МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Сумський державний університет

Факультет електроніки та інформаційних технологій

Кафедра комп'ютерних наук

«До захисту допущено»

В.о. завідувача кафедри

Ігор ШЕЛЕХОВ

(підпис)

01 червня 2024 р.

КВАЛІФІКАЦІЙНА РОБОТА

на здобуття освітнього ступеня бакалавр

зі спеціальності 122 – Комп'ютерні науки,

освітньо-професійної програми «Інформатика»

на тему: «Вебсайт інтернет-магазину чоловічого взуття»

здобувача групи ІН – 02 Борухи Артема Володимировича

Кваліфікаційна робота містить результати власних досліджень. Використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело.

Артем БОРУХА

(підпис)

Керівник,

асистент кафедри комп'ютерних наук

кандидат фізико-математичних наук Олександр ВЛАСЕНКО

(підпис)

Суми – 2024

Сумський державний університет
Факультет електроніки та інформаційних технологій
Кафедра комп'ютерних наук

«Затверджую»
В.о. завідувача кафедри
_____ Ігор ШЕЛЕХОВ
(підпис)

ІНДИВІДУАЛЬНЕ ЗАВДАННЯ НА КВАЛІФІКАЦІЙНУ РОБОТУ

на здобуття освітнього ступеня бакалавра

зі спеціальності 122 – Комп'ютерні науки, освітньо-професійної програми «Інформатика»
здобувача групи ІН-02 Борухи Артема Володимировича

1. Тема роботи: «Вебсайт інтернет-магазину чоловічого взуття» затверджена наказом по СумДУ від *«22» квітня 2024 р. № 0414-VI*
2. Термін здачі здобувачем кваліфікаційної роботи *до 01 червня 2024 року*
3. Вхідні дані до кваліфікаційної роботи _____
4. Зміст розрахунково-пояснювальної записки (перелік питань, що їх належить розробити) *1) Аналіз проблеми предметної області, постановка й формування завдань дослідження. 2) Огляд технологій, що використовуються для створення інтернет-магазинів. 3) Розробка сайту для продажу чоловічого взуття. 4) Аналіз результатів.*
5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень)_____
6. Консультанти до проєкту (роботи), із значенням розділів проєкту, що стосується їх

Розділ	Консультант	Підпис, дата	
		Завдання видав	Завдання прийняв

7. Дата видачі завдання «08» квітня 2024 р.

Завдання прийняв до виконання _____ (підпис) Керівник _____ (підпис)

КАЛЕНДАРНИЙ ПЛАН

№ п/п	Назва етапів кваліфікаційної роботи	Термін виконання	Примітка
1	*Аналіз проблеми предметної області, постановка й формування завдань дослідження*		
2	*Огляд технологій, що використовуються для створення інтернет-магазинів*		
3	*Розробка сайту для продажу чоловічого взуття*		
4	*Аналіз результатів*		
5	*Оформлення пояснювальної записки до кваліфікаційної роботи*		

Здобувач вищої освіти _____ (підпис) Керівник _____ (підпис)

АНАТОЦІЯ

Записка: 69 ст., 54 рис., 2 табл., 24 використаних джерел

Обґрунтування актуальності теми роботи. Тема кваліфікаційної роботи є актуальною, оскільки присвячена розробці інтернет-магазину.

Розробка вебсайту для інтернет-магазину чоловічого взуття є надзвичайно актуальною з кількох причин, пов'язаних із сучасними тенденціями ринку, змінами у споживчій поведінці та технологічними нововведеннями.

Об’єкт дослідження – процес створення вебсайту інтернет-магазину чоловічого взуття.

Мета роботи – реалізація інформаційного та програмного забезпечення вебсайту інтернет-магазину чоловічого взуття.

Методи дослідження – аналіз існуючих рішень, технології існуючих рішень створення інтернет-магазинів.

Результати – розроблено інформаційну систему для продажу та покупки чоловічого взуття у мережі Інтернет. Створений програмний продукт зручний у використанні, інтерфейс та функціонал інтуїтивно зрозумілий.

ІНТЕРНЕТ-МАГАЗИН, ВЕБСАЙТ, JAVASCRIPT, MOKKY, VUE, CSS, HTML

ЗМІСТ

ВСТУП

У сучасному цифровому світі електронна комерція займає визначне місце у вирішенні потреб споживачів та розвитку бізнесу. З кожним роком зростає популярність інтернет-магазинів, які стають важливим інструментом для продажу товарів та послуг. У цьому контексті розробка та ефективне функціонування вебсайтів для інтернет-магазинів є ключовим завданням для бізнесу, спрямованим на досягнення успіху та конкурентоспроможності.

Сайт – це сукупність вебсторінок, оформлених в одному стилі та об'єднаних спільною концепцією. Усі вони мають унікальні адреси, але масивність даних пов'язана доменним ім'ям. Користувач сприймає їх як єдине ціле. Є можливість переходу між сторінками одного сайту [1].

Актуальність роботи полягає в тому, що українці переважно купують товари через онлайн-магазини. Згідно з дослідженням компанії Gradus Research, найчастіше в інтернет-магазинах громадяни України купують одяг і взуття (56%) та електроніку (52%). 38% замовляють онлайн засоби гігієни і догляду за собою, а 28% - ліки [2]. Зважаючи на ці дані, потреба у розвитку комерційного онлайн-сегменту лише зростає. Тому постала необхідність у розробці сайту для інтернет-магазину.

Інтернет-магазин – це складний ресурс, який крім каталогу з товарами включає цілу систему обслуговування клієнта. Користувач може не тільки зв'язатися з менеджером компанії, але й оформити замовлення, вибрати форму оплати та доставки покупки. Інтернет-магазини оснащуються такою функцією, як віртуальний кошик. Це розділ, в який покупець може відкладати товари, що сподобалися, щоб потім повернутися до їх розгляду [3].

Мета роботи. У кваліфікаційній роботі представлено аналіз сучасних тенденцій у сфері електронної комерції та вебдизайну, виокремлені основні вимоги до функціоналу та дизайну вебсайту інтернет-магазину взуття. На основі отриманих знань та враховуючи передовий досвід у цій галузі, буде розроблено

концепцію та створено прототип вебсайту, а також реалізовано його програмну частину.

Об'єктом дослідження є процес розробки вебсайту для інтернет-магазину взуття.

Новизна роботи полягає у розробці клієнто-орієнтованого вебсайту, який буде допомагати в управлінні процесом продажу взуття через інтернет.

Структура роботи складається зі вступу, аналізу програмних продуктів-аналогів, постановки задачі, вибору мов програмування, фреймворків, бібліотек для реалізації поставлених задач, практичної реалізації та тестування використання вебсайту, висновків, списку використаних джерел та додатків.

1 АНАЛІЗ ВІДОМИХ РІШЕНЬ

1.1 Сучасні інтернет-магазини

Зараз, у XXI ст. люди максимально заощаджують свій час, роблячи максимальну кількість покупок онлайн. Перевагами сучасних інтернет-магазинів можна виокремити наступні:

- економія часу та зусиль: не потрібно стояти в черзі каси, для того щоб оплатити всі куплені товари. Користувач може робити замовлення прямо зі свого будинку або роботи не витрачаючи час на дорогу до торгового центру;
- зручність здійснення покупок не виходячи з дому;
- великий асортимент товарів: продавці виставляють всі наявні товари на різних маркетплейсах або у власних магазинах. Це дає змогу обирати товар з безлічі різних моделей, з можливістю порівняти характеристики та ціну;
- хороші знижки: багато інтернет-магазинів і маркетологів пропонують покупцям знижки, адже немає необхідності платити гроші за оренду приміщень [4].

Як уже було згадано у вступі, 56% покупців купляють одяг та взуття онлайн. Згідно дослідження на сайті Торгсофта, список топ-10 найпопулярніших сайтів для взуття в Україні наступний:

- Intertop.ua;
- Sezon;
- Mida.style;
- KASANDRA;
- Rieker;
- KACHOROVSKA;
- Respect;
- Bartek;
- HVOYA;
- STEPTER [5].

Для порівняння інтерфейсу користувача та зручності використання розглянемо наступні: Intertop.ua, Sezon та Mida.style, оскільки вони найпопулярніші у опитаній аудиторії, тому є логічним є звернути увагу на те, що цінує аудиторія. Головними критеріями при перевірці будуть наступні: сортування за ціною, сортування за типом, фільтрація за кольором та брендом.

Першим аналогом інтернет-магазину взуття є магазин «Intertop.ua» [6]. Варто відзначити розмежування товарів за 4 головними категоріями: чоловічий, жіночий, дитячий одяг та товари для дому (рис. 1.1). Користувач відразу виконує перехід на необхідну категорію та без зайвих нюансів може обрати товар.

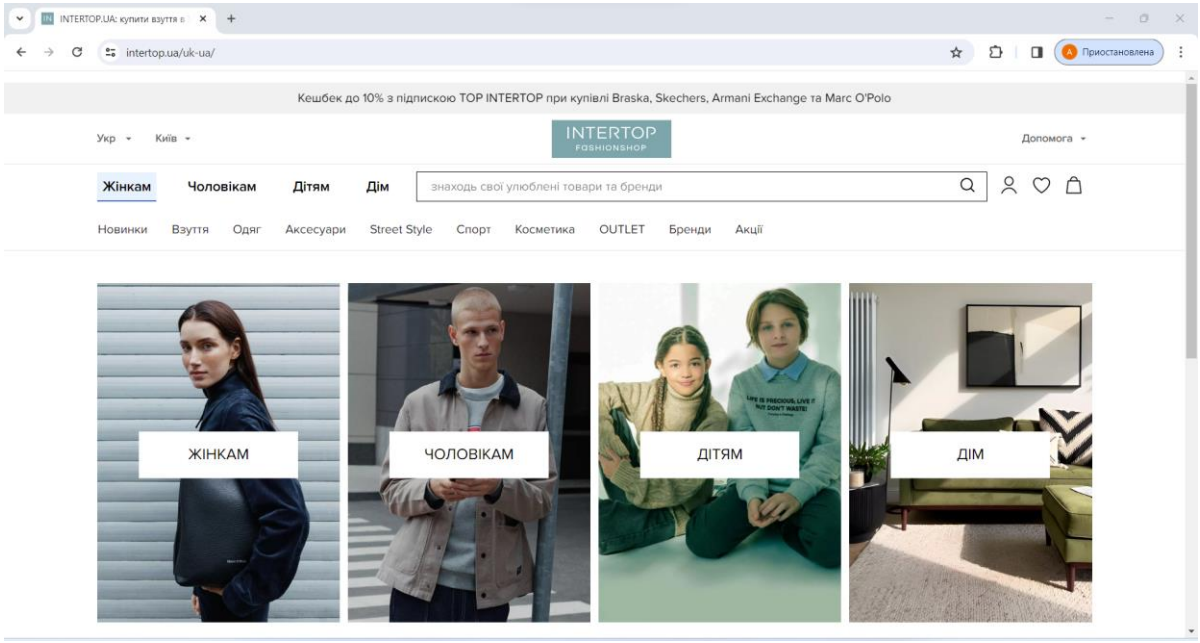

Рисунок 1.1 – Головна сторінка інтернет-магазину Intertop.ua

Наступною перевагою є можливість авторизуватися. Для цього потрібно увести телефон або Email та пароль. Користувач зі свого облікового запису може переглянути замовлення, адреси минулих доставок, бонуси (нараховуються на наступні покупки при замовлені), бонуси та улюблені бренди (рис. 1.2).

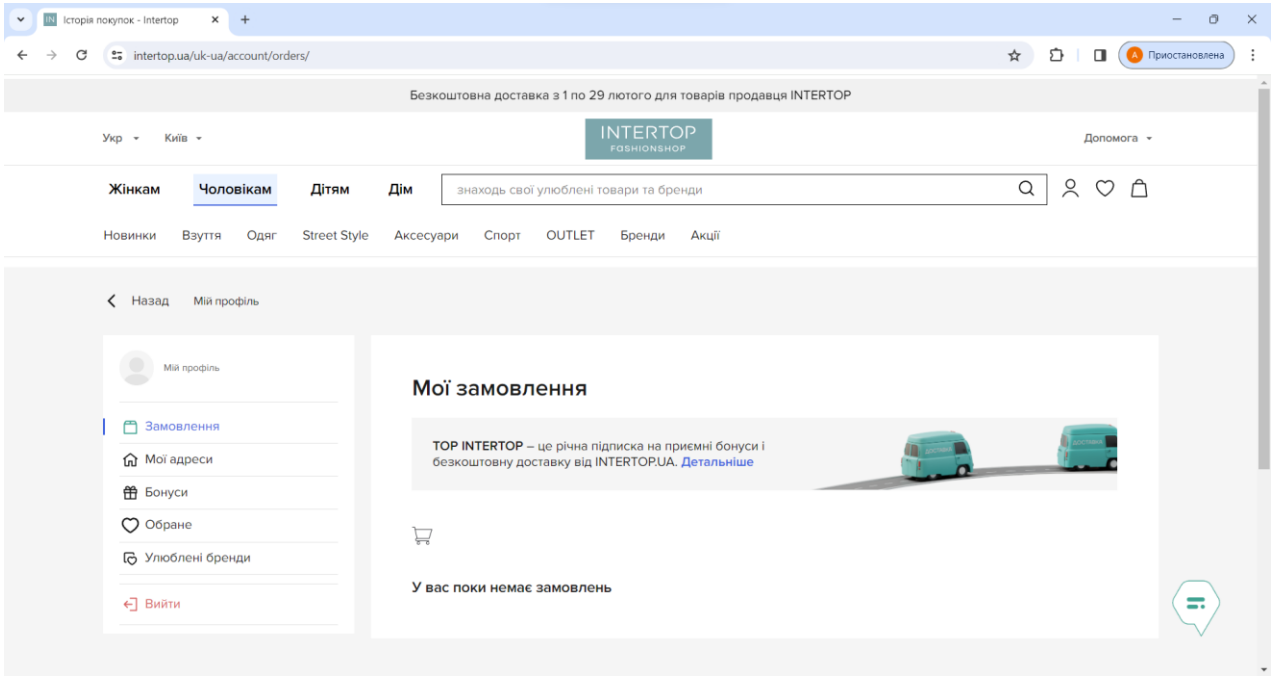

Рисунок 1.2 – Обліковий запис користувача інтернет-магазина Intertop.ua

Недоліком є занадто велика кількість повідомлень про знижки та рекламні пропозиції: з’являються при переході на кожну нову категорію (рис. 1.3)

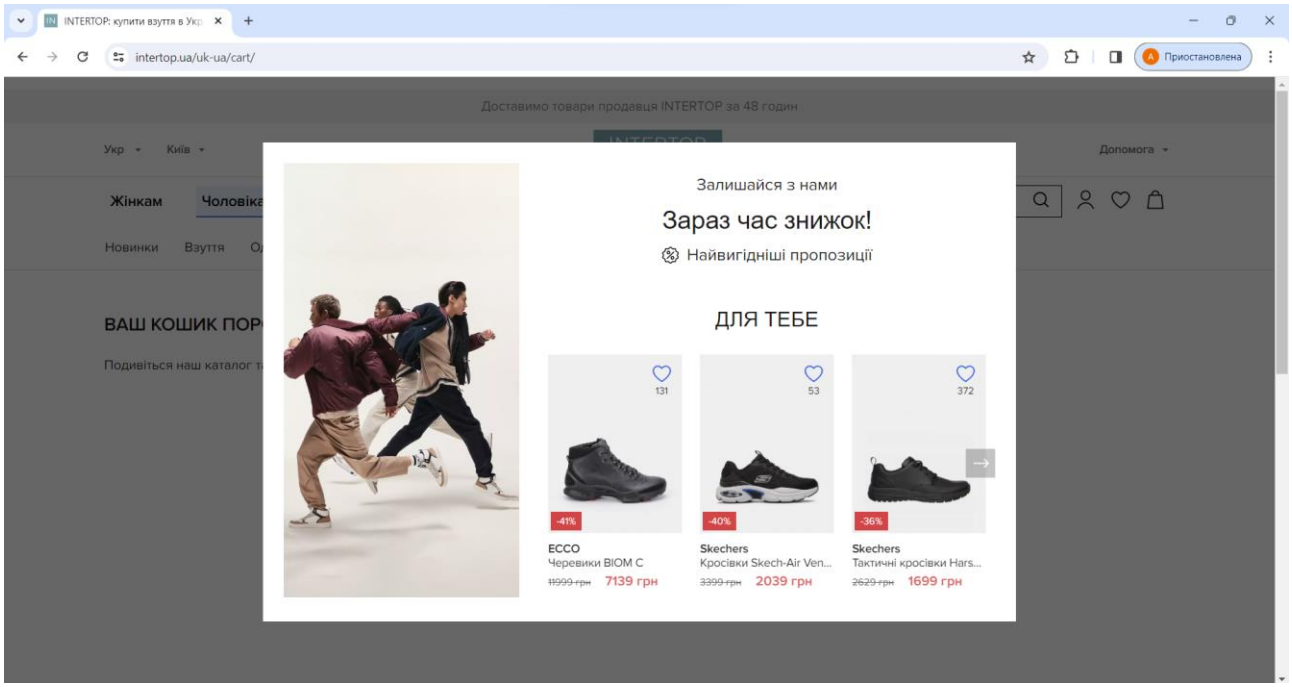

Рисунок 1.3 – Повідомлення про знижки

Варто зауважити, що незважаючи на популярність серед українських покупців, вебсайт некоректно відображає контент при розширенні 330 х 720 px та при меншому не повністю відображає посилання на соціальні мережі (рис. 1.4 – 1.5).

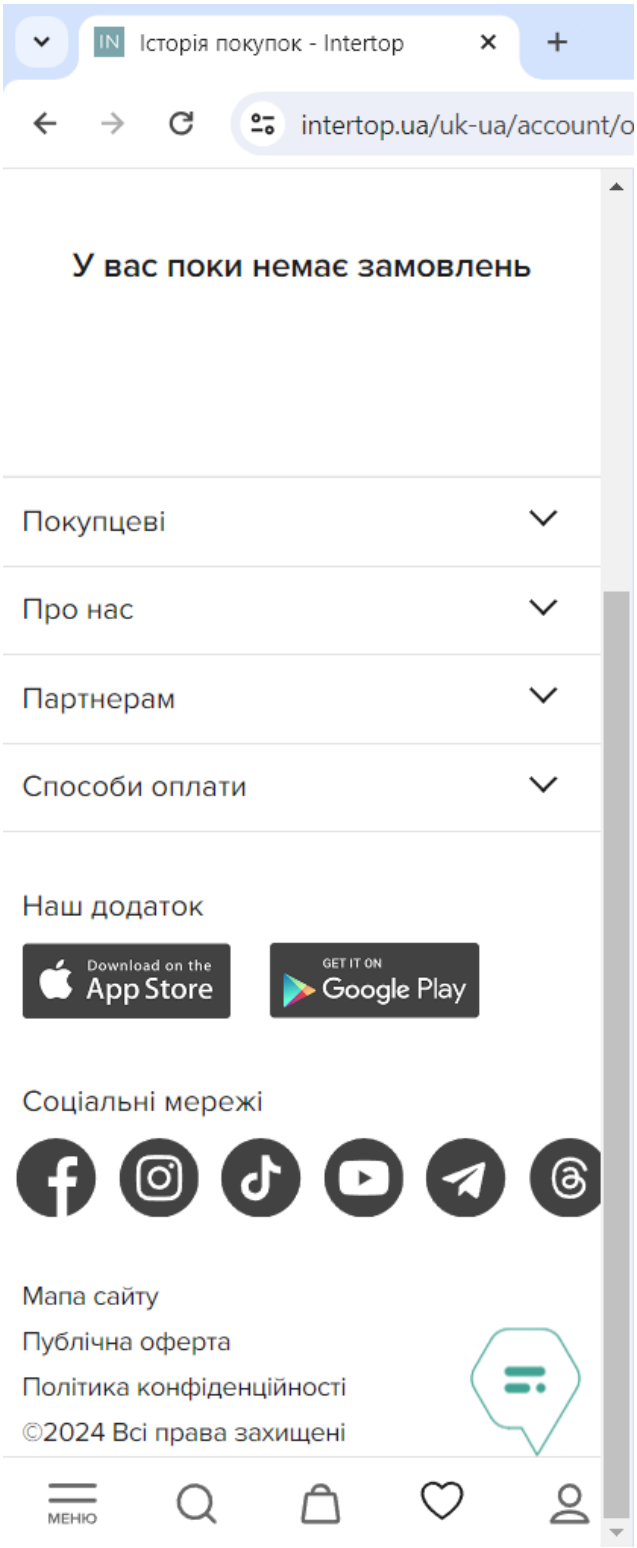

Рисунок 1.4 – Некоректне відображення при розширенні 330 х 720 px

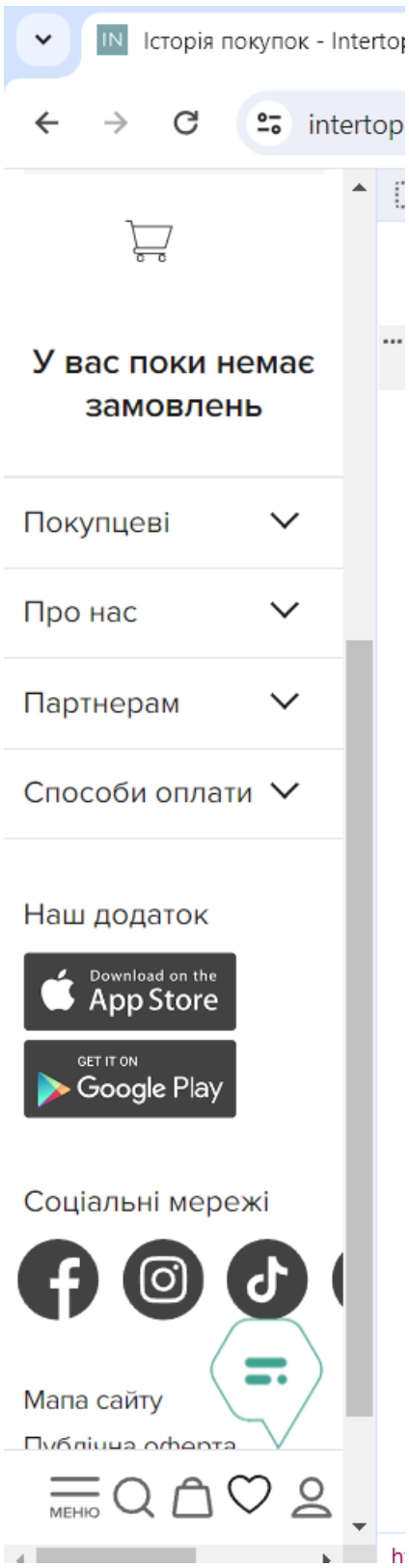

Рисунок 1.5 – Некоректне відображення при розширенні 200 x 720 px

Другим аналогом інтернет-магазину є Sezon.ua [7]. Аналогічно як і на інтернет-магазині Intertop сайт має достатньо інформації стосовно асортименту та знижок (рис. 1.6).

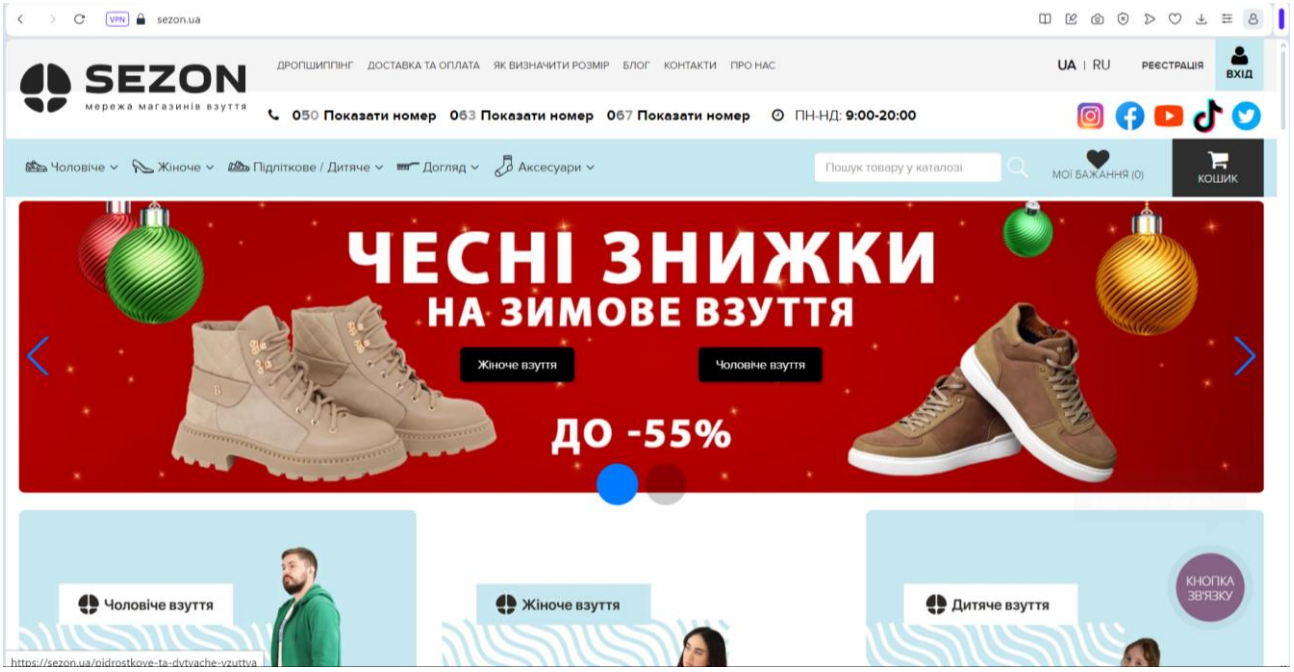

Рисунок 1.6 – Головна сторінка інтернет-магазину Sezon.ua

Відмінною особливістю даного вебсайту є наявність статті «Як визначити розмір взуття» (рис. 1.7).

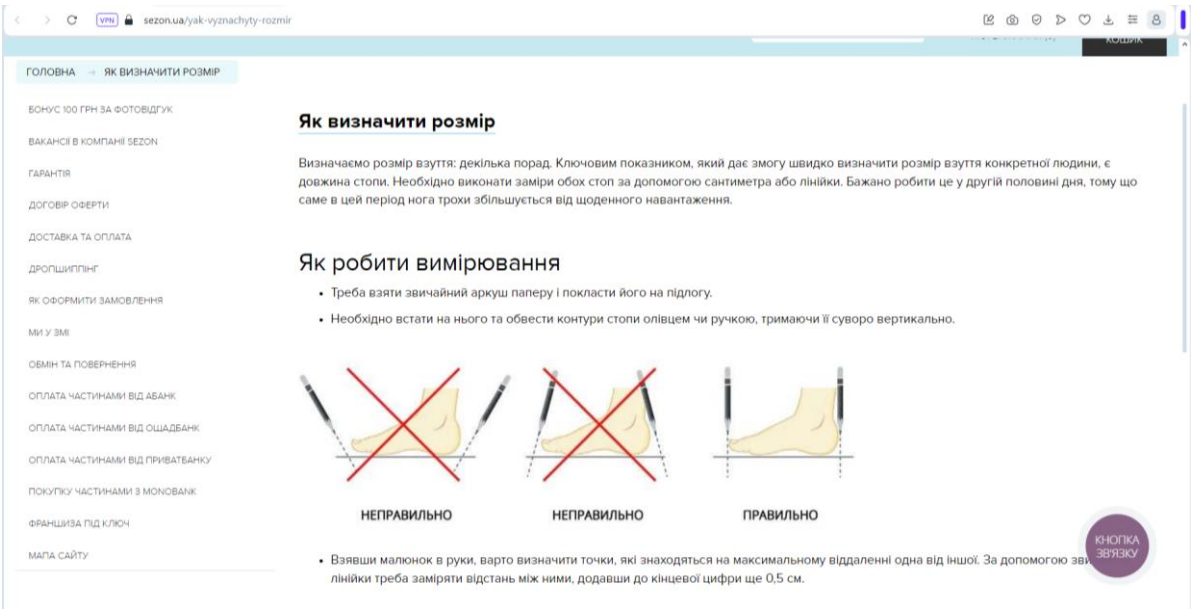

Рисунок 1.7 – Окрема сторінка для статті «Як визначити розмір взуття»

Обсяг продукції для продажу становить більше 500 найменувань, у зв’язку із чим передбачено використання фільтрів, за допомогою яких простіше та швидше обирати майбутню покупку.

Покупець може використати наступні фільтри (рис. 1.8):

- розмір;
- ціна;
- матеріал;
- колір;
- сезон;
- стиль;
- підбір;
- застібка;
- країна виробник;
- утеплювач;
- акційний товар.

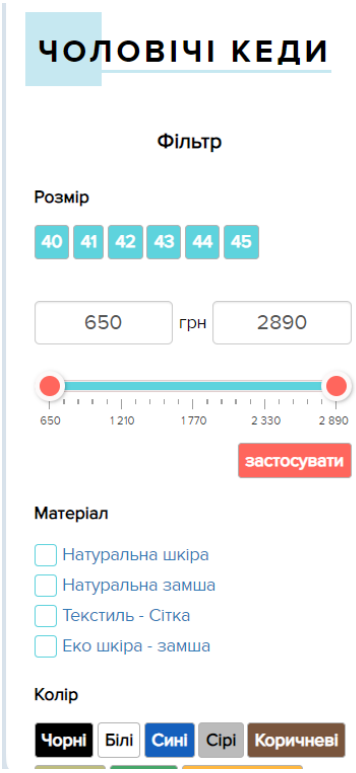

Рисунок 1.8 – Фільтри для категорії «Чоловіче взуття»

Єдиним недоліком сайту інтернет-магазину Sezon.ua є відсутність англомовної версії. У час, коли багато іноземців перебувають на території

Україні, необхідно зробити умови, аби вони могли скористатись послугами будь-якого магазину на рівних умовах з українцями (рис. 1.9).

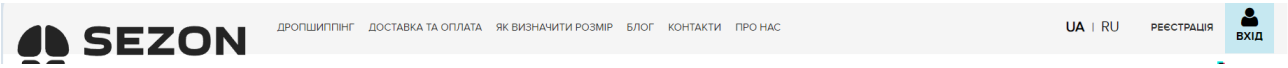

Рисунок 1.9 – Відсутність англомовної версії сайту

Третім розглянутим сайтом є інтернет-магазин Mida.style [8]. Аналогічно до двох попередніх магазинів найбільше місця займає банер з інформацією про актуальні знижки та новинки сезону (рис. 1.10).

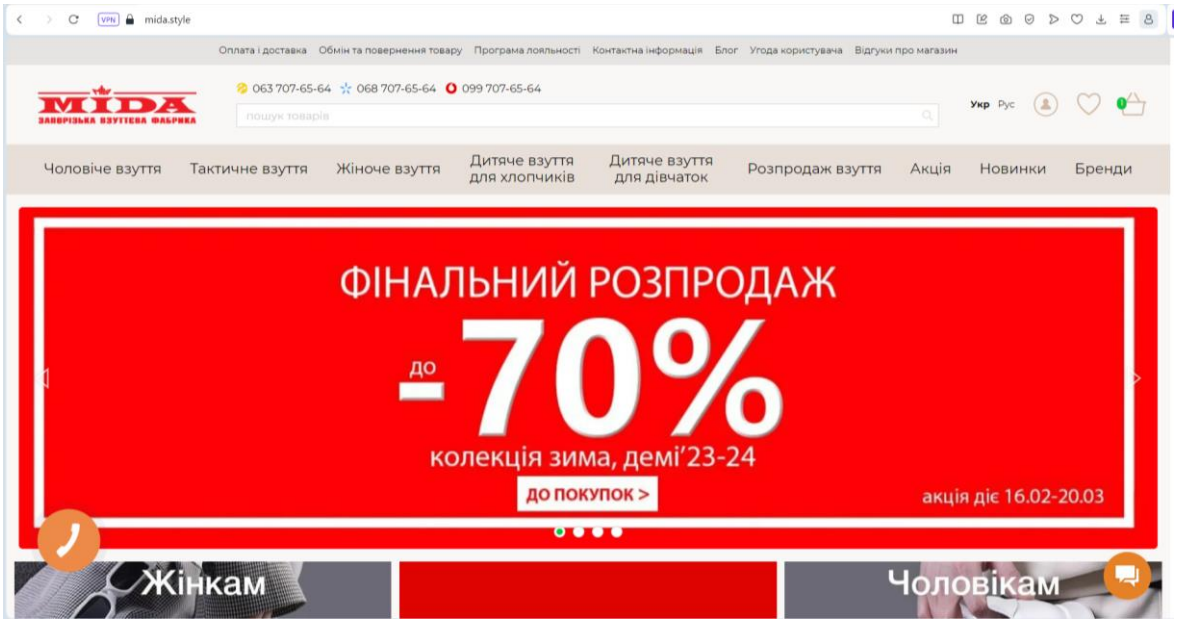

Рисунок 1.10 – Головна сторінка інтернет-магазину Mida.style

У оформленні усіх елементів переважає червоний колір. Згідно дослідження психологів, даний колір – сигнал тривоги, неспокою, тому можливі проблеми в продажу товару.

Користувач може зробити фільтрацію товару за приналежністю за статтю, ціною, видом (тактичні кросівки, мокасини, кеди, туфлі та ін.), сезоном, розміром, стилем, кольором, матеріалом верху, внутрішнім матеріалом, застібкою, типом підошви, висотою підбору та типом носка (рис. 1.11). Аналізуючи даний перелік, можна зрозуміти, що підбір взуття максимально

зручний, оскільки покупець може увести усі свої критерії та отримати підбір, який наявний в асортименті.

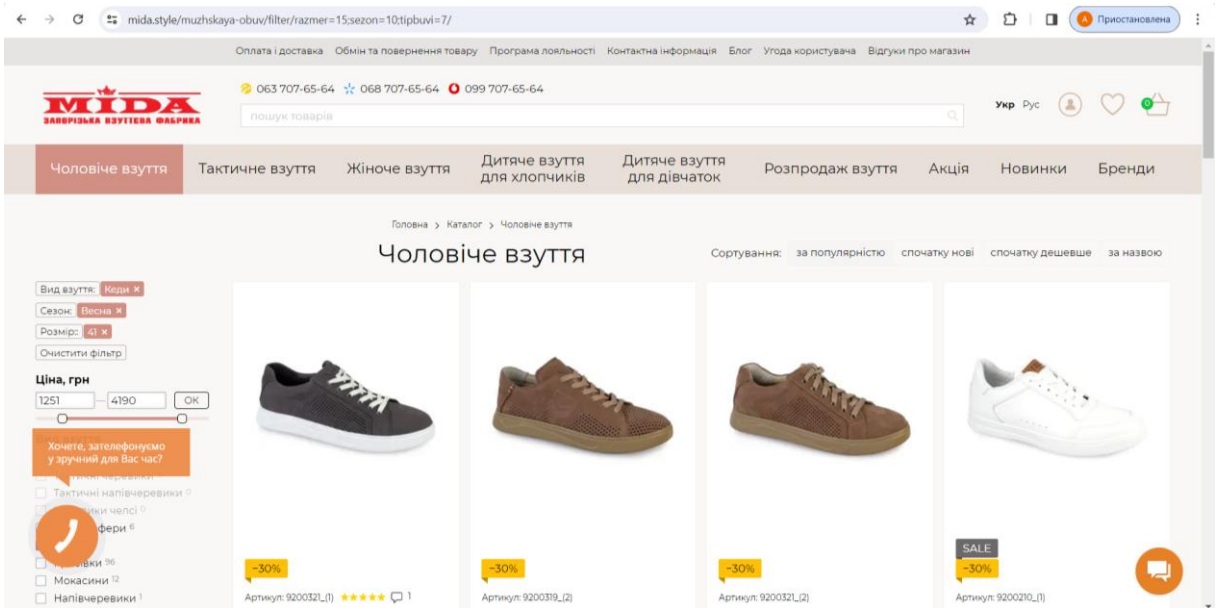

Рисунок 1.11 – Приклад фільтрації товару на сайті Mida.style

Згідно аналізу аналогів було сформовано порівняльну таблицю розглянутих прикладів та майбутнього інтернет-магазину. Результати аналізу представлено у таблиці 1.1.

Таблиця 1.1 – Порівняльна характеристика аналогів

Критерій	Назва ресурсу			
	Intertop.ua	Sezon.ua	Mida.style	Додаток який буде розроблено
Використання ресурсу без реєстрації	+	+	+	+
Зручність навігації	+	-	-	+
Належність до одного бренду	-	-	-	-
Адаптивність сайту	+	+	+	+
Форма зворотного зв’язку	+	+	+	+
Сортування	+	+	+	+
Фільтрація	+	+	+	+

У результаті даного порівняльного аналізу вирішено розробити вебдодаток, у якому буде виправлено недоліки аналогів із таблиці 1.1. Також у розробці будуть збережені переваги вищезазначених сайтів: сортування, фільтрація, зручність навігації та використання ресурсу без реєстрації.

1.2 Постановка задачі

Мета створення інтернет-магазину полягає в створенні зручної платформи для клієнтів з можливістю перегляду доступного асортименту кросівок, ознайомлення з їх характеристиками та зручним замовленням обраних товарів. Основне завдання – забезпечення користувачам швидкого та зручного доступу до інформації про товари та процесу замовлення.

Актуальність створення такого магазину обумовлена зростанням популярності онлайн-покупок та попитом на спортивне взуття, зокрема кросівки.

Для досягнення цієї мети передбачається виконання таких етапів:

1. Аналіз вимог: визначення потреб та вимог клієнтів щодо функціональності та інтерфейсу магазину, а також вивчення конкурентного середовища.

2. Проектування: розробка структури та дизайну магазину, включаючи розташування товарів, функціональність фільтрації та сортування, а також організацію процесу замовлення.

3. Розробка програмного забезпечення: реалізація програмного коду, який відповідає вимогам щодо функціональності магазину та забезпечує безперебійну роботу платформи.

4. Тестування: перевірка роботи сайту на різних пристроях та браузерах, виявлення та усунення помилок.

Ці кроки допоможуть забезпечити успішний розвиток та функціонування інтернет-магазину кросівок.

2 ПРОЄКТУВАННЯ СИСТЕМИ

2.1Технології проєктування веборієнтованих систем

Переважна частина сучасних вебдодатків, зокрема інтернет-магазинів, базується на архітектурі клієнт-сервер. У основі клієнт-серверної архітектури лежать два компоненти: клієнт і сервер (рис. 2.1).

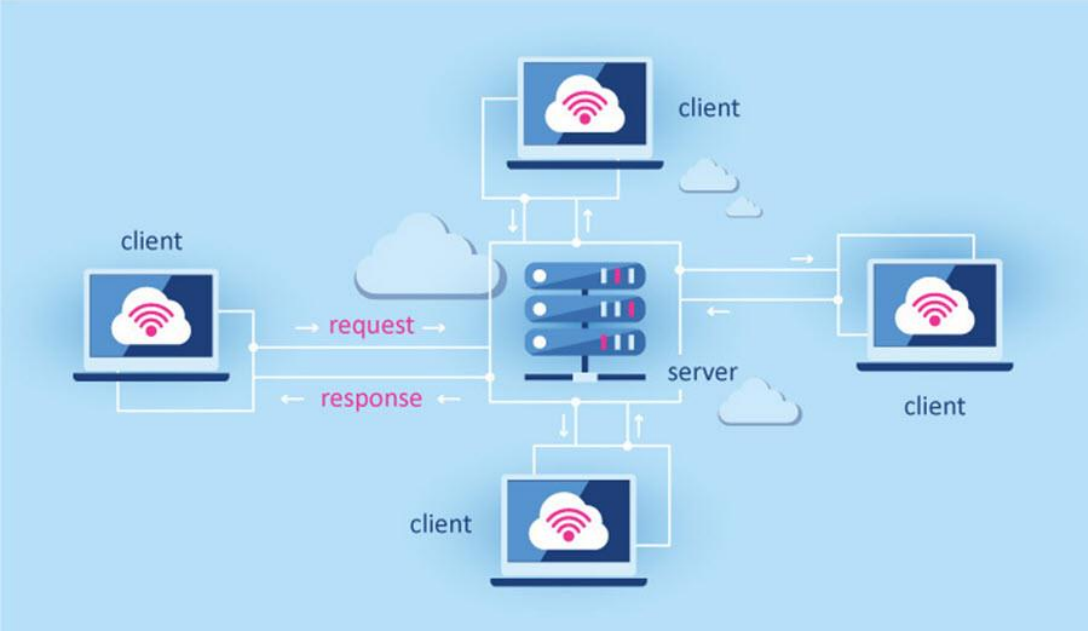

Рисунок 2.1 – Принцип клієнт-серверної архітектури

Клієнт – комп'ютер на стороні користувача, який відправляє запит до сервера для надання інформації або виконання певних дій.

Сервер – більш потужний комп'ютер або обладнання, призначене для вирішення певних завдань з виконання програмних кодів, виконання сервісних функцій за запитом клієнтів, надання користувачам доступу до певних ресурсів, зберігання інформації і баз даних [9].

У даному випадку клієнтом буде відвідувач сайту інтернет-магазину – майбутній покупець, а сервером – обладнання хостингу, на якому буде розміщено сайт.

Модель такої системи полягає в тому, що клієнт відправляє запит на сервер, де він обробляється, і готовий результат відправляється клієнтові. Сервер може

обслуговувати кілька клієнтів одночасно. Якщо одночасно приходить більше одного запиту, то вони встановлюються в чергу і виконуються сервером послідовно. Іноді запити можуть мати пріоритети. Запити з більш високими пріоритетами повинні виконуватися раніше [1]. Важливо, щоб серверна частина мала можливість обслуговувати декілька запитів одночасно, оскільки від цього першочергово залежить взаємодія із покупцями. Тому варто звернути увагу на це.

Аналізуючи вищезазначені вимоги було вирішено використати фреймворк Vue мови програмування JavaScript.

Vue – це фреймворк, який працює на JavaScript, створений для розробки користувацьких інтерфейсів. Він працює на базі звичайного HTML, CSS та JavaScript, з можливостями декларативно програмувати користувацькі інтерфейси будь-якої складності на основі компонентів [10].

Окрім цього буде застосовано наступний стек технологій:

- HTML та CSS – мова розмітки гіпертексту та каскадні таблиці стилів [11];
- PHP – мова програмування, яка спрощує процес створення динамічних вебсторінок [12];
- Vue Router – бібліотека маршрутизації фреймворку Vue [13];
- TailwindCSS – CSS-фреймворк [14];
- Axios – HTTP-клієнт для браузера та node.js на основі Promise [15];
- ESLint – лінтер, для NodeJS застосунків [16].

2.2 Інформаційна модель системи

Карта інтернет-магазину кросівок являє собою динамічні vue-сторінки, які оформленні мовою розмітки гіпертексту HTML та каскадною таблицею стилів CSS. Через vue-файли відбувається під’єднання до бази даних, яка розміщена в хмарному сервісі Mokky Dev. На рисунку 2.2 зображено загальну структуру вебсайту.

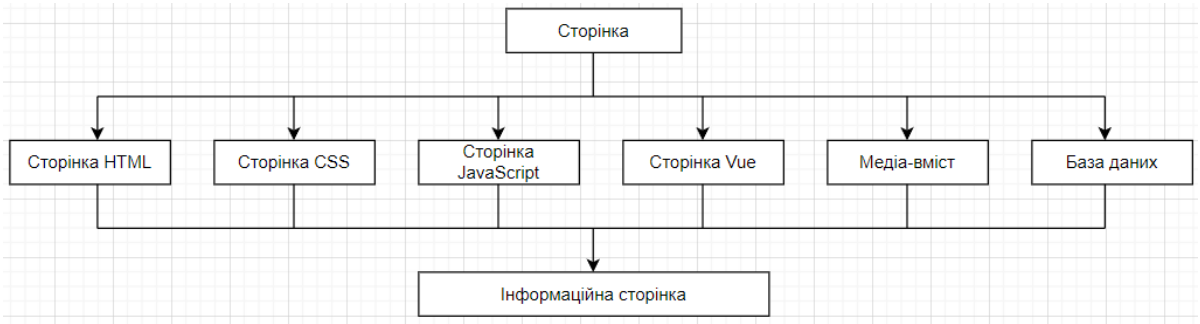

Рисунок 2.2 – Фізична структура сайту

На рис. 2.2 зображено необхідні компоненти для повноцінного функціоналу розробленого програмного продукту. Виключивши будь-який із компонентів система почне працювати некоректно.

2.3 Структура бази даних

Для збереження інформації про товари та її відображення на вебсторінці необхідно використати базу даних. У даному випадку використано хмарний сервіс Mokky.dev [17]. Mokky – це простий сервіс, за допомогою якого є можливість створювати API проєкта та зберігати дані для роботи із сайтом. Принцип, який лежить у механізмі роботи даного програмного рішення, базується на правилах роботи із базою даних MySQL, але суттєво спрощений. Тому було вирішено обрати його для налаштування інтернет-магазину.

Дані про об’єкти зберігаються у форматі JSON-файлу.

База даних буде складатися з 1 таблиці Sneakers та матиме відповідні поля:

- Id – порядковий номер запису у таблиці (Primary Key);
- title – заголовок та короткий опис товару;
- price – ціна за одиницю продукції;
- imageUrl – посилання на зображення.

Алгоритм налаштування складається із декількох кроків. Для роботи із даним сервісом потрібно авторизуватися. У разі виникнення питань стосовно роботи, то потрібно знайти інформацію на сторінці «Документація» (рис. 2.3).

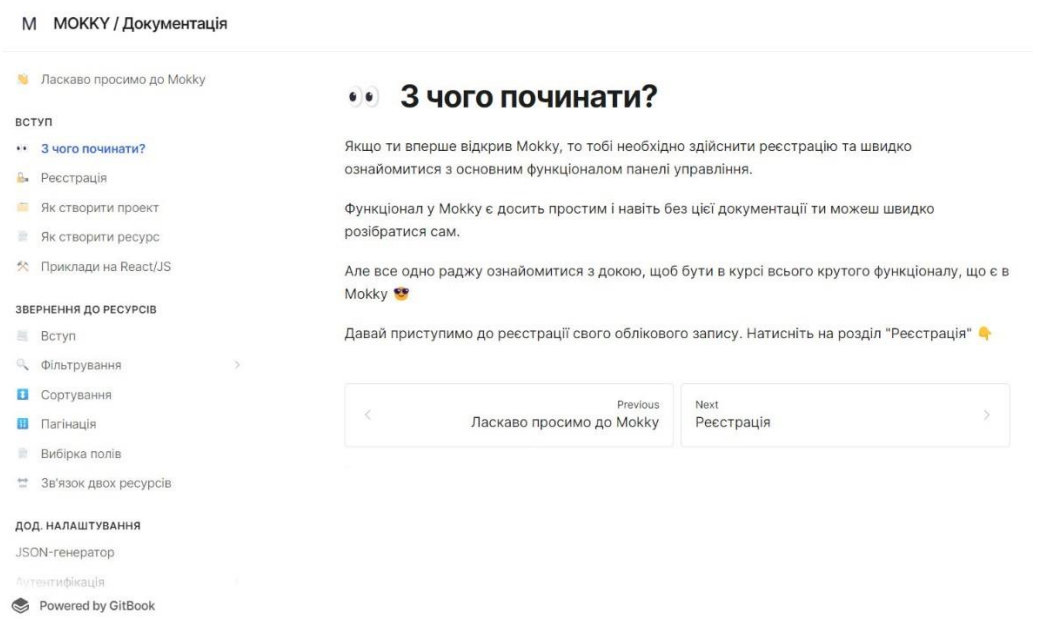

Рисунок 2.3 – Сторінка «Документація» для отримання інформації

Для виконання процесу авторизації потрібно натиснути на кнопку «Спробувати» (рис. 2.4).

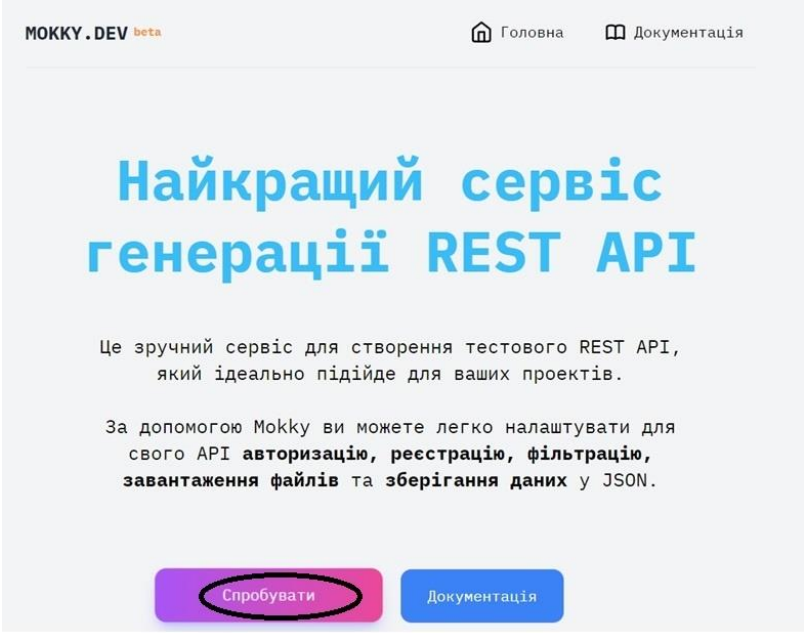

Рисунок 2.4 – Інформація на сторінці перед початком авторизації

При натисканні на виділений елемент керування відбувається перехід на форму для авторизації. Користувач має можливість створити обліковий запис безпосередньо на даному сервісі або використати обліковий запис Google чи GitHub (рис. 2.5).

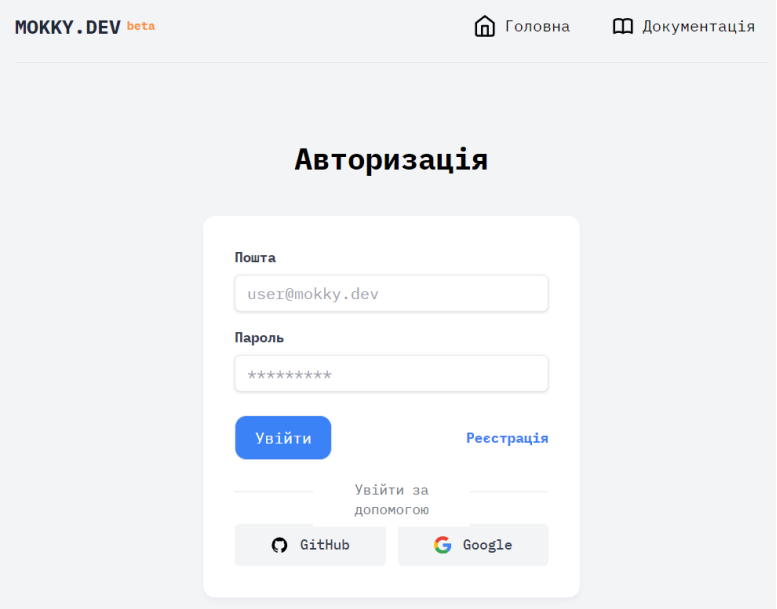

Рисунок 2.5 – Форма для авторизації користувача

Наступним кроком необхідно створити проєкт, у якому і буде розміщено файл із даними з інтернет-магазину. Натискаємо на кнопку «Створити проект» і уволимо необхідну назву (рис. 2.6 – 2.7).

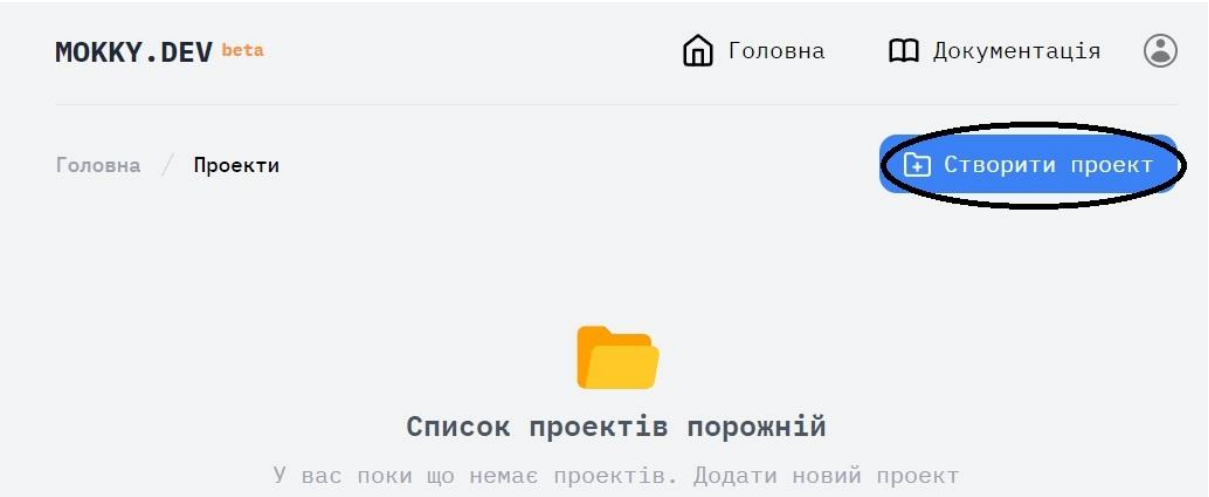

Рисунок 2.6 – Початок створення нового проєкту

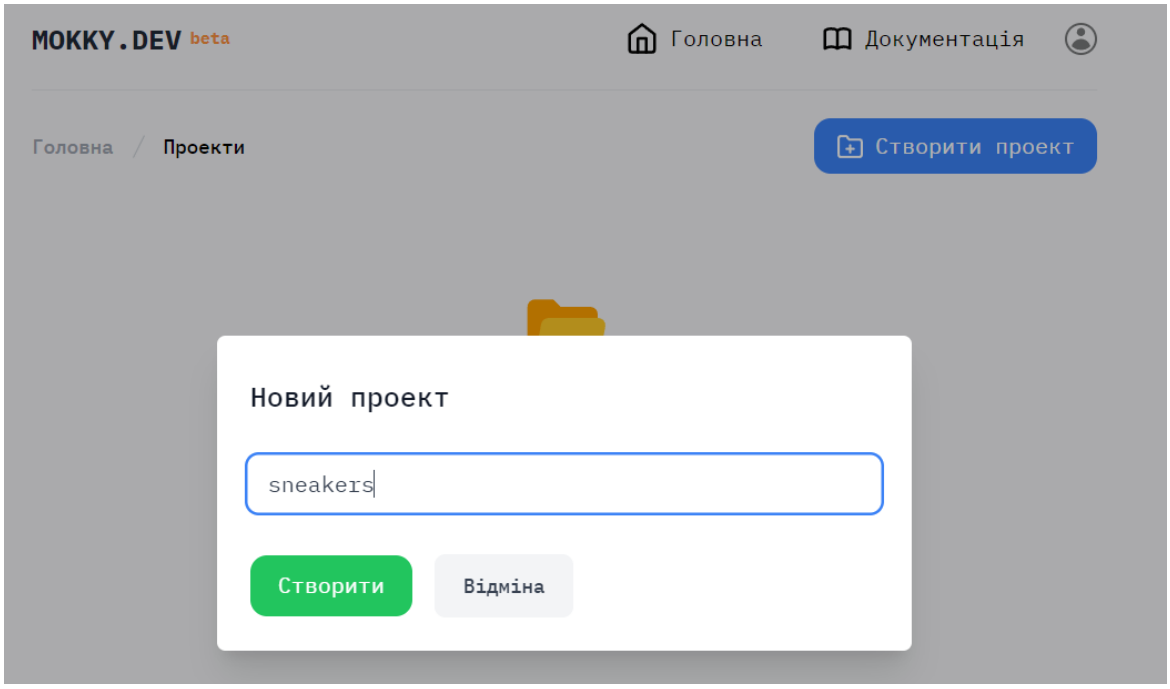

Рисунок 2.7 – Створення нового проєкту

Як уже було вищезазначено, наступним кроком буде створення ресурсу. Ресурс – це текстовий файл, який містить інформацію у вигляді масиву про товар чи будь-яку іншу інформацію, яку потрібно зберігати у структурованому вигляді.

Для створення даного файлу потрібно натиснути на кнопку «Створити ресурс» та увести назву (рис. 2.8).

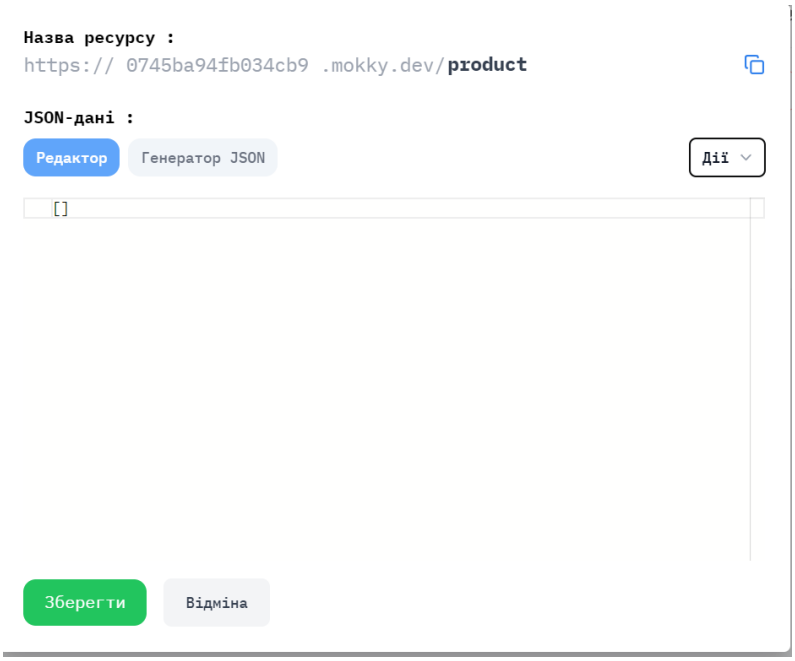

Рисунок 2.8 – Створення ресурсу

Інформація про товари буде міститися у JSON-файлі. Нижче наведено частину списку (рис. 2.9).

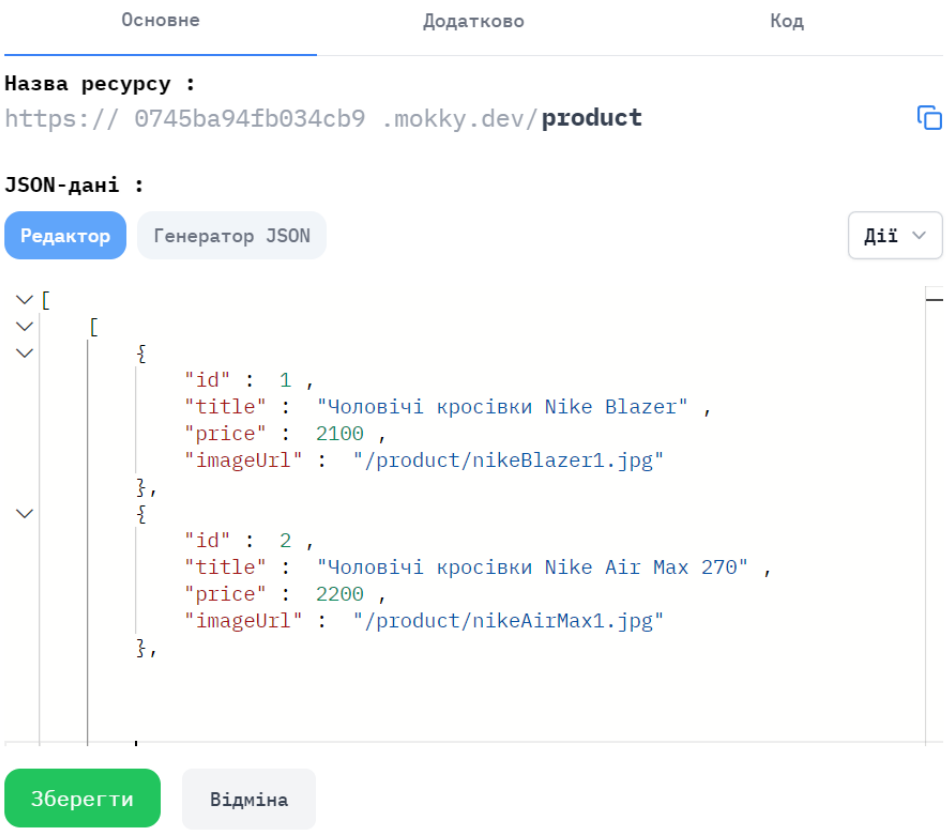

Рисунок 2.9 – Формат запису інформації про товар

Для під’єднання створеного ресурсу із програмним кодом потрібно скопіювати API-посилання усього проєкту на Mokky.dev та вставити його у файл-конфігурації майбутнього інтернет-магазину чоловічого взуття.

3 ПРОГРАМНА РЕАЛІЗАЦІЯ

3.1 Вибір засобів для програмної реалізації

При створенні будь-якого програмного продукту розробник обов’язково використає редактор коду. Редактор коду або текстовий редактор – це програмне забезпечення, яке використовується програмістами для написання та редагування коду програм. Він відіграє важливу роль у процесі розробки, оскільки він обробляє весь програмний код і допомагає програмісту зрозуміти його структуру.

Однією з найважливіших переваг редактора коду є те, що він значно полегшує процес розробки. Він відображає синтаксис у кольорі, автоматично форматує код, а також надає функції для здійснення багатьох задач, що повторюються, автоматично, що допомагає ефективніше використовувати час. Редактор коду може також надавати додаткові функції, такі як вбудований термінал, інтеграція з системами контролю версій, підтримка різних плагінів та інші корисні інструменти [18].

Згідно опитування компанії Stack Overflow у 2022 році найпопулярнішими середовищами розробки є Visual Studio Code, Visual Studio, Notepad++, Sublime Text, IntelliJ, Vim, Eclipse, Atom [19]. На рисунку 3.1 наведено узагальнену діаграму.

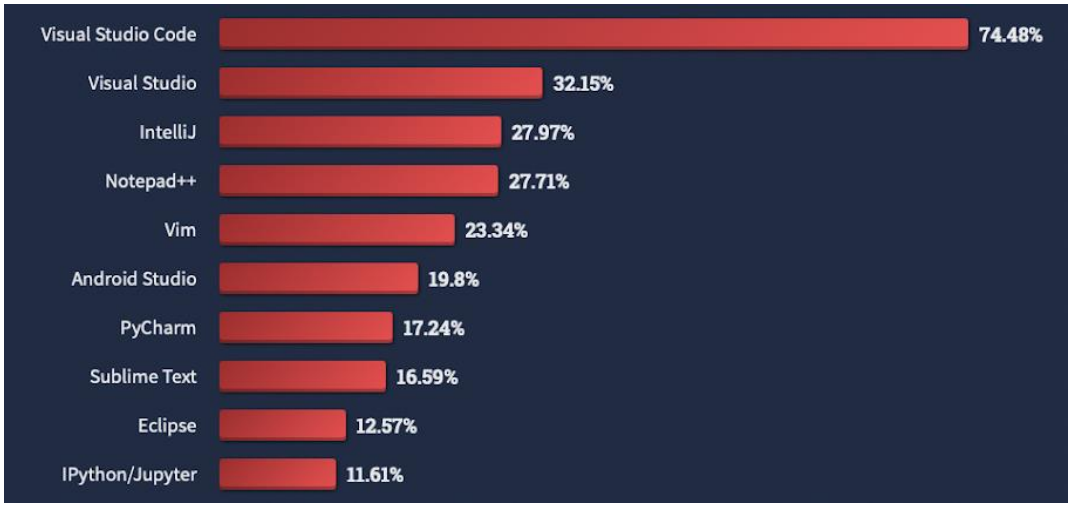

Рисунок 3.1 – Топ-10 найпоширеніших IDE [3]

Нижче проаналізовано 3 редактори: Visual Studio Code, Notepad++ та WebStorm.

Першою проаналізованою програмою став Visual Studio Code. Випущений компанією Microsoft на основі коду Atom, Visual Studio Code має частину функціоналу IDE (Integrated development environment) – інтегрованого середовища розроблення – потужної програми, що містить окрім текстового редактора коду ще ряд механізмів, які дозволяють проводити аналіз коду, запуск його та налагодження. У багатьох рейтингах безкоштовних HTML редакторів саме Visual Studio Code займає перше місце – розробники надають йому перевагу все частіше. Так, наприклад, за даними Stack Overflow, цей редактор у 2017 році використовували 24% веброзробників, а у 2018 році – вже 38,7%. На рис. 3.2 наведено інтерфейс даного середовища.

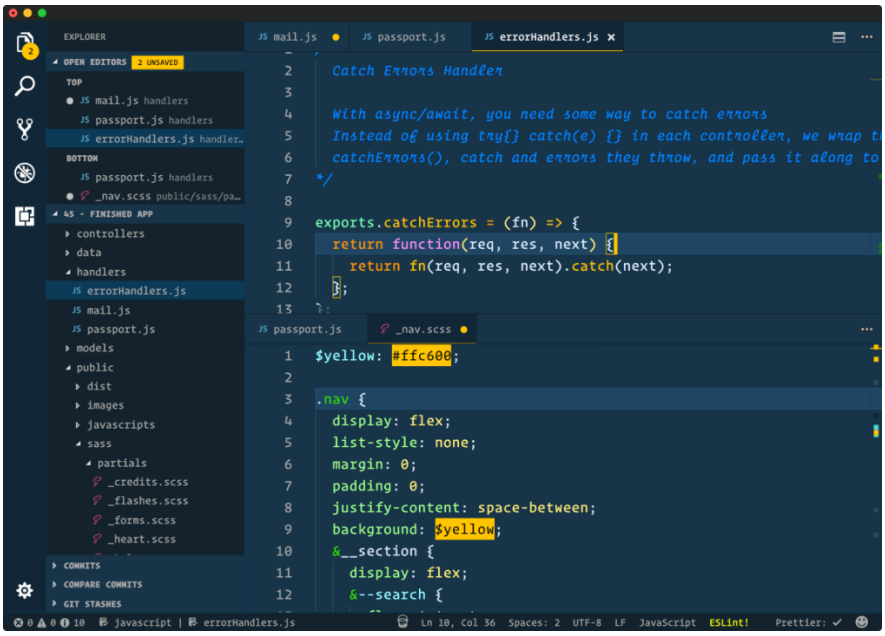

Рисунок 3.2 – Інтерфейс Visual Studio Code

Переваги Visual Studio Code:

- має значну частину функціоналу IDE;
- вбудований потужний механізм автозаповнення IntelliSense;
- значна кількість розширень та доповнень;
- інтегрований з Git "з коробки";

- є вбудований налагоджувач для коду JavaScript, TypeScript, Node.js;
- відкритий вихідний код додатку;
- Visual Studio Code розповсюджується безкоштовно.

Недоліки Visual Studio Code:

- серед недоліків розробники відзначають досить великий час запуску програми;
- пошук за проєктами здійснюється відносно повільно.

Наступним було розглянуто Notepad++. Notepad++ – це текстовий редактор, що займає небагато місця в оперативній пам’яті комп’ютера. Він розроблений для комп’ютерів під керуванням Windows. Користувачі Linux можуть використовувати його через Wine. Notepad++, випущений ще в 2003 році, є перевіреним та усталеним інструментом багатьох розробників, будучи зручним текстовим редактором для HTML коду. Цей редактор поширюється як безкоштовне програмне забезпечення, а його репозиторій доступний у GitHub. Notepad++ підтримує сторонні плагіни [20]. Інтерфейс зображено на рис. 3.3 [21].

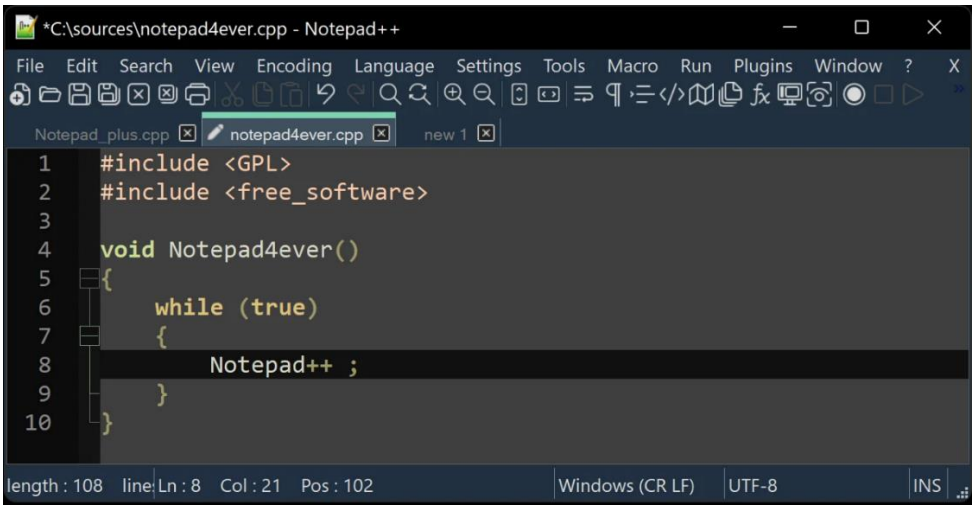

Рисунок 3.3 – Інтерфейс Notepad++

Основні переваги Notepad++:

- Notepad++ є простим, невимогливим до ресурсів інструментом;
- є портативна версія;

- функціонал програми легко розширюється безліччю плагінів. За бажанням такий плагін можна створити самому;
- інтерфейс програми також легко налаштовується;
- підтримується робота з великою кількістю вкладок одночасно;
- Notepad++ є на 100% безкоштовною програмою.

Недоліки Notepad++:

- переважна більшість користувачів цього текстового редактора HTML коду не знаходять у ньому недоліків. Однак можна відзначити деяку мінімалістичність інтерфейсу, яка не підходить ряду користувачів;
- також можна відзначити, що цей редактор не є IDE і не містить у собі його додатковий функціонал. З цієї причини багатьом користувачам доводиться використовувати якесь середовище розробки у якості додаткового інструменту до редактора Notepad++ [21].

Останнім з проаналізованих середовищ є WebStorm. WebStorm – це інтегроване середовище розробки (IDE), призначене для веброзробки. Воно розроблене компанією JetBrains і базується на платформі IntelliJ IDEA. WebStorm надає широкий набір інструментів для зручної роботи з вебтехнологіями, такими як HTML, CSS, JavaScript, і є відмінним вибором для фронтенд-розробників. На рисунку 3.4 наведено інтерфейс середовища WebStorm.

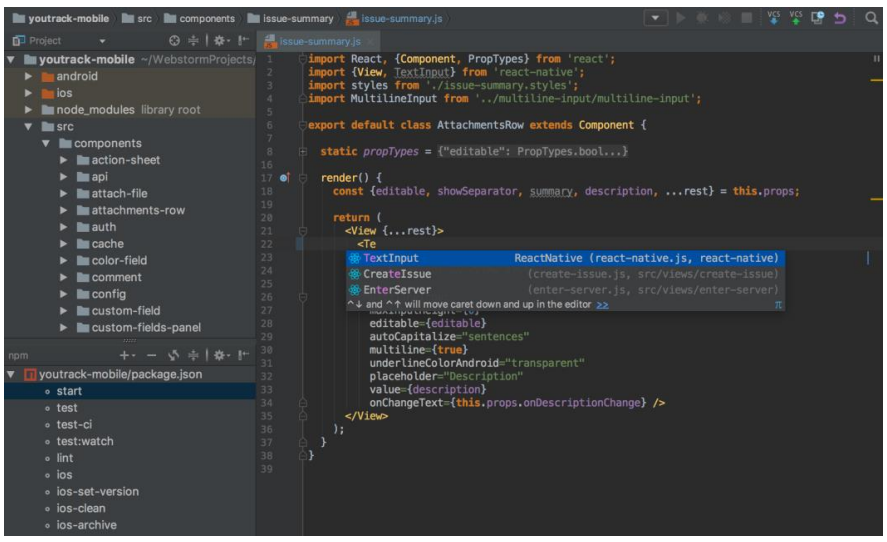

Рисунок 3.4 – Інтерфейс WebStorm

Основні можливості редактора WebStorm включають в себе:

- розумне автодоповнення коду для JavaScript, HTML та CSS, що полегшує написання коду;
- інтегровану підтримку систем контролю версій, таких як Git, що дозволяє зручно керувати версіями вашого проекту;
- вбудовану підтримку фреймворків і бібліотек, таких як Angular, React, Vue.js, що полегшує розробку за їхнім допомогою;
- можливість виконання і відлагодження коду безпосередньо в середовищі розробки;
- інструменти для автоматичного форматування коду та аналізу якості коду для підвищення продуктивності.

Недоліки WebStorm:

- властива всім IDE повільність у роботі та вимогливість до ресурсів;
- відносно складні налаштування;
- платна IDE, що розповсюджується за передплатою (безкоштовна ліцензія за студентською підпискою).

Вищезазначена і проаналізована інформація занесена до порівняльної таблиці 3.1.

Таблиця 3.1 – Порівняльна характеристика проаналізованих IDE

Критерій	Середовище розробки		
	Visual Studio Code	Notepad++	WebStorm
Вбудований відладчик	+	-	+
Мовна локалізація (українська)	+	+	+
Швидкодія	+	+	-
Простота у використанні	+	+	-
Зручність у масштабних проєктах	+	-	+
Безкоштовний доступ за студентською підпискою	+	+	+

Згідно із вищезазначеною інформацією вирішено обрати для розробки редактор коду Visual Studio Code, так як у ньому зібрано усі критерії із перевагами для розробки.

3.2 Розробка системи

Вебсайт розроблено у мінімалістичному стилі: наявні блоки для проведення сортування каталогу, сторінка із закладками та кошик. За потреби власника можливе розміщення рекламних блоків. На рис. 3.5 зображено головну сторінку сайту.

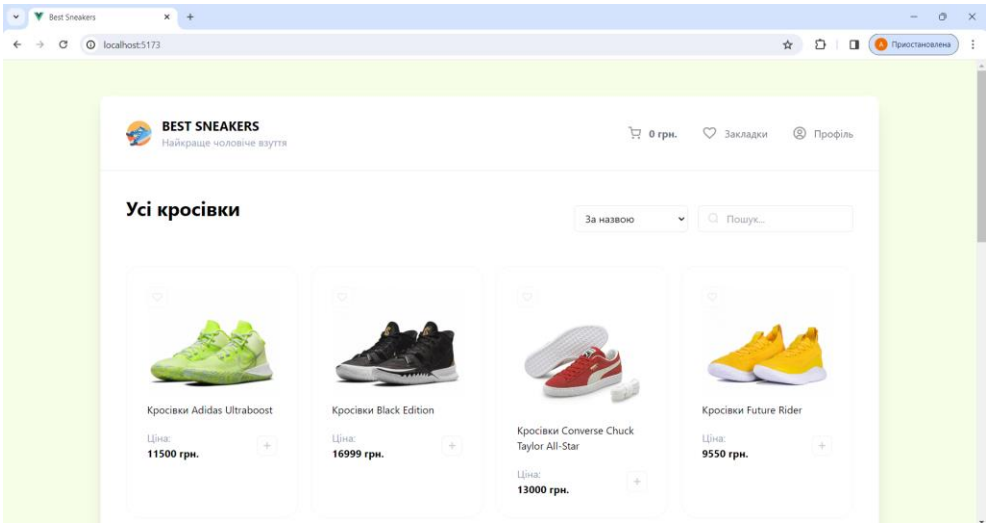

Рисунок 3.5 – Головна сторінка сайту

Розробку сайту можна поділити умовно на декілька частин:

- розробка шапки: згідно прототипу із пункту 2.4 сюди входить блок із назвою, корзина, закладки та профіль;
- розробка блоку для сортування та пошуку товару;
- розробка карток товару.

У ході реалізації вебдодатку використано бібліотеку TailwindCSS. Tailwind CSS – це фреймворк CSS, що є перш за все утилітою, який надає набір низькорівневих класів утиліт CSS, які можна використовувати для створення

будь-якого дизайну без будь-яких обмежень [22]. Перевагами використання даного готового рішення є:

1. швидкість: Tailwind CSS може допомогти вам писати CSS швидше, особливо для складних макетів;

2. послідовність: Tailwind CSS може допомогти вам створити більш послідовний дизайн у вашій команді;

3. гнучкість: Tailwind CSS дуже гнучкий і може використовуватися для створення будь-якого дизайну, від простого до складного;

4. швидкий апгрейд: код CSS Tailwind, як правило, легше підтримувати та оновлювати, ніж традиційний код CSS [23].

Перед початком роботи необхідно встановити утиліту Tailwind CSS для підключення до проєкту [24]. Для цього слід виконати наступні кроки:

1. Перейти на офіційний вебсайт (рис. 3.6) за посиланням: https://tailwindcss.com

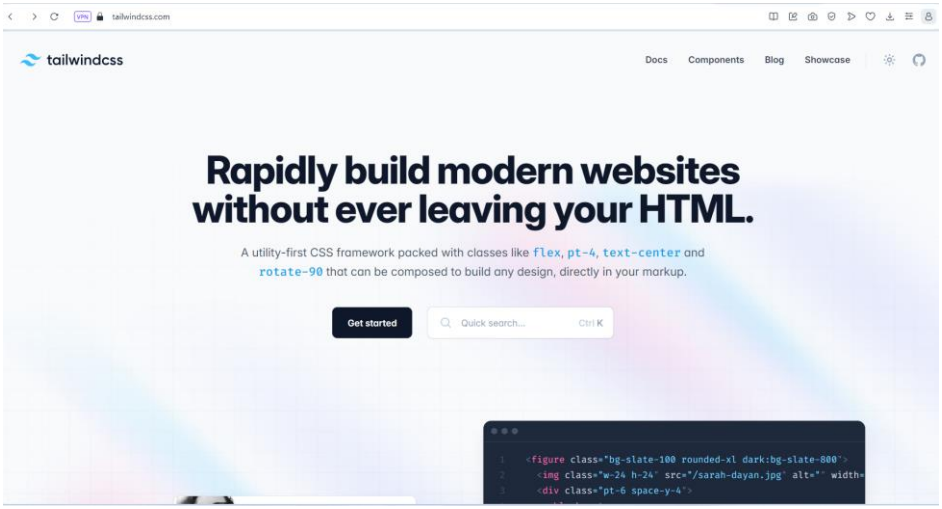

Рисунок 3.6 – Офіційний вебсайт Tailwind CSS для завантаження

2. Наступним кроком потрібно перейти на сторінку для вибору фреймоврку, із яким буде використано його у рамках проєкту. У даному випадку це Vite (рис. 3.7)

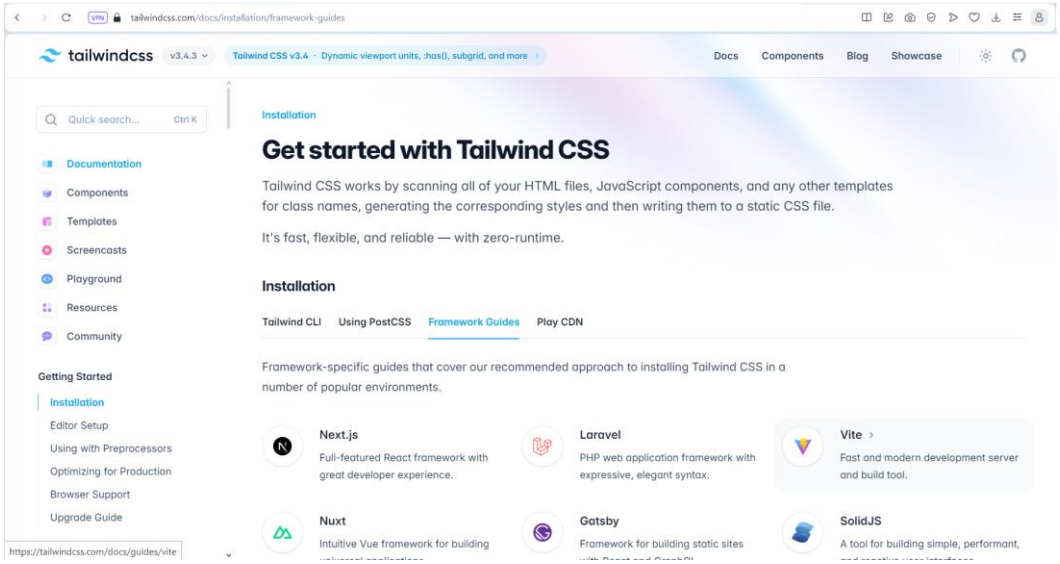

Рисунок 3.7 – Вибір фреймворку, із яким буде поєднано виконання на проєкті

3. Після завантаження у папці із проєктом з’являється 2 файли-конфігуратори: postcss.config.js та tailwind.config.js (рис. 3.8)

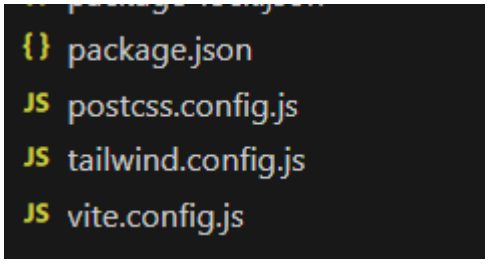

Рисунок 3.8 – Завантажено 2 файли із конфігурацією фреймворка Tailwind CSS

4. Інсталюємо Tailwind CSS увівши команди npm install -D tailwindcss postcss autoprefixer та npx tailwindcss init -p у Terminal редактора Visual Code.

5. У файлі уводимо команди для підключення встановлених стилів (рис. 3.9).

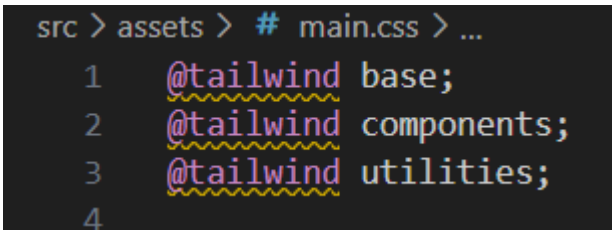

Рисунок 3.9 – Команди у файлі main.css

Наступними етапами є розробка компонентів інтернет-магазину. Код шапки знаходить у файлі Header.vue. На ній розміщається назва та логотип магазину, кошик із сумою покупки, закладки та профіль (рис. 3.10).

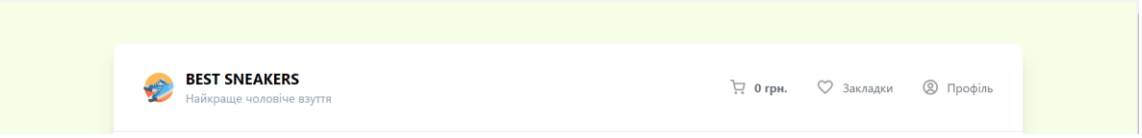

Рисунок 3.10 – Шапка сайту

Нижче наведено програмний код даного файлу:

```
<script setup>
defineProps({
  totalPrice: Number
})

const emit = defineEmits(['openDrawer'])
</script>

<template>
  <header class="flex justify-between border-b border-slate-200 px-10 py-8">
    <router-link to="/"
      ><div class="flex items-center gap-4">
        <img src="/logo.png" alt="Logo" class="w-10" />
        <div>
          <h2 class="text-xl font-bold uppercase">Best Sneakers</h2>
          <p class="text-slate-400">Найкраще чоловіче взуття</p>
        </div>
      </div></router-link
    >

    <ul class="flex items-center gap-10">
      <li
        @click="() => emit('openDrawer')"
        class="flex items-center cursor-pointer gap-3 text-gray-500 hover:text-black"
      >
        <img src="/cart.svg" alt="Cart" />
        <b>{{ totalPrice }} грн.</b>
      </li>

      <router-link to="/favorites">
        <li class="flex items-center cursor-pointer gap-3 text-gray-500 hover:text-black">
          <img src="/heart.svg" alt="Cart" />
```

```
            <span>Закладки</span>
          </li>
        </router-link>

        <li class="flex items-center cursor-pointer gap-3 text-gray-
500 hover:text-black">
          <img src="/profile.svg" alt="Cart" />
          <span>Профіль</span>
        </li>
      </ul>
    </header>
  </template>
```

Для розробки блоку для проведення сортування за ціною та пошуку товару (рис. 3.11) на сайті розроблено файл Home.vue. Нижче наведено програмний код:

```
<script setup>
import { reactive, watch, ref, onMounted } from 'vue'
import axios from 'axios'
import debounce from 'lodash.debounce'
import { inject } from 'vue'
import CardList from '../components/CardList.vue'

const { cart, addToCart, removeFromCart } = inject('cart')

const items = ref([])

const filters = reactive({
  sortBy: 'title',
  searchQuery: ''
})

const onClickAddPlus = (item) => {
  if (!item.isAdded) {
    addToCart(item)
  } else {
    removeFromCart(item)
  }
}

const onChangeSelect = (event) => {
  filters.sortBy = event.target.value
}

const onChangeSearchInput = debounce((event) => {
  filters.searchQuery = event.target.value
}, 300)

const addToFavorite = async (item) => {
  try {
```

```
          if (!item.isFavorite) {
            const obj = {
              item_id: item.id
            }

            item.isFavorite = true

            const { data } = await
axios.post(`https://604781a0efa572c1.mokky.dev/favorites`, obj)

            item.favoriteId = data.id
          } else {
            item.isFavorite = false
            await
axios.delete(`https://604781a0efa572c1.mokky.dev/favorites/${item.favorit
eId}`)
            item.favoriteId = null
          }
        } catch (err) {
          console.log(err)
        }
      }

      const fetchFavorites = async () => {
        try {
          const { data: favorites } = await
axios.get(`https://604781a0efa572c1.mokky.dev/favorites`)

          items.value = items.value.map((item) => {
            const favorite = favorites.find((favorite) => favorite.item_id
=== item.id)

            if (!favorite) {
              return item
            }

            return {
              ...item,
              isFavorite: true,
              favoriteId: favorite.id
            }
          })
        } catch (err) {
          console.log(err)
        }
      }

      const fetchItems = async () => {
        try {
          const params = {
            sortBy: filters.sortBy
```

```
        }

        if (filters.searchQuery) {
          params.title = `*${filters.searchQuery}*`
        }

        const { data } = await
axios.get(`https://604781a0efa572c1.mokky.dev/items`, {
          params
        })

        items.value = data.map((obj) => ({
          ...obj,
          isFavorite: false,
          favoriteId: null,
          isAdded: false
        }))
      } catch (err) {
        console.log(err)
      }
    }

    onMounted(async () => {
      const localCart = localStorage.getItem('cart')
      cart.value = localCart ? JSON.parse(localCart) : []

      await fetchItems()
      await fetchFavorites()

      items.value = items.value.map((item) => ({
        ...item,
        isAdded: cart.value.some((cartItem) => cartItem.id === item.id)
      }))
    })

    watch(cart, () => {
      items.value = items.value.map((item) => ({
        ...item,
        isAdded: false
      }))
    })

    watch(filters, fetchItems)
    </script>

    <template>
      <div class="flex justify-between items-center">
        <h2 class="text-3xl font-bold mb-8">Усі кросівки</h2>

        <div class="flex gap-4">
```

```
            <select  @change="onChangeSelect"  class="py-2  px-3  border
rounded-md outline-none">
              <option value="name">За назвою</option>
              <option value="price">По ціні (дешевші)</option>
              <option value="-price">По ціні (дорожчі)</option>
            </select>

            <div class="relative">
              <img class="absolute left-4 top-3" src="/search.svg" />
              <input
                @input="onChangeSearchInput"
                class="border  rounded-md  py-2  pl-11  pr-4  outline-none
focus:border-gray-400"
                type="text"
                placeholder="Пошук..."
              />
            </div>
          </div>
        </div>

        <div class="mt-10">
          <CardList :items="items" @add-to-favorite="addToFavorite" @add-
to-cart="onClickAddPlus" />
        </div>
      </template>
```

Центральним об’єктом на будь-якому інтернет-магазині є асортимент продажу. Назва та ціна товару записана у json-файл на сервісі Mokky.dev. Його перевагою є легкість в користуванні навіть без спеціальних навичок роботи порівняно із роботою СУБД. Програмний код карти товару знаходиться у файлах Card.vue та CardList.vue.

Вміст файлу Card.vue:

```
<script setup>
defineProps({
  id: Number,
  title: String,
  imageUrl: String,
  price: Number,
  isFavorite: Boolean,
  isAdded: Boolean,
  onClickFavorite: Function,
  onClickAdd: Function
})
</script>

<template>
  <div
```

```
        class="relative bg-white border border-slate-100 rounded-3xl p-8
cursor-pointer transition hover:-translate-y-2 hover:shadow-xl"
      >
        <img
          v-if="onClickFavorite"
          :src="!isFavorite ? '/like-1.svg' : '/like-2.svg'"
          alt="Like 1"
          class="absolute top-8 left-8"
          @click="onClickFavorite"
        />

        <img :src="imageUrl" alt="Sneaker" />

        <p class="mt-2">{{ title }}</p>

        <div class="flex justify-between mt-5">
          <div class="flex flex-col">
            <span class="text-slate-400">Ціна:</span>
            <b>{{ price }} грн.</b>
          </div>

          <img
            v-if="onClickAdd"
            @click="onClickAdd"
            :src="!isAdded ? '/plus.svg' : '/checked.svg'"
            alt="Plus"
          />
        </div>
      </div>
    </template>
```

Вміст файлу CardList.vue:

```
    <script setup>
    import Card from './Card.vue'

    defineProps({
      items: Array,
      isFavorites: Boolean
    })

    const emit = defineEmits(['addToFavorite', 'addToCart'])
    </script>

    <template>
      <div class="grid grid-cols-4 gap-5" v-auto-animate>
        <Card
          v-for="item in items"
          :key="item.id"
          :id="item.id"
          :title="item.title"
          :imageUrl="item.imageUrl"
          :price="item.price"
```

```
            :onClickFavorite="isFavorites    ?    null    :    ()    =>
emit('addToFavorite', item)"
            :onClickAdd="isFavorites  ?  null  :  ()  =>  emit('addToCart',
item)"
            :isFavorite="item.isFavorite"
            :isAdded="item.isAdded"
          />
        </div>
      </template>
```

Результат створення області із товаром зображено на рисунку 3.11.

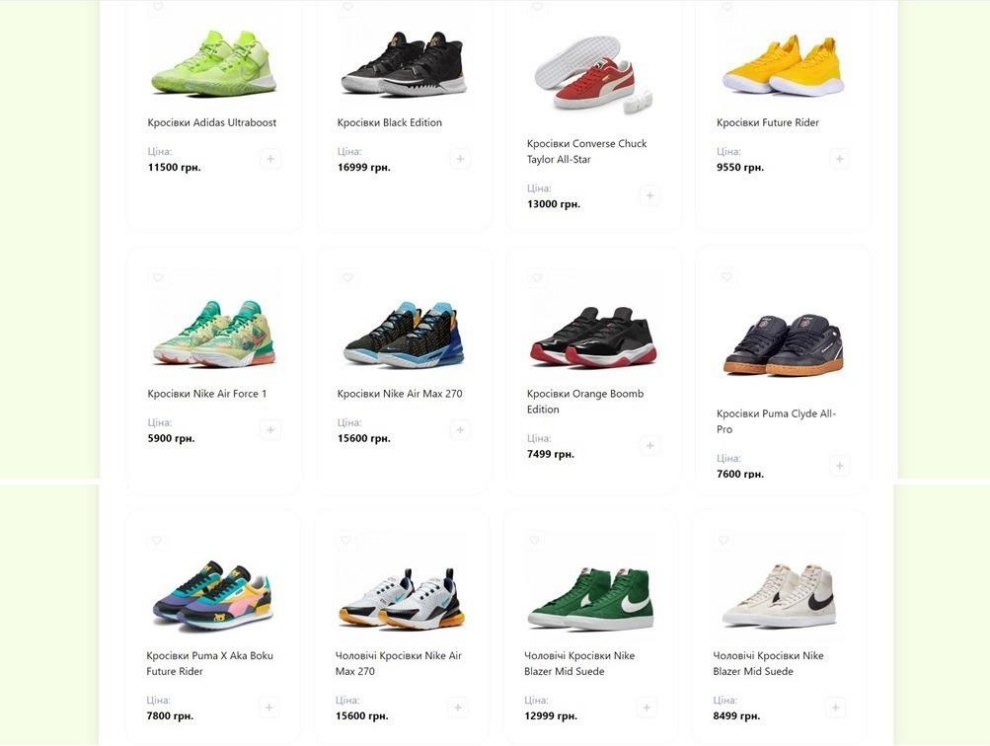

Рисунок 3.11 – Відображення товару на сайті

Після вибору бажаної продукції покупцем, вона переміщається до кошика. Для реалізації розроблено 2 окремі файли. Стилізація кошику прописана у DrawerHead.vue, робота скрипту – у файлі Drawer.vue. Процес додавання товару у кошик відбувається після натиснання кнопки «+» у правому нижньому куті позиції товару (рис. 3.12).

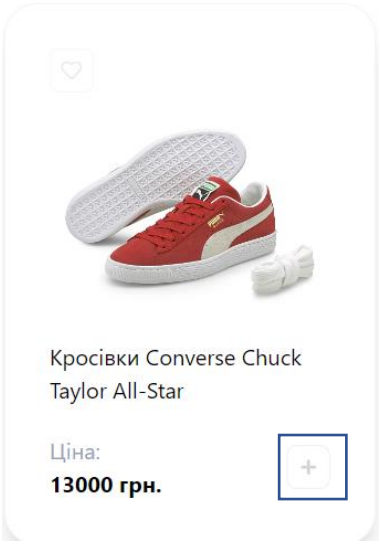

Рисунок 3.12 – Кнопка для додавання товару у кошик

У результаті колір кнопки стає зеленим, а сума виводиться у шапці сайту поряд із відповідною піктограмою (рис. 3.13).

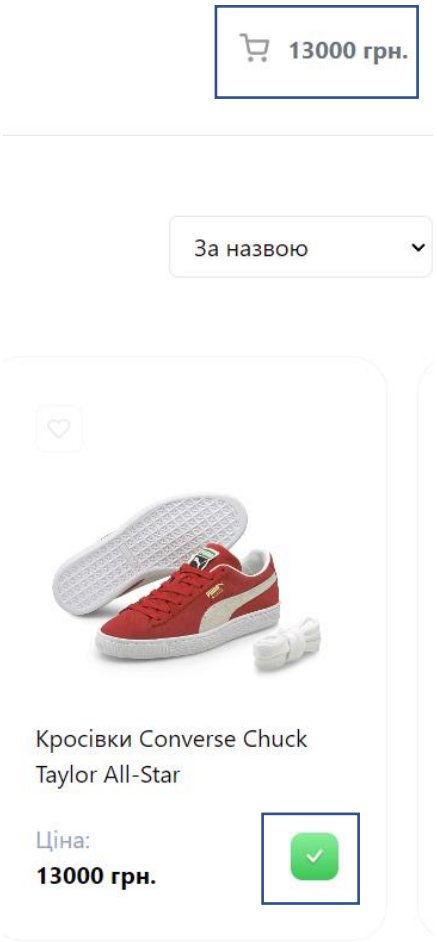

Рисунок 3.13 – Товар додано у кошик, інформація оновлена

Для переходу у кошик потрібно натиснути на вищазазначену кнопку на шапці. У відкритій вкладці з’являється інформація про додану продукцію, її вартість та податок 5% (рис. 3.14).

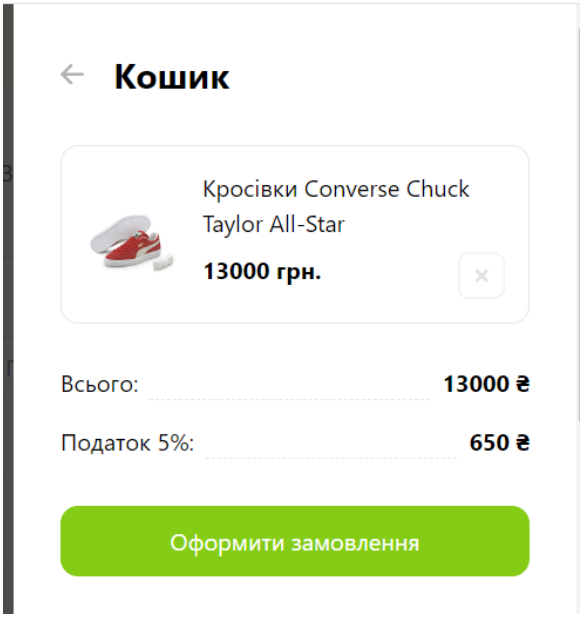

Рисунок 3.14 – Вигляд вкладки кошик із доданим товаром

Замовлення виконується натисканням на кнопку «Оформити замовлення». Зміна кольору її заливки на сірий свідчить про успішне виконання даної операції (рис. 3.15).

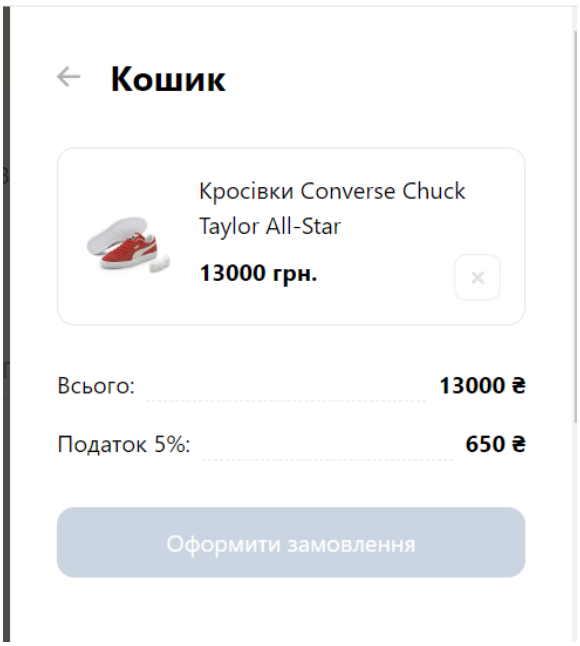

Рисунок 3.15 – Успішне виконання операції замовлення

Програмний код вищезазначених операцій міститься у файлі Drawer.vue:

```
<script setup>
import axios from 'axios'
import { ref, computed, inject } from 'vue'

import DrawerHead from './DrawerHead.vue'
import CartItemList from './CartItemList.vue'
import InfoBlock from './InfoBlock.vue'

const props = defineProps({
  totalPrice: Number,
  vatPrice: Number
})

const { cart, closeDrawer } = inject('cart')

const isCreating = ref(false)
const orderId = ref(null)

const createOrder = async () => {
  try {
    isCreating.value = true

    const { data } = await axios.post(`https://0745ba94fb034cb9.mokky.dev/product`, {
      items: cart.value,
      totalPrice: props.totalPrice.value
    })

    cart.value = []

    orderId.value = data.id
  } catch (err) {
    console.log(err)
  } finally {
    isCreating.value = false
  }
}

const cartIsEmpty = computed(() => cart.value.length === 0)
const buttonDisabled = computed(() => isCreating.value || cartIsEmpty.value)
</script>

<template>
  <div class="fixed top-0 left-0 h-full w-full bg-black z-10 opacity-70"></div>
  <div class="bg-white w-96 h-full fixed right-0 top-0 z-20 p-8">
    <DrawerHead />
```

```
          <div v-if="!totalPrice || orderId" class="flex h-full items-
center">
            <InfoBlock
              v-if="!totalPrice && !orderId"
              title="Кошик порожній"
              description="Оберіть хоча б 1 пару взуття."
              image-url="/package-icon.png"
            />
            <InfoBlock
              v-if="orderId"
              title="Замовлення сформовано!"
              :description="`Ваше замовлення #${orderId} незабаром
передамо кур'єру`"
              image-url="/order-success-icon.png"
            />
          </div>

          <div v-else>
            <CartItemList />

            <div class="flex flex-col gap-4 mt-7">
              <div class="flex gap-2">
                <span>Всього:</span>
                <div class="flex-1 border-b border-dashed"></div>
                <b>{{ totalPrice }} ₴</b>
              </div>

              <div class="flex gap-2">
                <span>Податок 5%:</span>
                <div class="flex-1 border-b border-dashed"></div>
                <b>{{ vatPrice }} ₴</b>
              </div>

              <button
                :disabled="buttonDisabled"
                @click="createOrder"
                class="mt-4 transition bg-lime-500 w-full rounded-xl py-3
text-white disabled:bg-slate-300 hover:bg-lime-600 active:bg-lime-700
cursor-pointer"
              >
                Оформити замовлення
              </button>
            </div>
          </div>
        </div>
      </template>
```

У випадку, коли кошик порожній, то виводиться сума 0 грн біля відповідної кнопки. Також при переході безпосередньо на вкладку користувач

бачить інформацію про необхідність замовлення хоча б 1 пари взуття (мінімальний обсяг замовлення) (рис. 3.16).

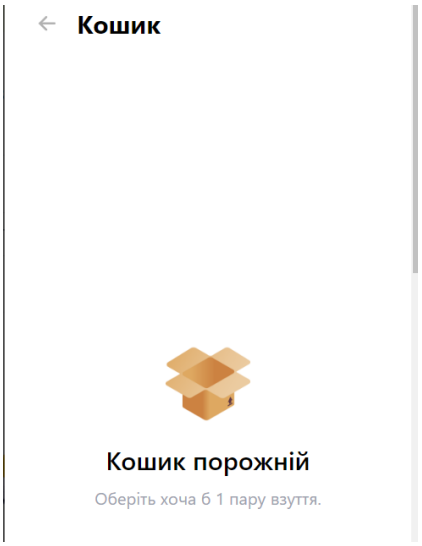

Рисунок 3.16 – Виведення інформації про порожній кошик

Після створення фронтенд-частини інтернет-магазину необхідно протестувати систему на коректну роботу.

3.3 Тестування вебзастосунку

Після повноцінної розробки застосунку потрібно виконати тестування. Так як функціонал сайту відносно невеликий, то можна виконати ручне (manual) тестування.

Алгоритм підготовки до тестування наступний:

1. у терміналі Visual Code необхідно увести 2 команди: ‘npm install’, ‘npm run dev’;

2. після уведення команд система автоматично згенерує локальну адресу. Потрібно натиснути на неї або скопіювати та вставити у пошуковий рядок браузера (рис. 3.17);

Рисунок 3.17 – Локальна адреса розробленого вебсайту

3. результатом переходу за вищезазначеним посиланням буде відкритий інтернет-магазин чоловічого взуття (рис. 3.18);

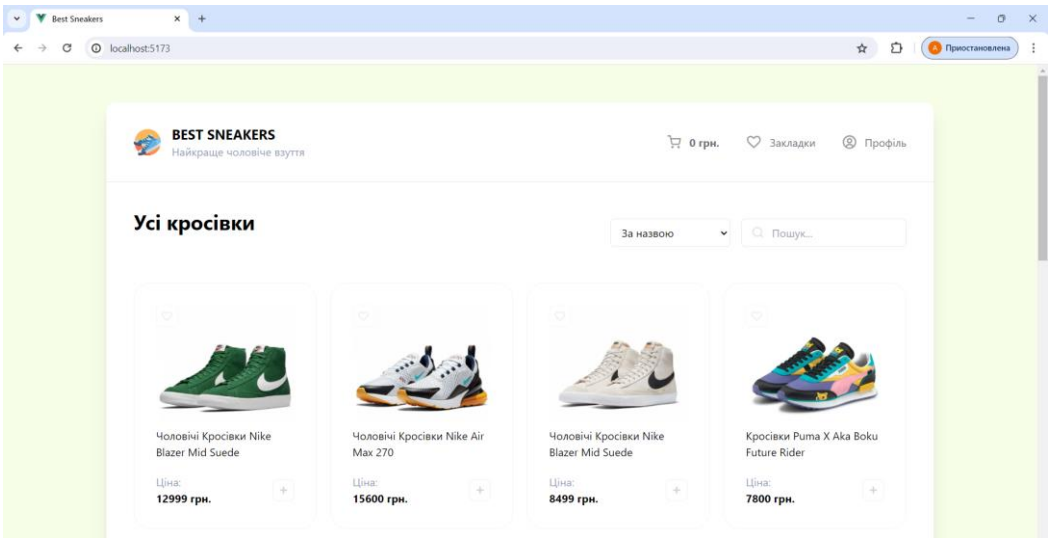

Рисунок 3.18 – Відкритий інтернет-магазин чоловічого взуття

Увесь процес тестування поділяється на 6 етапів: тестування Кошику, сортування за назвою, сортування за ціною (дешевщі), сортування за ціною (дорожчі), пошук товару, робота вкладки «Закладки» та адаптивність.

Етап 1. Тестування вкладки «Кошик»:

1. Потрібно натиснути на карточці із бажаним товаром на позначку «+». Після натискання змінюється позначка на «✓» (рис. 3.19).

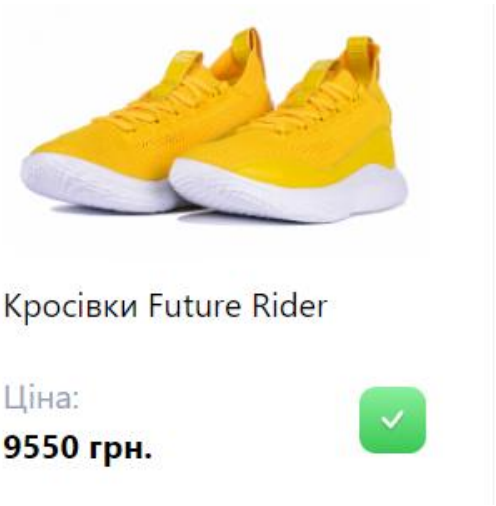

Рисунок 3.19 – Відбулася зміна позначки після натискання

2. Переходимо на вкладку «Кошик».

3. У результаті виконання даної послідовності товар з'явився у вікні «Кошик» (рис. 3.20) та загальна сума відображення на вкладці швидкого доступу (рис. 3.21).

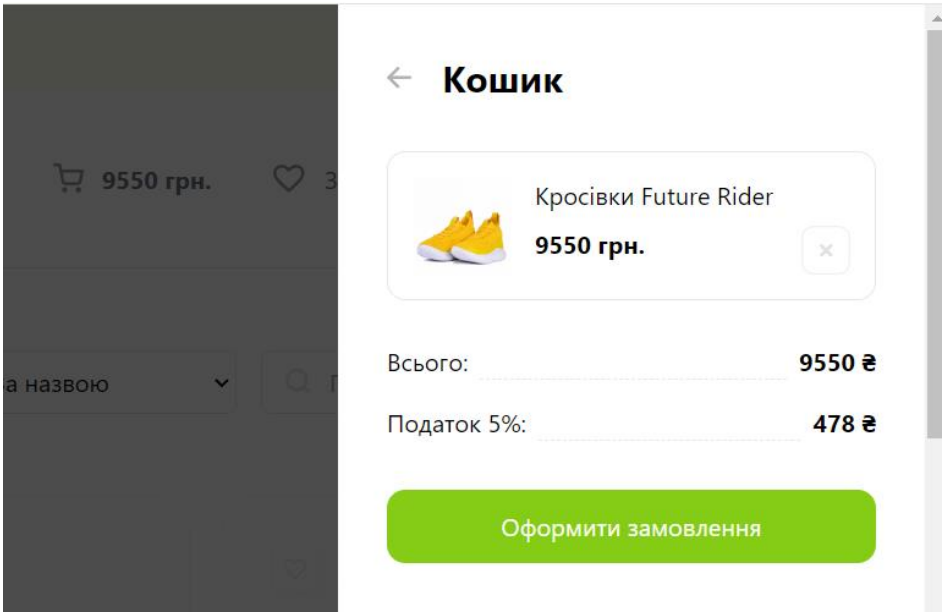

Рисунок 3.20 – Товар з'явився у вікні «Кошик»

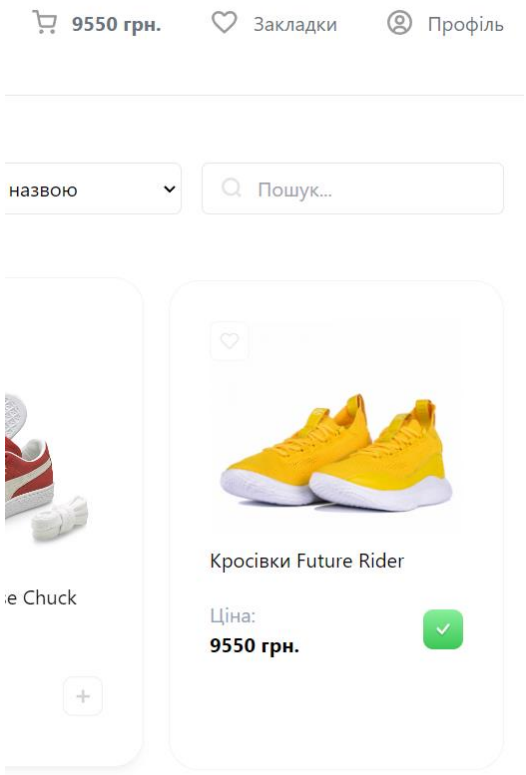

Рисунок 3.21 – Загальна сума покупок відображається на вкладці

4. Для підтвердження придбання товару потрібно натиснути на кнопку «Оформити замовлення» (рис. 3.22).

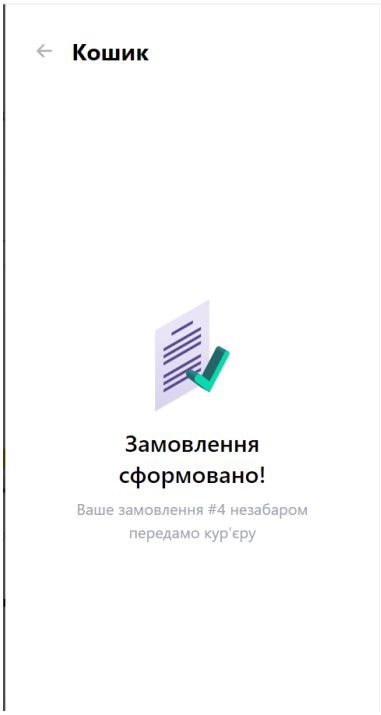

Рисунок 3.22 – Підтвердження замовлення товару

У результаті тестування функцій вкладки «Кошик» було перевірено коректність проведення замовлення покупцем. Помилок на даному етапі не виявлено.

Етап 2. Тестування сортування за назвою:

1. Для проведення даного упорядкування товару потрібно вибрати у розкривному списку критерій «За назвою» (рис. 3.23). За замовчуванням обрано саме даний метод сортування товару у алфавітному порядку.

2. Для перевірки коректної роботи потрібно упорядкувати будь-яким іншим тип сортування (рис. 3.24) та знову обрати сортування за назвою (рис. 3.25).

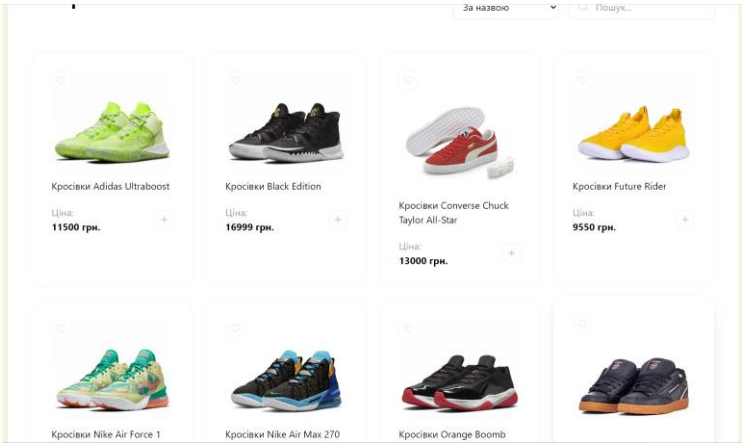

Рисунок 3.23 – Сортування товару за назвою

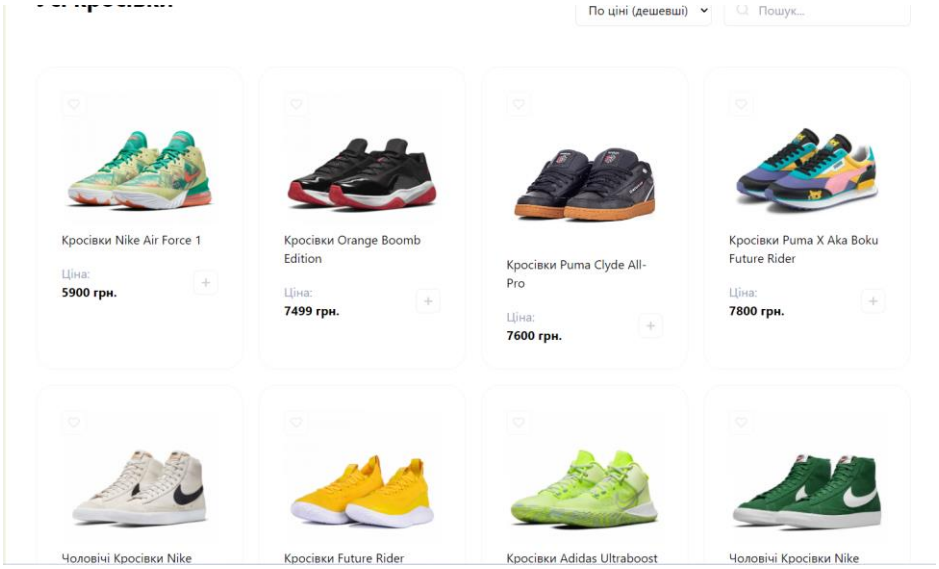

Рисунок 3.24 – Сортування за ціною (дешевші)

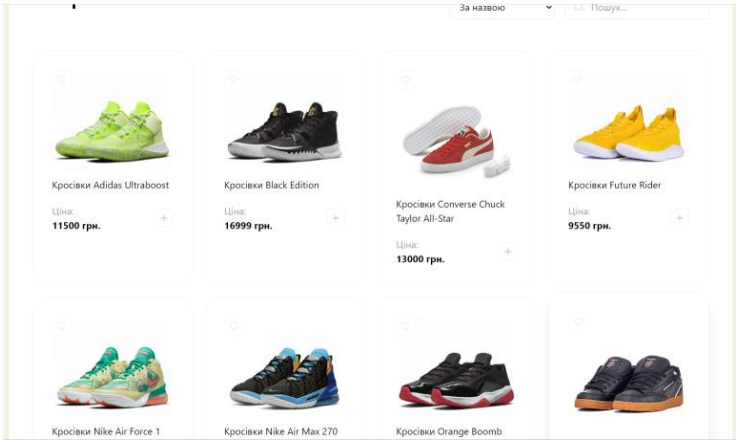

Рисунок 3.25 – Сортування за назвою

У результаті тестування сортування за назвою було перевірено упорядкування товару у алфавітному порядку. Помилок на даному етапі не виявлено.

Етап 3. Тестування сортування за ціною (дешевші):

1. На початковому етапі асортимент відсортовано в алфавітному порядку (рис. 3.25). Для проведення упорядкування товару за ціною потрібно вибрати у розкривному списку критерій «За ціною (дешевші)» (рис. 3.26).

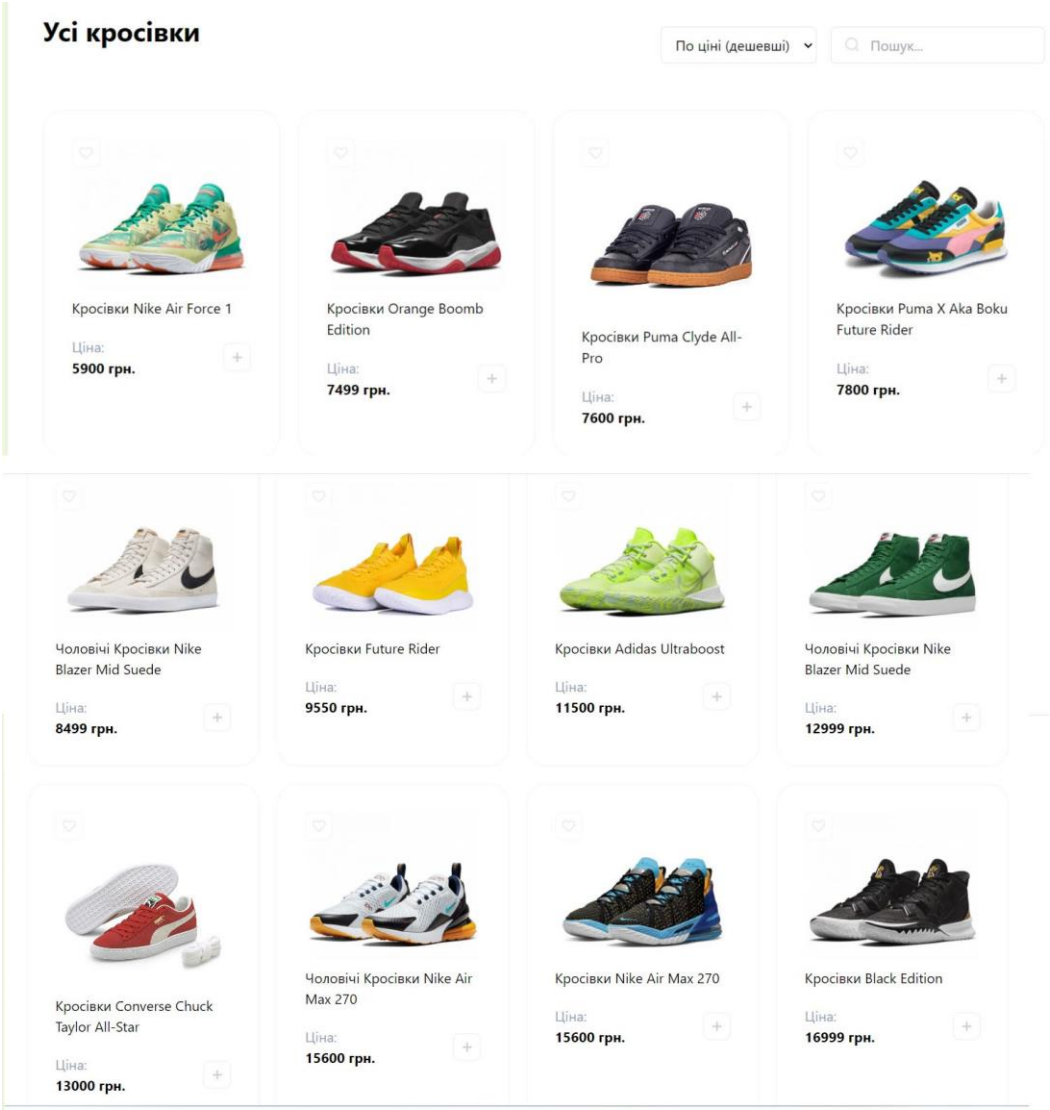

Рисунок 3.26 – Сортування за ціною (дешевші)

У результаті тестування сортування за ціною було перевірено упорядкування товару від дешевних до дорожчих найменувань Помилок на даному етапі не виявлено.

Етап 4. Тестування сортування за ціною (дорожчі):

1. На початковому етапі асортимент відсортовано в алфавітному порядку (рис. 3.25). Для проведення упорядкування товару за ціною потрібно вибрати у розкривному списку критерій «За ціною (дорожчі)» (рис. 3.27).

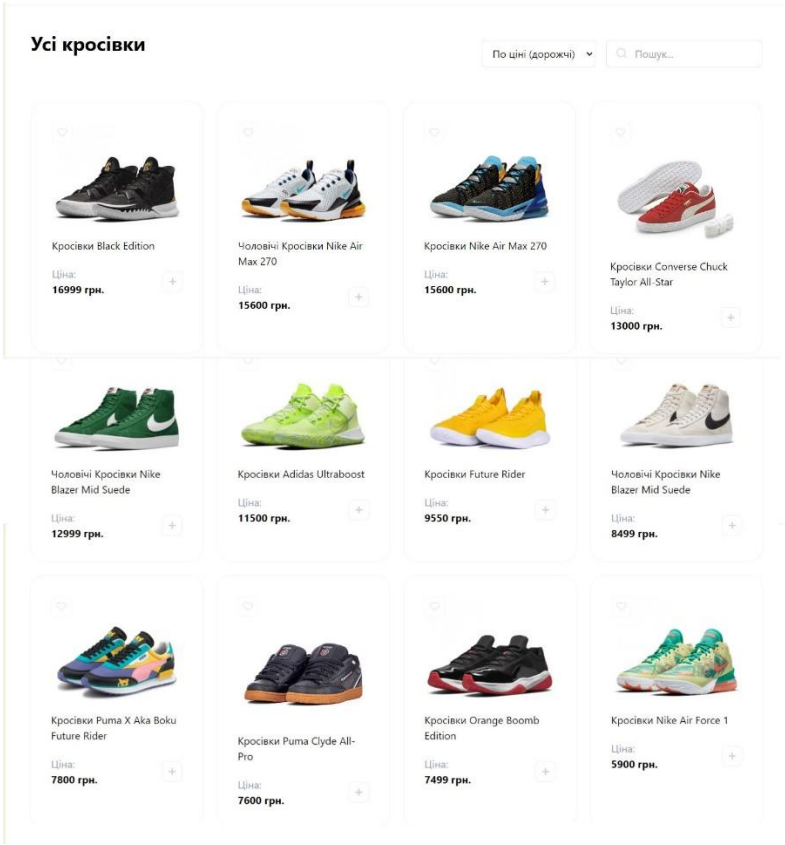

Рисунок 3.27 – Сортування на ціною (дорожчі)

У результаті тестування сортування за ціною було перевірено упорядкування товару від дорожчих до дешевших найменувань. Помилок на даному етапі не виявлено.

Етап 5. Пошук товару:

1. Пошук бажаного найменування взуття здійснюється через відповідне поле «Пошук». Потрібно увести запит у нього і система знайде відповідний

товар. На рис. 3.28 відображено процес пошуку товару за умови, що він є в асортименті.

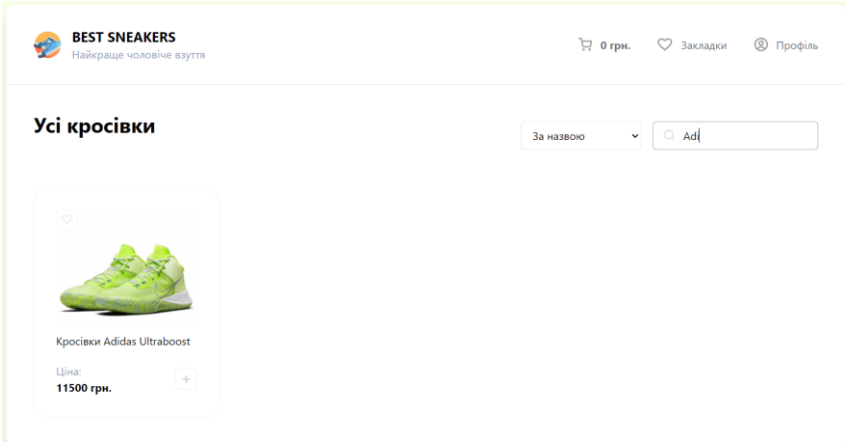

Рисунок 3.28 – Пошук товару, який є в наявності

2. На рис. 3.29 відображено пошук товару за умови, що він відсутній в асортименті. Так як відсутнє взуття марки reebok, то нічого не відображається.

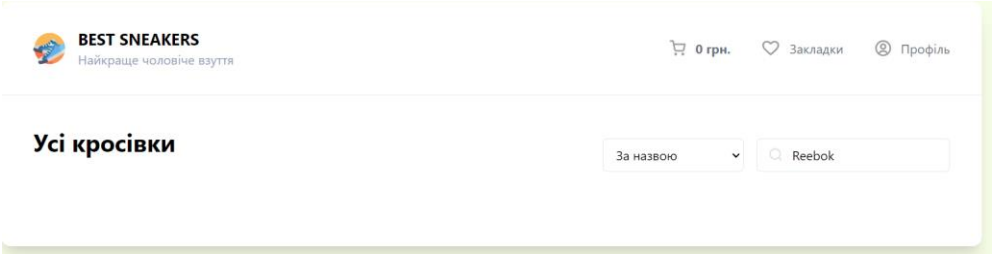

Рисунок 3.29 – Пошук товару, який відсутній в наявності

У результаті даного етапу тестування було перевірено пошук товару. Помилок на даному етапі не виявлено.

Етап 6. Тестування вкладки «Закладки»:

1. Потрібно натиснути на карточці із бажаним товаром на позначку у лівому верхньому куті. Після натискання змінюється її колір заливки на червоний (рис. 3.30).

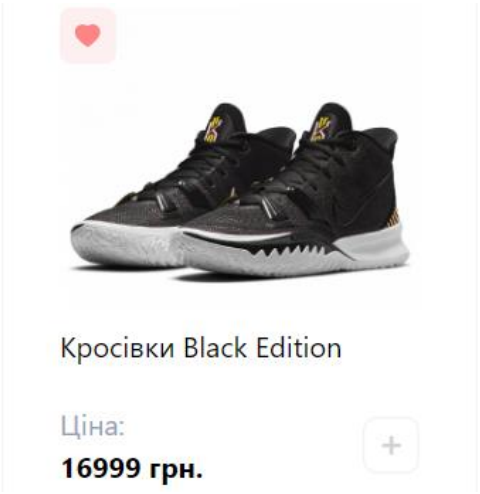

Рисунок 3.30 – Товар обрано для поміщення у закладки

2. Необхідно перейти безпосередньо на вкладку «Закладки» для перевірки, що позиція товару дійсно додана (рис. 3.31).

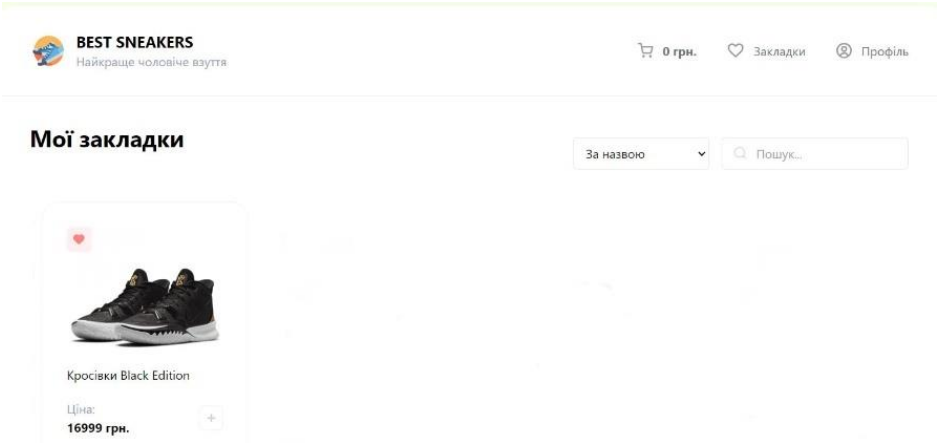

Рисунок 3.31 – Позиція товара додана в улюблені

Використання даної функції необхідне у тому випадку, коли покупець хоче проаналізувати взуття, яке сподобалось, та порівняти ціни/зображення на нього.

У результаті тестування функцій вкладки «Закладки» було перевірено коректність додавання. Помилок на даному етапі не виявлено.

Етап 7. Тестування адаптивності вебсайту:

Перевірка адаптивності буде проводитися на 3 точках переходу: 1200px, 992px та 768px.

1. У браузері Google Chrome встановлюється роздільна здатність 1200px. Результат відображено на рисунку 3.32.

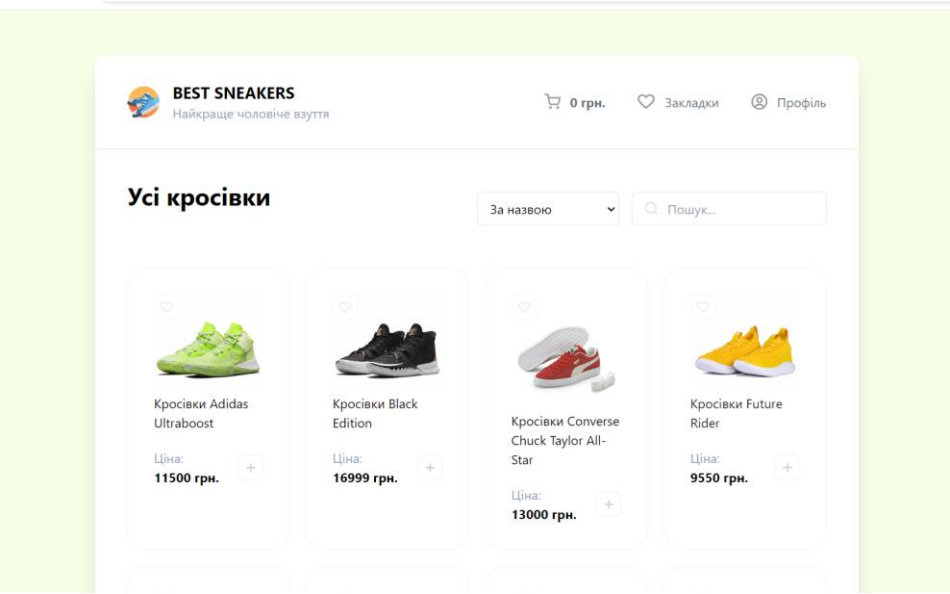

Рисунок 3.32 – Перевірка з роздільною здатністю 1200px

2. У браузері Google Chrome встановлюється роздільна здатність 992px. Результат відображено на рисунку 3.33.

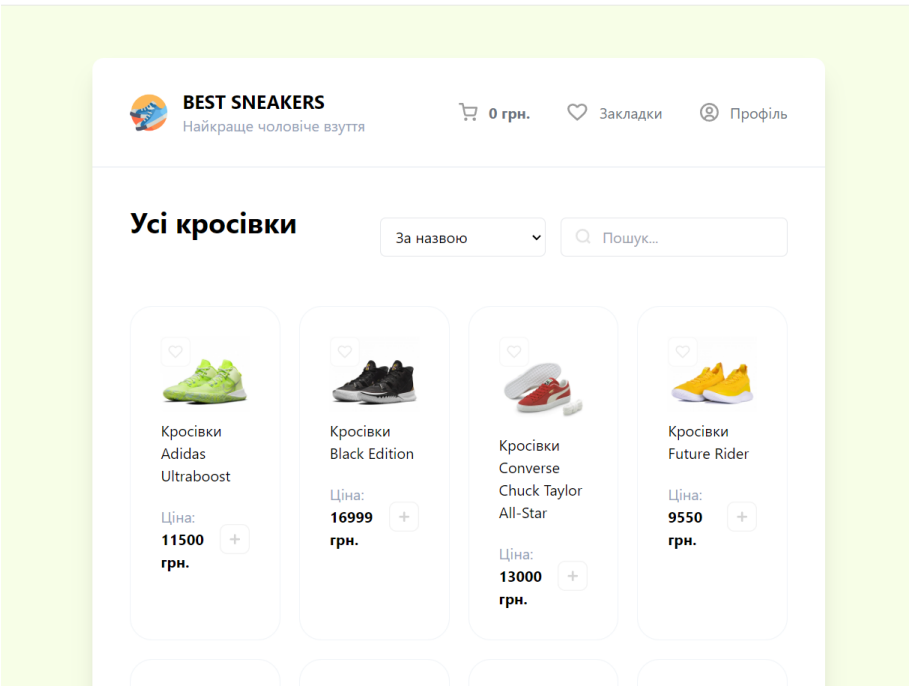

Рисунок 3.33 – Перевірка з роздільною здатністю 992px

3. У браузері Google Chrome встановлюється роздільна здатність 768px. Результат відображено на рисунку3.34.

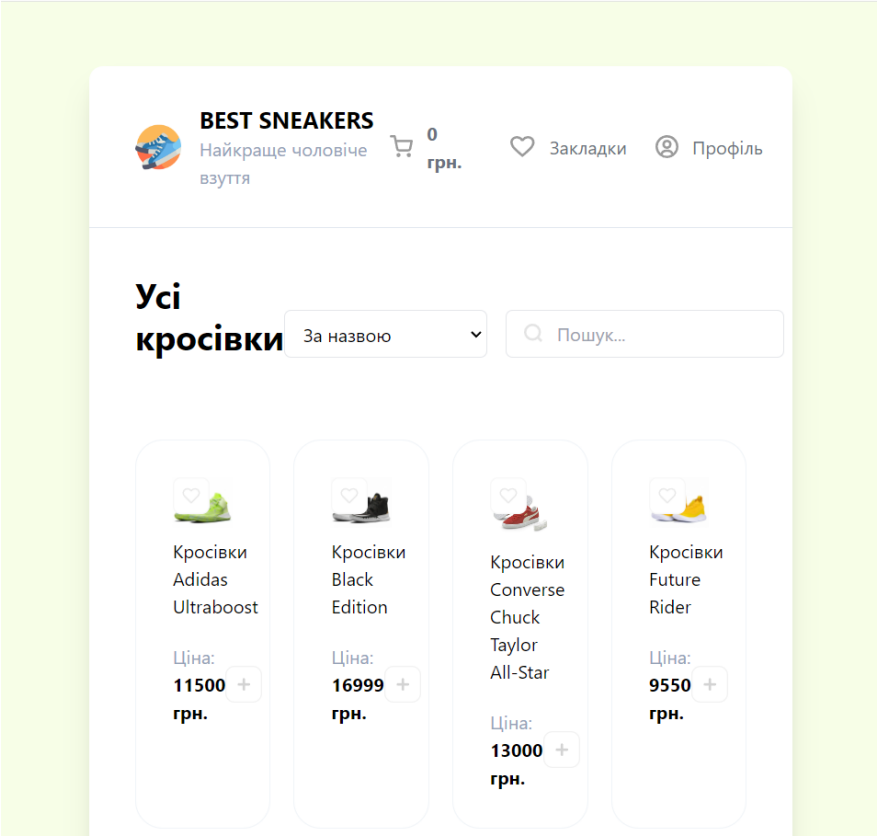

Рисунок 3.34 – Перевірка з роздільною здатністю 768px

У результаті тестування адаптивності вебсайту було перевірено коректність відображення об'єктів на різних роздільних здатностях екранів. Помилок на даному етапі не виявлено.

Отже, у результаті виконання розділу 3 було обрано засіб для програмної реалізації, розроблено програмний код та протестовано створений вебдодаток.

ВИСНОВКИ

У процесі виконання кваліфікаційної роботи було розроблено інформаційне та програмне забезпечення інтернет-магазину чоловічого взуття. Метою проєкту було створити сайт для пошуку та покупки взуття. Основним завданням було створення комфортного середовища для перегляду інформації про товари.

У процесі розробки було реалізовано декілька етапів. Перший етап включав аналіз магазинів-конкурентів, переваги та недоліки при взаємодії користувача та системи. Другий – у проєктуванні веборієнтованої системи, дизайну інтерфейсу, інформаційної моделі, включаючи розташування товарів, функціональність фільтрації та сортування, а також організацію процесу замовлення. На третьому етапі було обрано засоби для програмування та програмно реалізовано, використовуючи інструменти HTML, CSS, JavaScript, Vue. У рамках четвертого – виконано manual-тестування розробленого вебдодатку.

Після успішного тестування систему було розгорнуто на сервері та запущено в експлуатацію. Користувачі змогли почати використовувати створений інтернет-магазин для перегляду інформації про товар, перевірки наявності та замовлення чоловічого взуття.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Сайт Freehost.ua: Що таке сайт? [Електронний ресурс]. – Режим доступу: https://freehost.com.ua/ukr/faq/wiki/chto-takoe-sajt/. 29.01.2024

2. Сайт Бізнес: Онлайн, офлайн або все одразу? В Україні представили нове дослідження споживацьких звичок — публікуємо головні цифри. [Електронний ресурс]. – Режим доступу: https://biz.nv.ua/ukr/markets/hto-i-shcho-kupuye-v-ukrajini-onlayn-doslidzhennya-novini-ukrajini-50170428.html#:~:text=В%20Україні%20представили%20нове%20дослідження%20споживацьких%20звичок%20—%20публікуємо%20головні%20цифри&text=Поділитися%3A,як%20онлайн%2C%20так%20і%20офлайн. 29.01.2024

3. Сайт Freehost.ua: Що таке сайт? [Електронний ресурс]. – Режим доступу: https://freehost.com.ua/ukr/faq/wiki/chto-takoe-sajt/. 29.01.2024

4. Переваги покупок в Інтернеті [Електронний ресурс] – Режим доступу: https://www.0512.com.ua/news/3548676/perevagi-pokupok-v-internet-magazini. 02.02.24

5. ТОП-10 магазинів взуття в Україні [Електронний ресурс] – Режим доступу: https://torgsoft.ua/articles/stati/magazyny-vzuttja-ukrajiny/. 02.02.24

6. Інтернет-магазин Intertop [Електронний ресур] – Режим доступу: https://intertop.ua/uk-ua/. 02.02.24

7. Інтернет-магазин Sezon [Електронний ресур] – Режим доступу: https://sezon.ua. 02.02.24

8. Інтернет-магазин Mida [лектронний ресур] – Режим доступу: https://mida.style. 02.02.24

9. Клієнт-серверна архітерктура [Електронний ресурс] – Режим доступу: https://training.qatestlab.com/blog/technical-articles/client-server-architecture/. 15.03.24

10. Технічна документація про Vue.js [Електронний ресурс] – Режим доступу: https://ua.vuejs.org/guide/introduction.html. 15.03.24

11. Що таке html? [Електронний ресурс] – Режим доступу: https://css.in.ua/article/shcho-take-css_3. 15.03.24

12. Що таке PHP? [Електронний ресурс] – Режим доступу: https://freehost.com.ua/ukr/faq/wiki/chto-takoe-php/. 15.03.24

13. Дорожня карта: як вивчати Vue.js у 2023 році. [Електронний ресурс] – Режим доступу: https://spacelab.ua/articles/dorozhnya-karta-yak-vivchati-vuejs-u-2023-roci/. 15.03.24

14. Rapidly build modern websites without ever leaving your HTML. [Електронний ресурс] – Режим доступу: https://tailwindcss.com. 15.03.24

15. Технічна інформація про HTTP-клієнт для браузера на node.js. [Електронний ресурс] – Режим доступу: https://axios-http.com/uk/. 15.03.24

16. ESLint: Що? Чому? Де? Як? [Електронний ресурс] – Режим доступу: https://dev.to/dmytrych/eslint-shcho-chomu-die-iak-2c83. 15.03.24

17. Технічна документація про сервіс Mokky.dev [Електронний ресурс] – Режим доступу: https://mokky.gitbook.io/welcome. 15.03.24

18. 10 кращих редакторів коду. [Електронний ресурс] – Режим доступу: https://mate.academy/blog/front-end-and-js/top-10-text-editors/. 15.03.24

19. Stack Overflow Developer Survey 2022: JavaScript — найпопулярніша мова, користування Docker зростає, AWS випереджає інші хмари. [Електронний ресурс] – Режим доступу: https://dou.ua/forums/topic/38869/. 15.03.24

20. Топ 10 найкращих HTML редакторів. [Електронний ресурс] – Режим доступу: https://itvdn.com/ua/blog/article/top10-html. 15.03.24

21. Технічна документація про Notepad++. [Електронний ресурс] – Режим доступу: https://notepad-plus-plus.org. 15.03.24

22. Front-end Digest № 21: оптимізація продуктивності, плюси та мінуси Tailwind CSS та управління станами в React. [Електронний ресурс] – Режим доступу: https://dou.ua/forums/topic/46196/. 15.03.24

23. Tailwind CSS: Pros and Cons. [Електронний ресурс] – Режим доступу: https://medium.com/readyt.owork-org/tailwind-css-pros-and-cons-f1a8fdb1fb47. 15.03.24

24. Технічна документація про Tailwind CSS. [Електронний ресурс] – Режим доступу: https://tailwindcss.com. 15.03.24

ДОДАТКИ

Додаток А. Програмний код шапки сторінки

Header.vue:

```
<script setup>
defineProps({
  totalPrice: Number
})

const emit = defineEmits(['openDrawer'])
</script>

<template>
  <header class="flex justify-between border-b border-slate-200 px-10 py-8">
    <router-link to="/"
      ><div class="flex items-center gap-4">
        <img src="/logo.png" alt="Logo" class="w-10" />
        <div>
          <h2 class="text-xl font-bold uppercase">Best Sneakers</h2>
          <p class="text-slate-400">Найкраще чоловіче взуття</p>
        </div>
      </div></router-link
    >

    <ul class="flex items-center gap-10">
      <li
        @click="() => emit('openDrawer')"
        class="flex items-center cursor-pointer gap-3 text-gray-500 hover:text-black"
      >
        <img src="/cart.svg" alt="Cart" />
        <b>{{ totalPrice }} грн.</b>
      </li>

      <router-link to="/favorites">
        <li class="flex items-center cursor-pointer gap-3 text-gray-500 hover:text-black">
          <img src="/heart.svg" alt="Cart" />
          <span>Закладки</span>
        </li>
      </router-link>

      <li class="flex items-center cursor-pointer gap-3 text-gray-500 hover:text-black">
        <img src="/profile.svg" alt="Cart" />
        <span>Профіль</span>
      </li>
    </ul>
  </header>
</template>
```

Додаток Б. Програмний код картки товару

Card.vue:

```
<script setup>
defineProps({
  id: Number,
  title: String,
  imageUrl: String,
  price: Number,
  isFavorite: Boolean,
  isAdded: Boolean,
  onClickFavorite: Function,
  onClickAdd: Function
})
</script>

<template>
  <div
    class="relative bg-white border border-slate-100 rounded-3xl p-8 cursor-pointer transition hover:-translate-y-2 hover:shadow-xl"
  >
    <img
      v-if="onClickFavorite"
      :src="!isFavorite ? '/like-1.svg' : '/like-2.svg'"
      alt="Like 1"
      class="absolute top-8 left-8"
      @click="onClickFavorite"
    />

    <img :src="imageUrl" alt="Sneaker" />

    <p class="mt-2">{{ title }}</p>

    <div class="flex justify-between mt-5">
      <div class="flex flex-col">
        <span class="text-slate-400">Ціна:</span>
        <b>{{ price }} грн.</b>
      </div>

      <img
        v-if="onClickAdd"
        @click="onClickAdd"
        :src="!isAdded ? '/plus.svg' : '/checked.svg'"
        alt="Plus"
      />
    </div>
  </div>
</template>
```

CardItem.vue:

```
<script setup>
const emit = defineEmits(['onClickRemove'])
```

```
defineProps({
  id: Number,
  title: String,
  imageUrl: String,
  price: Number
})
</script>

<template>
  <div class="flex items-center border border-slate-200 p-4 rounded-xl gap-4">
    <img class="w-16 h-16" :src="imageUrl" :alt="title" />

    <div class="flex flex-col flex-1">
      <p>{{ title }}</p>

      <div class="flex justify-between mt-2">
        <b class="flex-1">{{ price }} грн.</b>
        <img
          @click="emit('onClickRemove')"
          class="opacity-40 hover:opacity-100 cursor-pointer transition"
          src="/close.svg"
        />
      </div>
    </div>
  </div>
</template>
```

Додаток В. Програмний код списку товару

CardList.vue:

```
<script setup>
import Card from './Card.vue'

defineProps({
  items: Array,
  isFavorites: Boolean
})

const emit = defineEmits(['addToFavorite', 'addToCart'])
</script>

<template>
  <div class="grid grid-cols-4 gap-5" v-auto-animate>
    <Card
      v-for="item in items"
      :key="item.id"
      :id="item.id"
      :title="item.title"
      :imageUrl="item.imageUrl"
      :price="item.price"
      :onClickFavorite="isFavorites ? null : () => emit('addToFavorite', item)"
      :onClickAdd="isFavorites ? null : () => emit('addToCart', item)"
      :isFavorite="item.isFavorite"
      :isAdded="item.isAdded"
    />
  </div>
</template>
```

Додаток Г. Програмний код кошику

Drawer.vue:

```
<script setup>
import axios from 'axios'
import { ref, computed, inject } from 'vue'

import DrawerHead from './DrawerHead.vue'
import CartItemList from './CartItemList.vue'
import InfoBlock from './InfoBlock.vue'

const props = defineProps({
  totalPrice: Number,
  vatPrice: Number
})

const { cart, closeDrawer } = inject('cart')

const isCreating = ref(false)
const orderId = ref(null)

const createOrder = async () => {
  try {
    isCreating.value = true

    const { data } = await axios.post(`https://0745ba94fb034cb9.mokky.dev/product`, {
      items: cart.value,
      totalPrice: props.totalPrice.value
    })

    cart.value = []

    orderId.value = data.id
  } catch (err) {
    console.log(err)
  } finally {
    isCreating.value = false
  }
}

const cartIsEmpty = computed(() => cart.value.length === 0)
const buttonDisabled = computed(() => isCreating.value || cartIsEmpty.value)
</script>

<template>
  <div class="fixed top-0 left-0 h-full w-full bg-black z-10 opacity-70"></div>
  <div class="bg-white w-96 h-full fixed right-0 top-0 z-20 p-8">
    <DrawerHead />

    <div v-if="!totalPrice || orderId" class="flex h-full items-center">
      <InfoBlock
        v-if="!totalPrice && !orderId"
```

```
            title="Кошик порожній"
            description="Оберіть хоча б 1 пару взуття."
            image-url="/package-icon.png"
          />
          <InfoBlock
            v-if="orderId"
            title="Замовлення сформовано!"
            :description="`Ваше замовлення #${orderId} незабаром передамо кур'єру`"
            image-url="/order-success-icon.png"
          />
        </div>

        <div v-else>
          <CartItemList />

          <div class="flex flex-col gap-4 mt-7">
            <div class="flex gap-2">
              <span>Всього:</span>
              <div class="flex-1 border-b border-dashed"></div>
              <b>{{ totalPrice }} ₴</b>
            </div>

            <div class="flex gap-2">
              <span>Податок 5%:</span>
              <div class="flex-1 border-b border-dashed"></div>
              <b>{{ vatPrice }} ₴</b>
            </div>

            <button
              :disabled="buttonDisabled"
              @click="createOrder"
              class="mt-4 transition bg-lime-500 w-full rounded-xl py-3 text-white disabled:bg-slate-
300 hover:bg-lime-600 active:bg-lime-700 cursor-pointer"
            >
              Оформити замовлення
            </button>
          </div>
        </div>
      </div>
    </template>
```

DrawerHead.vue:

```
    <script setup>
    import { inject } from 'vue'

    const { closeDrawer } = inject('cart')
    </script>

    <template>
      <div class="flex items-center gap-5 mb-8">
        <svg
          @click="closeDrawer"
```

```
        class="opacity-30 cursor-pointer rotate-180 hover:opacity-100 transition hover:-translate-x-1"
        width="16"
        height="14"
        viewBox="0 0 16 14"
        fill="none"
        xmlns="http://www.w3.org/2000/svg"
      >
        <path
          d="M1 7H14.7143"
          stroke="black"
          stroke-width="2"
          stroke-linecap="round"
          stroke-linejoin="round"
        />
        <path
          d="M8.71436 1L14.7144 7L8.71436 13"
          stroke="black"
          stroke-width="2"
          stroke-linecap="round"
          stroke-linejoin="round"
        />
      </svg>
      <h2 class="text-2xl font-bold">Кошик</h2>
    </div>
  </template>
```

Додаток Д. Програмний код головної сторінки

index.html:

```
<!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>Best Sneakers</title>
  </head>
  <body>
    <div id="app"></div>
    <script type="module" src="/src/main.js"></script>
  </body>
</html>
```

main.js:

```
import './assets/main.css'

import { createApp } from 'vue'
import { createRouter, createWebHistory } from 'vue-router'
import { autoAnimatePlugin } from '@formkit/auto-animate/vue'
import App from './App.vue'

import Home from './pages/Home.vue'
```

```
import Favorites from './pages/Favorites.vue'

const app = createApp(App)

const routes = [
  { path: '/', name: 'Home', component: Home },
  { path: '/favorites', name: 'Favorites', component: Favorites }
]

const router = createRouter({
  history: createWebHistory(),
  routes
})

app.use(router)
app.use(autoAnimatePlugin)

app.mount('#app')
```

main.css:

```
@tailwind base;
@tailwind components;
@tailwind utilities;

body {
  background-color: rgb(248 253 245);
}
```

Home.vue:

```
<script setup>
import { reactive, watch, ref, onMounted } from 'vue'
import axios from 'axios'
import debounce from 'lodash.debounce'
import { inject } from 'vue'
import CardList from '../components/CardList.vue'

const { cart, addToCart, removeFromCart } = inject('cart')

const items = ref([])

const filters = reactive({
  sortBy: 'title',
  searchQuery: ''
})

const onClickAddPlus = (item) => {
  if (!item.isAdded) {
    addToCart(item)
  } else {
    removeFromCart(item)
  }
}
```

```
const onChangeSelect = (event) => {
  filters.sortBy = event.target.value
}

const onChangeSearchInput = debounce((event) => {
  filters.searchQuery = event.target.value
}, 300)

const addToFavorite = async (item) => {
  try {
    if (!item.isFavorite) {
      const obj = {
        item_id: item.id
      }

      item.isFavorite = true

      const { data } = await axios.post(`https://604781a0efa572c1.mokky.dev/favorites`, obj)

      item.favoriteId = data.id
    } else {
      item.isFavorite = false
      await axios.delete(`https://604781a0efa572c1.mokky.dev/favorites/${item.favoriteId}`)
      item.favoriteId = null
    }
  } catch (err) {
    console.log(err)
  }
}

const fetchFavorites = async () => {
  try {
    const { data: favorites } = await axios.get(`https://604781a0efa572c1.mokky.dev/favorites`)

    items.value = items.value.map((item) => {
      const favorite = favorites.find((favorite) => favorite.item_id === item.id)

      if (!favorite) {
        return item
      }

      return {
        ...item,
        isFavorite: true,
        favoriteId: favorite.id
      }
    })
  } catch (err) {
    console.log(err)
  }
}
```

```
const fetchItems = async () => {
  try {
    const params = {
      sortBy: filters.sortBy
    }

    if (filters.searchQuery) {
      params.title = `*${filters.searchQuery}*`
    }

    const { data } = await axios.get(`https://604781a0efa572c1.mokky.dev/items`, {
      params
    })

    items.value = data.map((obj) => ({
      ...obj,
      isFavorite: false,
      favoriteId: null,
      isAdded: false
    }))
  } catch (err) {
    console.log(err)
  }
}

onMounted(async () => {
  const localCart = localStorage.getItem('cart')
  cart.value = localCart ? JSON.parse(localCart) : []

  await fetchItems()
  await fetchFavorites()

  items.value = items.value.map((item) => ({
    ...item,
    isAdded: cart.value.some((cartItem) => cartItem.id === item.id)
  }))
})

watch(cart, () => {
  items.value = items.value.map((item) => ({
    ...item,
    isAdded: false
  }))
})

watch(filters, fetchItems)
</script>

<template>
  <div class="flex justify-between items-center">
    <h2 class="text-3xl font-bold mb-8">Усі кросівки</h2>
```

```
            <div class="flex gap-4">
              <select @change="onChangeSelect" class="py-2 px-3 border rounded-md outline-none">
                <option value="name">За назвою</option>
                <option value="price">По ціні (дешевші)</option>
                <option value="-price">По ціні (дорожчі)</option>
              </select>

              <div class="relative">
                <img class="absolute left-4 top-3" src="/search.svg" />
                <input
                  @input="onChangeSearchInput"
                  class="border rounded-md py-2 pl-11 pr-4 outline-none focus:border-gray-400"
                  type="text"
                  placeholder="Пошук..."
                />
              </div>
            </div>

            <div class="mt-10">
              <CardList :items="items" @add-to-favorite="addToFavorite" @add-to-
cart="onClickAddPlus" />
            </div>
          </template>
```

Додаток Е. Програмний код сторінки «Обране» (закладки)

Favorites.vue:

```
<script setup>
import { ref, onMounted } from 'vue'
import axios from 'axios'

import CardList from '../components/CardList.vue'

const favorites = ref([])

onMounted(async () => {
  try {
    const { data } = await axios.get(
      'https://604781a0efa572c1.mokky.dev/favorites?_relations=items'
    )

    favorites.value = data.map((obj) => obj.item)
  } catch (err) {
    console.log(err)
  }
})
</script>

<template>
  <h2 class="text-3xl font-bold mb-8">Мої закладки</h2>

  <CardList :items="favorites" is-favorites />
</template>
```