МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ АВІАЦІЙНИЙ УНІВЕРСИТЕТ

ДОПУСТИТИ ДО ЗАХИСТУ
Завідувач випускової кафедри
_____Аліна САВЧЕНКО
«___»_____2024 р.

КВАЛІФІКАЦІЙНА РОБОТА
(ПОЯСНЮВАЛЬНА ЗАПИСКА)

ЗДОБУВАЧА ОСВІТНЬОГО СТУПЕНЯ **“БАКАЛАВР”**

Тема: «Вебзастосунок інтернет-магазину взуття на основі технологій Vue JS, Node JS»

Виконавець: Артем ВАНЗИТЛЕР

Керівник: к.т.н., доцент Олена ТОЛСТІКОВА

Нормоконтролер: к.т.н., доцент Вікторія СИДОРЕНКО

КИЇВ 2024

НАЦІОНАЛЬНИЙ АВІАЦІЙНИЙ УНІВЕРСИТЕТ

Факультет *комп'ютерних наук та технологій*

Кафедра *комп'ютерних інформаційних технологій*
Спеціальність *122 «Комп'ютерні науки»*
Освітньо-професійна програма *«Інформаційні технології проєктування»*

ЗАТВЕРДЖУЮ:
завідувач кафедри КІТ
_____Аліна САВЧЕНКО
(підпис)
«____»_____2024 р.

ЗАВДАННЯ

на виконання кваліфікаційної роботи

Ванзитлера Артема Сергійовича
(прізвище, ім'я, по батькові здобувача вищої освіти в родовому відмінку)

1. Тема кваліфікаційної роботи: «Вебзастосунок інтернет-магазину взуття на основі технологій Vue JS, Node JS»,
затверджена наказом ректора від «05» квітня 2024 р. № 517/ст.
2. Термін виконання роботи: з 06 травня 2024 року по 16 червня 2024 року.
3. Вихідні дані до роботи: Вебзастосунок інтернет-магазину взуття на основі технологій Vue JS, Node JS.
4. Зміст пояснювальної записки: 1) Аналіз предметної області. 2) Вибір технологій та інструментів розробки. 3) Розробка вебзастосунка.
5. Перелік обов'язкового ілюстративного матеріалу: слайди презентації MS PowerPoint.

6. Календарний план-графік

№з/п	Завдання	Термін виконання	Підпис керівника
1.	Аналіз предметної області. Написання 1 розділу, представлення керівнику.	06.05.2024-10.05.2024	
2.	Вибір технологій та інструментів розробки. Написання 2 розділу, представлення керівнику.	13.05.2024-17.05.2024	
3.	Розробка вебзастосунка. Написання 3 розділу, представлення керівнику.	20.05.2024-24.05.2024	
4.	Загальне редагування та друк пояснювальної записки.	27.05.2024-31.05.2024	
5.	Проходження нормоконтролю, перепліт пояснювальної записки.	03.06.2024-05.06.2024	
6.	Розробка тексту доповіді. Оформлення графічного матеріалу для презентації.	06.06.2024-10.06.2024	

7. Дата видачі завдання «06» травня 2024 р.

Керівник кваліфікаційної роботи _____ Олена ТОЛСТІКОВА
(підпис керівника)

Завдання прийняв до виконання _____ Артем ВАНЗИТЛЕР
(підпис здобувача вищої освіти)

РЕФЕРАТ

Пояснювальна записка до кваліфікаційної роботи «Вебзастосунок інтернет-магазину взуття на основі технологій Vue JS, Node JS» представлена на 58 сторінках, містить 25 рисунків, 13 літературних джерел.

Об’єкт дослідження – процес розробки вебзастосунка інтернет-магазину взуття на основі технологій Vue JS, Node JS.

Предмет дослідження – вебзастосунок інтернет-магазину взуття на основі технологій Vue JS, Node JS

Мета роботи – розробити вебзастосунок інтернет-магазину взуття на основі технологій Vue JS, Node JS.

Методи дослідження – логічний, синтезу, аналізу, порівняльний, обробка літературних джерел та моделювання.

Наукова новизна даної кваліфікаційної роботи полягає у використанні новітніх технологій, які розширюють можливості вебзастосунка, роблять його масштабованим, надійним і дешевшим у розробці ніж вебзастосунки на застарілих технологіях.

Результат роботи: Вебзастосунок інтернет-магазину, який надалі може бути використаний як в освітніх цілях, так і для практичного застосування у створенні на його основі інших інтернет-магазинів.

Для розробки вебзастосунка знайдено та використано найефективніші програмні засоби, а також розроблено зручний користувацький інтерфейс.

Ключові слова: ВЕБЗАСТОСУНОК, ІНТЕРНЕТ-МАГАЗИН, JavaScript, Vue JS, Node JS, PostgreSQL.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СКОРОЧЕНЬ, ТЕРМІНІВ

CSS	−	Cascading Style Sheets – каскадні таблиці стилів
HTML	−	HyperText Markup Language – мова розмітки гіпертексту
SQL	−	Structured query language – мова структурованих запитів
HTTP	−	Hyper Text Transfer Protocol – протокол передачі гіпертексту
API	−	Application Programming Interface – прикладний програмний інтерфейс
HTTP	−	HyperText Transfer Protocol – протокол передачи гипертекста
DOM	–	Document Object Model – об'єктна модель документа

ВСТУП

Сучасний світ вимагає від бізнесу швидко адаптуватися до зміни обставин. З постійними повітряними тривогами, ризиками, що пов'язані з веденням офлайн бізнесу, стають все більш актуальними онлайн шляхи продажу товарів.

В наш час, коли більшість рутинних задач перейшли в онлайн, сфера торгівлі не стала винятком. Все більше людей обирають зручність та економність онлайн-покупок, де можна легко порівнювати ціни, читати відгуки та знаходити широкий асортимент товарів. Цей тренд веде до поступового відмирання традиційної моделі торгівлі з фізичними магазинами та сприяє бурхливому розвитку онлайн-шопінгу.

Для виконання індивідуального завдання було обрано взуттєве виробництво, тому що останнім часом попит на цю продукцію зростає, асортимент продукції постійно збільшується, а у підприємців є потреба розширювати ринок збуту, не витрачаючи великі кошти на відкриття фізичних магазинів у важкі часи.

З розвитком технологій активно зростає ринок електронної комерції. Згідно зі статистикою, обсяг світового ринку електронної комерції щорічно зростає на 10-15%. В Україні онлайн-шопінг стає все більш популярним, особливо після пандемії COVID-19. Люди цінують зручність та економність онлайн-покупок, можливість порівнювати ціни та вибирати товари з широкого асортименту.

Актуальність теми кваліфікаційної роботи «Вебзастосунок інтернет-магазину взуття на основі технологій Vue JS, Node JS» ґрунтується, з одного боку, на задоволенні потреб користувача в якісних, легких, а головне безпечних онлайн-покупках, а з іншого – на потребі підприємців відкривати нові канали для продажу продукції без ризиків втрати великих інвестицій. Це створює велику потребу в розробці недорогих, функціональних та зручних інтернет-магазинів. Дана кваліфікаційна робота відповідає актуальним тенденціям розвитку вебзастосунків та відкриває можливості для подальшого дослідження та

розвитку в області розробки вебзастосунків інтернет-магазинів.

Об’єктом дослідження – процес розробки вебзастосунка інтернет-магазину взуття на основі технологій Vue JS, Node JS.

Предмет дослідження – вебзастосунок інтернет-магазину взуття на основі технологій Vue JS, Node JS.

Мета кваліфікаційної роботи – розробити вебзастосунок інтернет-магазину взуття на основі технологій Vue JS, Node JS.

Відповідно до поставленої мети роботи визначено основні **завдання дослідження**:

– Проаналізувати предметну область вебзастосунків інтернет-магазинів;

– Дослідити основні етапи створення вебзастосунків;

– Виконати аналіз конкурентів;

– Проаналізувати і вибрати технології та інструменти для розробки вебзастосунка;

– Виконати проєктування вебзастосунка інтернет-магазину взуття;

– Виконати розробку вебзастосунка інтернет-магазину взуття.

Для досягнення поставленої мети й виконання завдань використано наступні методи: логічний, синтезу, аналізу, порівняльний, обробка літературних джерел та моделювання.

Наукова новизна даної кваліфікаційної роботи полягає у використанні новітніх технологій, які розширюють можливості вебзастосунка, роблять його масштабованим, надійним і дешевшим у розробці ніж вебзастосунки на застарілих технологіях.

Практичне значення отриманих результатів. Результати кваліфікаційної роботи надалі можуть бути використані як в освітніх цілях, так і для практичного застосування у створенні інших інтернет-магазинів.

РОЗДІЛ 1
АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1. Дослідження поняття вебзастосунок та інтернет-магазин

Вебзастосунок, також відомий як вебдодаток, - це програмне забезпечення, яке працює у веббраузері.

На відміну від звичайних програм, які потрібно завантажувати та встановлювати на комп'ютер, вебзастосунки працюють в Інтернеті. Це робить їх доступними з будь-якого пристрою, який має підключення до Інтернету, такі як комп'ютери, планшети, смартфони.

Ключові характеристики вебзастосунків:

– Доступність: До вебзастосунків можна отримати доступ з будь-якого пристрою, який має підключення до Інтернету та веббраузер. Це робить їх дуже зручними для використання, адже не потрібно турбуватися про сумісність програмного забезпечення або операційну систему.

– Простота оновлення: Вебзастосунки постійно оновлюються на сервері, тому користувачу не потрібно завантажувати та встановлювати оновлення вручну. Це гарантує, що користувач завжди використовує найновішу версію програмного забезпечення.

– Простота використання: Вебзастосунки прості у використанні та не потребують спеціальних знань чи навичок.

– Масштабованість: Вебзастосунки легко масштабуються, щоб обслуговувати велику кількість користувачів. Це робить їх ідеальними для використання у великих організаціях або для програм з великою базою користувачів.

– Зниження витрат: Вебзастосунки, як правило, є більш економічно вигідними, ніж традиційні програми, оскільки не потребують придбання ліцензій на програмне забезпечення та встановлення на кожному комп'ютер.

– Функціональність: Вебзастосунки можуть пропонувати широкий спектр функцій, таких як обробка текстових документів, електронні таблиці, презентації, управління проєктами, спілкування та багато іншого.

На сьогодні існує багато різних типів вебзастосунків, наприклад:

– Інтернет-магазини: Дозволяють користувачам купувати товари та послуги онлайн.

Соціальні мережі: Дозволяють користувачам спілкуватися з друзями та родиною.

– Вебпошта: Дозволяє користувачам надсилати та отримувати електронні листи.

– Онлайн-банкінг: Дозволяє користувачам керувати своїми банківськими рахунками онлайн.

– Вебсайти новин: Надають користувачам доступ до новин та інформації.

– Вебсайти освітніх закладів: Дозволяють студентам навчатися онлайн.

– Вебсайти урядових установ: Надають громадянам доступ до урядових послуг.

Вебзастосунок складається з трьох основних частин.

– Користувальницький інтерфейс – це частина вебзастосунка, яку бачить користувач та може з нею взаємодіяти. Вона складається зі сторінок, кожна з яких є HTML файлом. Стилів цих сторінок, які задаються за допомогою CSS. Та скриптів які створюються на мові JavaScript.

– Серверна частина – це частина вебзастосунка, яка відповідає за логіку його роботи, обробку даних, взаємодію з базою даних і працює на сервері. Логіка його роботи також написана на мові програмування JavaScript.

– База даних – це програма для організованого сховища структурованої інформації. Вона використовується для зберігання, організації та доступу до даних.

Вебзастосунки стали невід'ємною частиною нашого життя. Вони роблять наше життя простішим, зручнішим та цікавішим.

Інтернет-магазин (також відомий як вебмагазин або онлайн-магазин) - це вебзастосунок, який дозволяє користувачам купувати товари та послуги через Інтернет.

Інтернет-магазини стали популярним способом здійснення покупок завдяки своїй зручності, широкому асортименту товарів та конкурентним цінам.

Ключові характеристики інтернет-магазинів:

– Широкий асортимент товарів: Інтернет-магазини, як правило, пропонують набагато ширший асортимент товарів, ніж традиційні магазини. Це пов'язано з тим, що вони не обмежені фізичним простором і можуть пропонувати продукти з усього світу.

– Зручність: Інтернет-магазини можна відвідати з будь-якого місця з підключенням до Інтернету, 24 години на добу, 7 днів на тиждень. Це робить їх дуже зручними для покупок, особливо для людей з щільним графіком.

– Конкурентні ціни: Інтернет-магазини, як правило, пропонують більш конкурентні ціни, ніж традиційні магазини. Це пов'язано з тим, що їм не потрібно платити за оренду, зарплату персоналу та інші накладні витрати.

– Простота оформлення замовлення: Інтернет-магазини, як правило, пропонують простий та зручний процес оформлення замовлення. Ви можете вибрати товари, додати їх до кошика та оформити замовлення за лічені хвилини.

Найпопулярніші типи інтернет-магазинів:

– Універсальні магазини: Ці магазини пропонують широкий асортимент товарів, таких як одяг, електроніка, меблі, іграшки тощо.

– Спеціалізовані магазини: Ці магазини пропонують товари певної категорії, такі як спортивні товари, автозапчастини, косметика тощо.

Інтернет-магазини, що належать виробникам: Ці магазини продають продукти безпосередньо виробникам, що може призвести до більш низьких цін.

– Маркетплейси: Ці Вебсайти пропонують товари від різних продавців, що дає вам більше можливостей для порівняння цін і вибору найкращої пропозиції.

– Дисконтні магазини: Ці магазини пропонують товари за зниженими цінами, як правило, це залишки, уцінені товари або товари з минулих сезонів.

Інтернет-магазини пропонують багато переваг покупцям, тому вони стали популярним способом здійснення покупок.

Щоб отримати доступ для вебзастосунка інтернет-магазину потрібен веббраузер. Популярні веббраузери: Google Chrome, Mozilla Firefox, Safari, Microsoft Edge, Microsoft Internet Explorer. Завдяки використанню веб технологій до вебзастосунка можна отримати доступ з будь-якого пристрою, на якому є браузер.

1.2. Основні етапи створення вебзастосунка інтернет-магазину

Створення вебзастосунка інтернет-магазину – це складний процес, який потребує ретельного планування, дизайну та розробки.

Основними етапами створення вебзастосунка інтернет-магазину є:

– Аналіз:

1. визначення цілей та вимог інтернет-магазину;
2. проведення аналізу ринку;

– Вибір технологій та інструментів розробки:

– Проєктування та розробка:

1. створення прототипу інтерфейсу користувача;
2. розробка інтерфейсу користувача;
3. розробка серверної частини;
4. розробка логічної моделі бази даних;
5. розробка структури бази даних;

1.3. Визначення цілей та вимог до вебзастосунка інтернет-магазину

Визначення цілей та вимог до вебзастосунка інтернет-магазину є першим і найважливішим етапом життєвого циклу програмного продукту. На даному етапі чітко формулюється те, що продукт повинен робити, для кого він призначений і які проблеми він повинен вирішувати.

Ціль продукту – це загальне твердження, яке описує бажаний результат розробки. Вона повинна бути чіткою, вимірювальною, досяжною, релевантною та обмеженою за часом (SMART).

Термів виконання до 16.06.2024

Функціональні вимоги:

1. Інтерфейс користувача:

– Привабливий та зручний інтерфейс, що відповідає сучасним трендам.

– Швидке та плавне завантаження сторінок.

– Зручна система фільтрації та пошуку взуття за назвою

– Можливість додавати взуття до кошика та оформляти замовлення.

– Інтуїтивно зрозуміла навігація по сайту.

– Простий та зрозумілий процес оформлення замовлення.

– Швидкий та зручний пошук потрібного взуття.

– Можливість додавати товари в обрані

– Можливість переглядати список обраних товарів

– Можливість переглядати список замовлень

2. Серверна частина:

– Управління каталогом продукції:

– Додавання, редагування, отримання та видалення товарів.

– Додавання, редагування, отримання та видалення обраних товарів.

– Додавання, редагування, отримання та видалення замовлень.

– Фільтрація та сортування товарів.

Нефункціональні вимоги:

1. Продуктивність: вебзастосунок повинен бути швидким та плавним у роботі, сторінки повинні завантажуватися швидко, без затримок.

2. Безпека: вебзастосунок повинен бути захищений від хакерських атак та крадіжки даних.

3. Надійність: вебзастосунок повинен бути надійним та працювати без збоїв.

4. Масштабованість: вебзастосунок повинен бути масштабованим, щоб можна було легко збільшити його потужність у міру зростання кількості користувачів та товарів.

5. Зручність використання: вебзастосунок повинен бути простим та зручним у користуванні, інтерфейс користувача повинен бути інтуїтивно зрозумілим.

1.4. Огляд та аналіз конкурентів

Першим кроком в аналізі конкурентів є визначення конкурентів. Це включає як прямих, так і непрямих конкурентів. Прямі конкуренти - це інші онлайн-магазини взуття, які пропонують схожі продукти та послуги вашій цільовій аудиторії. Непрямі конкуренти - це магазини, які можуть продавати взуття як частину свого асортименту, наприклад, магазини одягу, аксесуарів та спортивні магазини.

На зараз існує велика кількість онлайн магазинів взуття. Одна частина має як онлайн, так і офлайн магазин. Інші орієнтуються тільки на онлайн продажі.

Було проаналізовано найпопулярніші інтернет-магазини взуття (рис. 1.1 – рис. 1.3)

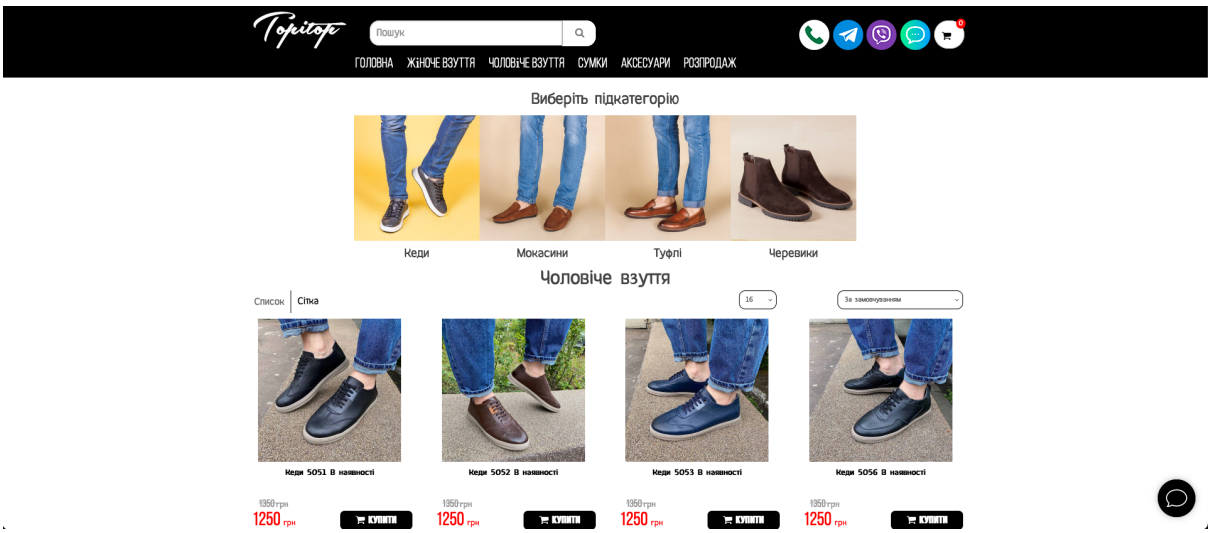

Рисунок 1.1. Інтернет-магазин topitop

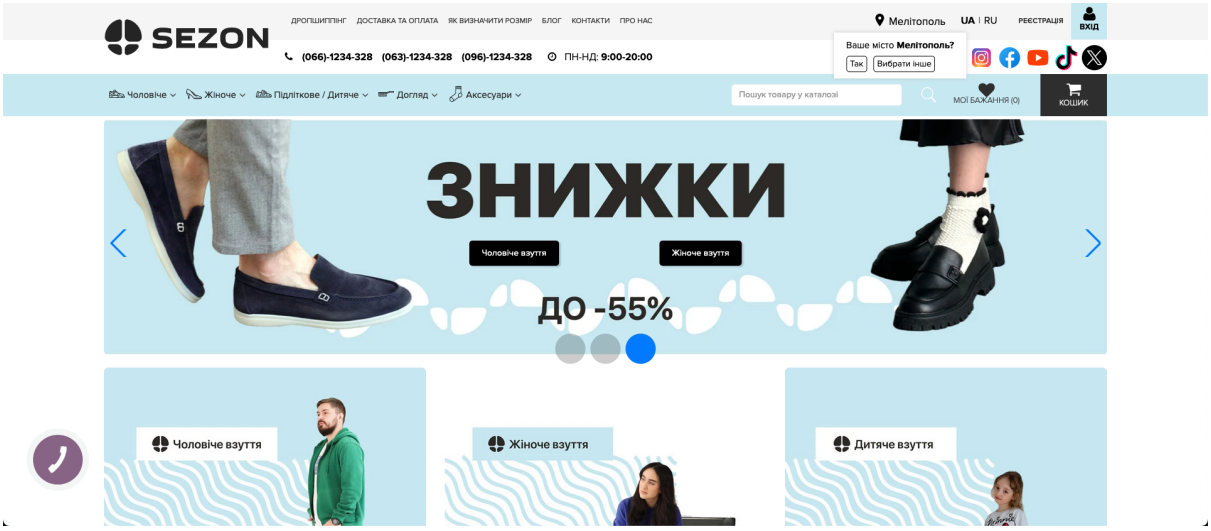

Рисунок 1.2. Інтернет-магазин sezon

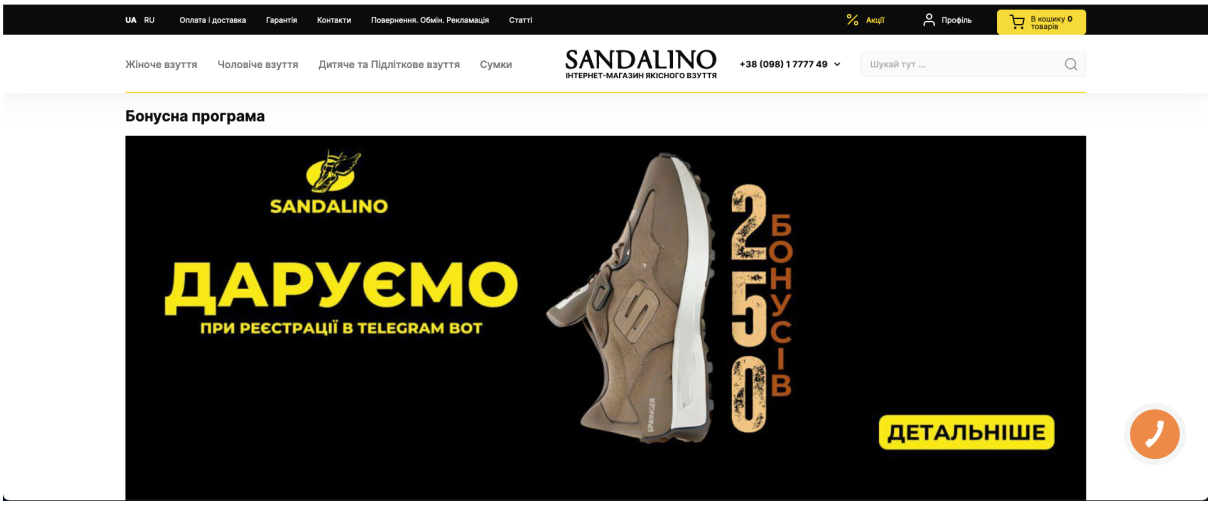

Рисунок 1.3. Інтернет-магазин sandalino

Сильні сторони:

– Широкий асортимент товарів: Ці онлайн-магазини пропонують широкий спектр взуття, що задовольнить потреби будь-якого клієнта.

– Велика база клієнтів: Завдяки своїй популярності, ці магазини мають значну кількість постійних та нових клієнтів.

Слабкі сторони:

– Складний дизайн: Інтерфейс сайтів може здатися заплутаним та незручним для користувачів, що може призвести до труднощів з навігацією та пошуком необхідних товарів.

– Надмірний контент: Занадто багато інформації та візуальних елементів може перевантажити користувачів та відволікти їх від головного - вибору та покупки взуття.

Конкурентною перевагою даного вебзастосунка інтернет-магазину взуття стане його легкий та зручний дизайн.

Це буде досягнуто завдяки:

– Інтуїтивно зрозумілому інтерфейсу: Користувачі зможуть легко знаходити потрібні їм товари та інформацію, не витрачаючи на це багато часу та зусиль.

– Швидкому завантаженню сторінок: Завдяки оптимізованому коду та мінімалістичному дизайну сторінки сайту будуть завантажуватися швидко, що забезпечить приємний користувацький досвід.

– Мінімалістичному дизайну: Відсутність зайвих елементів та візуального шуму дозволить зосередити увагу користувачів на продукції.

– Чіткій навігації: Проста та зрозуміла структура сайту допоможе користувачам швидко знаходити потрібні їм розділи та товари.

Завдяки цим факторам наш вебзастосунок стане зручнішим та приємнішим для користувачів у порівнянні з сайтами конкурентів.

Це, в свою чергу, призведе до:

– Збільшення конверсії: Більше користувачів, які відвідали наш сайт,

ймовірно, зроблять покупку.

– Підвищення лояльності клієнтів: Задоволені користувачі частіше повертаються на сайт та рекомендують його друзям та знайомим.

– Покращення іміджу бренду: Легкий та зручний дизайн сайту сформує позитивний імідж нашого бренду.

Таким чином, інвестування у розробку легкого та зручного дизайну вебзастосунка інтернет-магазину взуття дасть нам значну конкурентну перевагу та допоможе досягти успіху на ринку.

ВИСНОВКИ ДО РОЗДІЛУ 1

Під час розробки другого розділу було проведено аналіз предметної області, що включало:

– Дослідження поняття вебзастосунок та інтернет-магазин: визначення ключових характеристик вебзастосунків, типів вебзастосунків, трьох частин, з яких складається вебзастосунок, ключових характеристик інтернет-магазинів, типи інтернет-магазинів. Це дозволило краще зрозуміти специфіку теми.

– Визначено основні етапи створення вебзастосунків:

3. визначення цілей та вимог для інтернет-магазину;
4. проведено аналіз ринку;
5. вибір технологій та інструментів розробки;
6. проєктування та розробка.

– Визначення цілей та вимог до вебзастосунка інтернет магазину.

– Проаналізовано конкурентів:

1. вивчено наявні аналоги;
2. проведено аналізу їх сильних та слабких сторін;
3. визначено ключові відмінностей нашого проєкту.

Це дозволило зрозуміти який вебзастосунок стане кращим і буде заслуговувати увагу клієнтів.

РОЗДІЛ 2
ТЕХНОЛОГІЇ ТА ІНСТРУМЕНТИ РОЗРОБКИ

2.1. Мова програмування

JavaScript - це динамічна, високорівнева мова програмування, яка може використовуватися як для додавання інтерактивності до інтерфейсу вебзастосунка, так і використовуватися на сервері.

На відміну від мов, що компілюються, таких як C++ або Java, JavaScript інтерпретується. Це означає, що код JavaScript не перекладається в машинну мову перед виконанням, а замість цього виконується рядком за рядком браузером або іншим середовищем виконання JavaScript.

JavaScript використовує динамічну типізацію, що означає, що тип змінної не визначається явно при її оголошенні. Тип змінної визначається значенням, яке їй присвоюється. Це може призвести до більшої гнучкості, але також може призвести до помилок під час виконання, якщо не бути обережним.

JavaScript - це мова одно поточна, що означає, що вона може виконувати лише один код одночасно. Це може призвести до проблем з продуктивністю, якщо ваш код виконує багато обчислень. Однак існує ряд методів, які можна використовувати для покращення продуктивності однопотокового коду JavaScript, таких як використання Web Workers і асинхронного програмування.

JavaScript - це мова з множинною парадигмою, що означає, що вона підтримує різні стилі програмування, такі як імперативне, функціональне та об'єктноорієнтоване програмування. Це робить JavaScript гнучкою мовою, яку можна використовувати для вирішення широкого кола завдань.

JavaScript спочатку використовувався як мова для створення динамічного контенту на вебсторінках. Однак тепер він також використовується для розробки серверних застосунків за допомогою Node.js.

Переваги JavaScript:

− Універсальність: JavaScript підтримується всіма сучасними веббраузерами, що робить його ідеальним для створення вебконтенту, який можна переглядати на будь-якому пристрої.

− Простота вивчення: JavaScript порівняно простий для вивчення, особливо для людей, які вже знайомі з іншими мовами програмування.

− Гнучкість: JavaScript можна використовувати для створення широкого спектра вебконтенту, від простих анімацій до складних вебдодатків.

− Популярність: JavaScript - одна з найпопулярніших мов програмування у світі, що робить його чудовим вибором для початківців та досвідчених розробників.

Синтаксис JavaScript схожий на інші мови програмування, такі як C і Java. Він використовує змінні, оператори, умовні оператори, цикли та функції.

Інші мови програмування, які можна було б використати:

– PHP – це мова програмування, яка використовується для серверної веброзробки. Вона може бути гнучкою, але JavaScript пропонує більшу універсальність і ширший спектр можливостей.

– Java – це мова програмування, яка використовується для розробки складних вебзастосунків. Вона може бути потужною, але JavaScript пропонує більшу легкість у вивченні та швидкість розробки.

– C# – це мова програмування, яка використовується для розробки вебзастосунків на платформі .NET. Вона може бути надійною, але JavaScript пропонує більшу універсальність і ширший спектр можливостей.

Отже, завдяки своїй простоті у вивченні, великій спільноті та безлічі бібліотек та фреймворків JavaScript – це найкращий вибір.

2.2. Технологій для користувацького інтерфейсу

HTML (HyperText Markup Language) - це стандартизована мова розмітки документів, яка використовується для створення вебсторінок. Її розуміють і обробляють усі веббраузери, щоб відображати вебконтент на екрані користувача.

Основні функції HTML:

– Структура документа: HTML використовується для визначення структури вебсторінки, включаючи заголовки, абзаци, списки, таблиці, зображення та інші елементи.

– Гіперпосилання: HTML дозволяє створювати гіперпосилання, які ведуть до інших вебсторінок або ресурсів.

– Вбудовування контенту: HTML дозволяє вбудовувати в вебсторінки різні типи контенту, такі як зображення, відео, аудіо та інтерактивні елементи.

– Формування тексту: HTML пропонує теги для форматування тексту, такі як жирний, курсив, підкреслення, заголовки та вирівнювання.

Переваги HTML:

– Простота: HTML порівняно простий для вивчення та використання, навіть для початківців.

– Універсальність: HTML підтримується всіма веббраузерами, що робить його ідеальним для створення вебсторінок, які можна переглядати на будь-якому пристрої.

– Гнучкість: HTML можна використовувати для створення широкого спектра вебсторінок, від простих статичних сайтів до складних динамічних вебдодатків.

– Стандартизованість: HTML - це стандартизована мова, що гарантує її надійність та сумісність.

CSS (Cascading Style Sheets) - це мова опису зовнішнього вигляду вебсторінок. Вона використовується для контролю візуального оформлення HTML-сторінок.

Основні функції CSS:

– Відокремлення вмісту від оформлення: CSS дозволяє відокремлювати структуру вебсторінки (HTML) від її візуального оформлення (CSS). Це робить код чіткішим, організованим та легким для обслуговування.

– Контроль візуального оформлення: CSS дає широкий спектр можливостей для контролю візуального оформлення вебсторінок. Можна змінювати шрифти, кольори, розміри, межі, відступи, розташування та багато інших аспектів елементів HTML.

– Створення макетів: CSS використовується для створення макетів вебсторінок, тобто для розміщення елементів на сторінці та визначення їх взаємозв'язку.

– Адаптивність: CSS дозволяє створювати адаптивні вебсайти, які автоматично підлаштовуються під різні розміри екранів і пристрої.

– Доступність: CSS можна використовувати для покращення доступності вебсайтів для людей з обмеженими можливостями.

Переваги CSS:

– Покращена чіткість коду: CSS робить код HTML чіткішим і організованим шляхом відокремлення вмісту від оформлення.

– Зменшення дублювання коду: CSS дозволяє визначити стилі один раз і застосовувати їх до декількох елементів HTML, що економить час і код.

– Легкість обслуговування: CSS робить код HTML легшим для обслуговування, оскільки зміни візуального оформлення можна вносити в одному місці, а не редагувати код HTML для кожного елемента.

– Покращена продуктивність: CSS може покращити продуктивність вебсайтів, оскільки браузеру не потрібно повторно обробляти одні й ті ж стилі для різних елементів.

Tailwind CSS - це CSS-фреймворк з відкритим кодом, який використовує підхід "utility-first".

На відміну від традиційних CSS-фреймворків, які пропонують готові компоненти та стилі, Tailwind надає набір базових класів, які можна використовувати для стилізації будь-якого HTML-елемента.

Це надає гнучкість та контроль над зовнішнім виглядом своїх вебзастосунків, водночас економлячи час за рахунок використання готових класів замість написання власного CSS.

Ось деякі з ключових особливостей Tailwind CSS:

– Utility-first: Tailwind надає набір базових класів, які можна використовувати для стилізації будь-якого HTML-елемента. Ці класи охоплюють широкий спектр стилів, таких як розміри, кольори, відступи, шрифти, тіні тощо.

– Гнучкість: Розробники можуть використовувати комбінації класів Tailwind для створення будь-якого бажаного стилю. Це дає їм повний контроль над зовнішнім виглядом своїх вебзастосунків.

– Простота використання: Tailwind легко вивчити та використовувати, навіть для розробників, які не мають досвіду роботи з CSS. Класи мають описові назви, які чітко вказують на їх функціональність.

– Модульність: Tailwind можна використовувати модульно, додаючи лише ті класи, які потрібні для вашого проєкту. Це робить його легким та ефективним фреймворком.

– Спільнота: Tailwind має активну та зростаючу спільноту розробників, які створюють плагіни, інструменти та навчальні матеріали.

Переваг використання Tailwind CSS:

– Швидкість розробки: Tailwind може допомогти вам писати CSS швидше та ефективніше, завдяки використанню готових класів замість написання власного CSS.

– Гнучкість: Tailwind дає вам повний контроль над зовнішнім виглядом ваших вебзастосунків. Ви можете створювати будь-який бажаний стиль, використовуючи комбінації класів.

– Масштабованість: Tailwind добре підходить для проєктів будь-якого розміру, від невеликих вебзайтів до складних вебзастосунків.

– Співпраця: Tailwind полегшує співпрацю над проєктами, адже розробники використовують спільну мову стилізації.

Це робить Tailwind CSS більш лаконічним та читабельним, а також значно спрощує процес розробки користувацького інтерфейсу.

Vue.js - це фреймворк JavaScript з відкритим вихідним кодом для створення користувальницьких інтерфейсів. Він відрізняється від деяких інших фреймворків тим, що є прогресивною. Це означає, що Vue можна використовувати для створення як простих компонентів, так і складних додатків. Він пропонує декларативний та компонентний підхід до розробки інтерфейсів, що робить його легким у вивченні, гнучким та масштабованим для проєктів різної складності.

Структура проєкту Vue.js може варіюватися залежно від його розміру та складності. Однак існує загальна структура, яку рекомендується використовувати для більшості проєктів (рис. 2.1).

Рисунок 2.1. Приклад структури проєкту на Vue.js

Коренева директорія:

– index.html: Цей файл є основним HTML-файлом вебзастосунка. Він містить посилання на JavaScript-файли та CSS-стилі проєкту.

– main.js: Цей файл JavaScript є точкою входу вебзастосунка. Він імпортує та ініціалізує компоненти Vue та налаштовує маршрутизацію.

– package.json: Цей файл містить метадані проєкту, також залежності та сценарії npm.

Директорія src:

– assets: Ця директорія містить статичні активи проєкту, такі як зображення, шрифти та CSS-файли.

– components: Ця директорія містить компоненти Vue проєкту. Кожен компонент повинен мати свій файл .vue.

– App.vue: Цей файл є головним компонентом вебзастосунка. Він описує загальну структуру інтерфейсу користувача.

– router.js: Цей файл JavaScript налаштовує маршрутизацію вебзастосунка. Він визначає, як різні URL-адреси відображаються різними компонентами.

– store.js: Цей файл JavaScript (необов'язково) налаштовує Vuex, систему управління станом для Vue.js.

Основні принципи Vue.js:

– Прогресивність: Vue.js можна використовувати як бібліотеку для додавання інтерактивності до наявних вебсторінок або як повноцінний фреймворк для масштабних односторінкових додатків. Ця гнучкість дозволяє розробникам інтегрувати Vue.js у проєкти будь-якого розміру.

– Компонентна архітектура: Vue.js розділяє інтерфейс на повторно використовувані компоненти. Кожен компонент має свій шаблон (HTML-подібний синтаксис для опису структури), логіку JavaScript та стилі CSS. Цей підхід сприяє модульності, організованості та повторному використанню коду.

– Шаблони з прив'язкою даних: Vue.js використовує декларативний підхід до керування DOM. Розробники визначають структуру інтерфейсу за допомогою шаблонів, а потім прив'язують дані JavaScript до цих шаблонів. Vue.js

автоматично оновлює DOM, коли дані змінюються, забезпечуючи реактивність інтерфейсу.

– Віртуальний DOM: Vue.js використовує віртуальний DOM для оптимізації оновлень інтерфейсу. Він порівнює віртуальне DOM-дерево з реальним DOM-деревом у браузері та застосовує лише мінімальні необхідні зміни до реального DOM, що покращує продуктивність.

Переваги використання Vue.js:

– Легкість у навчанні: Vue.js має порівняно простий API та невеликий розмір, що робить його легким для освоєння навіть для розробників без досвіду роботи з фреймворками.

Гнучкість: Vue.js можна використовувати як бібліотеку або фреймворк, що дозволяє адаптувати його до потреб проєкту.

– Реактивність: Двостороння прив'язка даних забезпечує автоматичне оновлення інтерфейсу при зміні даних, що спрощує розробку динамічних інтерфейсів.

– Компонентна модель: Компонентна архітектура сприяє модульності, повторному використанню коду та легшому управлінню складністю.

– Підвищена продуктивність: Реактивність та віртуальний DOM забезпечують швидкий та ефективний процес оновлення інтерфейс

– Велика спільнота та екосистема: Vue.js має активну спільноту та багату екосистему бібліотек та інструментів, що полегшує розробку.

Інші фреймворки та бібліотеки для створення користувацьких інтерфейсів:

– React – це ще один популярний фреймворк JavaScript, який пропонує схожі можливості, що й Vue.js. Однак React може бути складнішим для вивчення та використання.

– Angular – складніший фреймворк JavaScript, який пропонує більше функцій, але також може бути складніший для вивчення та використання.

– jQuery – це бібліотека JavaScript, яка використовується для маніпулювання DOM. Вона може бути корисною для створення простих вебзастосунків, але не так гнучка та масштабована, як Vue.js.

Vue.js пропонує гнучкий та ефективний підхід до розробки користувальницьких інтерфейсів. Його легкість у навчанні, реактивність, компонентна модель та велика спільнота роблять його популярним вибором для розробників різного рівня.

2.3. Технологій для сервера

Зазвичай JavaScript використовується для створення інтерактивних елементів на вебсторінках. Node.js розширює можливості JavaScript, дозволяючи йому працювати на серверах, комп'ютерах, які обробляють запити від веббраузерів та інших програм.

Node.js - це кросплатформене середовище виконання JavaScript з відкритим вихідним кодом, побудоване на двигуні V8 Chrome. Воно дозволяє запускати JavaScript поза браузером, що робить його ідеальним для створення серверних застосунків.

Node.js працює в одному процесі, не створюючи новий потік для кожного запиту. Node.js надає набір примітивів асинхронного вводу/виводу у своїй стандартній бібліотеці, які запобігають блокуванню коду JavaScript, і взагалі, бібліотеки в Node.js написані з використанням неблокуючих парадигм, що робить поведінку блокування скоріше винятком, ніж нормою.

Коли Node.js виконує операцію вводу/виводу, наприклад, читання з мережі, доступ до бази даних або файлової системи, замість того, щоб блокувати потік і витрачати цикли процесора на очікування, Node.js відновлює операцію, коли відповідь повертається. Це дозволяє Node.js обробляти тисячі паралельних з'єднань з одним сервером, не створюючи тягаря керування паралельністю потоків, що може бути значним джерелом помилок.

У Node.js можна без проблем використовувати нові стандарти ECMAScript, оскільки не потрібно чекати, поки всі користувачі оновлять свої браузери - ви самі вирішуєте, яку версію ECMAScript використовувати.

Node.js - це потужний і універсальний інструмент, який можна використовувати для створення широкого спектра застосунків.

Основні характеристики Node.js:

– Подіє орієнтований: Node.js використовує модель подій для обробки асинхронних запитів. Це робить його дуже ефективним для обробки великої кількості одночасних з'єднань.

– Легкий: Node.js має невеликий розмір і потребує мінімальних ресурсів.

– Масштабований: Node.js можна легко масштабувати для обробки великих навантажень.

– Гнучкий: Node.js можна використовувати для створення широкого спектра застосунків, від вебсайтів і вебдодатків до інструментів командного рядка та IoT-пристроїв.

Його легкість, масштабованість і гнучкість роблять його популярним вибором для розробників, які хочуть створювати високопродуктивні та ефективні програми.

2.4. Технологій бази даних

SQL (Structured Query Language) - це мова програмування для зберігання та обробки інформації в реляційній базі даних. Реляційна база даних зберігає інформацію в табличній формі, де рядки й стовпці представляють різні атрибути даних і різні зв'язки між значеннями даних. Розробник може використовувати оператори SQL для зберігання, оновлення, видалення, пошуку та отримання інформації з бази даних. Розробник також може використовувати SQL для підтримки та оптимізації продуктивності бази даних.

Вона дозволяє розробникам створювати, читати, оновлювати та видаляти дані в базі даних, а також виконувати складні операції з ними.

Основні можливості SQL:

– Створення таблиць: SQL дозволяє створювати таблиці в базі даних, визначаючи їх структуру та тип даних для кожного стовпця.

– Вставлення даних: SQL дозволяє вставляти нові записи в таблиці бази даних.

– Вибірка даних: SQL дозволяє вибирати дані з таблиць бази даних за допомогою різних умов та фільтрів.

– Оновлення даних: SQL дозволяє оновлювати наявні записи в таблицях бази даних.

– Видалення даних: SQL дозволяє видаляти записи з таблиць бази даних.

– Сортування та групування даних: SQL дозволяє сортувати та групувати дані в таблицях бази даних за різними критеріями.

– Об'єднання таблиць: SQL дозволяє об'єднувати дані з декількох таблиць бази даних.

Створення підзапитів: SQL дозволяє використовувати підзапити для вкладення складних операцій з даними.

– Забезпечення прав доступу: SQL дозволяє контролювати доступ користувачів до даних та операцій над ними.

Переваги використання SQL:

– Стандартизована мова: SQL є стандартизованою мовою, що робить її доступною для розробників на різних платформах.

– Потужність: SQL є потужною мовою, яка дозволяє виконувати складні операції з даними.

– Гнучкість: SQL можна використовувати для роботи з різними типами реляційних баз даних.

– Ефективність: SQL оптимізована для роботи з великими обсягами даних.

SQL є потужною та гнучкою мовою, яка використовується для управління даними в реляційних базах даних. Вона має просту структуру, легка у вивченні та використовується в широкому спектрі застосунків.

PostgreSQL - це вільна та відкрита реляційна система керування базами даних, яка пропонує широкий спектр функцій для зберігання, керування та аналізу даних. Вона відома своєю надійністю, масштабованістю, гнучкістю та відповідністю стандартам SQL.

Ось ключові характеристики PostgreSQL:

– Відповідність стандартам: PostgreSQL суворо дотримується стандарту SQL:2008, що робить її сумісною з іншими системами керування базами даних та полегшує розробникам перехід на неї.

Надійність: PostgreSQL славиться своєю надійністю та стійкістю до збоїв. Вона використовується у багатьох критично важливих застосунках, де час простою є неприпустимим.

– Масштабованість: PostgreSQL може масштабуватися від невеликих робочих станцій до великих серверних систем, обробляючи терабайти даних та обслуговуючи тисячі одночасних користувачів.

Гнучкість: PostgreSQL підтримує широкий спектр типів даних, функцій та можливостей, що робить її гнучким інструментом для вирішення різних завдань керування даними.

– Безпека: PostgreSQL пропонує декілька рівнів безпеки, включаючи автентифікацію користувачів, шифрування даних та контроль доступу на основі ролей.

– Розширюваність: PostgreSQL має активну спільноту розробників, яка створює безліч розширень та інструментів для розширення її можливостей.

– Безкоштовність: PostgreSQL - це вільна та відкрита програмна платформа, що робить її доступною для користувачів та розробників з будь-яким бюджетом.

PostgreSQL використовується в широкому спектрі застосунків, включаючи:

– Вебзастосунки: PostgreSQL є популярним вибором для вебзастосунків, які потребують надійного та масштабованого зберігання даних.

– Мобільні додатки: PostgreSQL можна використовувати для зберігання та керування даними в мобільних додатках.

– Аналітика даних: PostgreSQL використовується для зберігання та аналізу великих обсягів даних, завдяки своїм потужним можливостям аналітики.

– Наукові дослідження: PostgreSQL використовується в наукових дослідженнях для зберігання та аналізу наукових даних.

– Системи вбудованих пристроїв: PostgreSQL можна використовувати в системах вбудованих пристроїв завдяки її невеликим розмірам та низьким вимогам до ресурсів.

Переваги використання PostgreSQL:

– Надійність та стійкість до збоїв: PostgreSQL - це надійна та стійка до збоїв система керування базами даних, що робить її ідеальним вибором для критично важливих застосунків.

Масштабованість: PostgreSQL може масштабуватися від невеликих робочих станцій до великих серверних систем, що робить її гнучким рішенням для будь-яких потреб.

– Гнучкість: PostgreSQL пропонує широкий спектр типів даних, функцій та можливостей, що робить її гнучким інструментом для вирішення різних завдань керування даними.

– Відповідність стандартам: PostgreSQL суворо дотримується стандарту SQL:2008, що робить її сумісною з іншими система керування базами даних та полегшує розробникам перехід на неї.

– Безпека: PostgreSQL пропонує декілька рівнів безпеки, що робить її безпечним вибором для зберігання конфіденційних даних.

– Безкоштовність: PostgreSQL - це вільна та відкрита програмна платформа, що робить її доступною для користувачів та розробників з будь-яким бюджетом.

Порівняння з іншими реляційними базами даних:

– MySQL – це популярна реляційна база даних, яка також є безкоштовною та відкритою. Однак MySQL не така масштабована та надійна, як PostgreSQL, і вона не пропонує такого ж набору функцій.

– Microsoft SQL Server – це платна реляційна база даних, яка пропонує широкий спектр функцій та високу продуктивність. Однак SQL Server не є безкоштовним, і він складний у налаштуванні та використанні.

– Oracle Database – це платна реляційна база даних, яка пропонує найширший спектр функцій та найвищу продуктивність. Однак Oracle Database дуже дорога, і вона може бути складною у налаштуванні та використанні.

PostgreSQL – це потужна, надійна, масштабована та безкоштовна реляційна база даних, яка є ідеальним вибором для створення вебзастосунків.

Завдяки своїй сумісності зі стандартом SQL, широкому набору функцій, активній спільноті та високій продуктивності PostgreSQL є ідеальним вибором для розробки вебзастосунків.

2.5. Інструменти розробки

Visual Studio Code (VS Code) є безкоштовним і відкритим редактором коду, розробленим Microsoft. Він швидко став одним з найпопулярніших редакторів завдяки своєму поєднанню потужних функцій, зручного інтерфейсу та крос-платформної підтримки (Windows, macOS, Linux).

Основні характеристики VS Code:

– Підсвічування синтаксису та автодоповнення: VS Code підтримує підсвічування синтаксису для багатьох мов програмування, що допомагає вам писати код швидше та уникати помилок. Він також пропонує автодоповнення коду, яке пропонує можливі завершення коду під час його написання.

– Інтеграція Git: VS Code має вбудовану підтримку Git, що дозволяє розробнику керувати своїм кодом без необхідності перемикатися на інший інструмент. Ви можете переглядати історію змін, робити коміти, а також push та pull коду безпосередньо з редактора.

– Розширення: VS Code має великий і активний маркетплейс розширень, які додають нові функції та можливості. Розробник може знайти розширення для покращення підтримки мов програмування, додавання інструментів для відладки, інтеграції з іншими сервісами та багато іншого.

– Інтелектуальне завершення коду (Intellisense): VS Code пропонує Intellisense, який є більш потужною функцією, ніж автодоповнення. Він може пропонувати завершення коду на основі контексту, типу змінної та інших факторів.

– Відладка: VS Code дозволяє вам відлагоджувати код безпосередньо в редакторі. Розробник може встановити точки зупинки, переглянути стек викликів і переглянути значення змінних під час виконання коду.

Інтерфейс користувача: VS Code має чистий інтерфейс користувача, який

можна змінити під себе. Розробник може змінювати тему редактора, налаштовувати макет вікна та приховувати або відображати різні панелі.

– Крос-платформна підтримка: VS Code працює на Windows, macOS і Linux, що робить його чудовим вибором для розробників, які використовують різні операційні системи.

– Легкий: VS Code є легким редактором, який швидко запускається і не вимагає великих ресурсів.

Переваги використання VS Code:

– Безкоштовний та відкритий код: VS Code є безкоштовним і відкритим кодом, що робить його доступним для всіх.

– Потужні функції: VS Code пропонує широкий спектр функцій, які роблять його ефективним інструментом для розробки.

– Зручний інтерфейс: VS Code має чистий інтерфейс користувача, що робить його приємним у використанні.

– Крос-платформна підтримка: VS Code працює на Windows, macOS і Linux.

– Великий маркетплейс розширень: Розробник може розширити можливості VS Code за допомогою великої кількості розширень.

– Активна спільнота: VS Code має активну спільноту розробників, які постійно додають.

2.6. Переваги обраних технологій

Ці технології є оптимальним вибором для розробки вебзастосунка, який буде:

1. Зручним у розробці:

– Vue.js пропонує декларативний та компонентний підхід, який робить код більш читабельним та легким у розумінні.

Node.js використовує JavaScript, який використовується для розробки користувальницьких інтерфейсів, що спрощує процес розробки серверної частини.

– PostgreSQL має зручний інтерфейс командного рядка та потужний API, що робить роботу з базою даних простою та ефективною.

2. Передовим у технологіях:

– Vue.js постійно оновлюється та підтримує найсучасніші вебстандарти.

– Node.js ґрунтується на JavaScript-рушії V8 з Chrome, що робить його надзвичайно швидким та ефективним.

– PostgreSQL – це надійна, масштабована та гнучка реляційна база даних, яка відповідає стандарту SQL:2008.

3. Легким для розширення:

– Vue.js має модульну архітектуру, що дозволяє легко додавати нові функції та можливості.

– Node.js підтримує широкий спектр бібліотек та інструментів, які роблять його гнучким та розширюваним.

PostgreSQL має потужний SQL-інтерфейс, який дозволяє легко розширювати та модифікувати схему бази даних.

ВИСНОВКИ ДО РОЗДІЛУ 2

Під час розробки другого розділу було проведено глибоке дослідження основних технологій, які використовуються для створення вебзастосунків. Це дослідження охопило широкий спектр технологій, включаючи фреймворки користувацьких інтерфейсів, серверної частини та бази даних. Було розглянуто їх особливості та переваги.

Після глибокого дослідження основних технологій, які використовуються для створення вебзастосунків, було обрано мову програмування JavaScript, технології Vue.js для створення реактивних користувальницьких інтерфейсів, Node.js для створення серверної частини вебзастосунка та PostgreSQL для роботи з реляційною базою даних. Vue.js та Node.js чудово доповнюють один одного, пропонуючи комплексне рішення для розробки вебзастосунків.

РОЗДІЛ 3
РОЗРОБКА ВЕБЗАСТОСУНКА

3.1. Проєктування макета інтерфейсу користувача

Проєктування макета інтерфейсу користувача – це процес створення візуального представлення інтерфейсу, яке описує розміщення елементів, їх розміри, шрифти, кольори та інші візуальні характеристики.

Макет інтерфейсу користувача – це статичне зображення або прототип, який дає уявлення про те, як буде виглядати інтерфейс кінцевого продукту.

Мета проєктування макета інтерфейсу користувача:

6. Візуалізувати інтерфейс: Макет інтерфейсу користувача допомагає дизайнерам та розробникам чітко уявити, як буде виглядати інтерфейс продукту.

7. Виявити проблеми на ранніх стадіях: Макет інтерфейсу користувача дозволяє виявити потенційні проблеми з дизайном на ранніх стадіях розробки, коли їх ще легко виправити.

8. Створити узгоджений інтерфейс: Макет інтерфейсу користувача допомагає створити узгоджений інтерфейс для всіх сторінок та компонентів продукту.

Макет даного сайту містить 6 прототипів:

1. Прототип головної сторінки (рис. 3.1) – описує головну сторінку вебзастосунка. На цю сторінку користувачі будуть потрапляти при відкриті сайту. Вона містить: логотип сайту (Logo) (посилання на головну сторінку), назву (посилання на головну сторінку), посилання на кошик, посилання на обрані товари, посилання на історію замовлень, назву сторінки, фільтри,

пошук і саме головне, товари. Кожний товар має назву, ціну, зображення товару, кнопку "Купити" та кнопку "Додати в обране".

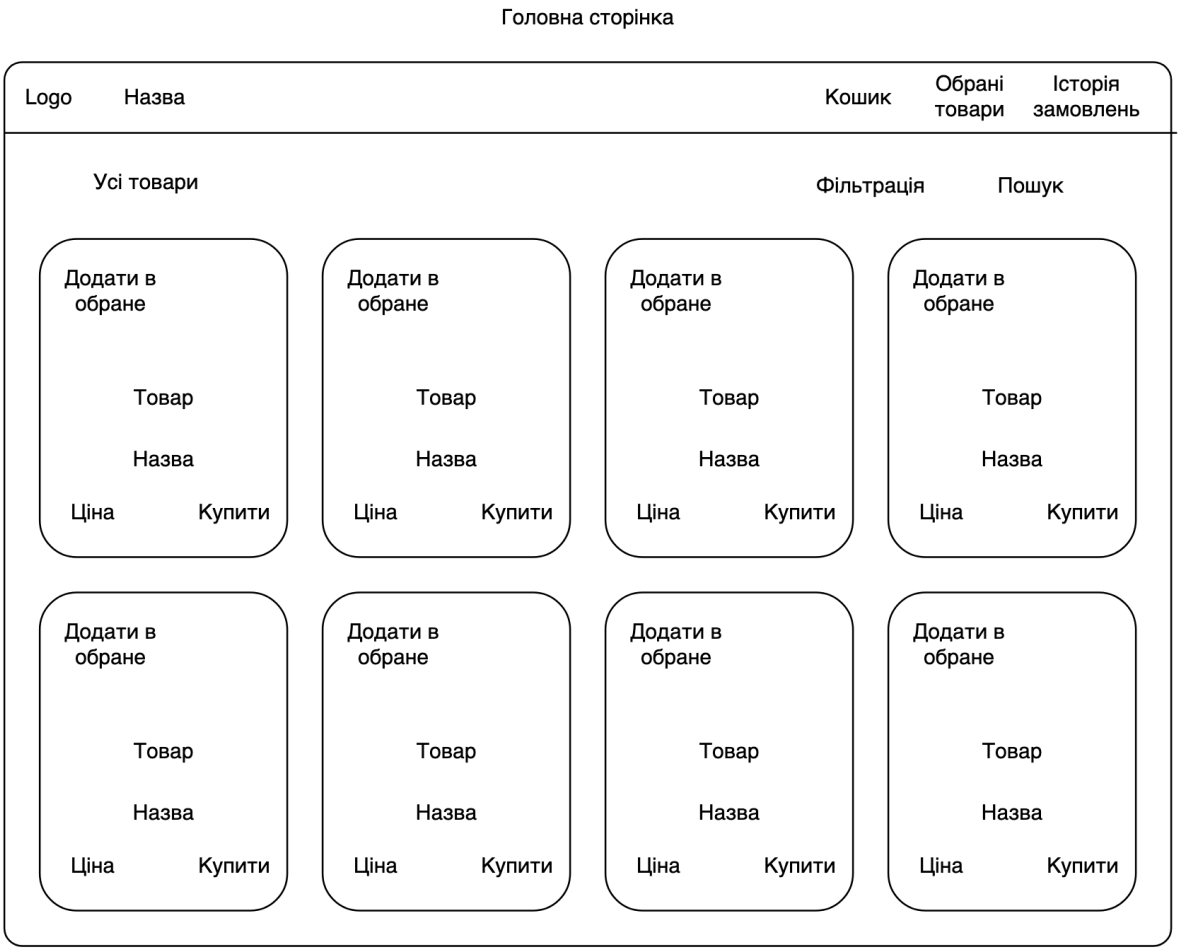

Рисунок 3.1. Макет головної сторіки

2. Прототип сторінки обраних товарів (рис. 3.2) – описує сторінку з товарами, які користувач додав в обране. На цю сторінку користувач може потрапити натиснувши на кнопку "Обрані товари" та будь-якій іншій сторінці. Вона містить: логотип сайту (Logo) (посилання на головну сторінку), назву (посилання на головну сторінку), посилання на кошик, посилання на обрані товари, посилання на історію замовлень, назву сторінки й обрані товари.

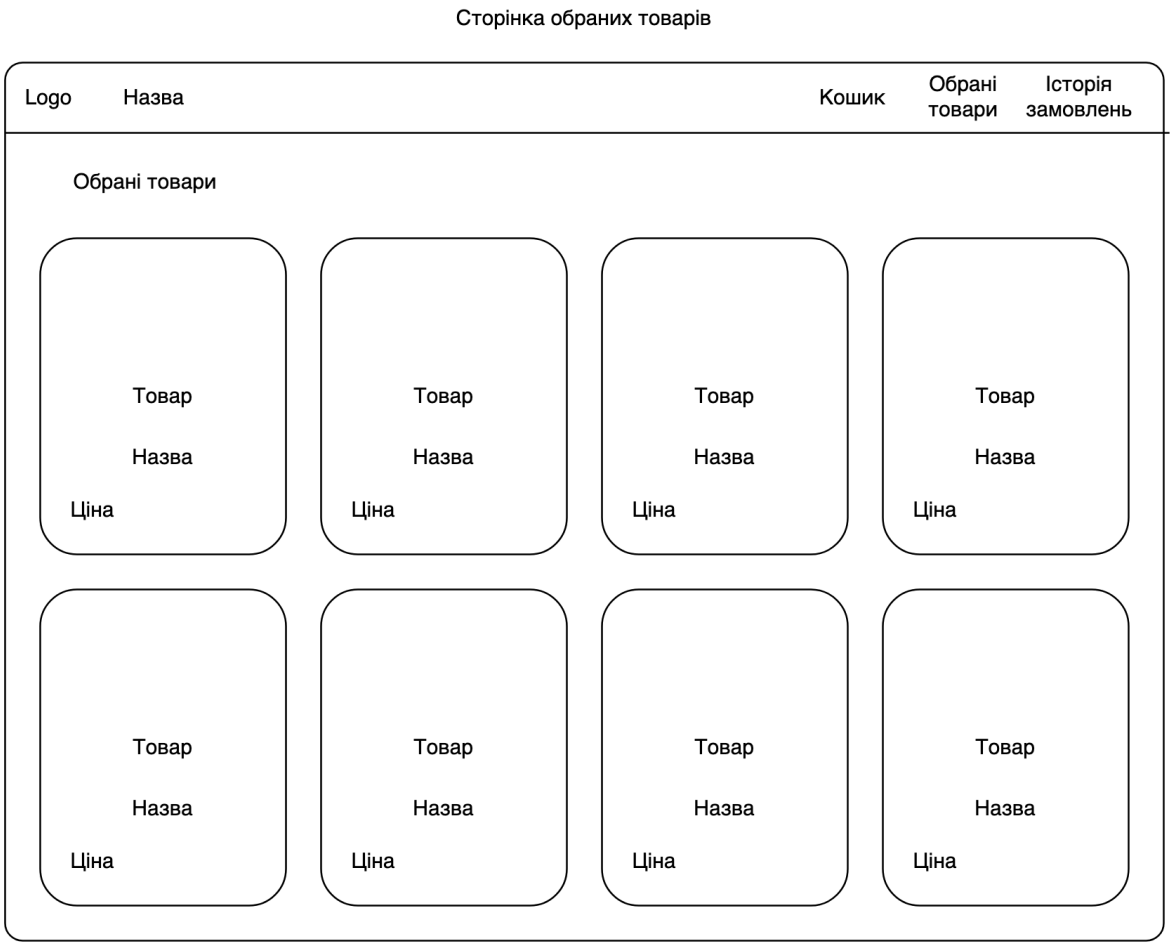

Рисунок 3.2. Макет сторінки обраних товарів

3. Прототип сторінки з історією замовлень (рис. 3.3) – описує сторінку з замовленнями, які користувач вже створив. Користувач може потрапити на цю сторінку використавши кнопку "Історія замовлень". Ця сторінка містить: логотип сайту (Logo) (посилання на головну сторінку), назву (посилання на головну сторінку), посилання на кошик, посилання на обрані товари, посилання на історію замовлень, назву сторінки й список створених замовлень.

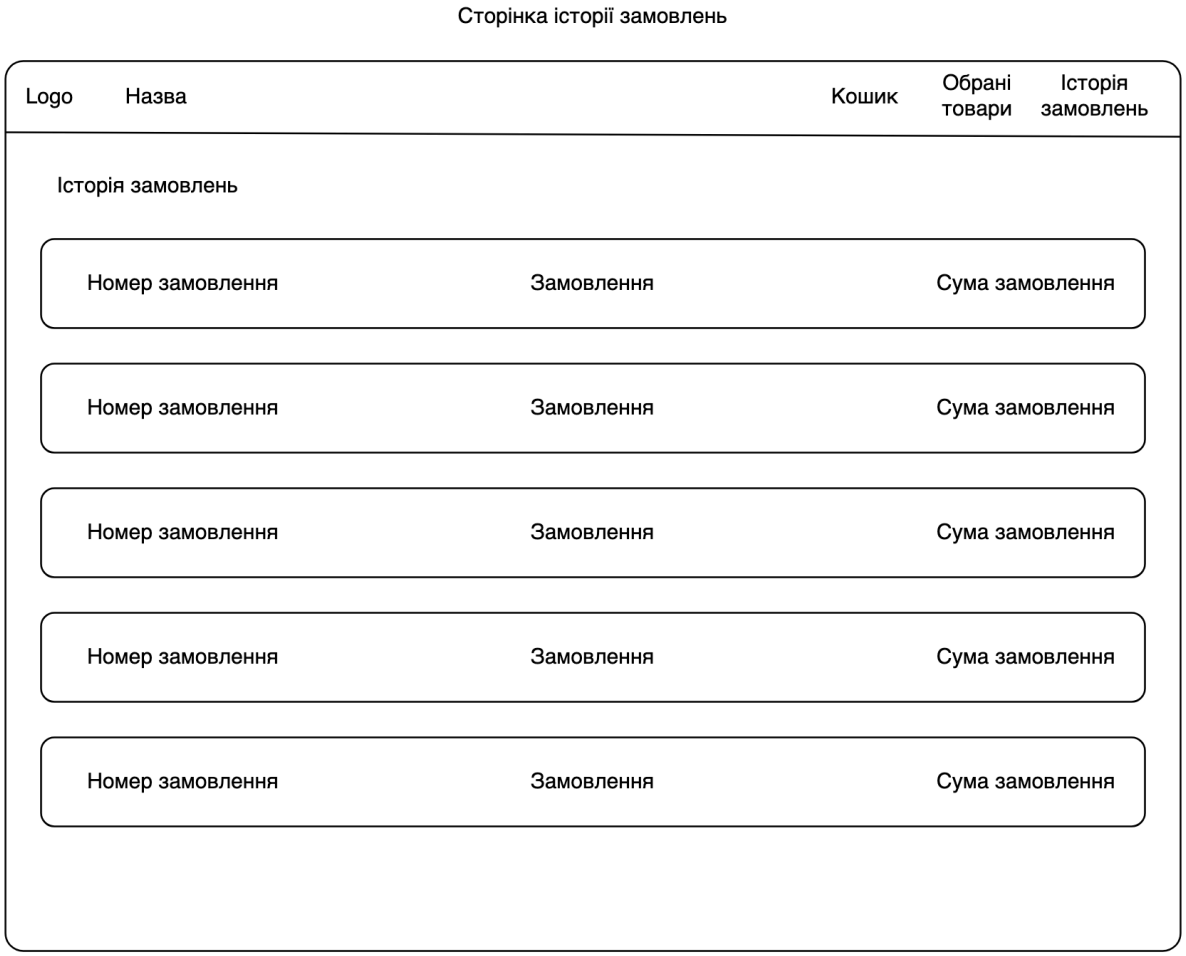

Рисунок 3.3. Макет сторінки історії замовлень

4. Прототип пустого кошика (рис. 3.4) – описує як має виглядати кошик, коли користувач не додав до нього ні одного товару. Користувач бачить цю сторінку якщо натисне на кнопку "Кошик" не додавши до нього ні одного товару. Кошик можна закрити натиснувши на сіру ділянку зліва від кошика або на кнопку "Закрити". В такому випадку кошик містить лице повідомлення, що кошик пустий та рекомендує користувачу додати в нього хоча б один товар.

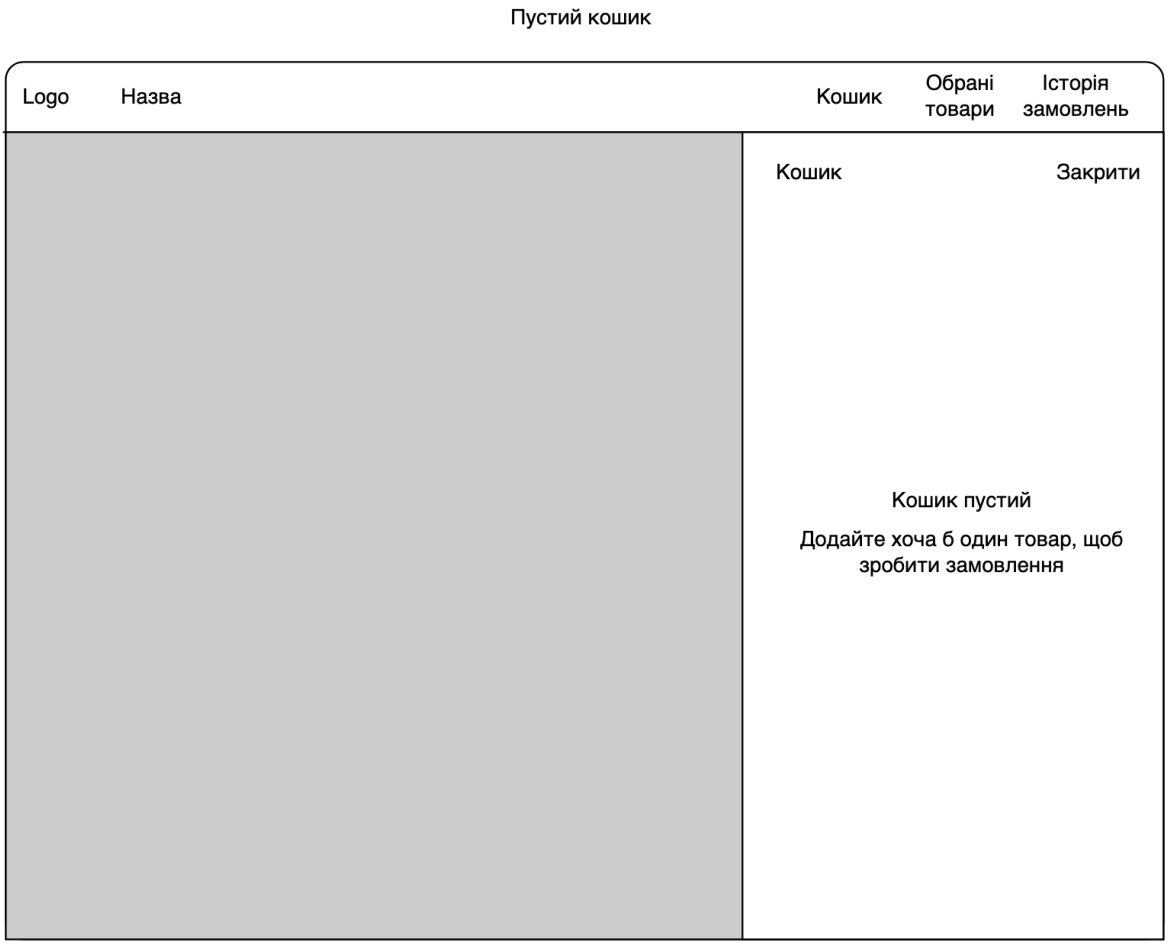

Рисунок 3.4. Макет порожнього кошика

5. Прототип кошика з товарами (рис. 3.5) – описує вигляд кошика з товарами. Користувач бачить цю сторінку якщо натисне на кнопку "Кошик" після того, як додасть хоча б один товар у кошик. Кошик можна закрити натиснувши на сіру ділянку зліва від кошика або на кнопку "Закрити". Кошик містить список товарів з їх ціною та можливістю прибрати їх з кошика, загальну суму кошика та кнопку "Замовити".

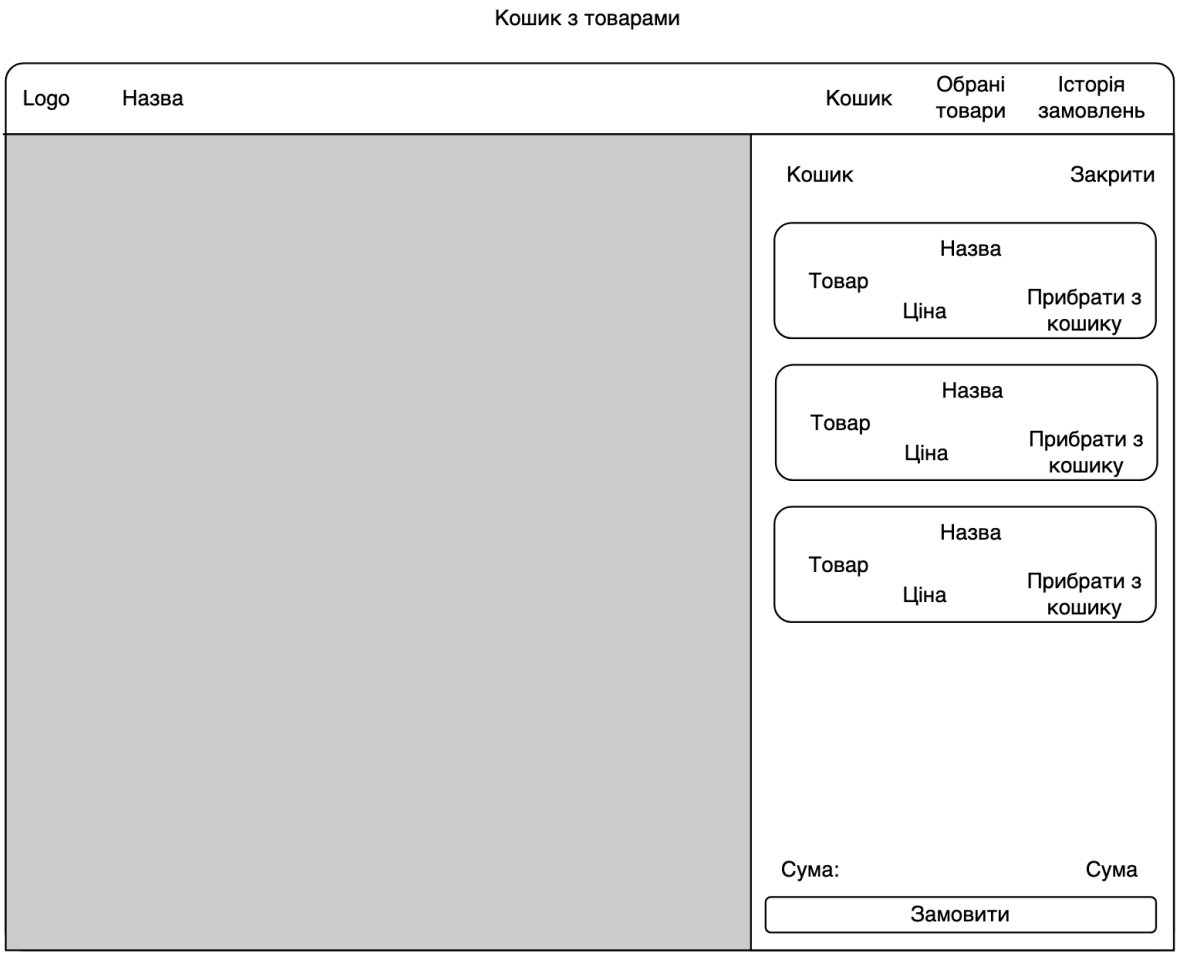

Рисунок 3.5. Макет кошика з товарами

6. Прототип кошика після оформлення замовлення (рис. 3.6) – описує вигляд кошика після натискання кнопки "Замовити" клієнтом. Він має надпис з номером замовлення та інформацією, що з покупцем зв’яжеться консультант.

Рисунок 3.6. Макет зі створеним замовленням

3.2. Створення користувацького інтерфейсу

Першим кроком у розробці користувацького інтерфейсу є ретельне вивчення макета. Це допоможе чітко зрозуміти цілі інтерфейсу, його візуальний стиль, взаємодію користувачів та загальну структуру.

Після того, як макет зрозумілий, настав час перетворити його на код HTML та CSS. HTML використовується для структурування контенту інтерфейсу. Це включає такі елементи, як заголовки, абзаци, кнопки, зображення та форми. CSS використовується для стилізації зовнішнього вигляду інтерфейсу. Це включає кольори, шрифти, розміри елементів, відстані та інші візуальні аспекти. В цьому випадку замість написання власних CSS буде використана Tailwind CSS.

JavaScript використовується для додавання динамічних елементів до інтерфейсу та реалізації взаємодії користувачів. В цьому випадку буде

використовуватися фреймворк Vue.js.

Окрім цього Vue.js дозволяє розбити один суцільний HTML файл на окремі компоненти. Це дозволяє зручніше і швидше працювати з кодом і дозволяє перевикористовувати цілі компоненти без потреби копіювання коду.

Замість використання окремих HTML файлів для кожної сторінки, було використано новітню технологію Vue router.

Vue router - це офіційний маршрутизатор для JavaScript-фреймворку Vue.js. Він дозволяє створювати односторінкові вебпрограми з динамічною зміною контенту без перезавантаження сторінки. Це означає що при переході по різним сторінкам замість завантаження кожного разу нового HTML файлу, буде відбуватися заміна контенту у даному файлі. Це дозволяє створити ефект моментального завантаження сторінки.

Проєкт містить лише один HTML файл (рис. 3.7). Він містить блоковий елемент з ідентифікатором «app». В цьому блоковому елементі Vue JS буде відображати головний компонент.

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8">
    <link rel="icon" href="/favicon.ico">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Vite App</title>
  </head>
  <body>
    <div id="app"></div>
    <script type="module" src="/src/main.js"></script>
  </body>
</html>
```

Рисунок 3.7. HTML файл вебзастосунка

Користувацький інтерфейс складається з окремих компонентів.

Першим було розроблено компонент загального вікна, який буде містити усі інші компоненти (рис. 3.8).

```
<template>
  <div>
    <div class="bg-white w-4/5 m-auto rounded-xl shadow-xl mt-10">
      <div class="p-10">
        <router-view></router-view>
      </div>
    </div>
  </div>
</template>
```

Рисунок 3.8. Код головного компонента

Замість використання окремих HTML файлів для кожної сторінки, було використано новітню технологію Vue router.

Vue router - це офіційний маршрутизатор для JavaScript-фреймворку Vue.js. Він дозволяє створювати односторінкові вебпрограми з динамічною зміною контенту без перезавантаження сторінки. Це означає що при переході по різним сторінкам замість завантаження кожного разу нового HTML файлу, буде відбуватися заміна контенту у даному файлі. Це дозволяє створити ефект моментального завантаження сторінки. За допомогою тегу «router-view» задається місце, де повині відображати ці сторінки (див. рис. 3.8).

Наступним компонентом було розроблено компонент навігаційної панелі (рис. 3.9). Він містить назву сайту, його логотип, загальну вартість кошика, посилання на кошик, посилання на обрані товари та посилання на історію замовлень. Цей компонент буде використовуватися на усіх сторінках.

Рисунок 3.9. Навігаційна панель

Наступною компонентою є компонента товарів (рис. 3.10). Ця компонента відповідає за виведення інформації про товар (назва товару, ціна

товару, фото товару) та функціонал для взаємодії з ним (додати в обрані товари або прибрати з обраних товарів, додати в кошик або прибрати з кошика).

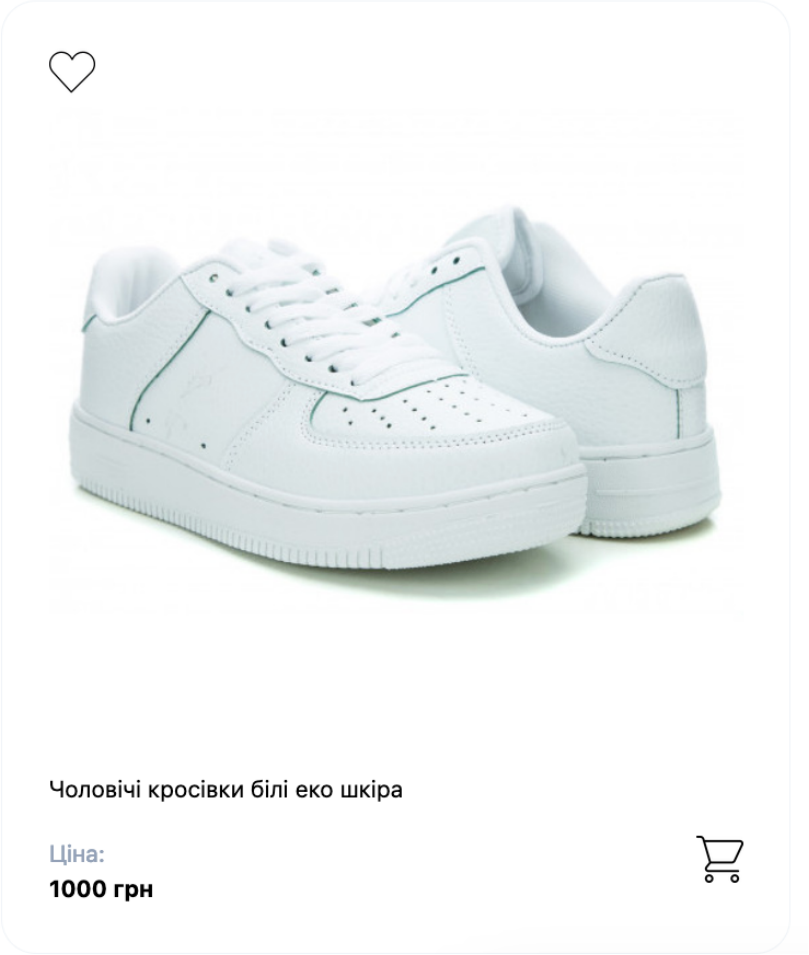

Рисунок 3.10. Компонента товару

Наступною стала компонента головної сторінки (рис. 3.11). Ця компонента окрім свого нового функціонала, ще використовує:

– компоненту товарів для відображення списку товарів, які вона отримує з сервера;

– компоненту навігаційної панелі.

Завдяки цьому, було створено першу сторінку даного інтернет-магазину.

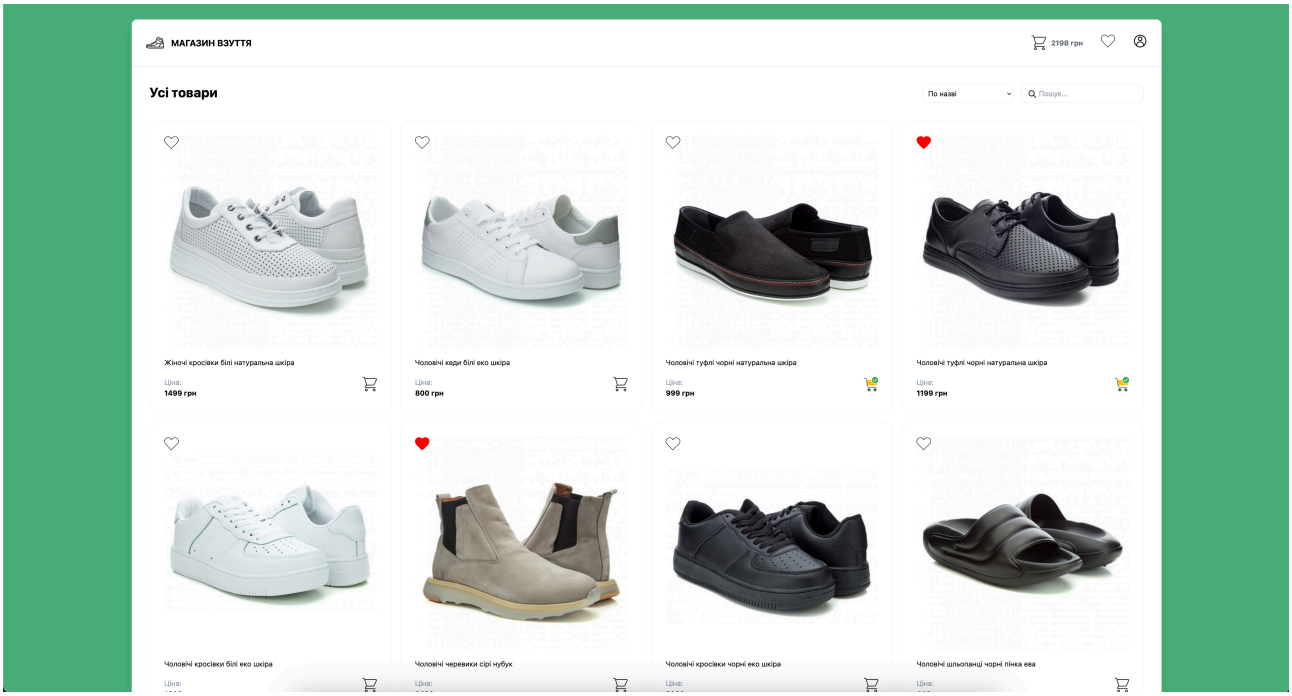

Рисунок 3.11. Компонента головної сторінки

Після компоненти головної сторінки було розроблено компоненту обраних товарів (рис. 3.12). Вона використовує ті ж самі компоненти, що і компонента головної сторінки, але з іншим функціоналом.

Рисунок 3.12. Компонента обраних товарів

Далі було розроблено компоненту елемента кошика (рис. 3.13). Ця компонента буде використовувати в компоненті кошик для виведення інформації про товари, які користувач додав у кошик. Ця компонента виводить назва товару, ціна товару, фото товару і кнопку для видалення товару з кошика.

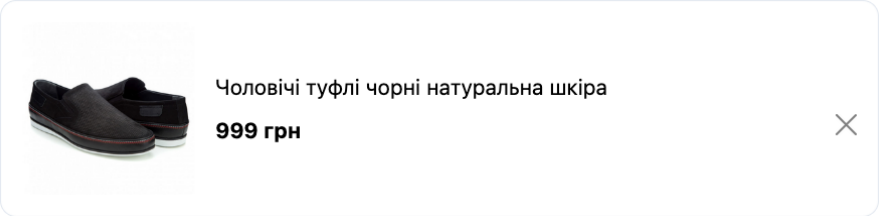

Рисунок 3.13. Компонента елемента кошика

Потім було розроблено компоненту кошика. Ця компонента має три різних стани, і в залежності від того в якому вона стані, вона має різний вигляд.

1. Кошик пустий (рис. 3.14). Виводиться повідомлення по це.

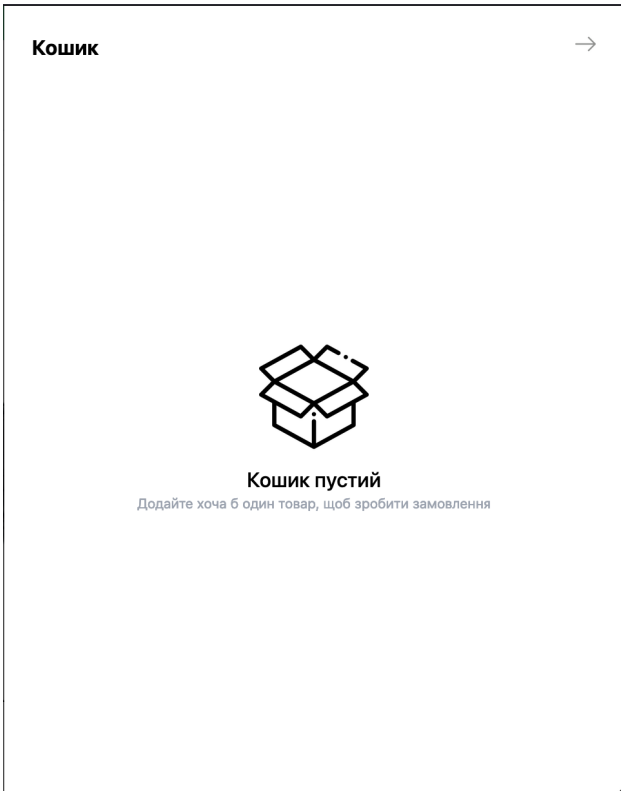

Рисунок 3.14. Порожній кошик

2. У кошика є один або більше товарів (рис. 3.15). Виводиться список компонентів елементів кошика, загальна сума і кнопка для оформлення замовлення.

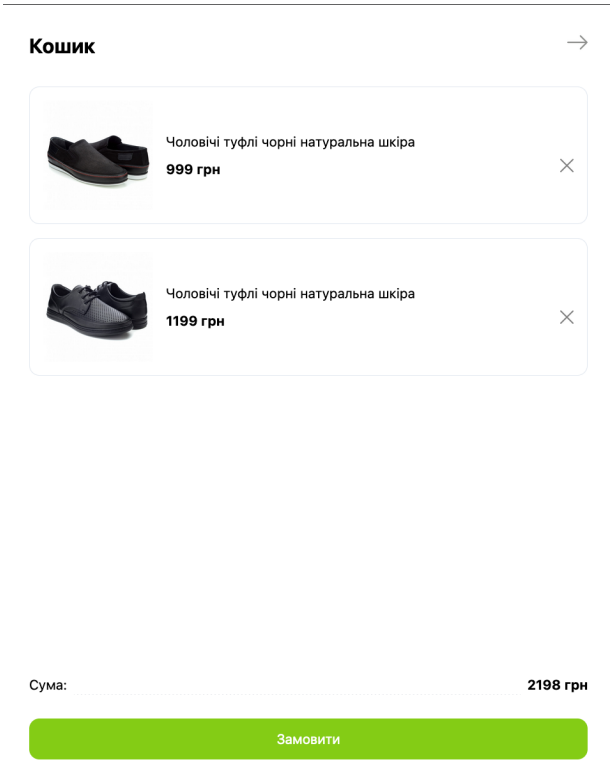

Рисунок 3.15. Кошик з товарами

3. Замовлення оформлене (рис. 3.16). Виводиться повідомлення по це.

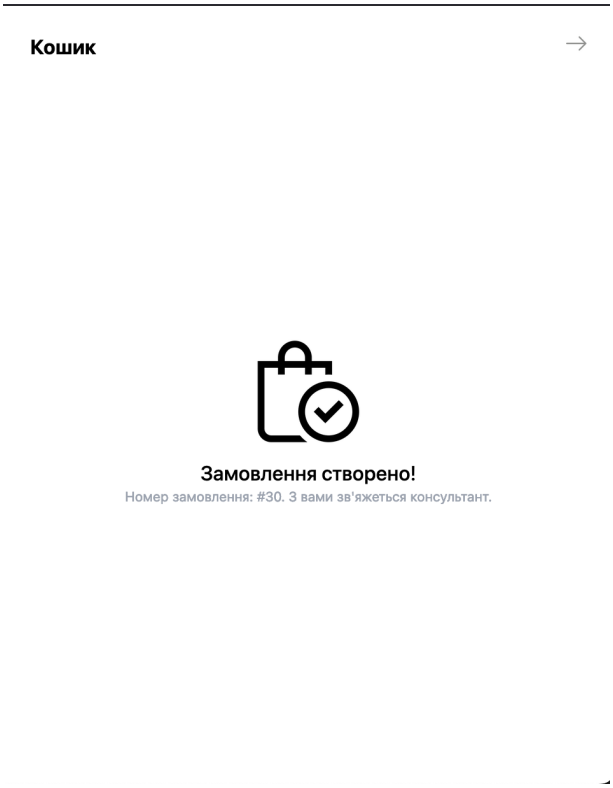

Рисунок 3.16. Кошик після створення замовлення

На відміну від товарів, обраних товарів та замовлень, які зберігаються на сервері, вміст кошика зберігається у localStorage.

localStorage - це API вебпереглядача, який використовується для зберігання даних на стороні клієнта. Ці дані зберігаються у вебпереглядачі користувача і доступні протягом усього сеансу, навіть після закриття та повторного відкриття браузера.

Це покращує зручність використання сайту і не навантажує сервер. Окрім того, localStorage використовує браузер для шифрування даних, тому вони захищені від несанкціонованого доступу

3.3. Створення серверної частини

Створення серверної частини – це процес розробки програмного забезпечення, яке працює на сервері та забезпечує логіку та дані для вебзастосунка.

Ця частина проєкту реалізована на мові JavaScript, з використанням Node.js.

Всього реалізовано 3 окремі CRUD (Create, Read, Update, Delete) API. для товарів, обраних товарів та замовлень. Тобто для кожного з цих сутностей є можливість створити (Create), прочитати (Read), оновити (Update), видалити (Delete). Це чотири основні операції, які використовуються для управління даними в більшості систем управління базами даних та програмного забезпечення.

1. Створити (Create): Ця операція використовується для додавання нових записів до бази даних. А в коді виконує SQL команду CREATE, щоб створити новий запис в потрібній таблиці.

2. Читати (Read): Ця операція використовується для отримання даних з бази даних. У коді використовує відповідну SQL команду SELEC T, щоб отримати дані з потрібної таблиці.

3. Оновити (Update): Ця операція використовується для зміни наявних записів у базі даних. Реалізує SQL команду UPDATE, щоб оновити запис в потрібній таблиці.

4. Видалити (Delete): Ця операція використовується для видалення записів з бази даних. Наприклад, ви можете використовувати команду DELETE в SQL, щоб видалити запис з потрібної таблиці.

CRUD - це простий, але потужний набір операцій, який дозволяє створювати програми, які можуть ефективно керувати даними.

Серверна частина реалізована з 2 окремих частин, які відповідають за різні задачі.

Перша це роутер (рис. 3.17), це код яких відстежує запити за спеціальними HTTP адресами, і вже після отримання запиту викликає контролер.

```
const Router = require('express')

const router = new Router()

const productsController = require('../controlllers/products')

router.get('/products', productsController.getProducts);
router.post('/product', productsController.addProduct);
router.put('/product', productsController.updateProduct);
router.delete('/product/:id', productsController.deleteProduct);

module.exports = router
```

Рисунок 3.17. Роутер товарів

Контролер (рис. 3.18) вже містить методи для опрацювання бізнес-логіки та виконує необхідні маніпуляції з базою даних

```
const db = require('../db')

You, 1 second ago | 1 author (You)
class ProductsController {
  async addProduct(req, res) {
    const { url, title, price } = req.body
    const newProduct = await db.query('INSERT INTO product (url, title, price) VALUES ($1, $2, $3) RETURNING *', [url, title, pric
    res.json(newProduct.rows[0])
  }
  async getProducts(req, res) {
    const columnMap = { 'title': "title ASC", 'price-low': 'price ASC', 'price-high': 'price DESC' }

    let params = []
    let q_start = `SELECT * FROM product `
    let q_where = ``
    if (req.query.searchQuery) {
      q_where = ` WHERE title LIKE $${params.length + 1}`
      params.push(`%${req.query.searchQuery}%`)
    }
    let q_end = ` ORDER BY ${columnMap[req.query.sortBy]}`

    const products = await db.query(q_start + q_where + q_end, params)
    res.send(products.rows)
  }
  async updateProduct(req, res) {
    const { url, title, price } = req.body
    const updProduct = await db.query('UPDATE product set url = $1, title = $2, price = $3) RETURNING *', [url, title, price])
    res.json(updProduct.rows[0])
  }
  async deleteProduct(req, res) {
    const { id } = req.params
    const deletedProduct = await db.query('DELETE FROM product WHERE id = $1  RETURNING *', [id])
    res.json(deletedProduct.rows[0])
  }
}

module.exports = new ProductsController()
```

Рисунок 3.18. Контролер товарів

3.4. Розробка логічної моделі бази даних

Логічна модель бази даних – це абстрактне представлення структури даних, яке описує що зберігається в базі даних, а також зв'язки між цими даними.

Логічна модель бази даних не залежить від конкретної системи управління базами даних та використовується для чіткого опису інформаційної структури предметної області без урахування технічних деталей реалізації (рис. 3.19).

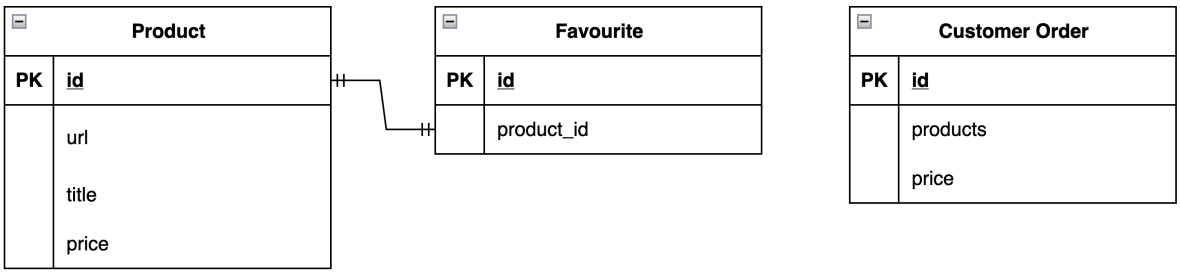

Рисунок 3.19. Логічна модель бази даних

3.3. Сутність "Product" зберігає інформацію про товари магазину. Вона має такі атрибути: id - ідентифікатор продукту, url – посилання на картинку цього товару, title – назва товару, price – ціна товару

3.4. Сутність "Favourite" використовується для збереження інформації про обрані товари. Вона має такі атрибути: id - ідентифікатор, product_id – ідентифікатор товару, який додали в обране.

3.5. Сутність "Customer Order" зберігає інформацію про замовлення. Вона має такі атрибути: id - ідентифікатор продукту, products – масив продуктів, які є в цьому замовленні, price – повна ціна замовлення.

3.5. Створення бази даних

Створення бази даних – це процес розробки структури для зберігання, організації та доступу до даних. Це фундамент, що дозволяє вебзастосунку ефективно керувати інформацією. Так, як у проєкті використовується PostgreSQL, яка є потужною та популярною системою керування реляційними базами даних, для створення таблиць та взаємодії з ними потрібно використовувати SQL. Тому для початку було написано запити на створення таблиць (рис. 3.20).

1. Таблиця product містить 4 поля: id – унікальний ідентифікатор, url – рядок з максимальною довжиною 255 символів, який зберігає посилання на зображення, title – рядок з максимальною довжиною 255 символів, який зберігає назву товару, price – числове поле, яке зберігає ціну товару.

2. Таблиця favourite містить 2 поля: id – унікальний ідентифікатор, product_id – числове поле, яке зберігає унікальний ідентифікатор відповідного товару.

3. Таблиця customer_order містить 3 поля: id – унікальний ідентифікатор, products – JSONB поле, яке зберігає об’єкт з товарами, які користувач замовив, price – числове поле, яке зберігає загальну вартість замовлення.

```sql
create table product (
  id serial primary KEY,
  url varchar(255),
  title varchar(255),
  price integer
);

create table favourite (
  id serial primary KEY,
  product_id integer,
  FOREIGN KEY (product_id) REFERENCES product (id)
);

create table customer_order (
  id serial primary KEY,
  products jsonb,
  price integer
);
```

Рисунок 3.20. Запити для створення таблиць

Для доступу до бази даних з сервера використовується бібліотека ‘pg’ (рис. 3.21).

```js
const Pool = require('pg').Pool;

const pool = new Pool({
  user: 'artem',
  password: 'root',
  host: 'localhost',
  port: 5432,
  database: "shoe_store"
});

module.exports = pool;
```

Рисунок 3.21. Код для підключення до бази даних

3.6. Взаємодія користувацького інтерфейсу з сервером

Взаємодія користувацького інтерфейсу з сервером – це процес, за допомогою якого UI надсилає запити до сервера та отримує від нього відповіді.

Цей процес включає такі етапи:

1. Користувач взаємодіє з UI. Користувач може натиснути кнопку, ввести текст або виконати іншу дію.

2. UI надсилає HTTP-запит до сервера. користувацький інтерфейс використовує HTTP-метод, такий як GET, POST, PUT або DELETE, щоб надіслати запит до сервера. Запит може містити введені користувачем дані.

Сервер обробляє запит. Сервер отримує HTTP-запит від користувацького інтерфейсу. Сервер декодує запит, щоб отримати метод, URL-адресу, заголовки та дані запиту. Сервер виконує відповідну логіку для обробки запиту.

1. Сервер надсилає HTTP-відповідь користувацькому інтерфейсу. Сервер використовує HTTP-код, такий як 200 OK, 404 Not Found або 500 Internal Server Error, щоб надіслати відповідь користувацькому інтерфейсу. Відповідь може містити дані у форматі HTML або JSON.

2. UI оновлюється на основі відповіді сервера. UI отримує HTTP-відповідь від сервера. UI декодує відповідь, щоб отримати код стану, заголовки та дані відповіді. UI оновлюється відповідно до інформації, отриманої у відповіді.

ВИСНОВКИ ДО РОЗДІЛУ 3

Під час розробки третього розділу було проведено ретельне проєктування та розробка вебзастосунка інтернет-магазину.

Завдяки впровадженню новітніх технологій, таких як Vue.js, Node.js та PostgreSQL, вдалося створити сучасний та інтуїтивно зрозумілий інтерфейс, який гарантує користувачам приємний та комфортний досвід. Швидку та економічну серверну інфраструктуру. Надійну базу даних.

Вебзастосунок пропонує широкий спектр функцій, які дозволяють:

- проглядати усі товари;
- фільтрувати товари;
- шукати товари по назву;
- додавати товари в обране і видаляти їх з обраного;
- переглядати список обраних товарів;
- додавати товари в кошик та прибирати їх;
- переглядати кошик та бачити загальну суму товарів в ньому;
- створювати замовлення;
- переглядати все створені замовлення.

Даний вебзастосунок стане незамінним інструментом для продажу взуття та допоможе магазинам досягти успіху.

Вебзастосунок було протестовано у Mozilla Firefox та Google Chrome.

ВИСНОВКИ

Результатом виконання дипломного проєкту (кваліфікаційної роботи) є універсальний вебзастосунок інтернет-магазину. Він відповідає усім сучасним потребам бізнесу та є гарним інструментом для покращення будь-якого бізнесу з продажу взуття.

Під час виконання дипломного проєкту (кваліфікаційної роботи) було виконано усі поставлені цілі, а саме:

– Проаналізовано предметну область вебзастосунків інтернет-магазинів;

– Досліджено основні етапи створення вебзастосунків;

– Виконано аналіз конкурентів;

– Проаналізовано і вибрати технології та інструменти для розробки веброзробки;

– Виконано проєктування вебзастосунка інтернет-магазину взуття;

– Виконано розробку вебзастосунка інтернет-магазину взуття.

Після ретельного вивчення усіх доступних та відповідаючих потребам технологій, було обрано:

– мову програмування JavaScript, за її не оцінені переваги у різноманітті фреймворків та бібліотек, зручності та простоті використання. А саме головне за можливість розробляти як користувацький інтерфейс, так і серверну частину, використовуючи лише одну мову програмування;

– фреймворк Vue.js, як один із новітніших фреймворків цієї категорії з найкращими показниками у легкості у вивченні, гнучкості та масштабованості для проєктів різної складності;

– CSS фреймворк Tailwind CSS, як більш лаконічний та читабельний, а також значно спрощующий процес розробки UI.

– середовище виконання JavaScript Node.js тому, що він є найпоширенішим і найзручнішим для створення серверних застосунків;

– SQL, тому що вона є потужною та гнучкою мовою, яка використовується для управління даними в реляційних базах даних. Вона має просту структуру,

легка у вивченні та використовується в широкому спектрі застосунків.

– реляційну систему керування базами даних PostgreSQL, за її найкращу надійність, масштабованість та зручність використання серед безкоштовних реляційних систем керування базами даних.

– редактор коду Visual Studio Code, як найкращій з безкоштовних інструментів розробки проєкту будь-якого масштабу та складності.

Вебзастосунок вражає своїм елегантним, простим у використанні та інтуїтивно зрозумілим інтерфейсом. Він розроблений таким чином, щоб кожен користувач, незалежно від досвіду роботи з комп'ютером, міг легко знайти те, що шукає, та здійснити необхідні дії.

Завдяки швидкому та масштабованому серверу, а також надійній базі даних, вебзастосунок гарантує:

– Безперебійну роботу: Ви можете бути впевнені, що вебзастосунок буде працювати безперебійно, навіть при великому навантаженні.

– Високу продуктивність: Сторінки завантажуються швидко, а дії виконуються блискавично, що робить роботу з вебзастосунком приємною та економною.

Завдяки цьому вебзастосунку покупці зможуть легко та швидко знаходити та купувати взуття, що відповідає їхнім потребам та стилю. Магазини ж отримають зручний та економічний канал для розширення аудиторії та збільшення продажів. Вебзастосунок гнучкий та адаптується до будь-якого магазину взуття. Можна без проблем інтегрувати його з наявною системою магазину, а також додати необхідні функції, такі як чат-бот, програма лояльності або віртуальний помічник.

В рамках цього проєкту було здобуто цінні знання та практичні навички. Було набуто навички обрання оптимальних інструментів для роботи, планування й реалізації проєкту. Ці знання та навички стануть моїм надійним фундаментом для подальшого професійного розвитку та успішної кар'єри.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Робсон Е., Фрімен Е. Head First. Програмування на JavaScript. Фабула, 2022. 672 с.

2. Sufyan bin Uzayr, Mastering Vue.js: A Beginner's Guide. Taylor & Francis, 2022. 250 p.

3. John Scott, Node.js Design Patterns, Manning Publications, 2022. 352 p.

4. MDN Web Docs – HTML: HyperText Markup Language URL: https://developer.mozilla.org/en-US/docs/Web/HTML (дата звернення: 06.05.2024).

5. CSS Tutorial URL: https://www.w3schools.com/css/ (дата звернення: 06.05.2024).

6. Документація Tailwind CSS URL: https://v2.tailwindcss.com/docs (дата звернення: 06.05.2024).

7. Сучасний підручник з JavaScript URL: https://uk.javascript.info/ (дата звернення: 07.05.2024).

8. Learn Node.js URL: https://nodejs.org/en/learn/getting-started/introduction-to-nodejs (дата звернення: 08.05.2024).

9. Visual Studio Code URL: https://code.visualstudio.com/ (дата звернення: 08.05.2024).

10. What is SQL? URL: https://aws.amazon.com/what-is/sql/ (дата звернення: 09.05.2024).

11. PostgreSQL Tutorial URL: https://www.postgresqltutorial.com/ (дата звернення: 10.05.2024).

12. PostgreSQL Documentation URL: https://www.postgresql.org/docs/current/index.html (дата звернення: 10.05.2024).

13. Положення про кваліфікаційні роботи (проєкти) здобувачів вищої освіти Національного авіаційного університету. СМЯ НАУ П 03.01(10) – 03 – 2024