість створити замовлення на основі доданих в кошик товарів. Після створення замовлення воно закріплюється за клієнтом і відображається в його особистому кабінеті.

Нефункціональні вимоги для клієнтської частини сайту:

– система повинна захищати від несанкціонованого входу в облікові записи користувачів;

– всі паролі користувачів повинні зберігатися в захешованому вигляді в базі даних;

– сайт повинен швидко завантажуватися та працювати без помилок;

– оптимізовані запити до бази даних;

– проект повинен бути готовим до масштабування.

Функціональні вимоги для адміністративної частини сайту:

– адміністратори повинні мати можливість авторизації в системі;

– адміністратори повинні мати можливість додавати, переглядати, змінювати та видаляти товари, категорії, групи товарів, замовлення тощо;

– адміністратори повинні мати можливість відповідати на залишені користувачами відгуки до товарів.

Нефункціональні вимоги для клієнтської частини сайту:

– система повинна захищати від несанкціонованого входу в облікові записи користувачів;

– система повинна працювати без будь яких помилок.

Після опису функціональних вимог, для полегшення роботи при розробці слід все описане відобразити на uml діаграмі. В моєму випадку представлю до уваги uml діаграму зв’язків класів, яка відображає в собі функціонал додавання відгуків до товарів рисунок 2.1.

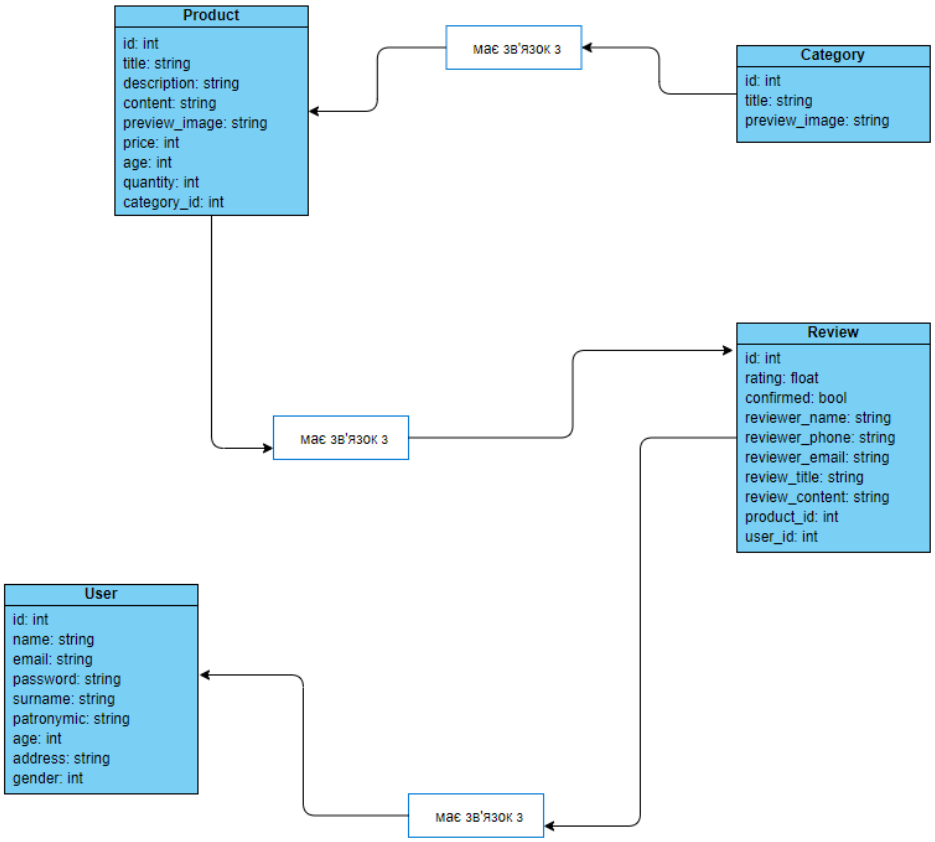

Рисунок 2.1 – Діаграма зв’язків класів функціоналу додавання відгуків

Тепер можна переходити до проектування дизайну інтерфейсу користувача.

В даному прикладі описано які блоки повинна містити «Головна» сторінка сайту:

– меню навігації повинне містити категорії товарів, каталог, посилання на головну, авторизація, персональний кабінет, кошик, список бажаного;

– головні банери;

– блок з інформаційними банерами;

– блок з найкращими товарами;

– блок з останніми доданими товарами;

– блок найпопулярніших товарів;

– блок покупками за категоріями;

– футер.

Зараз існує безліч діаграм які полегшують подальшу розробку. Тут і діаграми функціонування бази даних, і діаграми станів, діаграми прав користувача тощо, цей список можна продовжувати ще довго. Під час проектування системи довелось скласти діаграму варіантів використання (рис. 2.2), яка дуже допомогла при розробці, адже дала чітке загальне розуміння як саме повинен функціонувати сайт між покупцем та бізнес логікою.

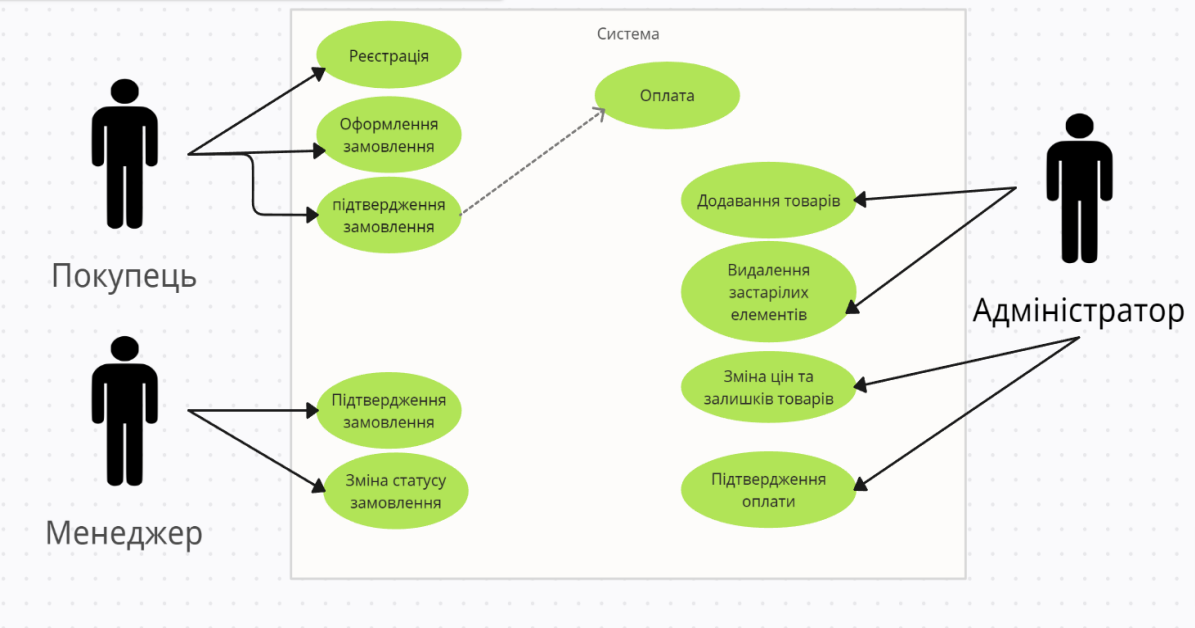

Рисунок 2.2 – Діаграма варіантів використання

2.2 Вибір засобів, методів і алгоритмів вирішення поставленого завдання

В сучасному світі перед веб розробниками стоїть безліч різноманітних завдань, від створення простеньких сайтів візиток до серйозних високо навантажених бізнес проектів. Але скільки завдань стільки й потенційних рішень як їх правильно реалізувати. Зараз існує безліч інструментів які допомагають при розробці та полегшують написання коду. Розробнику важливо ще перед початком роботи визначитись які саме інструменти будуть ним використовуватись, як правильно їх між собою поєднати, які в них переваги та

недоліки та що не менш важливо наскільки добре сам розробник розуміється у вибраних ним технологіях.

Так як у мене проект чітко поділяється на дві логічні частини:

– клієнтську частину;

– адміністративну частину.

Було прийнято рішення обрати фреймворк для кожної з цих частин. Важливо аби обрані технології доповнювали одне одне та працювали як одне ціле.

Вибір впав на Laravel для написання адміністративної частини та Vue для написання клієнтської. В сучасному світі це доволі актуально, адже дає велику гнучкість при розробці [5].

Laravel – це один із найпопулярніших php фреймворків. На ньому побудовані такі українські сайти як:

– online.ua інформаційне агенство новин україни та світу;

– тсн телевізійна служба новин;

– forbes.ua досить популярне видання;

– безліч інших менш відомих але не менш надійних проектів.

Усі ці сайти функціонують без будь яких проблем. Щодня їх відвідують тисячі та десятки тисяч людей як в україні так і по всьому світу.

Laravel має безліч переваг, ось декілька з них:

– зручний синтаксис який забезпечує швидкісну розробку. Вбудовані елементи такі як міграції, роутинг, мідлвеєри, фабрики, сіди тощо прискорюють написання проекту, адже розробник не витрачає час на рутинні завдання, за нього це все робить Laravel;

– побудований на багатьох патернах проектування, важко не виділити саме mvc, це гарантує що дотримуючись правил розробки на Laravel ваш додаток буде архітектурно чистим;

– Laravel включає в себе модульну структуру. Це дозволяє легко інтегрувати в систему вже готові компоненти, до прикладу авторизація, робота з базою даних, кешування, посторінкова навігація тощо;

– Laravel має велику спільноту розробників, яка пише безліч статей та модулів що свідчить про актуальність даного фреймворку;

– при встановленні Laravel одразу ж має вбудовані інструменти що відповідають за безпекову складову сайту розробника. Laravel має захист від SQL ін’єкцій, XSS атак, CSRF атак, тощо;

– Laravel доволі гнучкий, це дозволяє його без будь яких проблем поєднувати з різними javascript фреймворками, такими як Vue, React тощо [6].

Перераховано кілька основних переваг Laravel, які свідчать про актуальність даного фреймворку на поточний час, та чому було обрано для розробки саме його.

Що стосовно Vue.js то це потужний, сучасний JavaScript фреймворк для розробки клієнтської частини сайту. На Vue побудовані такі відомі сайти як:

– bilibili.com китайський маркетплейс;

– shein.com відомий інтернет магазин одягу;

– openai.com сайт розробників Chat GPT;

– багато інших не менш відомих сайтів.

Це свідчить про те, що світові компанії гіганти надають перевагу Vue.js що робить цей фреймворк одним із найпопулярніших. Це й не дивно, адже Vue має безліч переваг серед яких важко не виділити:

– низький поріг входу. Фреймворк легкий до вивчення та використання, має детально описаний та зрозумілий api, це робить його популярним серед малодосвідчених розробників;

– Vue прекрасний тим що дозволяє автоматично відслідковувати зміну даних та оновлювати відображення. Це знімає з розробника написання купи рутинного JavaScript коду;

– Vue має компоненту архітектуру, що дозволяє зручно ділити логічні блоки на окремі складові та оперувати ними без будь якого надлишкового коду;

– фреймворк має величезну кількість плагінів написаних спільнотою, що тільки полегшує розробку.

Було описано головні переваги та чому мій вибір впав саме на цей фреймворк.

Для більш комфортної розробки на Vue було прийнято рішення використати Vuex – становий управляючий патерн. Його основне завдання це розподілення фронтенд логіки по різних модулях, що робить компоненти Vue ще більш чистими та простішими до розуміння. Ось кілька основних переваг Vuex:

– централізоване сховище стану всього додатку. Дані зберігаються в одному місці що полегшує управління та підтримку ними;

– модульна система. Можливість розділити сховище на окремі модулі, кожен з яких буде мати свої власні методи, гетери, сетери, тощо;

– модулі можуть взаємодіяти між собою, тобто модуль А може виконувати методи та міняти стан модуля Б;

– Vuex дозволяє змінювати стан модуля лише через мутації і ніяк більше;

– Vuex має власні браузерні розширення які допомагають відслідковувати зміни в стані і полегшують процес відлагодження;

– підтримує асинхронні операції.

Якщо узагальнити, то Vuex допомагає управляти складним (великим) станом додатку та забезпечує чистий та ефективний код доступний до розширення.

Після того як фреймворки було обрано, настав час обрати підходящу базу даних, серце розроблюваного магазину.

База даних є важливою складовою будь якого сучасного веб застосунку. Вона повинна забезпечувати надійне зберігання та доступ до даних. В сучасному світі є безліч різновидів баз даних, але одними з найпоширеніших являються

саме реляційні. Вони використовують табличну структуру для зберігання даних, а мова SQL допомагає зручно оперувати цими даними:

– читати;
– змінювати;
– додавати;
– видаляти;
– робити вибірки різного рівня важкості із потрібним сортуванням;
– пришвидшувати роботу використовуючи індекси;
– створювати власні тригери;
– тощо.

Було обрано базу даних MySql, адже це одна з найпоширеніших реляційних баз, що використовує мову запитів SQL. Її основні переваги полягають в:

– великій популярності та підтримці. Вона є однією з найбільш популярних бд у світі, а отже має широку спільноту розробників;

– її безкоштовності. БД повністю відкрита та не потребує оплати за використання;

– можливості резервного копіювання та відновлення, що дозволяє більш убезпечити дані;

– швидкості бази даних, що дозволяє ефективно отримувати та обробляти великі обсяги даних;

– легкості інтеграції з різними системами, що робить цю базу даних досить гнучкою.

Перед початком розробки важливо чітко розуміти як повинна функціонувати система, для цього її потрібно візуалізувати. Тут допомогло створення власних діаграм.

Діаграми полегшують подальшу розробку, адже програміст чітко бачить перед собою набір сутностей які йому потрібно реалізувати.

Висновки до розділу 2

Спираючись на все описане в цьому розділі, можна дійти висновку що важливо не тільки писати код, а й правильно спроектувати систему. Це убезпечить розробника від майбутніх тупикових ситуацій і допоможе уникнути костилів при розробці.

Правильно спроектована система надає розробнику чіткий план дій, візуалізований план реалізації поставлених завдань. Для досягнення цього можна використовувати різні діаграми та описи певної функціональності сайту.

РОЗДІЛ 3
РОЗРОБКА ІНТЕРНЕТ МАГАЗИНУ НА LARAVEL ТА VUE.JS

3.1 Практична реалізація об’єкта проектування

В процесі створення адміністративної частини сайту було використано фреймворк Laravel який використовується з мовою php у середовищі розробки PhpStorm [7].

Написання коду починається з розробки основної функціональності, в моєму випадку це наповнення товарів. Для цього першочергово потрібно описати міграцію яка створить відповідну таблицю під продукти в моїй базі даних (лістинг 3.1). Для того щоб створити міграцію було використано команду php artisan make:model Product -m ця команда автоматично створить модель і міграцію до неї.

Лістинг 3.1 – Демонстрація коду міграції таблиці продуктів

```
<?php

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

class CreateProductsTable extends Migration
{
    /**
     * Run the migrations.
     *
     * @return void
     */
    public function up()
    {
        Schema::create('products', function (Blueprint $table) {
            $table->id();
            $table->string('title');
            $table->text('description');
            $table->text('content');
            $table->string('preview_image');

            $table->integer('price');
            $table->integer('quantity');
            $table->boolean('is_published')->default(true);
```

```
            $table->foreignId('category_id')->nullable()->index()->constrained('categories');

            $table->timestamps();
        });
    }

    /**
     * Reverse the migrations.
     *
     * @return void
     */
    public function down()
    {
        Schema::dropIfExists('products');
    }
}
```

Кінець лістингу 3.1

Після того як міграція описана потрібно її виконати, для цього виконано консольну команду php artisan migrate та перевіряю чи створилась потрібна таблиця в базі даних (рис. 3.1).

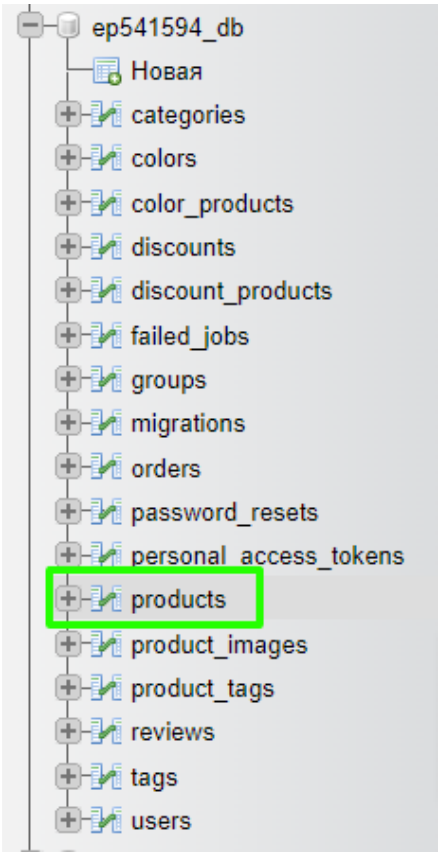

Рисунок 3.1 – Створена таблиця товарів

Тепер потрібно створити контролер який буде відповідати за обробку даних. Для цього виконаю команду: php artisan make:controller ProductController –resource. Ця команда створить ресурсний контролер, тобто контролер в якому одразу є заготовки під головні 7 методів роботи CRUD в Laravel:

– Index. Вивід списку товарів;

– Create. Форма створення нового товару;

– Store. Серверний код, метод обробник який безпосередньо зберігає товар в системі;

– Show. Відображення конкретного товару;

– Edit. Форма редагування конкретного товару;

– Update. Серверний код, метод обробник який безпосередньо оновлює товар в системі;

– Destroy. Видалення конкретного товару [8].

Усі ці методи представлені до ознайомлення в офіційній документації Laravel (рис. 3.2), і безпосередньо є конвенцією фреймворку.

Actions Handled by Resource Controllers

Verb	URI	Action	Route Name
GET	/photos	index	photos.index
GET	/photos/create	create	photos.create
POST	/photos	store	photos.store
GET	/photos/{photo}	show	photos.show
GET	/photos/{photo}/edit	edit	photos.edit
PUT/PATCH	/photos/{photo}	update	photos.update
DELETE	/photos/{photo}	destroy	photos.destroy

Рисунок 3.2 – Методи для роботи з ресурсними контролерами в Laravel

Тепер коли ми створили контролер, потрібно описувати методи в ньому, першим ділом потрібно описати метод create який відповідає за форму додавання товару (лістинг 3.2).

Лістинг 3.2 – Код метода create

```
    public function create()
    {
        $tags = Tag::all();
        $colors = Color::all();
        $categories = Category::all();
        $discounts = Discount::all();
        $groups = Group::all();

        return view('product.create', compact('tags', 'colors',
'categories', 'discounts', 'groups'));
    }
```

Кінець лістингу 3.2

Тепер потрібно створити view яка буде відповідати за відображення форми форми додавання товару. Файл потрібно розмістити в директорії resources (рис. 3.3), тут розміщуються усі шаблони та все що стосується верстки в Laravel.

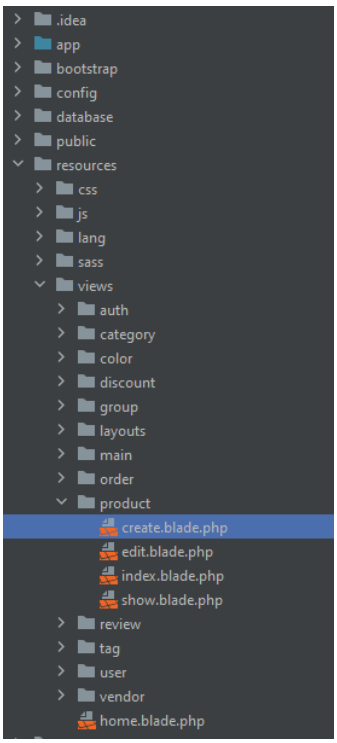

Рисунок 3.3 – Розміщення файлу верстки форми додавання товару

Тепер потрібно відобразити сторінку в браузері. Для цього потрібно створити відповідний роут в файлі web.php (лістинг 3.3). Розроблена форма додавання нового товару має вигляд як на рисунку 3.4.

Лістинг 3.3 – Роут для відображення сторінки в браузері

```
Route::resource('products',
App\Http\Controllers\Product\ProductController::class);
```

Кінець лістингу 3.3

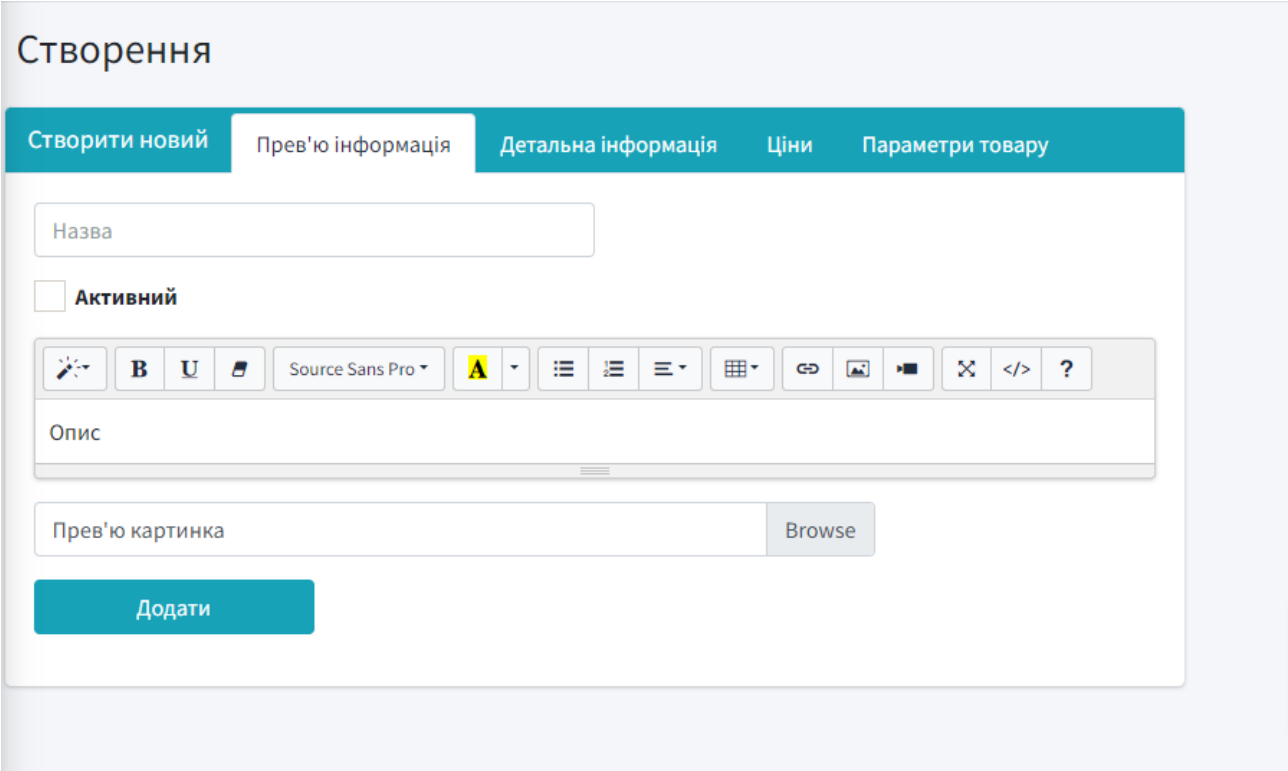

Рисунок 3.4 – Форма додавання товару

Отже, форма реалізована, наступний крок це обробка даних які вона віддає, і на їх основі створити новий товар в системі. Для цього було реалізовано метод store в контролері товарів (лістинг 3.4).

Лістинг 3.4 – Код методу store

```
        public function store(StoreRequest $request):
RedirectResponse
            {
                $data = $request->validated();
```

```
            $data['preview_image'] = Storage::disk('public')-
>put('/images', $data['preview_image']);

            $tagsIds = [];
            $colorsIds = [];
            $discountsIds = [];
            $productImages = [];
            $data['is_published'] =
(bool)isset($data['is_published']);
            $data['is_bestseller'] =
(bool)isset($data['is_bestseller']);
            if (isset($data['tags'])) {
                $tagsIds = $data['tags'];
                unset($data['tags']);
            }
            if (isset($data['colors'])) {
                $colorsIds = $data['colors'];
                unset($data['colors']);
            }

            if (isset($data['discounts'])) {
                $discountsIds = $data['discounts'];
                unset($data['discounts']);
            }

            if (isset($data['product_images'])) {
                $productImages = $data['product_images'];
                unset($data['product_images']);
            }

            $product = Product::firstOrCreate([
                'title' => $data['title']
            ], $data);

            foreach ($productImages as $productImage) {
                $currentImagesCount =
ProductImage::where('product_id', $product->id)->get()->count();

                if ($currentImagesCount > 3) break;
                $filePath = Storage::disk('public')-
>put('/images', $productImage);
                ProductImage::create([
                    'product_id' => $product->id,
                    'file_path' => $filePath
                ]);
            }
            $product->tags()->attach($tagsIds);
            $product->colors()->attach($colorsIds);
            $product->discounts()->attach($discountsIds);
            return redirect()->route('products.index');
        }
```

Кінець лістингу 3.4

Як бачимо в лістингу, дані з форми проходять валідацію стрічкою `$request->validated();` Це конвенція Laravel, адже фреймворк передбачає роботу з валідацією в окремих файлах, так званих реквестах. Для того щоб створити реквест потрібно виконати команду php artisan make:request Product\StoreRequest. Файл створиться у відповідній директорії (рис. 3.5).

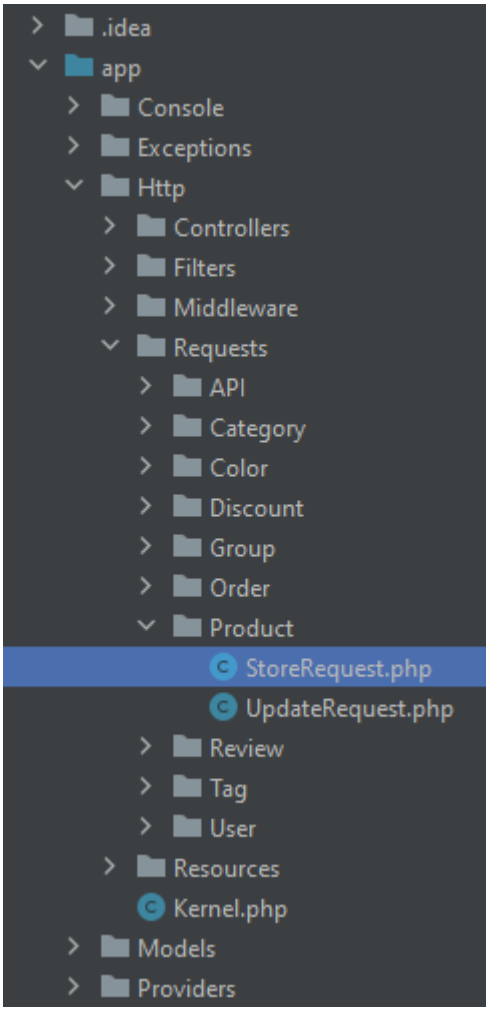

Рисунок 3.5 – Розміщення StoreRequest в файловій структурі Laravel

В StoreRequest прописуються правила для обробки даних з форми, поля які очікуються, тип даних для полів та повідомлення в разі помилки (лістинг 3.5).

Лістинг 3.5 – Код файлу StoreRequest

```
<?php

namespace App\Http\Requests\Product;
```

```
use Illuminate\Foundation\Http\FormRequest;

class StoreRequest extends FormRequest
{
    /**
     * Determine if the user is authorized to make this request.
     *
     * @return bool
     */
    public function authorize()
    {
        return true;
    }

    /**
     * Get the validation rules that apply to the request.
     *
     * @return array
     */
    public function rules()
    {
        return [
            'title' => 'required|string',
            'description' => 'required|string',
            'content' => 'required|string',
            'preview_image' => 'required|file',
            'price' => 'required|numeric',
            'quantity' => 'required|numeric',
            'is_published' => 'nullable',
            'is_bestseller' => 'nullable',
            'category_id' => 'required|integer',
            'group_id' => 'required|integer',
            'tags' => 'nullable|array',
            'colors' => 'nullable|array',
            'discounts' => 'nullable|array',
            'product_images' => 'nullable|array'
        ];
    }

    public function messages()
    {
        return [
        ];
    }
}
```

Кінець лістингу 3.5

Тепер форма додавання нового товару повністю готова, і можна створювати свій перший товар.

Коли перші товари наповнено, можна перейти і до їх відображення в клієнтській частині сайту. Для прикладу приведу реалізацію блоку бестселерів на головній сторінці сайту (рис. 3.6).

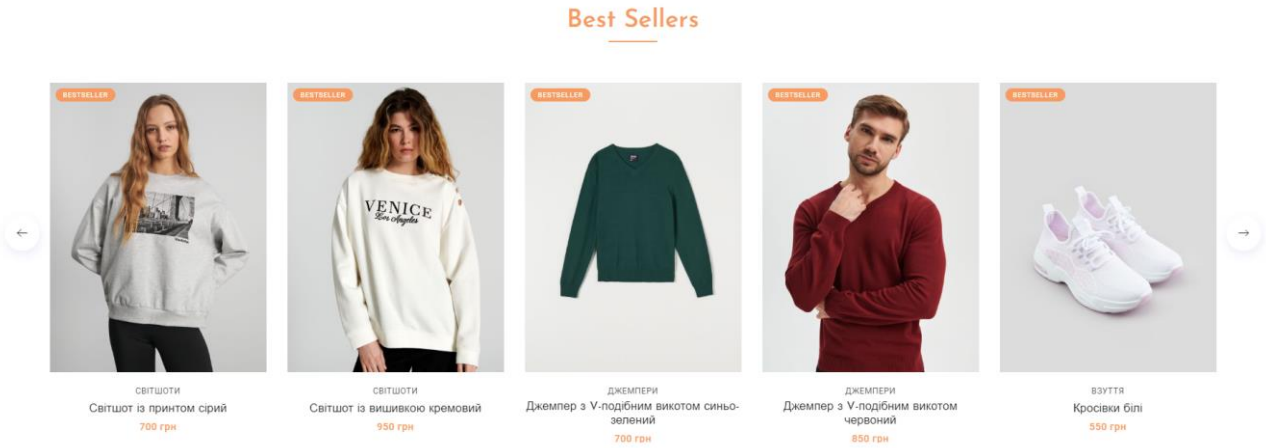

Рисунок 3.6 – Блок бестселерів на головній

Першочергово слід створити компонент [9]. Компонент створено та додано відповідну верстку, стилі та скрипти плагінів які задіяні в цьому блоці (лістинг 3.6).

Лістинг 3.6 – Верстка блоку бестселерів

```
        <div class="row">
          <div class="tab-content" id="pills-tabContent-two">
            <div class="tab-pane fade show active" id="pills-best-
sellers" role="tabpanel"
                 aria-labelledby="pills-best-sellers-tab">
              <div id="bestsellerBlock" class="">
                <div v-for="bestseller in bestsellers"
class="products-three__inner">
                  <div class="products-three-single w-100">
                    <div class="products-three-single-img">
                      <router-link class="d-block" :to="{name:
'products.show', params: {id: bestseller.id}}">
                        <img :src="bestseller.image_url"
class="first-img"
                             alt=""/>
                        <img :src="bestseller.image_url" alt=""
                             class="hover-img"/>
                      </router-link>
```

```
                        <div class="products-grid-one__badge-box">
<span
                            class="bg_black badge
discount">bestseller</span></div>
                        <a
@click.prevent="this.$store.dispatch('Cart/addToCart', {id:
bestseller.id, isSingle: true})"
                           class="addcart btn--primary style2"> Add
To Cart </a>
                        <div class="products-grid__usefull-links">
                          <ul>

                          </ul>
                        </div>
                      </div>
                      <div class="products-three-single-content
text-center"><span>{{ bestseller.category.title }}</span>
                        <h5>
                          <router-link :to="{name: 'products.show',
params: {id: bestseller.id}}">{{
                              bestseller.title
                            }}
                          </router-link>
                        </h5>
                        <p>{{ bestseller.price_with_discounts }}
грн</p>
                      </div>
                    </div>
                  </div>
                </div>
              </div>
```

Кінець лістингу 3.6

Наступним кроком слід отримати дані стосовно бестселерів від бекенду, тобто від Laravel. Для цього було створено відповідний роут в файлі api.php в Laravel (лістинг 3.7).

Лістинг 3.7 – Роут в файлі api.php

```
Route::get('/bestsellers',
[\App\Http\Controllers\API\Product\ProductController::class,
'getBestsellers']);
```

Кінець лістингу 3.7

Тепер потрібно створити відповідний контролер з методом, який буде віддавати товари бестселери (лістинг 3.8).

Лістинг 3.8 – Код метода який віддає товари бестселери

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

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

Кінець лістингу 3.8

Наступний крок це створення api запиту на стороні vue.js за допомогою якого vue отримає потрібні товари і вони автоматично підставляться в блок бестселерів (лістинг 3.9). Api запити відправлялись за допомогою Axios [10].

Лістинг 3.9 – Api запит на вибірку бестселерів

```
getBestsellers() {
      this.axios.get('http://2933833.ep541594.web.hosting-
test.net/admin/public/api/bestsellers')
          .then(response => {
            this.bestsellers = response.data.data;
          })
          .then(() => {
            this.$nextTick(() => {
              let bestsellerBlock =
document.getElementById('bestsellerBlock');
              if (bestsellerBlock) {
                bestsellerBlock.classList.add('catagory-slider-
three')
                $(document).trigger('initJqueryPlugins');
              }
            });
          })
    }
```

Кінець лістингу 3.9

Ось і все, тепер блок бестселерів наповнюється потрібними товарами та працює коректно, а адміністратори можуть з легкістю маніпулювати товарами які відображаються в цьому блоці.

3.2 Розробка бази даних

Важливим кроком є процес реалізації бази даних. Першим кроком потрібно її спроектувати, продумати які сутності вона повинна в себе включати, які зв'язки між таблицями повинні бути реалізовані, які поля повинна містити кожна з таблиць. Для цього було розроблено діаграму майбутньої бази даних (рис. 3.7).

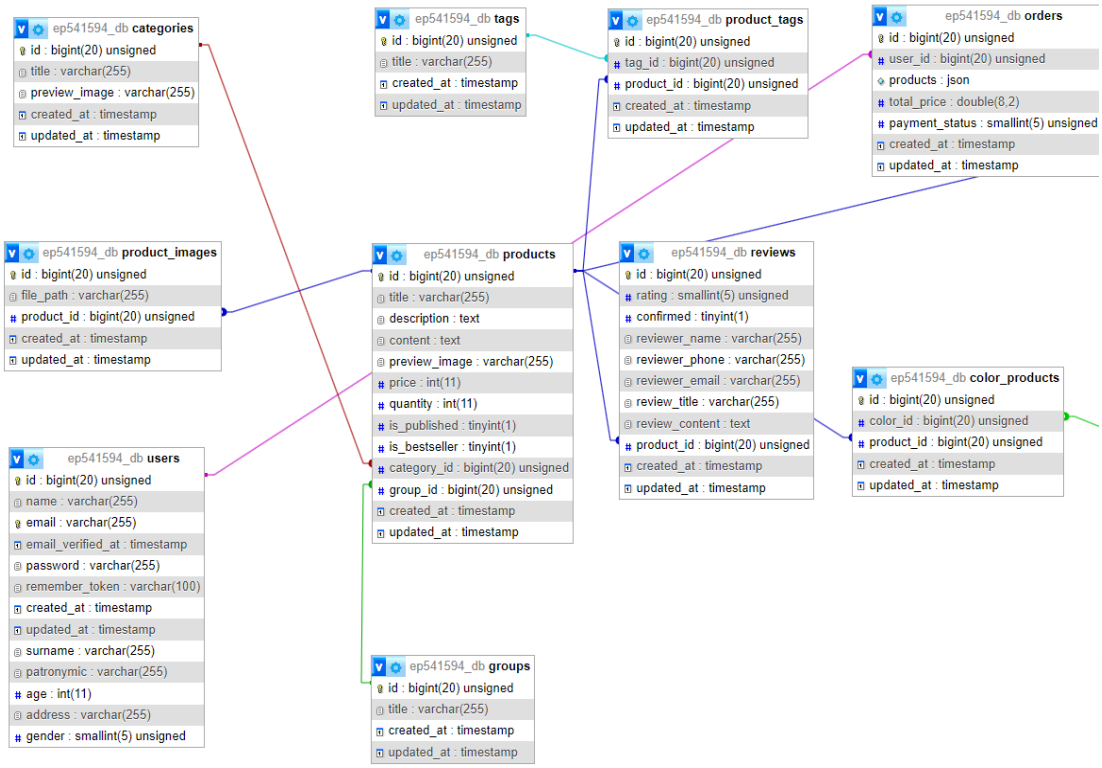

Рисунок 3.7 – Діаграма майбутньої бази даних

Для додавання цих таблиць не потрібно писати SQL код на створення чи використовувати програму PhpMyAdmin, в Laravel передбачено міграції для цього всього діла [11]. Це дуже зручно та просто в використанні. Міграції дозволяють швидко як додавати зміни так і відкатувати їх. Для прикладу

продемонструю міграцію таблиці відгуків (лістинг 3.10), для створення якої скористався командою php artisan make:migration create_reviews_table.

Лістинг 3.10 – Код міграції відгуків

```
<?php

use Illuminate\Database\Migrations\Migration;

use Illuminate\Database\Schema\Blueprint;

use Illuminate\Support\Facades\Schema;

class CreateReviewsTable extends Migration
{
    /**
     * Run the migrations.
     *
     * @return void
     */
    public function up()
    {
        Schema::create('reviews', function (Blueprint $table) {

            $table->id();
            $table->unsignedSmallInteger('rating')->default(1);
            $table->boolean('confirmed')->default(false);

            $table->string('reviewer_name');
            $table->string('reviewer_phone');
            $table->string('reviewer_email');

            $table->string('review_title');
            $table->text('review_content');

            $table->foreignId('product_id')->index()-
>constrained('products');


            $table->timestamps();
        });
    }

    /**
     * Reverse the migrations.
     *
     * @return void
     */
    public function down()
```

```
    {
        Schema::dropIfExists('reviews');
    }
}
```

Кінець лістингу 3.10

По аналогії було описано і створено решту таблиць (рис. 3.8). Серед яких таблиця тегів, товарів, категорій, користувачів, груп товарів, тощо [12].

Кожна таблиця важлива, адже зберігає відведені для неї дані, які в свою чергу забезпечують сайт контентом. Дані для кожної з таблиць менеджер сайту може редагувати, додавати або ж видаляти в адміністративній частині сайту.

Рисунок 3.8 – Повний список таблиць бази даних

3.3 Тестування та налагодження інтернет магазину

В процесі розробки можуть виникати різні підводні камені які спричиняють ті чи інші баги. Розробник повинен бути уважним до таких моментів, повинен проводити тести різних типів складності аби переконатись що розроблюваний ним функціонал працює коректно та без будь яких помилок [13].

Перше на що слід звернути увагу це консоль розробника, вмикається клавішою f12 практично в усіх браузерах, консоль повинна бути чиста та без

будь яких помилок чи зайвих логів (рис. 3.9). Адже якщо ігнорувати помилки такого типу це в майбутньому може призвести до нестабільної роботи сайту і певний функціонал зможе відпрацьовувати неналежним чином або ж взагалі не працювати.

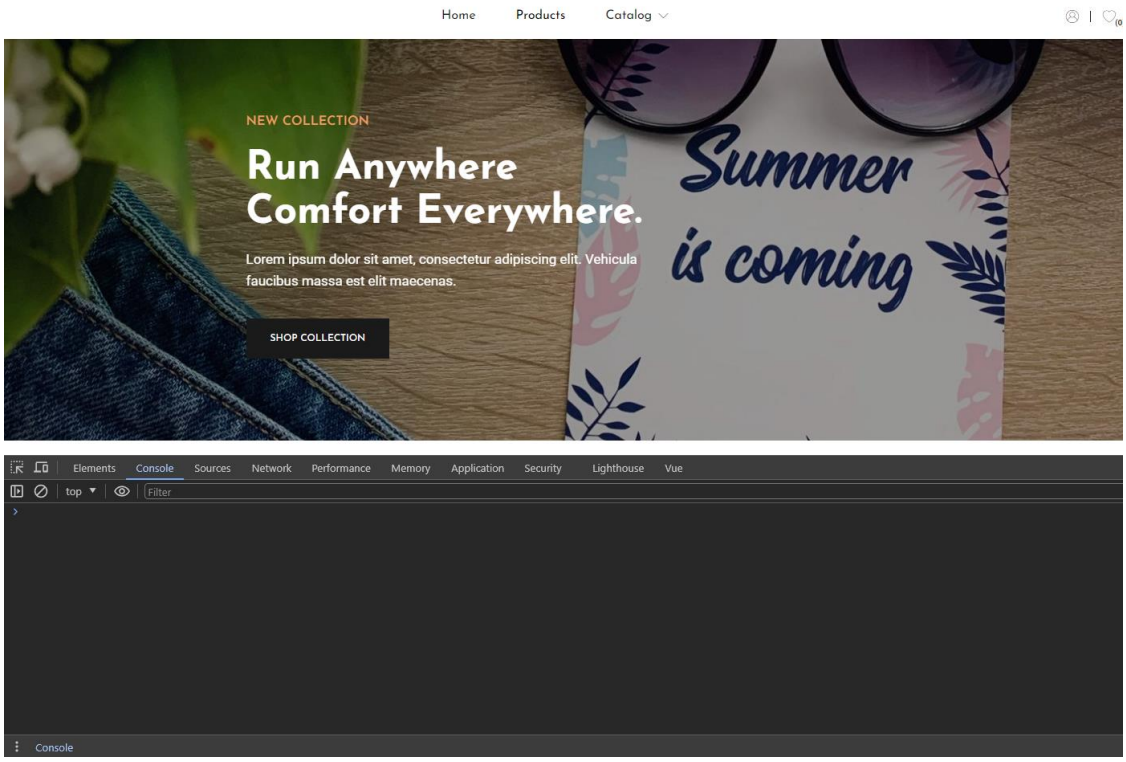

Рисунок 3.9 – Приклад чистої консолі розробника

Наступне на що потрібно звернути увагу це вкладка network в тій же самій консолі розробника [14]. Потрібно переконатись чи всі серверні запити мають успішне виконання, або ж код 200 (рис. 3.10).

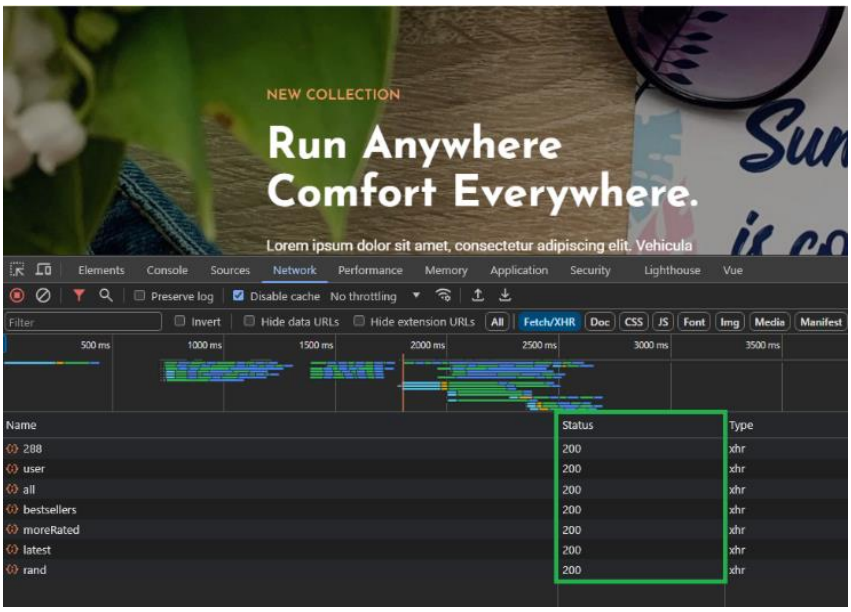

Рисунок 3.10 – Приклад коректного відпрацювання усіх серверних запитів

Для тестування Vue було встановлено відповідне розширення тестувань для браузера [15]. Розширення офіційне, від розробників фреймворку та допомагає зручно переглядати структуру компонентів (рис. 3.11) та стану веб застосунку (рис. 3.12).

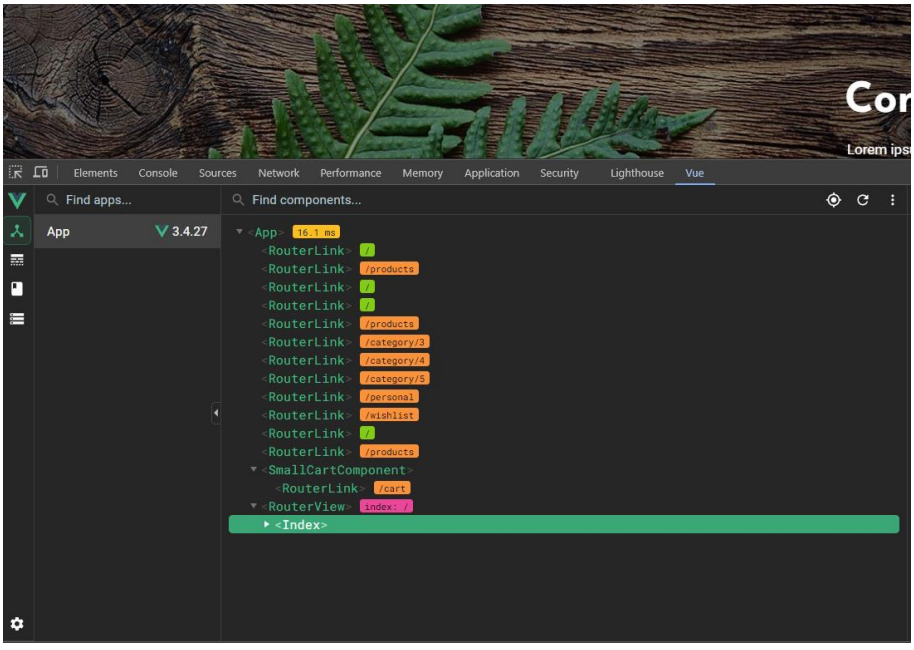

Рисунок 3.11 – Перегляд структури компонентів за допомогою розширення Vue

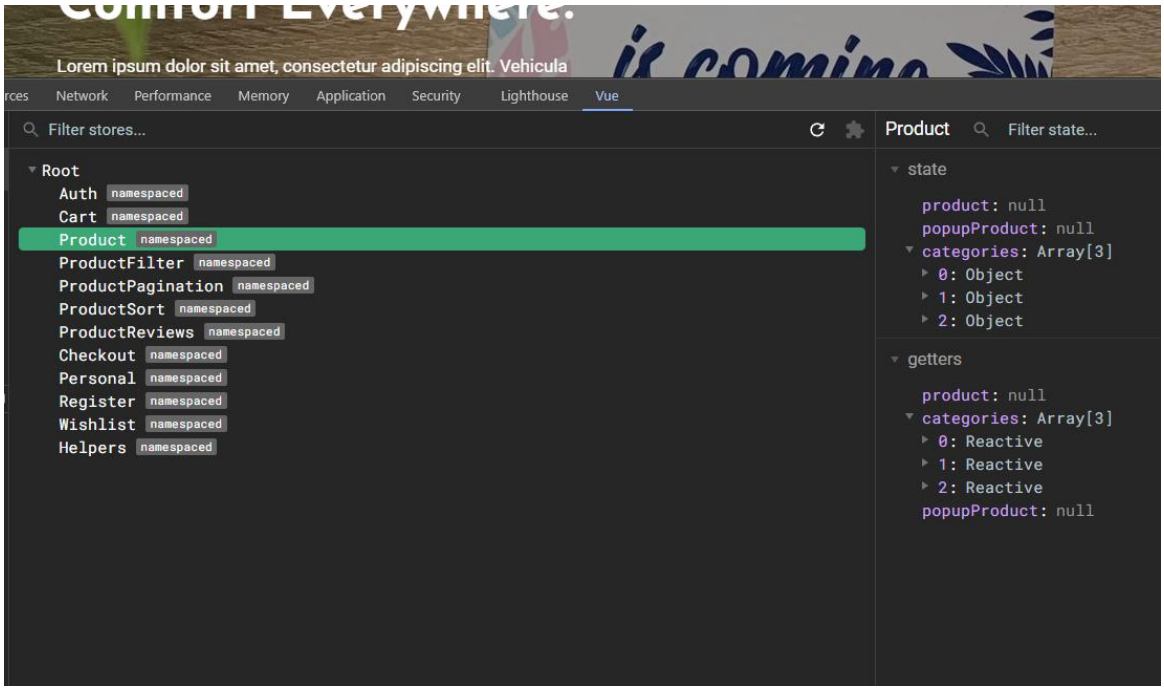

Рисунок 3.12 – Перегляд стану веб застосунку за допомогою розширення Vue

Також було перевірено і основну функціональність сайту:

– додавання товарів в кошик;
– додавання товарів в список бажаного;
– редагування кількості товарів в кошику;
– оформлення замовлення;
– наповнення сайту контентом;
– наповнення сайту товарами.

Проблем не було виявлено, що свідчить про добре виконану роботу, гарний та структурований код [16].

ВИСНОВКИ

Основною метою було успішне виконання усіх поставлених завдань, якість та вчасність виконаних робіт.

Головних цілей було досягнуто, в процесі розробки було виконано великий об’єм робіт.

Першочергово було проведено аналіз предметної області. Досліджено актуальність обраної теми та веб технологій. Досліджено архітектуру обраних фреймовркв, контролери, компоненти, роути, міграції, тощо.

Наступним кроком було проаналізовано сайти конкурентів(AGER, LUREX, MII STYL). На них виявлено проблеми, яких було уникнено при розробці інтернет магазину. Серед найбільш поширених проблем важко не відмітити:

- не сучасний або ж застарілий дизайн;
- відсутність кросбраузерності;
- погана оптимізація;
- постійні баги;
- велика навантаженість на сервер.

Наступним етапом було обрано необхідні для розробки інструменти:

- Laravel для адміністративної частини;
- Vue для клієнтської частини;
- MySql для роботи з БД.

Обрані інструменти дуже добре працюють в поєднанні між собою, проблем чи підводних каменів при розробці не виникало.

Подальшими діями було налаштування середовища, а саме вибір потрібного хостингу, налаштування версії php та увімкнення необхідних розширень для коректного функціонування застосунку, встановлення пакетного менеджера composer, встановлення node.js та пакетного менеджера npm.

Наступним кроком була розробка адміністративної частини сайту, аби була змога наповнювати товари, категорії, тощо. При розробці було створено

дизайн адміністративної частини, розроблено верстку, додано роути, контролери, ресурси.

Після завершення роботи з адміністративною частиною, було розроблено клієнтську частину, розроблено дизайн сайту, створено верстку та розбито її на компоненти Vue, за допомогою Vuex отримав потрібні мені дані від Laravel та використав їх для функціонування магазину.

Розробка велась легко, гармонічно та без будь яких проблем. В процесі розробки глухих кутів не виникало.

Наступним кроком було тестування розробленого продукту. Було перевірено таку функціональність:

- додавання товарів в кошик;
- додавання товарів в список бажаного;
- редагування кількості товарів в кошику;
- оформлення замовлення;
- наповнення сайту контентом;
- наповнення сайту товарами.

В результаті виконання кваліфікаційної роботи бакалавра, було створено повноцінний інтернет магазин. Було докладено максимальних зусиль для його валідного функціонування та легкості в експлуатації.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Інтернет магазин AGER. URL: https://ager.ua/uk/ (дата звернення: 05.02.24).

2. Google speed test. URL: https://pagespeed.web.dev/ (дата звернення: 05.02.24).

3. Інтернет магазин LUREX. URL: https://lurex.in.ua/uk (дата звернення: 10.11.23).

4. Інтернет магазин MII STYL. URL: https://miistyl.ua/uk/ (дата звернення: 10.02.24).

5. Документація Laravel. URL: https://laravel.com/docs/ (дата звернення: 10.02.24).

6. Документація Vue. URL: https://vuejs.org/ (дата звернення: 03.03.24).

7. Документація PHP. URL: https://www.php.net/ (дата звернення: 19.03.24).

8. Робота з ресурсними контролерами в Laravel. URL: https://laravel.com/docs/11.x/controllers (дата звернення: 20.03.2024).

9. Приклад створення кастомних компонентів Vue. URL: https://blog.avislab.com/flask-vue/example6/ (дата звернення: 25.03.2024).

10. Документація бібліотеки Axios. URL: https://my-js.org/docs/cheatsheet/axios/ (дата звернення: 07.04.2024).

11. Приклад роботи з міграціями в Laravel. URL: https://laravel.su/docs/10.x/migrations (дата звернення: 12.04.2024).

12. Правила для коректного проектування БД. URL: http://surl.li/ukynm (дата звернення: 02.03.2024)

13. Керівництво з поетапного тестування сайтів. URL: http://surl.li/ukynx (дата звернення: 02.05.2024)

14. Опис вкладки network в консолі розробника. URL: http://surl.li/ukynz (дата звернення: 12.05.2024)

15. Браузерне Vue розширення для тестувань. URL: http://surl.li/ukyoe (дата звернення: 20.05.2024)

16. Основні положення для чистого коду. URL: https://foxminded.ua/shcho-take-chystyi-kod/ (дата звернення: 22.05.2024)

ДОДАТКИ

Додаток А

Код Product контролера який відповідає за вибірку товарів по api

```
<?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

Код ProductMinResource який віддає по api підготовлені дані

```
<?php

namespace App\Http\Resources\Product;

use App\Http\Resources\Category\CategoryResource;
use App\Http\Resources\Color\ColorResource;
use App\Http\Resources\Discount\DiscountResource;
use App\Http\Resources\Tag\TagResource;
use App\Models\Product;
use Illuminate\Http\Resources\Json\JsonResource;

class ProductMinResource extends JsonResource
{
    /**
     * Transform the resource into an array.
     *
     * @param  \Illuminate\Http\Request  $request
     * @return
array|\Illuminate\Contracts\Support\Arrayable|\JsonSerializable
     */
    public function toArray($request)
    {
        $priceWithDiscounts = $this->price;
        foreach ($this->discounts as $discount) {
            $priceWithDiscounts -= $discount->discount_sum;
        }

        return [
            'id' => $this->id,
            'title' => $this->title,
            'description' => $this->description,
            'content' => $this->content,
            'image_url' => $this->imageUrl,
            'price' => $this->price,
            'price_with_discounts' => $priceWithDiscounts,
            'quantity' => $this->quantity,
            'is_published' => $this->is_published,
            'category' => new CategoryResource($this->category),
            'tags' => TagResource::collection($this->tags),
            'colors' => ColorResource::collection($this->colors),
            'discounts' => DiscountResource::collection($this-
>discounts),
        ];
```