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»

WEB-ПРОГРАМУВАННЯ ТА РОЗРОБКА ІНТЕРНЕТ-МАГАЗИНУ ІЗ ВИКОРИСТАННЯМ СУЧАСНИХ ФРЕЙМВОРКІВ ДЛЯ РОЗРОБКИ

WEB PROGRAMMING AND DEVELOPMENT OF AN INTERNET STORE USING MODERN DEVELOPMENT FRAMEWORKS

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

Виконав: здобувач вищої освіти
групи ІПЗс-31
Пилипчук Назарій Віталійович

Керівник:
д.т.н., професор
Андрущак Ігор Євгенович

Кваліфікаційну роботу
допущено до захисту
«8» 06 2023р.
Гарант освітньої програми:
к.т.н., доцент
Ліщина Наталія Миколаївна

Луцьк – 2023 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
Лішина Наталія Миколаївна
«02» 03 2023 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ ЗДОБУВАЧУ ВИЩОЇ ОСВІТИ

Пилипчуку Назарію Віталійовичу
(прізвище, ім'я, по батькові)

1. Розробка інтернет-магазину із використанням Nuxt 3, Vue 3.

Керівник роботи: д.т.н., професор Андрущак Ігор Євгенович
затверджені наказом закладу вищої освіти від «23» грудня 2022 р. № 961-01-02

2. Строк подання здобувачем вищої освіти кваліфікаційної роботи «8» червня 2023 р.

3. Вихідні дані до роботи: *вимоги до розробки програмного забезпечення, вимоги до функціонування програмного засобу, Технології програмування клієнтської частини – HTML, CSS, JavaScript, Vue, Nuxt.*

4. Зміст розрахунково-пояснювальної записки (перелік питань, що потрібно розробити):
Аналіз сучасного стану проблеми, існуючих методів і засобів її розв'язання, аналіз і вибір засобів проектування, опис функціонального наповнення об'єкта проектування, розробка додатку.

5. Перелік графічного матеріалу: 34 рис.

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис	
		завдання видав	завдання прийняв
Аналіз предметної області	*Андрущак І.Є.*		
Специфікація вимог до розробленої системи	*Андрущак І.Є.*		
Розробка об'єкта проєктування	*Андрущак І.Є.*		
Нормоконтроль	*Андрущак І.Є.*		
Гарант ОП	*Ліщина Н.М.*		
Показник запозичень тексту	5,4 %		
Академічна доброчесність	*Яцук А.А.*		

7. Дата видачі завдання *«2» лютого 2023 р.*

КАЛЕНДАРНИЙ ПЛАН

№ З/п	Назва етапів кваліфікаційної роботи бакалавра	Строк виконання етапів роботи	Прим
1	*Огляд літературних джерел по темі кваліфікаційної роботи бакалавра*	*05.02.2023 р.*	
2	*Аналіз проблеми розробки та впровадження об'єкту проєктування*	*27.02.2023 р.*	
3	*Обґрунтування вибору шляхів, технологій (алгоритмів) і засобів вирішення поставленого завдання*	*10.03.2023 р.*	
4	*Розробка функціонально-структурної схеми роботи об'єкта проєктування та проєктування бази даних*	*17.04.2023 р.*	
5	*Практична реалізація об'єкта проєктування та розробка бази даних*	*18.05.2023 р.*	
6	*Тестування та налагодження об'єкта проєктування*	*01.06.2023 р.*	
7	*Здача чистового варіанту кваліфікаційної роботи бакалавра на кафедру*	*08.06.2023 р.*	

Здобувач вищої освіти _____ Пилипчук К.В.
(підпис) (прізвище, ініціали)

Керівник кваліфікаційної роботи _____ Андрущак І.Є.
(підпис) (прізвище, ініціали)

Міністерство освіти і науки України
Луцький національний технічний університет

Рецензія
на кваліфікаційну роботу

Здобувач вищої освіти:
Пилипчук Назарій Віталійович

Тема: «Web-програмування та розробка інтернет-магазину із використанням сучасних фреймворків для розробки»

Коротка характеристика кваліфікаційної роботи:
За мету кваліфікаційної роботи було взято створення інтернет-магазину на основі останніх версій сучасних фреймворків основною мовою програмування яких є JavaScript.
Самостійні розробки і пропозиції автора:
Автором кваліфікаційної роботи було розроблено програмне забезпечення що повністю відповідає вимогам до розробки сучасного веб-сайту, що на його думку є важливим для сучасним показникам швидкодії, та як зразок було запропоновано використання таких можливостей як SSR для коректної роботи сайту із поганим мережевим з'єднанням.
Практичне значення роботи:
Практичне значення роботи доводить актуальність використання сучасних технологій для сьогодення, підтвердженням цього є кінцеві показники розробленого програмного продукту, та шляхи створення його функціоналу.
Недоліки:
Виконана робота відповідає сучасним вимогах розробки, проте ці рішення підтверджені документацією фреймворків, а це відповідає мінімалізму унікальності.
Загальний висновок:
У процесі розробки було створено загальну архітектуру проекту, включаючи маршрутизацію, управління станом, роботу з API та інші важливі функціональні частини. Були реалізовані ключові функції інтернет-магазину, такі як перегляд каталогу товарів, додавання товарів до кошика, оформлення замовлення та оплата. Під час розробки було використано сучасні практики програмування, такі як компонентна архітектура, реактивне програмування та асинхронні запити до сервера.

Рецензент: к.т.н. Кошелюк В.А.

Рецензент кваліфікаційної роботи _____
(підпис)

«8» червня 2023 р.

АНОТАЦІЯ

Пилипчук Н.В. Web-програмування та розробка інтернет-магазину із використанням сучасних фреймворків для розробки. Рукопис.

Кваліфікаційна робота бакалавра. Кваліфікаційна робота за спеціальністю 121 Інженерія програмного забезпечення. Луцький національний технічний університет. Луцьк, 2023.58 с.

Кваліфікаційна робота присвячена розробці інформаційної системи на прикладі інтернет-платформи для продажу товарів.

Ключові слова: Інтернет-магазин, JavaScript, Vue 3, Nuxt 3.

ANNOTATION

Pylypchuk N.V. Web programming and development of an online store using modern development frameworks. Manuscript.

Bachelor thesis. Bachelor thesis in the specialty 121 Software Engineering. Lutsk National Technical University. Lutsk, 2023. 58 p.

The thesis is devoted to the development of an Internet platform for the sale of goods.

Keywords: Online Store, JavaScript, Vue 3, Nuxt 3.

ЗМІСТ

ВСТУП

Актуальність вибраної теми передбачає спрощення і полегшення багатьох аспектів торгових відносин шляхом створення якісного програмного забезпечення для мінімалізації можливих проблем із таким. Чим більш цивілізованим ми стаємо, тим більше потребуємо використання передових технологій у повсякденному житті. Ми сподіваємося, що наша країна не відстає від західного світу, і наші громадяни також прагнуть відмовитися від звичного і прагнуть до кращого.

Робити покупки в Інтернеті неймовірно зручно. Лише кількома кліками ми можемо купувати продукти, не виходячи з дому, не відвідуючи звичайний магазин. Ми також можемо робити покупки в Інтернеті в будь-який час доби, що робить його чудовим вибором для людей із напруженим графіком.

Доступність – інтернет-магазини зробили покупки доступними для людей, які можуть не мати легкого доступу до звичайних магазинів, наприклад тих, хто живе в сільській місцевості або має проблеми з пересуванням. Це дозволяє людям купувати продукти, які вони раніше не мали змоги.

Різноманітність – широкий вибір товарів, від книг і одягу до електроніки та продуктів. Це означає, що покупці мають доступ до більшого вибору продуктів, що дозволяє їм знайти саме те, що вони шукають.

Конкурентоспроможні ціни – інтернет-магазини часто пропонують конкурентоспроможні ціни, що може допомогти покупцям заощадити гроші. Вони також можуть пропонувати ексклюзивні знижки та пропозиції, недоступні у звичайних магазинах.

Загалом, таке програмне забезпечення стало невід’ємною частиною нашого життя, забезпечуючи зручність, доступність та широкий вибір товарів за конкурентними цінами.

Завданням поставленим на кваліфікаційну роботу є:

- розробка інтернет-магазину для продажу товарів онлайн;

- підключення серверного API до проєкту;
- використання одного із сучасних фреймворків таких як React, Vue, Angular, приорітетом є останні версії.

Метою дослідження є аналіз продуктивності та масштабованості фреймворків. Аналізуються показники швидкодії, час відгуку, оптимізаційні можливості та здатність розширювати веб-додаток для обробки більшого обсягу даних. також зосереджується на сумісності фреймворків з іншими технологіями та інструментами, які використовуються веб-розробкою. Аналізується можливість інтеграції з базами даних, сторонніми API, сервісами хмарного зберігання та іншими системами.

Об'єкт дослідження – використання сучасних фреймворків для розробки веб-сайтів. Основною метою дослідження є вивчення та аналіз ефективності та призначення різних фреймворків, їхніх можливостей та потенціалу у процесі розробки веб-додатків.

Предмет дослідження – програмна реалізація веб додатку на базі Vue.js та використання таких технологій, як NUXT 3. Дослідження включає аналіз процесу розробки веб-сайту з використанням обраного фреймворку. Вивчення кроків, методологій, рекомендацій та інструментів, які надають фреймворки, допоможе зрозуміти ефективність та продуктивність розробки.

РОЗДІЛ 1
АНАЛІЗ ДОДАТКІВ ДЛЯ ЗАРЯДКИ ЕЛЕКТРОКАРІВ І ПОСТАНОВКА ЗАВДАНЬ НА КВАЛІФІКАЦІЙНУ РОБОТУ

1.1 Аналіз сучасного стану проблеми

Інтернет-магазини революціонізували наш спосіб здійснення покупок, і їх вплив на наше життя незаперечний. Ось деякі з ключових сильних і слабких сторін, можливостей і загроз, пов’язаних з онлайн-магазинами:

Сильні сторони:

– як згадувалося раніше, онлайн-магазини неймовірно зручні, дозволяючи покупцям купувати продукти, не виходячи з дому;

– велика різноманітність, інтернет-магазини пропонують широкий асортимент товарів, завдяки чому покупцям легше знайти те, що вони шукають;

– конкурентоспроможні ціни, магазини в інтернеті часто пропонують конкурентоспроможні ціни, що може допомогти покупцям заощадити гроші.

– доступність, додатки роблять покупки доступними для людей, які можуть не мати легкого доступу до звичайних магазинів.

Слабкі сторони:

– онлайн-магазинам може бракувати особистої взаємодії та дотику, які приходять під час покупок у звичайному магазині;

– затримки доставки можуть бути розчаруванням для покупців, яким швидко потрібні їхні товари;

– технічні проблеми, такі як збої на веб-сайті або проблеми з обробкою платежів, можуть ускладнити покупки в Інтернеті.

– Можливості:

– інтернет-магазини мають потенціал для охоплення глобальної аудиторії, розширюючи свою клієнтську базу та збільшуючи продажі;

– додатки можуть використовувати дані та аналітику, щоб пропонувати персоналізовані рекомендації покупцям, покращуючи їхній загальний досвід;

– оскільки все більше людей використовують свої смартфони для онлайн-покупок, у онлайн-магазинів з’являється можливість оптимізувати свій мобільний досвід.

– Загрози:

– інтернет-магазини стикаються з жорсткою конкуренцією з боку інших інтернет-магазинів, а також звичайних магазинів, які розширюють свою присутність в Інтернеті;

– веб-сторінки вразливі до кібератак і витоку даних, що може завдати шкоди їхній репутації та призвести до фінансових втрат;

– інтернет-магазини можуть зіткнутися з регуляторними проблемами, пов’язаними з конфіденційністю та безпекою даних, що може вплинути на їх здатність вести бізнес.

В цілому інтернет-магазини змінили галузь роздрібної торгівлі та пропонують численні переваги покупцям. Однак вони також стикаються з проблемами, пов’язаними з конкуренцією, кібербезпекою та регулюванням, які необхідно вирішити, щоб забезпечити їхній подальший успіх.

Існує кілька способів створити інтернет-магазин, залежно від ваших технічних знань і бюджету. Існує кілька популярних варіантів, які будуть наведені нижче.

Платформи електронної комерції, такі як Shopify, WooCommerce і BigCommerce, надають усі інструменти, необхідні для створення онлайн-магазину та керування ним. Ці платформи пропонують заздалегідь розроблені шаблони, обробку платежів, інтеграцію доставки та інші функції, які полегшують запуск онлайн-магазину.

Серед переваг E-commerce сайтів можна відзначити розширене охоплення. Веб-сайти електронної комерції дозволяють компаніям охоплювати ширшу аудиторію та продавати продукти клієнтам, які знаходяться в різних частинах світу. Такі сторінки електронної комерції доступні 24/7, що означає, що клієнти можуть робити покупки, коли захочуть, а не обмежені розкладом роботи

магазину. Веб-сайти електронної комерції вимагають менших накладних витрат, ніж звичайні магазини, що може призвести до більшої норми прибутку.

Проєкти електронної комерції вимагають менших накладних витрат, ніж звичайні магазини, що може призвести до більшої норми прибутку. Збільшення продажів, де веб-сайти комерції можуть збільшити продажі, пропонуючи ексклюзивні знижки, перехресні продажі та інші стратегії.

Недоліки E-commerce веб-сайтів:

– відсутність особистої взаємодії. Веб-сайтам електронної комерції не вистачає особистої взаємодії та дотику, який приносить покупка у звичайному магазині;

– збої на веб-сайті або проблеми з обробкою платежів, можуть ускладнити онлайн-покупки та сприятимуть втраті довіри клієнтів;

– веб-сайти електронної комерції вразливі до кібератак і витоку даних, що може зашкодити довірі клієнтів і призвести до фінансових втрат;

– сайти електронної комерції можуть зіткнутися з регуляторними проблемами, пов’язаними з конфіденційністю та безпекою даних, що може вплинути на їх здатність вести бізнес.

Зразок створюваного веб-сайту на одній із E-commerce платформ (рис. 1.1).

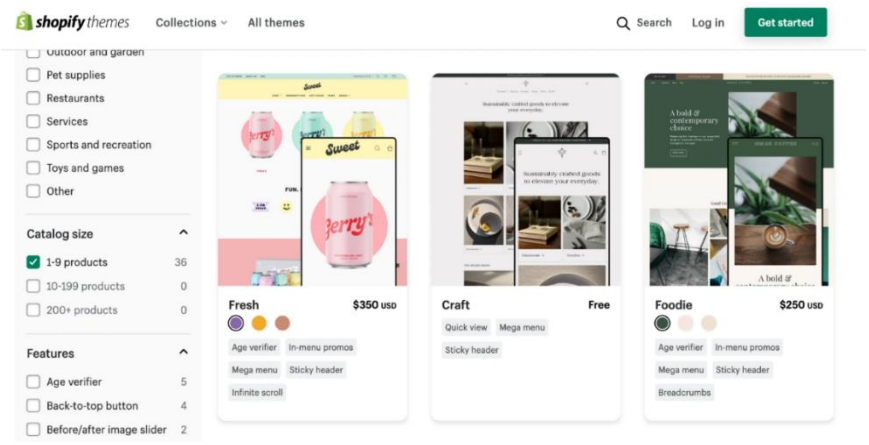

Рисунок 1.1 – Створення магазину на Shopify

Якщо у вас є досвід веб-розробки або бюджет, щоб найняти розробника, ви можете створити власний веб-сайт для свого інтернет-магазину. Ця опція забезпечує більшу гнучкість і контроль над дизайном і функціональністю вашого магазину.

Переваги сайтів на замовлення:

– веб-сайти пропонують унікальний дизайн, який можна адаптувати до потреб вашого бренду та бізнесу;

– спеціальні веб-сайти забезпечують більшу гнучкість і контроль над дизайном і функціональністю вашого сайту;

– сторінки можна масштабувати та налаштовувати відповідно до мінливих потреб вашого бізнесу;

– спеціальні додатки можна розробляти з урахуванням взаємодії з користувачем, забезпечуючи кращий досвід для відвідувачів і клієнтів;

– спеціальні веб-сайти можуть бути розроблені таким чином, щоб відображати ваш бренд і бізнес, що може сприяти розпізнаванню бренду та лояльності.

Недоліки власних сайтів:

– веб-сайти зазвичай дорожчі, ніж використання попередньо створених шаблонів або конструкторів веб-сайтів;

– користувальницькі веб-сайти розробляються довше, ніж готові шаблони або конструктори веб-сайтів;

– розробка спеціального веб-сайту потребує технічної експертизи, яка може стати перешкодою для компаній без власних розробників або бюджету для наймання сторонньої допомоги;

– веб-сайти потребують постійного обслуговування та оновлень, щоб гарантувати їх безпеку та належне функціонування;

– спеціальні веб-сайти можуть мати обмежені можливості підтримки, що може бути недоліком для підприємств, які потребують постійної підтримки та допомоги.

Онлайн-ринки, такі як Amazon, eBay і Etsy, дозволяють продавцям створювати власні вітрини на ринку. Цей варіант ідеально підходить для продавців, які хочуть охопити велику аудиторію без проблем зі створенням власного сайту.

Платформи соціальних мереж, такі як Facebook та Instagram, дозволяють продавцям створювати магазин у своєму профілі. Цей варіант ідеально підходить для продавців, які мають велику кількість підписників у соціальних мережах і хочуть продавати безпосередньо своїй аудиторії (рис. 1.2).

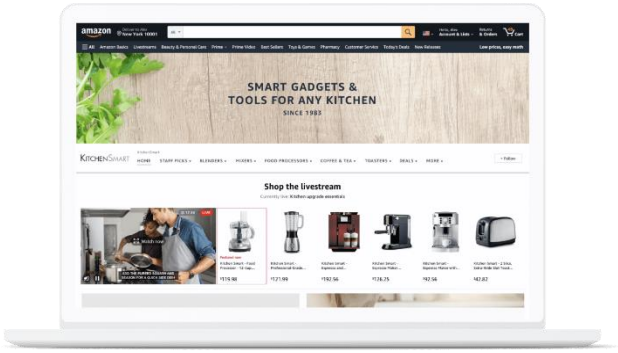

Рисунок 1.2 – Створення магазину на Amazon

Як інші способи створення сайту , магазини Instagram або Facebook мають свої переваги та недоліки. Такі платформи мають великі бази користувачів, а це означає, що існує велика потенційна аудиторія для ваших продуктів. Налаштувати магазин Instagram або Facebook відносно легко, і це можна зробити без будь-яких технічних знань. Магазини соціальних медіа можуть скористатися соціальним доказом, який стосується впливу інших на наші рішення. Якщо продукт популярний у соціальних мережах, це може підвищити сприйману цінність і бажаність продукту. Сторінки Instagram і Facebook можуть бути рентабельним способом продажу товарів, оскільки за відкриття магазину не стягується плата. Магазини в соціальних мережах пропонують інтегровані варіанти оплати та доставки, що полегшує процес покупки для клієнтів. Рисунок 1.3 відображає приклад інтернет-магазину у Instagram.

Недоліки магазинів Instagram або Facebook:

– такі платформи пропонують обмежені можливості налаштування, що може ускладнити створення унікального бренду;

– магазини Instagram і Facebook мають обмежену функціональність порівняно зі спеціальними веб-сайтами електронної комерції, що може обмежити можливість пропонувати розширені функції;

– магазини залежать від платформи, що означає, що вони підлягають будь-яким змінам або обмеженням, які накладає платформа;

– Instagram і Facebook пропонують обмежені дані та аналітику, що може ускладнити відстеження та аналіз поведінки клієнтів і продажів;

– алгоритми соціальних мереж можуть впливати на видимість ваших продуктів у стрічках користувачів, що може ускладнити охоплення цільової аудиторії.

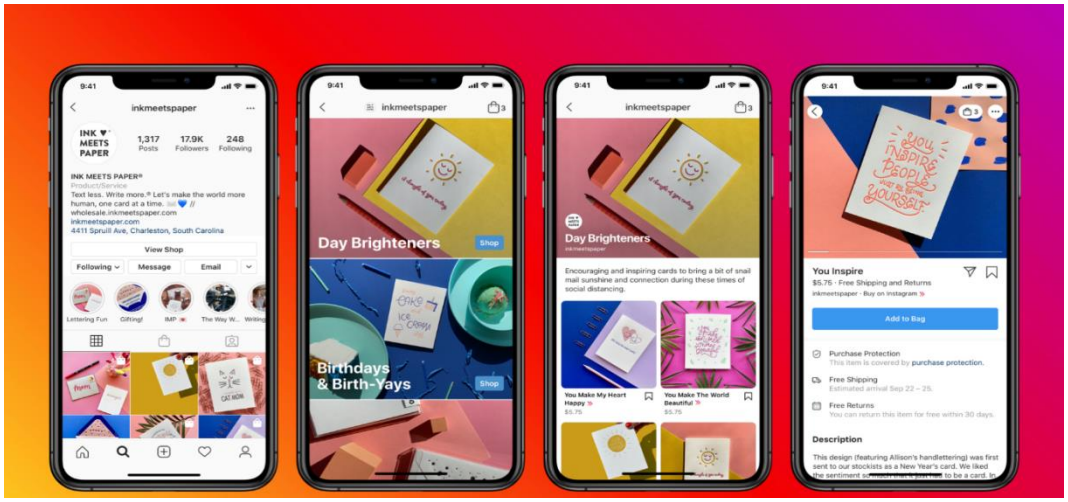

Рисунок 1.3 – Створення магазину за допомогою Instagram

Незалежно від того, який варіант ви виберете, важливо мати чітке уявлення про вашу цільову аудиторію, пропозицію продуктів і брендинг перед запуском вашого інтернет-магазину. Це гарантує, що ваш магазин розроблений відповідно до потреб ваших клієнтів і виділяється на переповненому онлайн-ринку. Варто

пам'ятати, якщо задумка сайту є унікальною, такий проєкт швидше за все можна реалізувати лише ручним написанням коду.

1.2 Постановка завдання на кваліфікаційну роботу бакалавра

Метою цього дослідження є порівняння та оцінка продуктивності різних сучасних фреймворків для розробки інтернет-магазину. Дослідження буде зосереджено на ключових функціях і функціональності популярних фреймворків, таких як Vue, Nuxt.

Для досягнення необхідної цілі необхідно дотримуватись наступного списку завдань:

– проаналізувати та порівняти функціонал фреймворків Vue, Nuxt;

– визначення сильних та слабких сторін кожного фреймворку для створення інтернет-магазину;

– вибір найбільш підходящої основи для створення інтернет-магазину;

– опис функціоналу;

– створення інтерфейсу;

– налаштування середовища розробки та технологій;

– розробка прототип інтернет-магазину, використовуючи кожен фреймворк, щоб продемонструвати їхні особливості та функціональність;

– тестування інтернет-магазину;

– розробка супутньої документації для програмного продукту.

Висновки до розділу 1

Було розглянуто найпопулярніші на сьогодні способи створення веб сторінок. Кожен з них має свою особливість. Завдяки перевагам та недолікам даних способів є можливість отримати досить якісний та сучасний продукт.

РОЗДІЛ 2
СПЕЦИФІКАЦІЯ ВИМОГ ДО РОЗРОБЛЮВАНОЇ СИСТЕМИ

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

Для проведення аналізу та визначення вимог до розроблюваного ПЗ необхідно:

– визначити предметну область і цілі ПЗ;

– проаналізувати потреби та вимоги користувачів та інших зацікавлених сторін;

– визначити функціональні та нефункціональні вимоги до ПЗ;

– вказати обмеження системи, такі як апаратне забезпечення, програмне забезпечення;

– створити випадки використання, діаграми послідовності та інші необхідні UML моделі, які описують поведінку ПЗ та взаємодію з користувачами;

– документація в стислій формі.

Керуючись ціллю та дослідженням предметної області інтернет-магазину, можна звертатись до декількох аспектів таких як, огляд літератури, він повинен забезпечувати поглиблений аналіз існуючої літератури про інтернет-магазини, включаючи їх історію, еволюцію та поточний стан ринку. Він також має охоплювати найкращі практики проектування та розробки онлайн-магазину, а також ключові характеристики та функціональні можливості успішних онлайн-магазинів. Методологія повинна описувати план дослідження, методи збору даних і методи аналізу даних, які будуть використовуватися в дослідженні. У ньому повинно бути обґрунтовано, як буде розроблено та розроблено інтернет-магазин, включаючи вибір відповідних технологій та інструментів. На сайті має бути повний каталог продукції з чітким описом, зображеннями та цінами на продукцію. Сайт повинен мати кошик для покупок, який дозволяє користувачам додавати продукти до свого кошика, переглядати вміст кошика та регулювати

кількість продуктів, які вони хочуть придбати. Безпечний процес оплати дозволить користувачам вводити платіжну інформацію та інформацію про доставку та вибирати бажаний спосіб оплати. Сайт повинен дозволяти користувачам відстежувати статус своїх замовлень, включаючи інформацію про доставку та дати доставки. Облікові записи клієнтів, що дозволятимуть користувачам контролювати свої облікові записи, у яких зберігається їх особиста та платіжна інформація, історія замовлень та інші деталі. Повинна бути присутньою функція пошуку, яка дозволяє користувачам швидко знаходити продукти, які вони шукають. Сайт повинен дозволяти користувачам залишати відгуки та оцінки продуктів, які вони придбали. Додаток повинен інтегруватися з платформами соціальних мереж, щоб дозволити користувачам ділитися продуктами зі своїми підписниками та друзями. У результатах повинні бути представлені результати дослідження, включаючи дизайн і розробку інтернет-магазину. Він має охоплювати особливості та функціональність онлайн-магазину, а також взаємодію з користувачем, продуктивність і безпеку. Результати також повинні включати порівняння інтернет-магазину з існуючими на ринку інтернет-магазинами. Висновок повинен підсумувати основні результати дослідження та підкреслити їх значення. Він також повинен надати рекомендації щодо майбутніх досліджень у сфері інтернет-магазинів. Система повинна бути розроблена для роботи з великою кількістю користувачів та повинна мати можливість обробляти запити та оновлювати їх в режимі реального часу. Обов’язкова сумісність із низкою веб-браузерів і пристроїв, включаючи настільні та мобільні пристрої. Взаємодія із користувачами на різних пристроях повинна бути узгодженою.

В цілому розроблена система має бути надійною, масштабованою та зручною для користувача. Вона має надавати низку функцій і функціональних можливостей для керування зарядними станціями та заряджанням електромобілів, а також повинен легко обслуговувати велику кількість користувачів і зарядних станцій.

2.2 Функціональні та нефункціональні вимоги

Варто розглянути функціональні та нефункціональні вимоги.

Функціональні вимоги:

– система повинна дозволяти клієнтам реєструвати обліковий запис, включно з наданням свого імені, електронної адреси та пароля.

– продукт мати вичерпний каталог продуктів, який включає зображення продуктів, описи, ціни та наявність;

– кошик для покупок повинен дозволяти клієнтам додавати та видаляти продукти зі свого кошика для покупок, переглядати загальну суму замовлення та переходити до оформлення замовлення;

– обробка платежів повинна дозволяти клієнтам оплачувати свої замовлення за допомогою різноманітних методів оплати, таких як кредитні картки, PayPal або інші онлайн-платіжні системи;

– система повинна дозволяти власнику магазину керувати та відстежувати замовлення клієнтів, включаючи перегляд деталей замовлення та обробку відшкодувань;

– клієнти матимуть доступ до служби підтримки клієнтів, включаючи контактні форми, поширені запитання та підтримку в чаті;

– система повинна надавати власнику магазину інструменти аналітики та звітності для відстеження відвідуваності веб-сайту, поведінки клієнтів і даних про продажі.

Нефункціональні вимоги:

– система повинна забезпечувати безпеку інформації про клієнтів, включно з використанням шифрування SSL для безпечних транзакцій і дотримання правил захисту даних;

– система повинна забезпечувати швидкий і оперативний досвід користувача, зі швидким часом завантаження сторінок і мінімальними простоями або помилками;

– розробка повинна бути в змозі обробляти великі обсяги трафіку та транзакцій і повинна бути масштабованою, щоб забезпечити зростання в майбутньому;

– система має бути розроблена таким чином, щоб вона була доступною для всіх користувачів, у тому числі для людей з обмеженими можливостями або з обмеженим доступом до Інтернету;

– додаток має бути простим у використанні та навігації, мати чіткий та інтуїтивно зрозумілий дизайн та функціональність;

– система має бути сумісна з низкою браузерів і пристроїв, включаючи мобільні пристрої та різні операційні системи;

– розробка повинна бути легкою для обслуговування та оновлення, мати доступ до технічної підтримки та регулярних оновлень програмного забезпечення.

Системні вимоги націлені на створення надійної системи інтернет магазину. Система даватиме можливість користувачам вибирати товар, оплачувати його та контролювати своїми даними через особистий кабінет користувача.

2.3 Моделі елементів програмного додатку у нотації UML

Щоб створити необхідні моделі, потрібно визначити ключові частини системи та те, як вони працюють разом, що робить кожна частина та як вони взаємодіють. Також потрібно визначити, як обмінюється інформацією між різними частинами системи, включаючи використовувані інтерфейси та структури даних.

Така модель допомагає зрозуміти сутність поставленого завдання та послідовність виконання залежностей.

Першою представлено діаграму послідовності для взаємодії користувача із веб-сайтом та сервером (рис. 2.1).

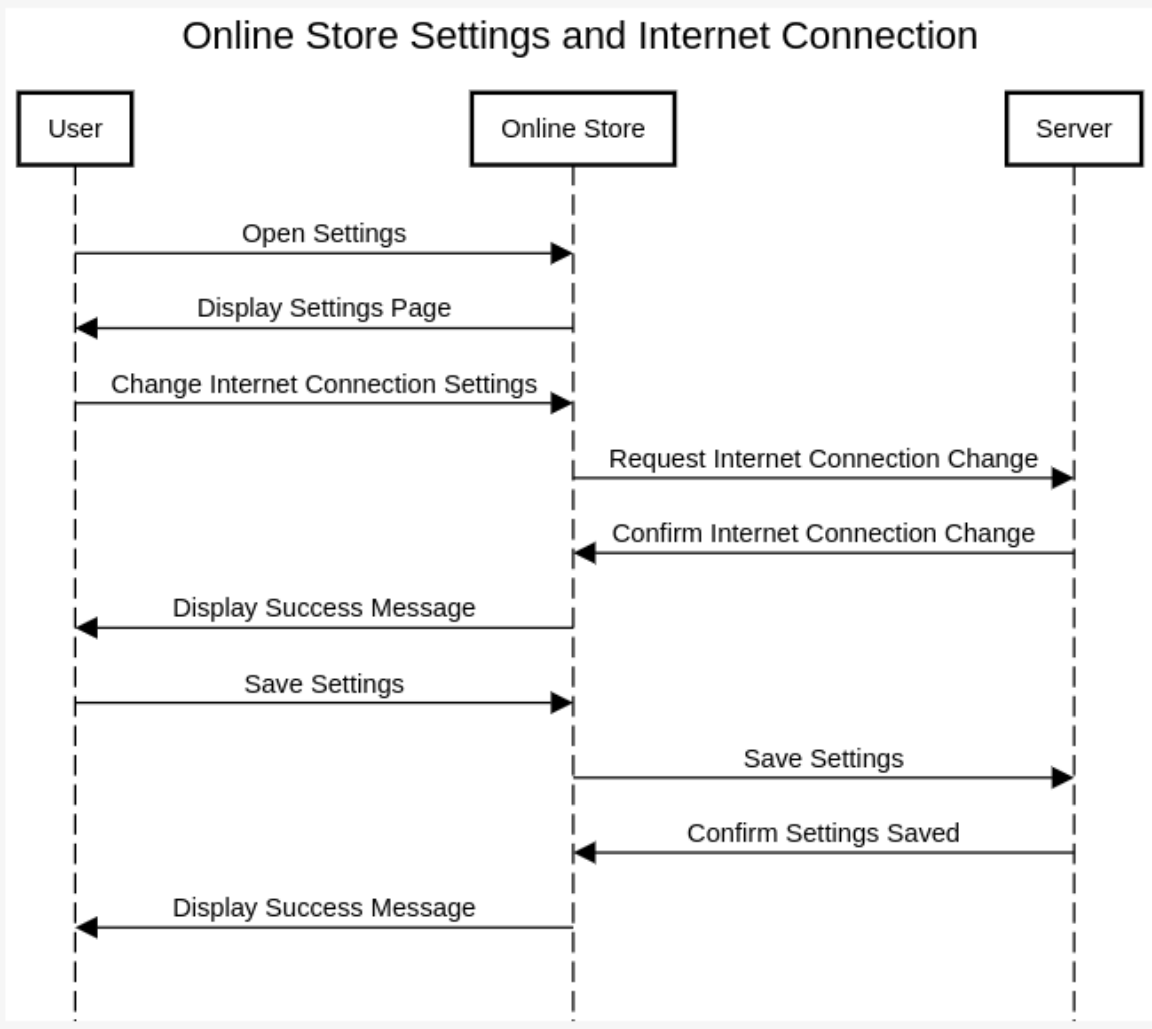

Рисунок 2.1 – Діаграма послідовностей для взаємодії користувача із веб-сайтом та сервером

Ця діаграма послідовності показує взаємодію між користувачем, онлайн-магазином і сервером під час покупки. Діаграма починається з того, що користувач переглядає продукти та вибирає продукт. Інтернет-магазин запитує інформацію про товар із сервера та відображає її користувачеві. Користувач додає товар у кошик і переходить до оформлення замовлення. Інтернет-магазин запитує інформацію про доставку із сервера та відображає форму для користувача. Користувач вводить інформацію про доставку та надсилає її на сервер. Сервер підтверджує інформацію про доставку, а інтернет-магазин відображає форму інформації про оплату. Користувач вводить платіжну інформацію та надсилає її

на сервер. Сервер підтверджує платіжну інформацію, а інтернет-магазин відображає підтвердження замовлення (рис. 2.2).

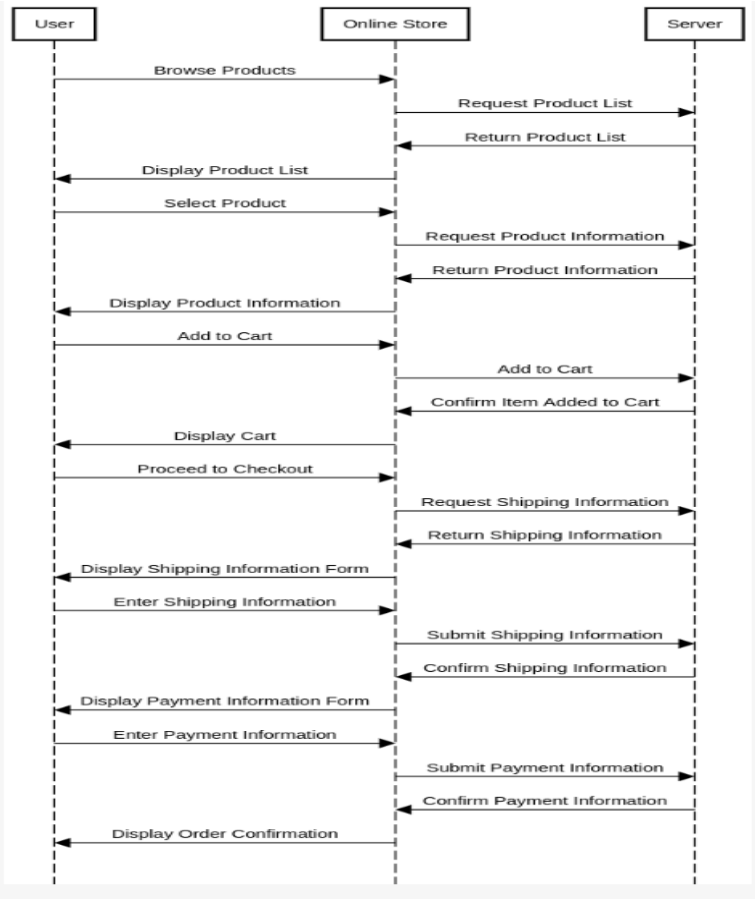

Рисунок 2.2 – Схема послідовності для взаємодії між користувачем, онлайн-магазином і сервером під час покупки

На рисунку 2.3 зображено діаграму послідовності яка показує відображає взаємодію між користувачем, онлайн-магазином і сервером під час процесу оплати. Діаграма починається з того, що користувач ініціює оформлення замовлення в інтернет-магазині. Інтернет-магазин запитує платіжну інформацію на сервері та відображає форму платіжної інформації для користувача. Користувач вводить платіжну інформацію та надсилає її в інтернет-магазин. Інтернет-магазин передає платіжну інформацію на сервер, і сервер обробляє платіж. Сервер оновлює статус замовлення та повертає інтернет-магазину

підтвердження оплати. Інтернет-магазин відображає користувачеві підтвердження оплати.

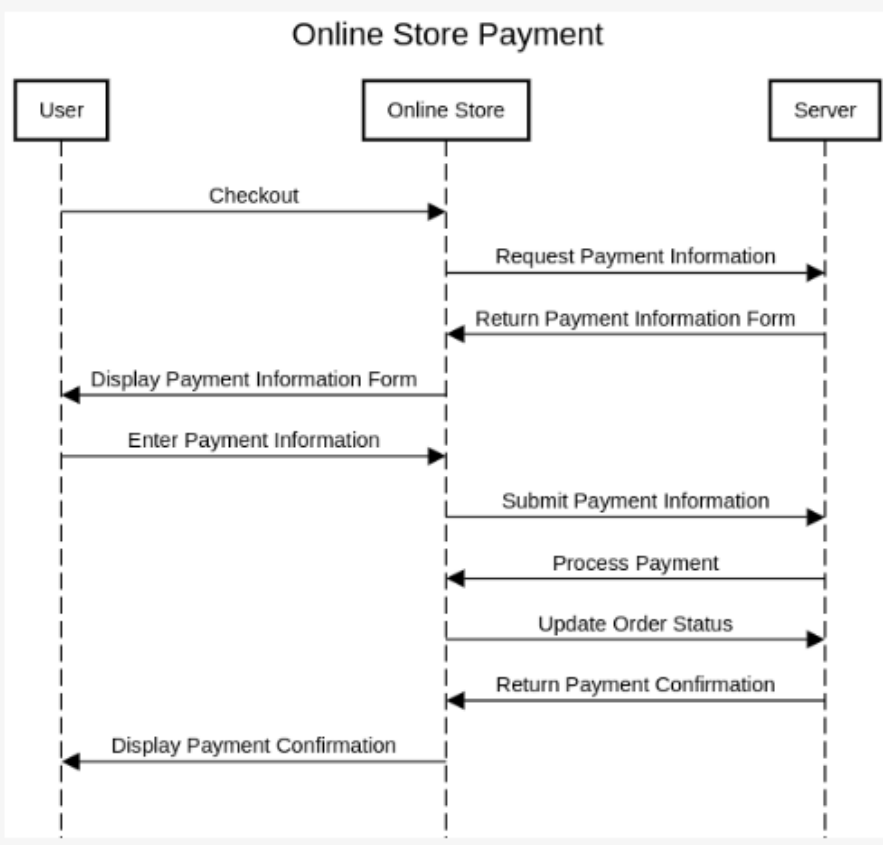

Рисунок 2.3 – Опрацювання платежу

2.4 Створення UI-дизайну

Figma – це хмарний інструмент для проектування, який дозволяє дизайнерам співпрацювати над проєктами дизайну інтерфейсу користувача (UI) у режимі реального часу. За допомогою Figma дизайнери можуть створювати та ділитися файлами дизайну, створювати прототипи інтерфейсів користувача та збирати відгуки від членів команди та зацікавлених сторін.

Користувачі можуть створювати інтерактивні прототипи з переходами та анімацією для імітації взаємодії з користувачем. Він пропонує ряд інструментів для проектування, включаючи векторне редагування, сітки макетів, компоненти дизайну та розширені можливості створення прототипів.

До основних функції Figma можна віднести функції співпраці в режимі реального часу Figma, такі можливості дозволяють декільком дизайнерам одночасно працювати над одним файлом дизайну, що полегшує співпрацю та повторення дизайнерських ідей. Хмарна платформа дизайну Figma дозволяє дизайнерам отримувати доступ до своїх проєктів з будь-якого місця, де є підключення до Інтернету, що полегшує співпрацю віддалених команд над проєктами дизайну інтерфейсу користувача. Вбудовані функції прототипування Figma дозволяють дизайнерам створювати інтерактивні прототипи своїх проєктів, полегшуючи тестування та вдосконалення своїх ідей. Figma дозволяє дизайнерам створювати багаторазові компоненти дизайну, які можна використовувати в кількох проєктах, полегшуючи підтримку узгодженості дизайну та прискорюючи процес проектування. Функції контролю версій Figma дозволяють дизайнерам відстежувати зміни у своїх файлах дизайну та повертатися до попередніх версій, якщо необхідно. Figma також інтегрується з іншими популярними інструментами проектування та розробки, забезпечуючи безперебійний робочий процес та інтеграцію з такими платформами, як Sketch, Zeplin і Jira. Дизайнери можуть створювати багаторазові елементи, такі як кнопки, піктограми та панелі навігації, що полегшує підтримку узгодженості та вносить глобальні зміни в дизайн.

Figma здобула популярність завдяки своїм функціям співпраці, простоті використання та здатності покращити робочий процес проектування. Він широко використовується окремими дизайнерами, командами дизайнерів і навіть організаціями для створення візуально приголомшливих і зручних дизайнів.

Вибраний графічний редактор є потужним інструментом для дизайнерів інтерфейсу користувача, які хочуть співпрацювати над дизайнерськими проєктами в режимі реального часу та оптимізувати свій процес розробки (рис. 2.4). Рисунок відображає головну сторінку сайту. Тут ми маємо можливість побачити інформацію що надає інтернет-магазин, а саме: основні сторінки, каталог, категорії товарів та інформацію про місцезнаходження інтернет-магазину.

На рисунку 2.5 зображено сторінку реєстрації, завдяки якій користувач створює профіль і отримує можливість робити покупки та відстежувати їх.

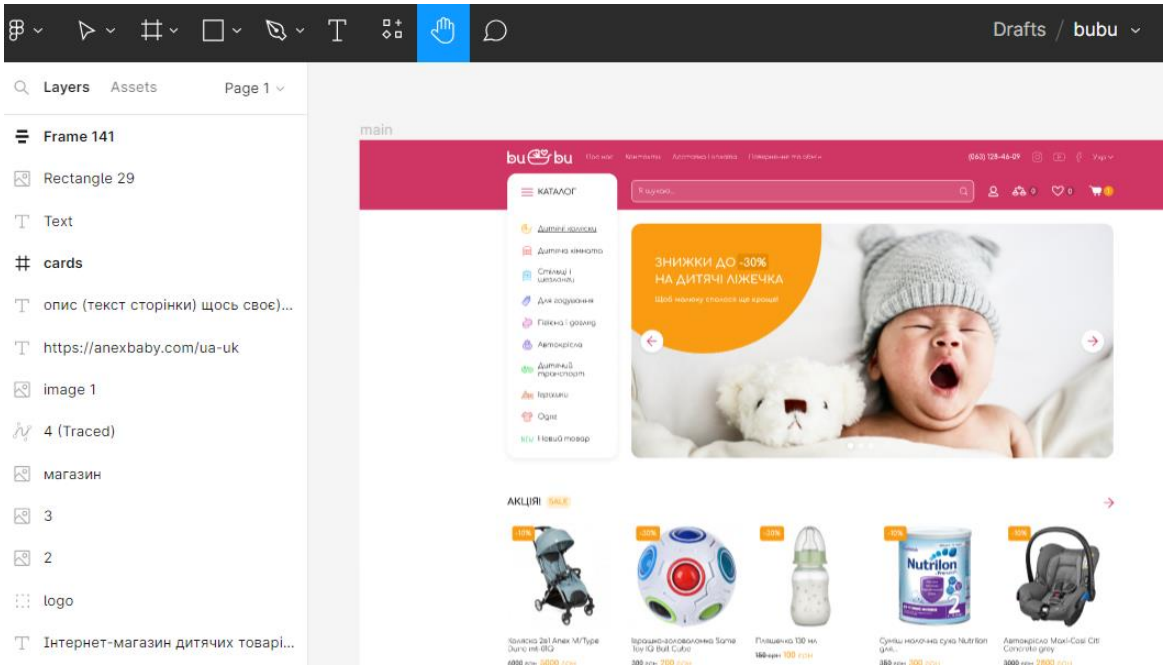

Рисунок 2.5 – Сторінка реєстрації

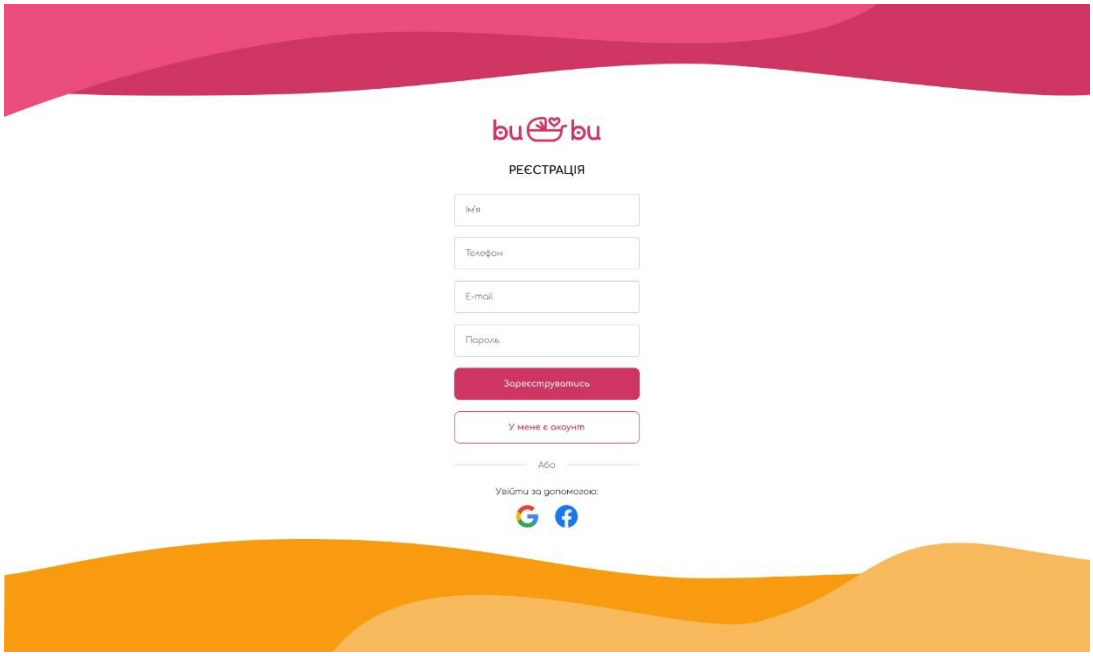

Рисунок 2.6 відображає форму входу на сайт.

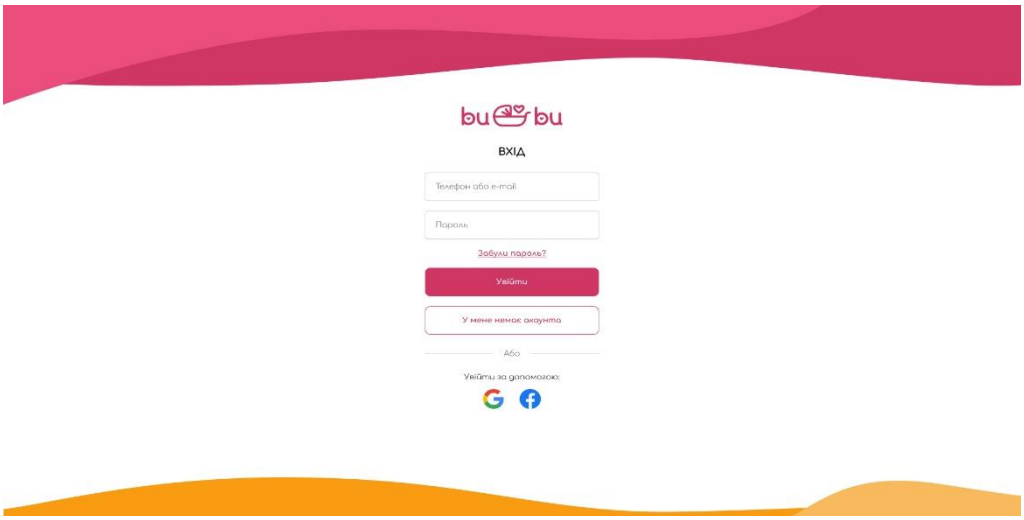

Рисунок 2.6 – Сторінка входу

На рисунку 2.7 відображено форму особистий кабінет користувача.

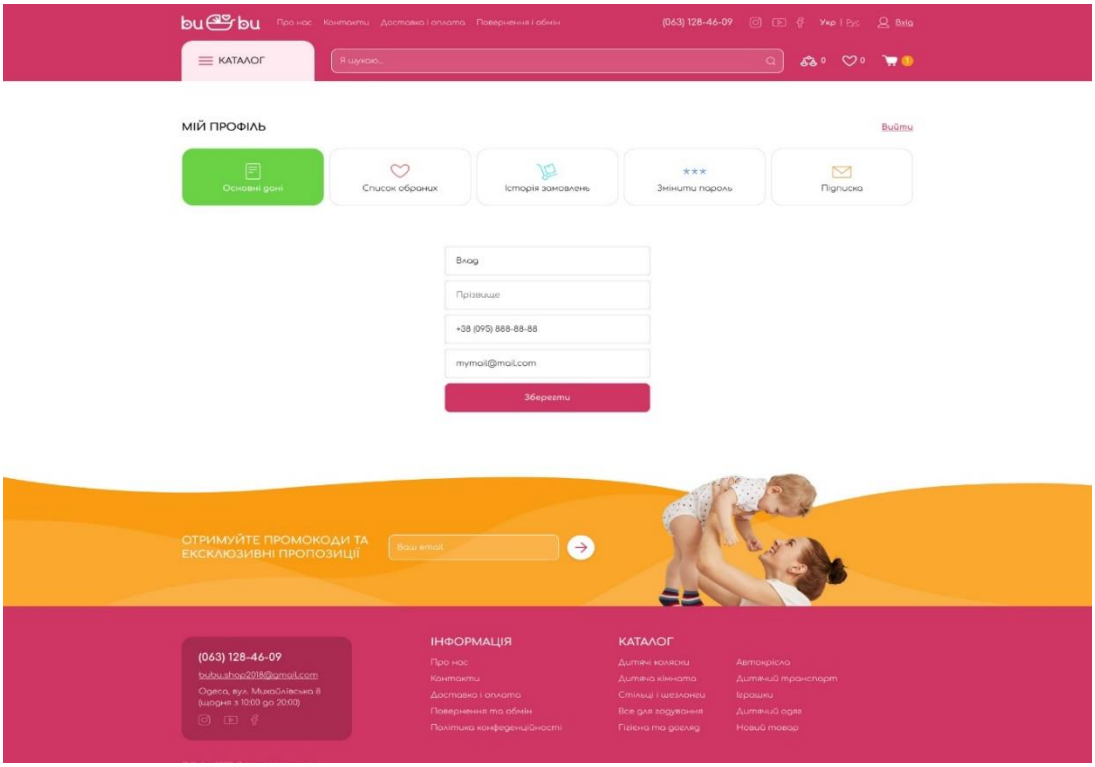

Рисунок 2.7 – Особистий кабінет користувача

Інтернет-магазин зазвичай має каталог товарів, який клієнти можуть переглядати. Цей каталог повинен містити докладний опис товару, зображення, ціни та інформацію про наявність .

Кошик для покупок є важливим компонентом інтернет-магазину, який дозволяє клієнтам додавати товари до свого кошика, а потім переходити до оформлення замовлення.

Процес оформлення повинен бути простим та інтуїтивно зрозумілим, із доступними різними варіантами оплати та доставки. Клієнти повинні мати можливість переглянути своє замовлення та внести зміни, перш ніж завершити покупку.

Інтернет-магазин повинен мати систему керування замовленнями, щоб відстежувати всі замовлення, відправлення та повернення. Ця система має бути простою у використанні та забезпечувати оновлення статусу замовлення в реальному часі.

Повинна бути присутня система для керування обліковими записами та даними клієнтів, включаючи історію замовлень, уподобання та особисту інформацію.

Управління запасами має вирішальне значення для будь-якого інтернет-магазину, щоб забезпечити постійну доступність продуктів і уникнути перепродажу. Він повинен мати можливість керувати рівнем запасів, відстежувати рух запасів і створювати звіти про запаси.

Загалом, інтернет-магазин повинен забезпечувати безперебійний та інтуїтивно зрозумілий досвід покупок для клієнтів, пропонуючи основні можливості управління та звітності для власника бізнесу. Така цифровізація власної справи спрощує документування та додає автоматизації. В подальшому проєкт можна модернізувати, додавши нові функції та можливості.

2.5 Вибір засобів, методів і алгоритмів вирішення поставленого завдання

Запорукою успішного створення веб-сайту стануть наступні чинники:

– магазин повинен мати всі необхідні функції, щоб клієнти могли легко переглядати та купувати товари. Це включає такі функції, як пошук і фільтрація, описи продуктів і зображення, кошик для покупок і процес оформлення замовлення;

– розробка повинна забезпечувати безперебійну та зручну взаємодію для клієнтів. Це включає в себе просту навігацію, інтуїтивно зрозумілий дизайн і адаптивний веб-дизайн для мобільних пристроїв;

– продукт має бути безпечним і захищати конфіденційні дані клієнтів, наприклад платіжну інформацію. Це включає впровадження шифрування SSL, сумісності з PCI та інших заходів безпеки;

– додаток має бути швидким і чуйним, з мінімальним часом завантаження сторінок і зображень продуктів. Це включає оптимізацію веб-сайту для забезпечення швидкості та продуктивності та використання надійного хостинг-провайдера;

– розробка повинна мати можливість обробляти зростаючий обсяг трафіку та продажів у міру зростання бізнесу. Це включає забезпечення того, що веб-сайт та інфраструктура хостингу можуть задовольняти підвищений попит, не впливаючи на продуктивність;

– магазин повинен мати можливість інтегруватися з іншими системами, такими як платіжні шлюзи, постачальники послуг доставки та системи управління запасами;

– додаток має передбачати налаштування та брендинг, щоб відповідати унікальній ідентичності та стилю бізнесу.

Vue.js, Angular і React – три популярні фреймворки JavaScript, які використовуються для створення веб-додатків. Хоча всі вони служать подібній меті, вони мають різні функції, підходи та пов’язані з ними спільноти. Вибір між Vue.js, Angular і React часто залежить від конкретних вимог проекту, досвіду команди розробників і особистих уподобань. Кожен фреймворк має свої сильні та слабкі сторони, і перед прийняттям рішення важливо враховувати такі фактори, як розмір проекту, складність, потреби в продуктивності та доступні ресурси.

Вибором для розробки став Vue.js 3 і Nuxt 3-ї версії. Причиною даного вибору були їхні переваги, що з’явились у останніх, оновлених версіях. У Vue 3 представлено кілька оптимізацій, які призводять до швидшого рендерингу та менших розмірів пакетів. Новий алгоритм візуалізації, відомий як засіб візуалізації «на основі шаблонів», ефективніший і швидший, ніж попередній засіб візуалізації «віртуального DOM» . Vue 3 має менший розмір пакета завдяки кращому струсу дерева та поділу коду, що призводить до меншого часу завантаження. API композиції – це нова функція, яка дозволяє розробникам структурувати свій код у більш модульний спосіб, який можна багаторазово використовувати, що спрощує організацію та обслуговування коду. Vue представляє налаштовувані директиви, що дозволяє розробникам створювати власні власні директиви, які можна використовувати в усій програмі. У фреймворку представлена нова система реактивності, яка полегшує розробникам створення реактивних властивостей даних і уникає поширених проблем із реактивністю. На рисунку 2.8 відображено логотип Vue.js.

Рисунок 2.8 – Логотип Vue

Nuxt 3 – остання версія фреймворку Nuxt.js, яка побудована на основі фреймворку Vue.js. Основною причиною його вибору були внутрішні налаштування, які суттєво спрощують розробку. У списку нижче деякі з переваг використання Nuxt 3:

– фреймворк пропонує кілька оптимізацій продуктивності, зокрема швидший час запуску, кращий рендеринг на стороні сервера та оптимізоване поділ коду;

– спрощена розробка, яку Nuxt 3 надає розробникам простий та інтуїтивно зрозумілий API для створення складних програм. Він містить потужну систему модулів, яка дозволяє розробникам легко додавати нові функції до своїх програм;

– покращена пошукова оптимізація забезпечує вбудовану підтримку візуалізації на стороні сервера, що може значно покращити пошукову оптимізацію вашої програми. Він також включає такі функції, як автоматичне створення карти сайту та динамічні мета-теги;

– потужна система проміжного програмного забезпечення. Вибраний фреймворк постачається з потужною системою проміжного програмного забезпечення, яке дозволяє розробникам легко додавати спеціальну логіку до запитів і відповідей своїх програм;

– Nuxt 3 розроблено з модульною архітектурою, яка дозволяє розробникам легко повторно використовувати компоненти у своїх програмах. Це полегшує створення та підтримку складних програм;

– покращено підтримку TypeScript, що полегшує розробникам написання безпечного коду;

– Nuxt містить вбудовану підтримку для розгортання вашої програми на популярних платформах хостингу, таких як Netlify і AWS;

– у загальному плані Nuxt 3 надає розробникам потужну та гнучку структуру для створення сучасних веб-додатків. Його зосередженість на продуктивності, простоті та модульності робить його чудовим вибором для

розробників, які прагнуть створювати масштабовані та підтримувані програми (рис.2.9).

Рисунок 2.9 – Логотип Nuxt 3

Laravel – це популярна веб-платформа PHP, яка надає потужні інструменти для створення веб-додатків, включаючи компоненти на стороні сервера. Серверні компоненти веб-додатку відповідають за обробку запитів від клієнтів, обробку даних і генерацію відповідей.

Переваги використання Laravel для серверної розробки:

– Laravel поставляється з багатьма вбудованими функціями, які спрощують розробку веб-додатків, включаючи інструменти автентифікації, маршрутизації та міграції бази даних;

– Laravel розроблений як модульний, що означає, що ви можете легко додавати нові компоненти та функціональні можливості до своєї програми за потреби;

– Laravel дотримується архітектури Model-View-Controller (MVC), яка розділяє логіку програми на окремі компоненти для полегшення обслуговування та масштабованості;

– Laravel Eloquent ORM (Object-Relational Mapping) полегшує роботу з базами даних, дозволяючи визначати схеми баз даних як класи PHP і використовувати їх для взаємодії з даними;

– механізм створення шаблонів Blade Laravel забезпечує потужний та інтуїтивно зрозумілий спосіб створення багаторазових компонентів перегляду, що полегшує створення динамічних веб-сторінок;

– Laravel містить вбудовані інструменти тестування, які спрощують написання та виконання автоматизованих тестів, гарантуючи, що ваша програма працює належним чином.

Надається потужна та гнучка структура для створення серверних компонентів для веб-додатків, що робить його популярним вибором серед розробників (рис.2.10).

Рисунок 2.10 – Логотип Laravel

Висновки до розділу 2

При спільному використанні Vue 3, Nuxt 3 і Laravel можуть сформувати потужний стек для створення сучасних веб-додатків. Laravel може служити серверним API для керування даними та бізнес-логікою, тоді як Vue 3 і Nuxt 3 обробляють зовнішній інтерфейс користувача та рендеринг. Можливості SSR і SSG Nuxt 3 можуть покращити пошукову оптимізацію та скоротити час початкового завантаження сторінки.

Поєднуючи Vue 3, Nuxt 3 і Laravel, ви отримуєте потужний повний стек для розробки, який забезпечує безперебійну розробку, ефективний рендеринг, покращене SEO та багату екосистему інструментів і бібліотек. Однак важливо враховувати вимоги вашого проєкту, досвід команди та конкретні випадки використання, щоб переконатися, що цей стек відповідає вашим потребам.

РОЗДІЛ 3
РОЗРОБКА ІНТЕРНЕТ-МАГАЗИНУ З ВИКОРИСТАННЯМ VUE 3 І NUXT 3

3.1 Практична реалізація об’єкта проектування

На початку написання коду потрібно визначитись із структурою проєкту та де будуть розміщені основні файли розроблюваного програмного забезпечення. Користуючись редактором коду VSCode було створено проєкт (рис. 3.1) та налаштовано його середовище. Основа проєкту створюється максимально просто, за допомогою команди, яку надає офіційна документація фреймворку.

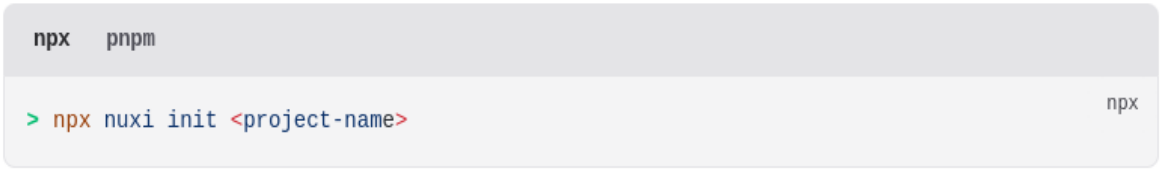

Рисунок 3.1 – Створення проєкту Nuxt

Наступним кроком стане налаштування самого проєкту та встановлення потрібних залежностей. Для цього мною був використаний пакетний менеджер NPM. Серед необхідних пакетів були встановлені такі як зображено на рисунку 3.2.

Рисунок 3.2 – Встановлення потрібних бібліотек

Такі залежності зазвичай є готовими рішеннями певних задач, які плануються виконати для свого проєкту. Для прикладу Swiper – система для гортання фото, у моєму випадку це будуть фото товару. Або ж object-to-form-data допомагає змінити формат об’єкту на такий, що буде відправлятись на back-end(прикріплена фотографія до відгуку на товар).

Vue.js – компонентно орієнтований фреймворк. Отже, наступним кроком стане створення компонентів (сторінок або їх окремих частин) для веб-сайту. Перед створенням таких компонентів важливою складовою є уважне ознайомлення із документацією. Назви таких компонентів мають певні правила або ж стандарти, яких склались з часом у спільноті розробників. Усі компоненти створено у папці components (рис. 3.3).

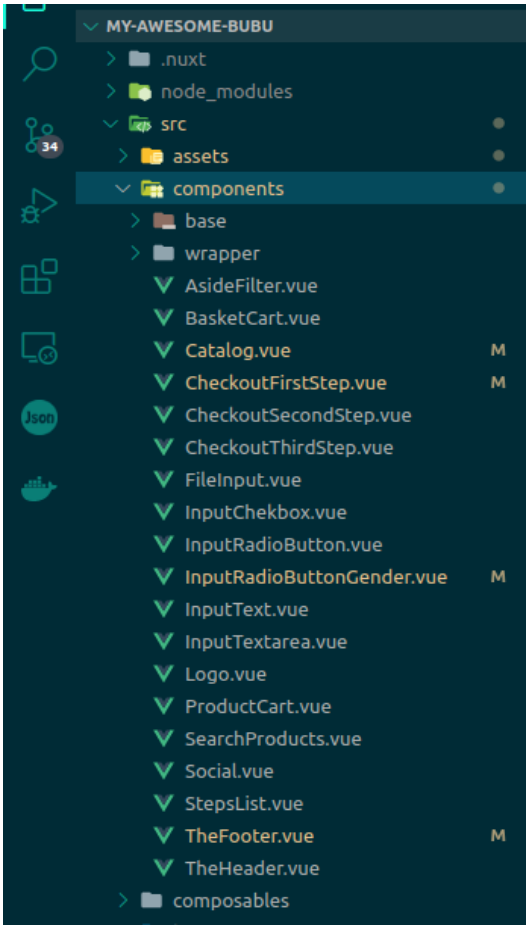

Рисунок 3.3 – Створення компонентів сайту

Після створених компонентів маємо можливість їх перевикористовувати, що і є перевагою компонентного підходу. Такі елементи матимуть свої властивості, під час кожного їх виклику, якщо такі дані потрібно змінити – використовується defineProps. Функція defineProps зазвичай використовується у функції налаштування компонента Vue 3. Це дозволяє вам явно визначити властивості, які ваш компонент очікує отримати. Визначаючи атрибути, ви надаєте чіткий інтерфейс для зв'язку між компонентами та полегшуєте розуміння того, як дані надходять у вашу програму. Використання компоненти на сторінці зображено нижче. Використовується компонента BasketCart яка отримує властивості basketItems.

Для маршрутизації між сторінками у Nuxt використовується тег NuxtLink, що прийшов на зміну звичному тегу <a> із HTML. Досить додати тільки атрибут to і шлях до сторінки на яку здійснюватиметься перехід. Автоматично застосовує активний клас до активного посилання, дозволяючи вам легко стилізувати посилання поточної сторінки, щоб вказати його активний стан (рис. 3.4, рис. 3.5).

```
<template>
    <main class="main">
        <div class="container">
            <div class="basket">
                <div class="basket__selects">
                    <div class="basket__head">
                        <h4 class="section__head">Кошик</h4>
                        <a href="" class="basket__link">Переглянуті товари</a>
                    </div>

                    <div class="basket__content">
                        <div class="basket__article">
                            <!--class="basket__categories basket__active" for reviewed pages page-->
                            <p class="basket__categories--item">Товар</p>
                            <p class="basket__categories--item">Кількість</p>
                            <p class="basket__categories--item">Сума</p>
                        </div>

                        <BasketCart @clickIsHere="clickHandler" :options="basketItems" />

                    </div>
                    </div>
                    <div class="basket__sum">
                        <div class="basket__order">
                            <div class="basket__enter">
                                <p class="basket__order--name">Сума:</p>
                                <p class="basket__order--sum">{{ sumTotalBasket }} грн</p>
                            </div>
        <p class="basket__order--name">Сума:</p>
        <p class="basket__order--sum">{{ sumTotalBasket }} грн</p>
    </div>
    <div class="basket__select">
        <nuxt-link to="/checkout" class="button button--order">Оформити замовлення</nuxt-link>
        <nuxt-link to="/catalog" class="button-light button--order">Продовжити покупки</nuxt-link>
    </div>
</div>
div>
```

Рисунок 3.5 – Використання NuxtLink

До теми маршрутизації також можна віднести і динамічний роутинг. Однією із особливостей Nuxt 3 є можливість динамічного роутингу (рис.3.6). Система маршрутизації на основі файлів дозволяє створювати динамічні маршрути, використовуючи структуру файлів і папок вашого проєкту. Структура проєкту нижче дає нам таку можливість, кожен товар матиме своє id.

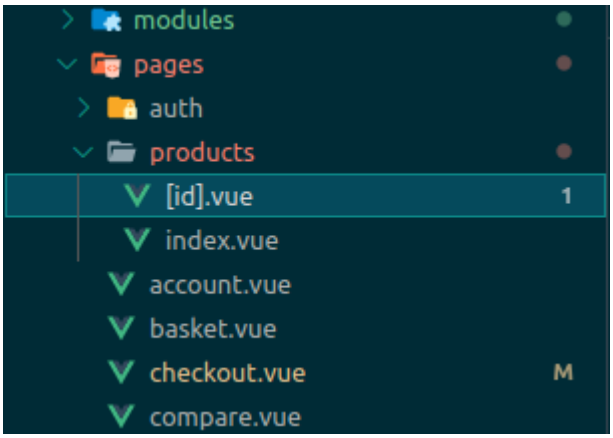

Рисунок 3.6 Використання динамічного роутингу Nuxt

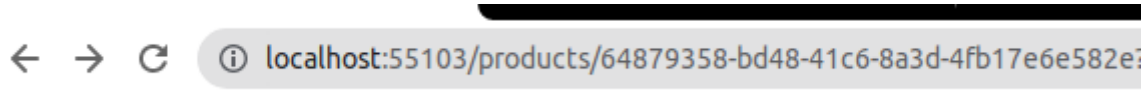

Рисунок 3.7 – Вигляд роуту в браузері

Якщо поглянути на використання цього методу у браузері, отримаємо такий результат у пошуковому рядку (рис. 3.7).

Практично на кожній сторінці розроблюваного додатку використовується компоненти TheHeader та TheFooter (рис. 3.8). Такі елементи відносяться до базових, через їх перевикористання майже на кожній із сторінок проєкту. Отже, основною ідеєю буде внесення їх до стандартного макету сторінки. Такий макет буде автоматично підвантажуватись на кожній із сторінок уже без обов'язкових додаткових властивостей чи налаштувань.

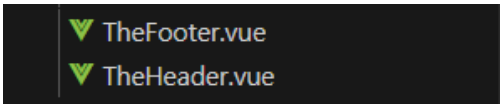

Рисунок 3.8 – Компонент поля вводу що використовує v-model

Одним із основних елементів в розробці інтернет-магазину є компонент картки товару, така картка буде використовуватись як для секції рекомендацій так і для загального каталогу товарів. Такий компонент було створено успішно.

Також функціонально реалізовано фільтри товарів які допомагають у сортуванні таких відповідно до поставлених критерій користувача, для прикладу критерії діапазону ціни або фільтрування по брендах.

Для гнучкості вибору методу доставки, який включає у себе метод оплати було створено 3 компоненти відповідно 3-м крокам верифікації. На першому кроці реалізовано можливість вибору служби доставки. Другий крок являє собою вибір спосіб доставки (відділення пошти, доставка додому, тощо). І наршеті, третім кроком є вибір платіжної системи для оплати вибраних товарів.

Для можливості відстеження покупок розроблено персональний кабінет із досить широким спектром можливостей. Користувач здатний керувати персональною інформацією, змінювати дані для авторизації для підтримки або актуалізації таких.

Важливим аспектом vue 3 є можливість використання v-model, а точніше – це двостороння директива зв’язування даних у Vue.js, яка дозволяє створювати зв’язок між вхідними даними форми та даними компонента. Це спрощує процес синхронізації даних між вхідними елементами та властивостями даних компонента (рис. 3.9).

```
<form class="checkout__department" v-if="changeDelItem === 'department'">
    <InputText customClass="input--checkout" name="city" placeholder="Виберіть місто" v-model="formData.city" />
    <InputText customClass="input--checkout" name="department" placeholder="Виберіть відділення" v-model="formData.department"/>
</form>
```

Рисунок 3.9 – Компонент поля вводу що використовує v-model

Якщо передати у v-model змінну, то її значенням буде результат поля вводу у моєму випадку (рис. 3.10).

```
<InputText customClass="profile__item" placeholder="Ім'я" name="name" v-model="formData.name"/>
<InputText customClass="profile__item" placeholder="Прізвище" v-model="formData.lastName"/>
<InputText customClass="profile__item" placeholder="Номер телефону" name="phone" v-model="formData.phone"/>
<InputText customClass="profile__item" placeholder="E-mail" name="email" v-model="formData.email"/>
<button  type="submit" class="button btn--signin">Зберегти</button>
```

Рисунок 3.10 – Використання v-model на сторінці

Для відправлення певної форми для перевірки на сервері, або передачі данних мною використовувався бібліотека VeeValidate. Він забезпечує простий і декларативний спосіб перевірки введених користувачем даних у формах. VeeValidate пропонує широкий спектр правил перевірки, налаштовувані повідомлення про помилки та різноманітні методи перевірки. Для його використання потрібен тег VeeFrom замість тегу form.

Часто перевірки vee-from за допомогою validation-schema допомагає уникнути введення зайвих символів або перевірити правильність їх використання, таку перевірки можна не налаштовувати вручну, відповідно потрібним даним. Vee Validate спрощує процес перевірки форм у програмах Vue.js, надаючи зручний спосіб переконатися, що введені користувачем дані відповідають певним критеріям перед надсиланням форми. Це допомагає покращити взаємодію з користувачем, скеровуючи користувачів надавати дійсні дані та відображати важливі повідомлення про помилки, коли перевірка не вдається. На рисунках 3.11 та 3.12 відображено вище сказане.

```
// Define a validation schema
const schema = yup.object({
    name: yup.string().required().min(6),
    email: yup.string().required().email(),
    password: yup.string().required().min(8),
    phone: yup.number().required().min(8),
});
```

Рисунок 3.11 – Налаштування схеми перевірки

```
<vee-form class="signin__form" @submit="postDataMethod" :validation-schema="schema">
    <InputText customClass="input--signin" placeholder="Ім'я" name="name" v-model="formData.name" />
    <InputText customClass="input--signin" placeholder="Телефон" name="phone"
        v-model="formData.phone" />
    <InputText customClass="input--signin" placeholder="E-mail" name="email" v-model="formData.email" />
    <InputText customClass="input--signin" placeholder="Пароль" type="password" name="password"
        v-model="formData.password" />
    <button type="submit" class="button btn--signin">Зареєструватись</button>
    <NuxtLink to="/auth" class="button-light btn--signin">У мене є акаунт</NuxtLink>
    <div class="sigin__or">
        <div class="signin__select">
            <p class="signin__select--text">Або</p>
        </div>
        <div class="signin__name">
            <p class="signin__name--head">Увійти за допомогою:</p>
            <div class="signin__icons">
                <a href="#" class="checkout__link"><img src="/img/google-logo.svg" alt="google-logo"
                        class="checkout__link--img"></a>
                <a href="#" class="checkout__link"><img src="/img/facebook-chekout.svg"
                        alt="facebook-chekout" class="checkout__link--img"></a>
            </div>
        </div>
    </div>
</vee-form>
```

Рисунок 3.12 Використання vee-form

Наступним етапом розробки було налаштування http запитів. Axios – це популярна бібліотека JavaScript, яка використовується для створення HTTP-запитів із веб-браузерів і середовищ Node.js. Він надає простий у використанні API для виконання асинхронних запитів HTTP, обробки перетворень запитів і відповідей, а також обробки помилок. Оскільки у Nuxt такою бібліотекою по замовчуванню є Axios то встановлювати нічого не потрібно було, лише налаштування (рис. 3.13).

```
import _ from "lodash";
export default defineNuxtPlugin(() => {

  const config = useRuntimeConfig(),
    token = useCookie('token'),
    route = useRoute(),
    router = useRouter();
  const axiosSettings = $fetch.create({

    headers: {
      Accept: "application/json",
      "Cache-Control": "no-cache",
      Authorization: `Bearer ${token.value}`,
      shost: `paw.shop`,
      scart: ``
    },
    baseURL: config.public.API_BASE_URL,
    async onRequest({ request, options }) {
      const cartId = useCookie("cartId")
      const token = useCookie("token");
      options.headers["Authorization"] = `Bearer ${token.value}`;
      options.headers["scart"] = `${cartId.value}`

      // Log request
      // console.log("[fetch request]", request, options);
    },
    async onRequestError({ request, options, error }) {
    },
```

Рисунок 3.13 – Налаштування Axios

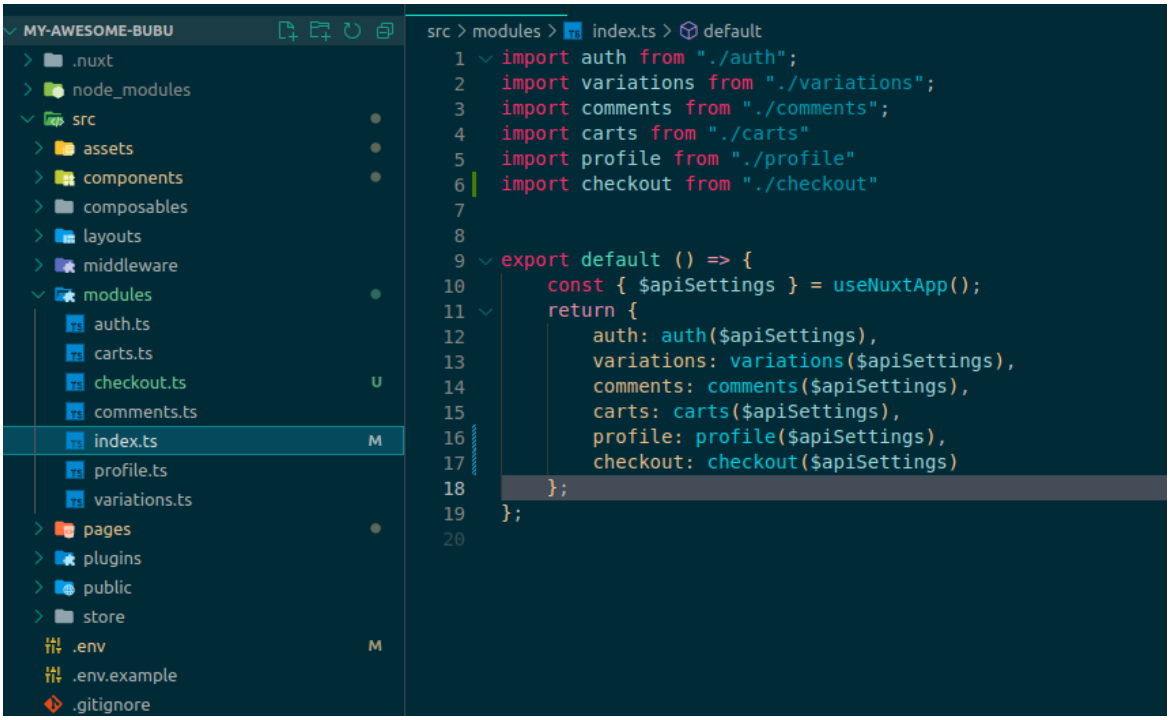

Рисунок 3.14 – Створення категорій запитів

Для кращої читабельності коду було розбито на окремі категорії запити http і об’єднано в одному файлі (рис. 3.14).

Кожен із таких запитів було написано відповідно до RestAPI. RESTful API (Representational State Transfer API) – це архітектурний стиль для розробки мережевих програм. Він базується на принципах архітектурного стилю REST і використовує протокол HTTP для зв'язку між клієнтами та серверами. RESTful API зазвичай використовуються для створення веб-служб, які дозволяють клієнтам взаємодіяти з ресурсами сервера. Один із таких запитів зображено на рисунку нижче (рис. 3.15).

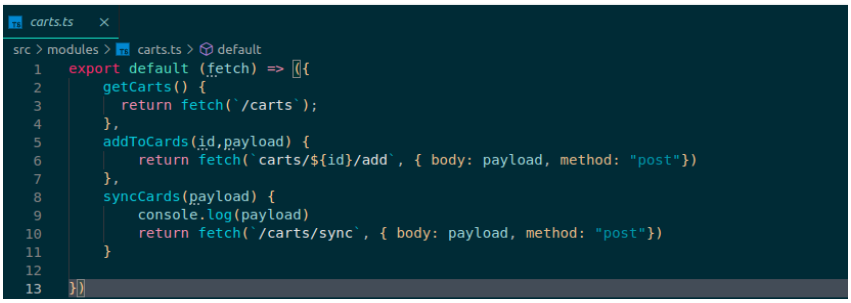

Рисунок 3.15 – Запит HTTP

Як можна побачити на фото, за правилами, кожен запит повинен бути названий інтуїтивно зрозуміло, відповідно до свого призначення. Запит GetCarts, наприклад, був використаний на сторінці корзини товарів, для отримання товарів, що було додано до кошику. Така функція має вигляд рисунку 3.16.

```
async function getCarts() {
    try {
        const shopCarts = await $api().carts.getCarts()
        basketItems.value = shopCarts.data.purchases
        console.log(basketItems)
        sumTotalBasket.value = shopCarts.total.total
        $snotify.success("success");
    }
    catch (e) {
        $snotify.error(e.data);
    }
}
```

Рисунок 3.16 – Надсилання запиту через фунцію на сторінці

Ключовою можливістю Nuxt є middleware, а точніше функції відіграють подібну роль, як і в інших фреймворках веб-розробки. Вони дозволяють перехоплювати та обробляти вхідні запити до того, як вони досягнуть компонентів сторінки або макета. Проміжне програмне забезпечення в Nuxt.js 3 можна використовувати для таких завдань, як автентифікація, отримання даних і захист маршруту. Так як авторизація відбувається шляхом збереження token значення, для коректної роботи token повинен бути видимим на всіх сторінках проєкту. Для цього його записують у middleware. У Nuxt 3 проміжне програмне забезпечення дозволяє визначати функції, які виконуються перед рендерингом сторінки або групи сторінок. У межах функції проміжного програмного забезпечення ви можете виконувати різні завдання, такі як перевірка автентифікації, виклики API, зміна сховища або перенаправлення користувача. Він надає можливість додавати спеціальну логіку або виконувати такі завдання, як перевірка автентифікації, вибірка даних або зміна контексту сторінки перед рендерингом (рис. 3.17).

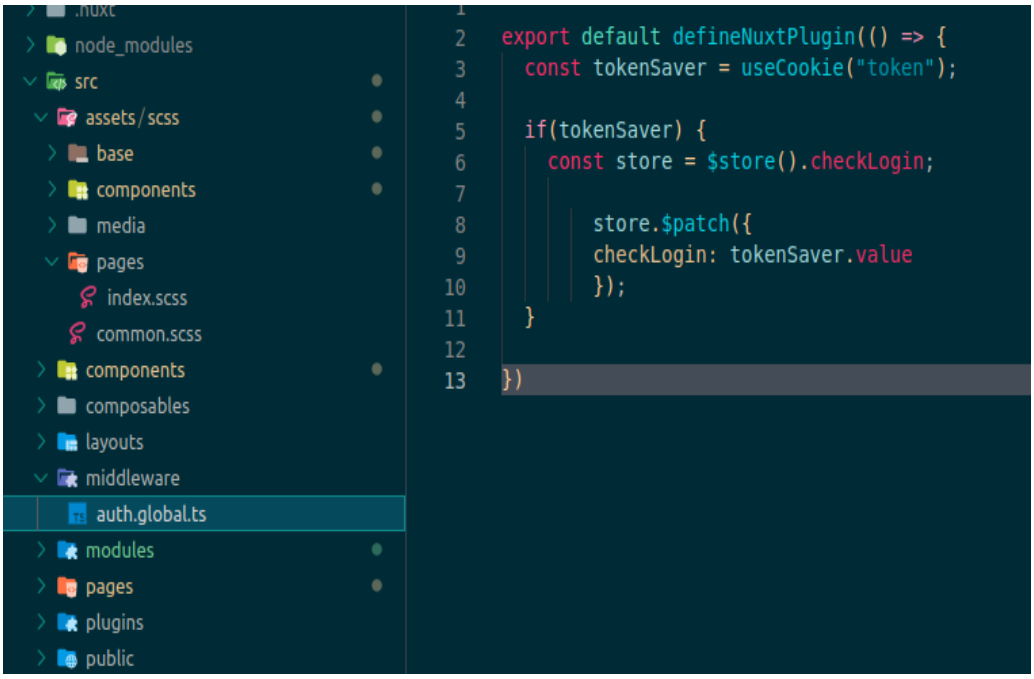

Рисунок 3.17 – Використання middleware

Стилізація відповідно макету здійснювалась за допомого препроцесора SCSS. Таку структуру можна побачити у папці assets/scss/components (рис. 3.18). SCSS (Sassy CSS) – це розширення CSS (каскадних таблиць стилів), яке додає кілька функцій і покращує стандартний синтаксис CSS. SCSS є популярним вибором для стилізації веб-додатків завдяки своїй розширеній функціональності та читабельності. Використовуючи SCSS у Vue 3, ви можете скористатися перевагами таких функцій, як змінні, вкладення, міксини тощо. Це може значно підвищити вашу продуктивність і зробити ваші стилі зручнішими для обслуговування.

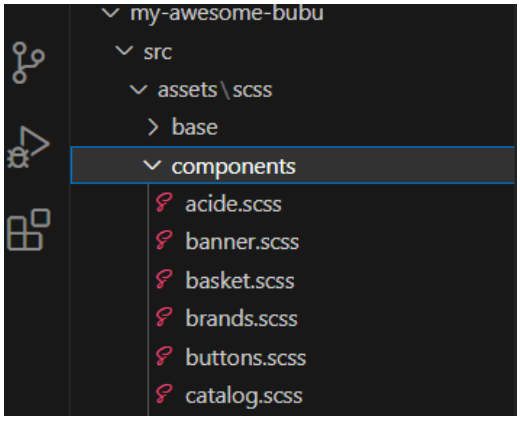

Рисунок 3.18 – Структура файлів стилізації проєкту

Оскільки однією із вимог до розробки на сьогодення я можливість користуватись веб-сторінкою з різних девайсів, було створено адаптивний дизайн сайту за допомогою медіа запитів. Адаптивний макет, також відомий як адаптивний макет, відноситься до проектування та реалізації веб-сайту або веб-додатку, які можуть налаштовувати та адаптувати свій макет на основі характеристик пристрою користувача та розміру екрана. Мета адаптивного макета – забезпечити оптимальну взаємодію з користувачем на різних пристроях, таких як настільні комп’ютери, ноутбуки, планшети та мобільні телефони. Такі запити мали наступний вигляд, відповідно до своєї контрольної точки. У моєму випадку згідно макету такими точками були: 1280px, 768px, 360px. Медіа-запити є фундаментальною функцією CSS, яка дозволяє застосовувати різні стилі на основі

характеристик пристрою або області перегляду. У Vue 3 ви можете використовувати медіа-запити в стилях ваших компонентів, щоб створювати адаптивні дизайни, які адаптуються до різних розмірів екрана (рис .3.19).

```scss
@media screen and (max-width: 768px) {

    .checkout {

        &__order {
            display: none;
        }
        &__content {
            flex-direction: column;
        }

        &__popup {
            box-shadow: 0px 4px 30px rgba(0, 0, 0, 0.07);
            display: block;
            padding: 30px;
            border-radius: 20px;
        }
    }
}
```

Рисунок 3.19 – Використання медіа запитів SCSS

MySQL – це система управління реляційними базами даних. Для реалізації веб-сайту до серверу, підключено базу даних MySQL (рис. 3.20)

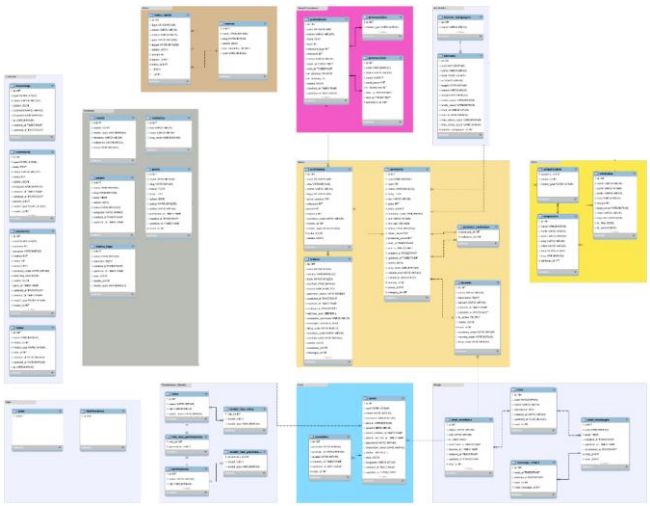

Рисунок 3.20 – Схема бази даних сайту

У серверній частині сайту база даних MySQL зберігає інформацію про користувачів, продукти, замовлення, тощо. API для авторизації та аутентифікації управляє реєстрацією та входом. API для продуктів дозволяє вивести, оновити дані. API для користувачів управляє CRUD операціями для облікових записів користувачів. API для замовлень управляє створенням та відслідковуванням замовлень. API для відгуків дає змогу залишити та передивитися відгуки.

3.2 Тестування та налагодження інформаційно-комп'ютерної системи

Під час веб-сайту було проведено ретельні процеси тестування та налагодження для переконання в її надійності та зручності для користувачів. Тестування веб-сайту, розробленого з використанням Vue 3, Nuxt 3, Laravel, може включало різні типи тестів для перевірки функціональності, надійності та коректності роботи розробленого додатку. Для потрібного тестування було використано наступні методи:

– юніт-тести перевіряють окремі компоненти, модулі або функції вашого коду для переконання, що вони працюють правильно. Ви можете використовувати фреймворки для тестування, такі як Jest або Mocha, для написання і запуску юніт-тестів для вашого Vue-або Laravel-коду;

– функціональні тести перевіряють як взаємодіє ваша програма з зовнішніми компонентами або API. Ви можете використовувати інструменти, такі як Selenium або Cypress, для автоматизованого тестування взаємодії користувача з вашим веб-сайтом;

– інтеграційні тести перевіряють як компоненти вашого веб-сайту взаємодіють між собою. Ви можете написати тести, які симулюють запити до вашого API, перевіряють правильність обробки даних та співпрацю різних модулів;

– тести користувальницького інтерфейсу перевіряють як користувачі взаємодіють з вашим веб-сайтом і перевіряють, чи працює інтерфейс правильно.

Ці тести можуть бути автоматизованими (наприклад, за допомогою Selenium або Cypress) або виконуватися вручну;

– PageSpeed означає вимірювання того, наскільки швидко веб-сторінка завантажується та відтворює її вміст. Це важливий аспект продуктивності веб-сайту та взаємодії з користувачем. Користувачі зазвичай залишають веб-сайти, які завантажуються повільно, тому оптимізація PageSpeed має вирішальне значення для утримання відвідувачів і покращення взаємодії (рис. 3.21).

Рисунок 3.21 – Тестування PageSpeed

Таким чином, тестування є невід'ємною частиною процесу розробки веб-сайтів, створених за допомогою Vue 3, Nuxt 3 і Laravel. Це веде до покращення якості, підвищеної надійності, покращення взаємодії з користувачем, економії коштів і часу, підвищення безпеки, впевненості в розгортанні та сприяє культурі постійного вдосконалення. Розставляючи пріоритети тестування, розробники можуть створити високопродуктивні, надійні та зручні веб-сайти, які відповідають очікуванням їхніх користувачів.

3.3 Розробка документації для програмного продукту

Ця документація призначена для розробників, які працюватимуть над проєктом онлайн-магазину. Вона містить інструкції та рекомендації щодо розробки, налаштування та підтримки цього проєкту.

Хороша документація може значно покращити досвід роботи з новими розробниками, які приєднуються до проєкту, і гарантувати, що кожен має чітке розуміння того, як працювати з проєктом Nuxt 3 і налаштовувати його.

Серверні вимоги:

– операційна система: Linux або Windows;

– рекомендовано використовувати Apache або Nginx у якості веб-серверу;

– мова програмування: PHP 7.4 або новіше;

– рекомендовано використовувати MySQL або PostgreSQL у якості бази даних.

– Клієнтські вимоги:

– підтримка сучасних веб-браузерів, таких як Google Chrome, Mozilla Firefox, Safari або Microsoft Edge.

– Фронтенд:

– використовуйте Vue 3 для розробки користувацького інтерфейсу;

– рекомендовано використовувати Nuxt.js для створення універсальних додатків.

Для створення серверної частини використовується Laravel для розробки серверної логіки та API. Забезпечення взаємодії між фронтендом і бекендом за допомогою HTTP-запитів.

У якості бази даних використовуйте MySQL або PostgreSQL для збереження даних про продукти, замовлення та користувачів.

Установка та налаштування:

– завантажте проєкт;

– склонуйте репозиторій з вихідним кодом проєкту на свій локальний комп'ютер.

– Налаштування середовища:

– встановіть необхідне програмне забезпечення, таке як PHP, Composer, Node.js та NPM;

– створіть конфігураційний файл для з'єднання з базою даних та іншими налаштуваннями;

– виконайте команду composer install для встановлення залежностей PHP;

– виконайте команду npm install для встановлення залежностей JavaScript.

– Запуск проєкту:

– виконайте команду php artisan serve для запуску локального веб-сервера;

– відкрийте браузер і перейдіть за адресою http://localhost:8000, щоб переглянути онлайн-магазин.

Пункт реєстрації та авторизація користувачів включає дозвіл користувачам створювати облікові записи та увійти в систему для здійснення покупок. Каталог товарів реалізований із можливістю перегляду списку товарів з їхніми характеристиками та цінами. Додайте функціонал пошуку та фільтрації товарів за категоріями, цінами тощо.

Кошик покупок:

– дозвольте користувачам додавати товари до кошика;

– реалізуйте функціонал збереження та видалення товарів з кошика;

– відображайте загальну суму товарів у кошику.

Оформлення замовлення дає дозвіл користувачам виконувати процес оформлення замовлення, включаючи введення адреси доставки та способу оплати. Підтверджуйте замовлення та надсилайте підтвердження на електронну пошту користувача.

Документація надає чіткі кроки, які потрібно виконати, і включає приклади, діаграми та кодові фрагменти для полегшення розуміння імплементації. Вона також наголошує на важливості дотримання кращих практик і рекомендацій, що сприяє ефективності, надійності та безпеці проєкту.

Висновки до розділу 3

Було проведено ініціалізацію проєкту. Встановлено необхідні для розробки бібліотеки та аналіз функціональних можливостей фреймворків. Для реалізації потрібного функціоналу використані всі найсучасніші методи виконання. У результаті створено готовий продукт із сучасними показниками швидкості та взаємодії з користувачем.

ВИСНОВКИ

У процесі розробки програмної системи на базі сучасних фреймворків, таких як Vue 3 та Nuxt 3, були встановлені важливі задачі, пов'язані з розробкою ефективного та функціонального інтернет-магазину. Було проведено аналіз сучасного стану предметної області розробки на Vue 3 та Nuxt 3, виявлені проблеми, з якими можуть зіткнутися розробники при роботі з цими фреймворками, та проведено порівняння із аналогами.

Один з основних об'єктів дослідження полягав у вивченні функціональних вимог до розробки інтернет-магазину. Були визначені вимоги до інтерфейсу користувача, технічні вимоги, процесні вимоги та нефункціональні вимоги. Ці вимоги визначили основні функції та характеристики програмної системи.

Для реалізації програмної системи були обрані методи та засоби, такі як Vue 3 та Nuxt 3, які відповідають вимогам та задачам розробки. Обґрунтування вибору цих методів та засобів було проведено з урахуванням їх потужності, гнучкості, широкої спільноти та підтримки.

Було розроблено архітектуру програмної системи, реалізовано її функціонал та проведено тестування для перевірки якості та коректності роботи системи. В результаті тестування було отримано позитивні результати, що підтверджують працездатність та відповідність системи вимогам.

Описуючи функції Vue 3 та Nuxt 3, було зазначено їх потужність, простоту використання та можливості для розширення функціоналу. Розробка на базі цих фреймворків дозволяє створювати інтерактивні та динамічні веб-додатки з покращеною продуктивністю та ефективним управлінням станом даних. Vue 3 надає зручний синтаксис та можливості компонентного підходу, що сприяє легкості розробки і підтримці коду. Використання Nuxt 3 дозволяє швидко побудувати універсальні додатки, які підтримують серверний та клієнтський рендеринг.

У результаті дослідження та розробки програмної системи на базі Vue 3 та Nuxt 3 було показано, що ці фреймворки є потужними і ефективними

інструментами для розробки сучасних веб-додатків. Вони дозволяють забезпечити швидку та зручну розробку, покращену продуктивність, масштабованість та надійність програмної системи.

Описані вимоги, обґрунтування вибору методів та засобів, а також реалізація та тестування програмної системи підтверджують важливість та доцільність використання Vue 3 та Nuxt 3 для розробки сучасних веб-додатків. Ці фреймворки надають розробникам сучасні та доступні інструменти, які сприяють швидкому розгортанню, покращенню користувацького досвіду та забезпеченню високої якості програмного продукту.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Vue.js документація URL: https://vuejs.org/guide/introduction.html (дата звернення: 22.04.2023).

2. The Intuitive Web Framework URL: https://nuxt.com/ (дата звернення: 23.04.2023).

3. Створення RESTful Web API за допомогою Node.js, Express: https://medium.com/@onejohi/building-a-simple-rest-api-with-nodejs-and-express-da6273ed7ca9 (дата звернення: 25.04.2023).

4. Методичні вказівки до виконання кваліфікаційної роботи бакалавра : метод. рек. / уклад.: Н. М. Ліщина, А. А. Ящук, О. М. Суринович. Луцьк, 2022. 56 с.

5. Matt Stauffer Laravel: Up & Running: A Framework for Building Modern PHP Apps 2nd Edition (дата звернення: 25.04.2023).

6. Vue.js Documentation URL: https://v3.vuejs.org/guide/introduction.html (дата звернення: 27.04.2023).

7. Nuxt.js - Tutorial URL: https://www.w3schools.com/whatis/whatis_nuxt.asp (дата звернення: 27.04.2023).

8. Advantages and disadvantages of Websocket URL: https://www.wallarm.com/what/websocket-vs-http-how-are-these-2-different (дата звернення: 28.04.2023).

9. SQL Tutorial URL: https://www.w3schools.com/sql/ (дата звернення: 28.04.2023).

10. MySQL Documentation URL: https://dev.mysql.com/doc/ (дата звернення: 28.04.2023).

11. PHP Manual URL: https://www.php.net/manual/en/ (дата звернення: 06.05.2023).

12. Адам Крут, Фредерік Джонсон. Coding HTML CSS JavaScript Made Easy. Web, Apps and Desktop. Flame Tree 2019 (дата звернення: 06.05.2023).

13. Vue.js 2 Cookbook: Build modern, interactive web applications with

Vue.js by Andrea Passaglia 2022 (дата звернення: 06.05.2023).

14. Laravel Tutorial URL: https://laracasts.com/series/laravel-8-from-scratch (дата звернення: 08.05.2023).

15. Хенчет Е., Листуон Б. Vue.js у дії. Україна 2019. (дата звернення: 08.05.2023).

16. Ерік Фрімен, Елізабет Робсон, Берт Бейтс, Кеті Сієрра. Книга Head First. Патерни проєктування. Україна 2022 (дата звернення: 09.05.2023).

17. Sequence Diagram Tutorial – Complete Guide with Examples URL: https://creately.com/guides/sequence-diagram-tutorial/ (дата звернення: 9.05.2023).

18. MDN Web Docs URL: https://developer.mozilla.org/en-US/ (дата звернення: 11.05.2023).

19. Аллен Тейлор SQL для чайників. Діалектика. Україна 2020 (дата звернення: 11.05.2023).

20. CSS Tutorial URL: https://www.w3schools.com/css/ (дата звернення: 13.05.2023).

21. Stack Overflow URL: https://stackoverflow.com/ (дата звернення: 14.05.2023).

ДОДАТКИ

ДОДАТОК А

	Chrome mount	Chrome update avg	Chrome update best	Chrome memory (mb)
Vue 2 template + with	93.46	23.17	16.44	11.9
Vue 3 template + with	77.43	9.18	7.69	4.5
Improvement over v2 (in-browser compiled)	20.70%	152.40%	113.78%	-62.18%
Vue 2 render fn (manual h, no this access)	90.9	15.75	8.18	10.2
Vue 3 render fn (manual h, no this access)	76.46	7.56	5.33	7.4
Improvement over v2 (manual render function)	18.89%	108.33%	53.47%	-27.45%
Vue 2 template no with	97.44	13.18	8.29	9.9
Vue 3 template no with	62.82	5.64	3.78	4.5
Improvement over v2 (pre-compiled)	55.11%	133.69%	119.31%	-54.55%
Vue 2 raw (no reactive state)	88.67	12.27	7.99	8.6
Vue 3 raw (no reactive state)	47.6	3.37	2.38	3.7
Improvement over v2 (raw, no reactive state)	86.28%	264.09%	235.71%	-56.98%

Порівняльна характеристика Nuxt.js

Feature / Version	Nuxt 2	Nuxt Bridge	Nuxt 3
Vue	2	2	3
Stability	😊 Stable	😌 Semi-stable	😬 Unstable
Performance	🏎️ Fast	🛩️ Faster	🚀 Fastest
Nitro Engine	❌	✅	✅
ESM support	🌙 Partial	👍 Better	✅
TypeScript	☑️ Opt-in	🚧 Partial	✅
Composition API	❌	🚧 Partial	✅
Options API	✅	✅	✅
Components Auto Import	✅	✅	✅
`<script setup>` syntax	❌	🚧 Partial	✅
Auto Imports	❌	✅	✅
Webpack	4	4	5
Vite	⚠️ Partial	🚧 Partial	🚧 Experimental
Nuxi CLI	❌ Old	✅ nuxi	✅ nuxi
Static sites	✅	✅	🚧

Порівняльна характеристика Vue.js