МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Чорноморський національний університет імені Петра Могили
Факультет комп'ютерних наук
Кафедра інженерії програмного забезпечення

ДОПУЩЕНО ДО ЗАХИСТУ

Завідувач кафедри інженерії програмного забезпечення, канд. техн. наук, доцент,

_____ Є. О. Давиденко

«____»_____2023 р.

КВАЛІФІКАЦІЙНА РОБОТА БАКАЛАВРА

ВЕБЗАСТОСУНОК ПРОДАЖУ ОДЯГУ НА ОСНОВІ VUE

Спеціальність «Інженерія програмного забезпечення»

121 – КРБ.1 − 408.19108250

Студентка _____В. О. Сабіна

підпис

« __»_____2023 р.

Керівник PHD, ст. викладач

_____ І. О. Кандиба

підпис

« __»_____2023 р.

Консультант канд. техн. наук, доцент

_____ А. О. Алек сєєва

підпис

« __»_____2023 р.

Миколаїв – 2023

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Чорноморський національний університет імені Петра Могили
Факультет комп'ютерних наук
Кафедра інженерії програмного забезпечення

ЗАТВЕРДЖУЮ

Завідувач кафедри інженерії програмного

забезпечення, канд.техн.наук, доцент,

_____Є.О. Давиденко

«____»_____2023 р.

ЗАВДАННЯ
на виконання кваліфікаційної роботи бакалавра

Видано студенту групи 408 факультету комп'ютерних наук

Сабіні Валерії Олександрівні.
(прізвище, ім'я, по батькові студента)

1. Тема кваліфікаційної роботи

«Вебзастосунок продажу одягу на основі Vue».

Затверджена наказом по ЧНУ від «17» березня 2023 р. № 60

2. Строк представлення кваліфікаційної роботи «____» _____ 2023р.

3. Очікуваний результат роботи та початкові дані, якщо такі потрібні

Вхідні дані до роботи − функціональні та нефункціональні вимоги до вебзастосунку продажу одягу на основі Vue. Результат – функціонуючий вебзастосунку продажу одягу на основі Vue.

4. Перелік питань, що підлягають розробці:

- Проаналізувати сучасні тенденції та вимоги користувачів до інтернет-магазинів для продажу одягу.
- Сформулювати вимоги до функціоналу та дизайну вебзастосунку.

- Розробка архітектури інтернет-магазину.
- Розробка функціоналу вебзастосунку, включаючи можливість пошуку та фільтрації товарів за різними параметрами, оформлення замовлення, оплату та доставку товарів.
- Протестувати розроблений вебзастосунок.

5. Перелік графічних матеріалів:

Презентація.

6. Завдання до спеціальної частини

Дослідження питань охорони праці, які безпосередньо пов'язані з діяльністю розробника програмного забезпечення .

7. Консультанти:

Консультант	Кафедра (організація)	Частина роботи
Алексєєва А.О., канд. техн. наук., доцент	Кафедра екології Медичного інституту ЧНУ ім. Петра Могили	Спеціальна частина з охорони праці

Керівник роботи PhD ст. викладач, Кандиба Ігор Олександрович
(посада, прізвище, ім'я, по батькові)

_____
(підпис)

Завдання прийнято до виконання

Сабіна Валерія Олександрівна

(прізвище, ім'я, по батькові студента)

_____
(підпис)

Дата видачі завдання «____» _____ 2023 р.

КАЛЕНДАРНИЙ ПЛАН

виконання кваліфікаційної роботи

Тема: «Вебзастосунок продажу одягу на основі Vue».

№	Найменування роботи	Початок	Закінчення	Примітки
1.	Розробка та затвердження завдання на виконання КРБ	26.10.2022	27.10.2022	Виконано
2.	Огляд літератури за темою роботи	30.10.2022	2.11.2022	Виконано
3.	Складання календарного плану КРБ	20.01.2023	22.01.2022	Виконано
4.	Аналіз предметної області	23.01.2023	25.01.2023	Виконано
5.	Розробка проєктних рішень	25.01.2023	26.01.2023	Виконано
6.	Моделювання та конструювання ПЗ	26.02.2023	28.02.2023	Виконано
7.	Тестування та апробація розробленого ПЗ, аналіз результатів тестування, розробка керівництва користувача	28.01.2023	28.02.2023	Виконано
8.	Розробка спеціальної частини з охорони праці	05.03.2023	10.03.2023	Виконано
9.	Відгук керівника КРБ	20.05.2023	21.05.2023	Виконано
6.	Оформлення КРБ та презентації	10.05.2023	22.05.2023	Виконано
7.	Попередній захист	06.07.2023	07.06.2023	Виконано
8.	Рецензування	26.05.2023	18.06.2023	Виконано
10.	Завершення оформлення КРБ та презентації	19.06.2023	21.06.2023	Виконано
14.	Захист кваліфікаційної роботи	27.06.2023	27.06.2023	Виконано

Розробила студентка _____Сабіна Валерія Олександрівна

«____» _____ 2023 р.

Керівник роботи PhD, ст. викладач Кандиба Ігор Олександрович

«____» _____ 2023 р

АНОТАЦІЯ

до кваліфікаційної роботи бакалавра

«Вебзастосунок продажу одягу на основі Vue»

Студентка 408 гр.: Сабіна Валерія Олександрівна

Керівник: PhD, ст. викладач Кандиба Ігор Олександрович

Метою роботи є вдосконалення інтернет-магазину для продажу продажу одягу, шляхом використання фреймворку Vue.

В роботі обґрунтовується актуальність дослідження та ставиться проблема, мета та завдання проєкту, визначається об'єкт та предмет дослідження.

При розробці проєкту була розкрита актуальність дослідження в обраному напрямку, поставлені цілі та завдання дослідження, визначені об'єкт та предмет дослідження, обґрунтовані основні проєктні рішення та вказана його теоретична та практична значущість.

У першому розділі «Аналіз предметної області» проводиться детальний аналіз структурних та функціональних особливостей предметної області. Також проводиться огляд існуючих аналогів та аналіз існуючих методів і засобів для вирішення поставлених завдань. Результатом цього розділу є специфікація вимог до програмного забезпечення.

Другий розділ «Моделювання вебзастосунку продажу одягу» присвячений розробці сценаріїв використання, проєктування системи і інтерфейсів користувача.

Третій розділ «Проектування вебзастосунку продажу одягу» описує ієрархію класів системи та вибір технологій та мов програмування. Також розглядається написання use cases - варіантів використання системи.

Четвертий розділ «Кодування програмного забезпечення» охоплює реалізацію програмного забезпечення, зокрема роботу зі зовнішнім JSON-

файлом як джерелом даних, програмну реалізацію функціональності та підготовку керівництва адміністратора та користувача.

Кваліфікаційна робота бакалавра «Вебзастосунок продажу одягу на основі Vue» складається зі 74 сторінок, включаючи 26 рисунків та 11 джерел. У роботі було використано різноманітні джерела, включаючи наукові статті, книги та Інтернет-джерела.

Процес розробки програмного забезпечення включав в себе використання таких технологій, як Vue.js [4], Vuex. Робота містить детальні описи проєктних рішень та класів системи, а також опис процесу тестування та виявлення проблем під час розробки. В цілому, кваліфікаційна робота бакалавра є досить об'ємною та детальною роботою, що демонструє здатність студентки Сабіни Валерії Олександрівни до розробки функціонального та безпечного інтернет-магазину одягу.

Отже, робота містить детальний опис проєкту інтернет-магазину одягу, включаючи аналіз вимог, проєктування системи, розробку та тестування програмного забезпечення. Результатом роботи є функціональний та безпечний інтернет-магазин одягу.

Ключові слова: інтернет-магазин, програмне забезпечення, електронна комерція, вебсервіси.

ABSTRACT

for the bachelor's qualification work

"Vue-based Clothing Sales Web Application"

Student of group 408: Sabina Valeriya Oleksandrivna

Supervisor: Senior Lecturer, PhD Kandyba Ihor Oleksandrovich

The purpose of the work is to improve the online store for clothing sales by using the Vue framework. The paper justifies the relevance of the research, defines the problem, objective, and tasks of the project, determines the object and subject of the study. During the project development, the relevance of the research in the chosen direction was revealed, the goals and objectives of the research were set, the object and subject of the study were defined, the main project decisions were substantiated, and its theoretical and practical significance was indicated.

The first chapter, "Analysis of the Subject Area," provides a detailed analysis of the structural and functional features of the subject area. It also includes a review of existing analogues and an analysis of existing methods and tools for solving the defined tasks. The result of this chapter is the specification of software requirements.

The second chapter "Modeling a web application for selling clothes" is devoted to the development of usage scenarios, system design and user interfaces.

The third chapter "Designing a web application for selling clothes" describes the hierarchy of system classes and the choice of technologies and programming languages. The writing of use cases - options for using the system is also considered.

The fourth chapter, "Software Coding," covers the implementation of the software, including working with an external JSON file as a data source, program implementation of functionality, and preparation of administrator and user guides. The bachelor's qualification work, "Vue-based Clothing Sales Web Application," consists of 74 pages, including 26 figures and 11 references. Various sources were used in the work, including scientific articles, books, and internet sources.

The software development process involved the use of technologies such as Vue.js and Vuex. The work contains detailed descriptions of project decisions and

system classes, as well as a description of the testing process and problem identification during development. Overall, the bachelor's qualification work is a comprehensive and detailed work that demonstrates the ability of student Sabina Valeriya Oleksandrivna to develop a functional and secure clothing online store. Therefore, the work provides a detailed description of the clothing online store project, including requirements analysis, system design, software development, and testing. The result of the work is a functional and secure clothing online store.

Keywords: online store, software, e-commerce, web services.

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ

ПЗ – Програмне забезпечення

API – Application Programming Interface

HTML – HyperText Markup Language (мова розмітки гіпертексту)

CSS – Cascading Style Sheets (каскадні таблиці стилів)

JS – JavaScript (мова програмування)

DFD – Data Flow Diagams

ВСТУП

Кваліфікаційна робота спрямована на практичне використання набутих навичок в процесі навчання шляхом розробки вебзастосунку для онлайн-магазину одягу, досліджується тема конструювання програмного забезпечення з використанням стеку вебтехнологій, що базується на вебфреймворку Vue.js [4].

У сучасному світі Інтернет відіграє важливу роль в нашому повсякденному житті. З його допомогою ми можемо робити багато різних речей, від спілкування з друзями до покупок та отримання інформації про навколишній світ. Тому, розробляючи вебзастосунок для магазину одягу, були досліджені не тільки технології, але і звички та поведінку користувачів в Інтернеті.

Головна технологія, яка використовується в проєкті – це Vue.js, вебфреймворк з легкою та зручною структурою компонентів, що дозволяє розробляти високоякісні веб-застосунки. Кожен компонент магазину одягу розробляється окремо, що дозволяє зручно керувати процесом розробки та зберігати код для майбутнього використання.

Досліджено сучасні технології та дизайнерські рішення, що дозволяють створювати вебзастосунки високої якості та забезпечувати кращий досвід користувача. Був обраний вебфреймворк Vue.js [4] для розробки кваліфікаційної роботи, оскільки він має багато переваг, таких як легкість, продуктивність та зручна структура компонентів. Vue.js дозволяє розробляти кожен компонент окремо, що робить його повторне використання більш ефективним. Крім того, наявність багатьох різноманітних плагінів дозволяє значно спростити процес розробки.

Отже, в рамках кваліфікаційної роботи був розробилений вебзастосунок для онлайн-магазину одягу з використанням сучасних технологій та дизайнерських рішень. В результаті роботи реалізовано основний функціонал магазину, зокрема відображення каталогу товарів, пошук, додавання товарів

до кошика, оформлення замовлення, оплата та доставка. Також були застосовані плагіни, що значно спрощували процес розробки.

Проєкт може бути використаний як основа для подальшого розширення функціональності та розвитку онлайн-магазину.

Об’єкт дослідження: Процес розробки вебзастосунку продажу одягу.

Предмет дослідження: програмні засоби для реалізації функціоналу інтернет-магазину та інтерфейс користувача для нього.

Мета роботи: вдосконалення інтернет-магазину для продажу продажу одягу, шляхом використання фреймворку Vue.

продажу одягу, шляхом використання фреймворку Vue.

Відповідно до мети визначено такі завдання:

1. проаналізувати сучасні тенденції та вимоги користувачів до інтернет-магазинів для продажу одягу;
2. сформулювати вимоги до функціоналу та дизайну вебзастосунку.
3. розробка архітектури інтернет-магазину;
4. розробка функціоналу вебзастосунку;
5. тестування розроблений вебзастосунку.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

При розробці вебзастосунку для онлайн-магазину одягу з використанням Vue.js, необхідно визначити конкретний перелік функцій, що дозволить проєктувати та розробляти систему з максимально ефективним та логічним виконанням поставлених завдань. Для цього необхідно провести аналіз предметної галузі та існуючих методів та засобів, які допоможуть досягнути мети проєкту. Крім того, важливо враховувати особливості Vue.js, які дозволяють розробляти веб-застосунки високої якості та забезпечувати найкращий досвід користувача.

1.1 Огляд існуючих аналогів

Виходячи з мети та завданнь роботи проаналізовано аналогічні застосунки, а також існуючі рішення та вебтехнології.

Назва: Gepur.com

Розробник (дистриб'ютор): Gepur LLC

Архітектура: 3-tier web application

Мова реалізації: PHP, JavaScript

Перелік функцій та характеристик:

1. Продаж жіночого одягу та аксесуарів.
2. Можливість онлайн-замовлення та доставки.
3. Огляд товарів з можливістю перегляду фотографій та відео.
4. Програма лояльності для постійних клієнтів.
5. Підтримка клієнтів у чаті та по електронній пошті.

Аналіз переваг та недоліків даного ПЗ:

Переваги:

1. Великий вибір жіночого одягу та аксесуарів.
2. Простий та зручний інтерфейс користувача.
3. Програма лояльності для постійних клієнтів.
4. Можливість швидкого онлайн-замовлення та доставки.

Недоліки:

1. Обмежений асортимент товарів порівняно з конкурентами.
2. Недостатній рівень інформації про товари на сайті.
3. Не кожен товар має детальну інформацію та відгуки користувачів.
4. Відсутність круглодобової підтримки клієнтів та обмежені години роботи служби підтримки.

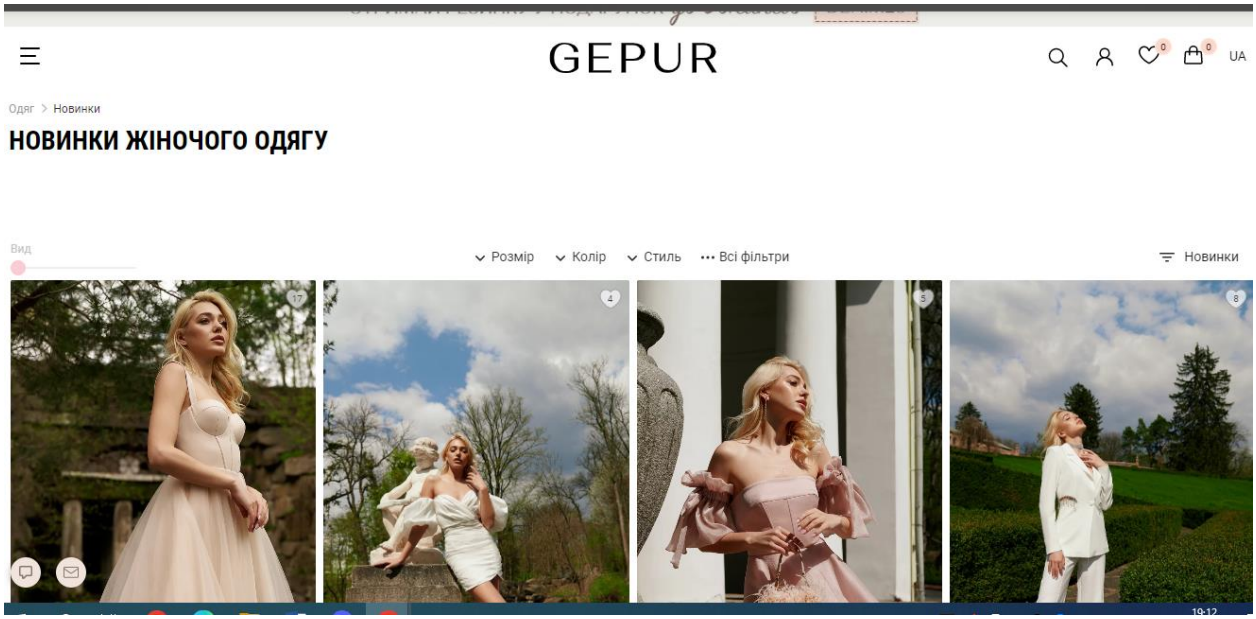

Рисунок 1.1 – головна сторінка магазину Gepur

Назва: Revolve.com

Розробник (дистриб'ютор): Revolve Group, LLC

Архітектура: 3-tier web application

Мова реалізації: PHP, JavaScript

Переваги:

1. Великий вибір модного одягу та аксесуарів від різних брендів.
2. Висока якість товарів та відповідність опису.
3. Доступність інформації про товари, включаючи фотографії та відео.
4. Міжнародна доставка та швидкий час обробки замовлень.

Недоліки:

1. Вищий ціновий діапазон порівняно з деякими іншими сайтами.
2. Обмежений вибір розмірів для деяких товарів.

3. Можуть виникати проблеми з поверненням товарів через процес та витрати.
4. Дуже погана адаптивність під різні пристрої.

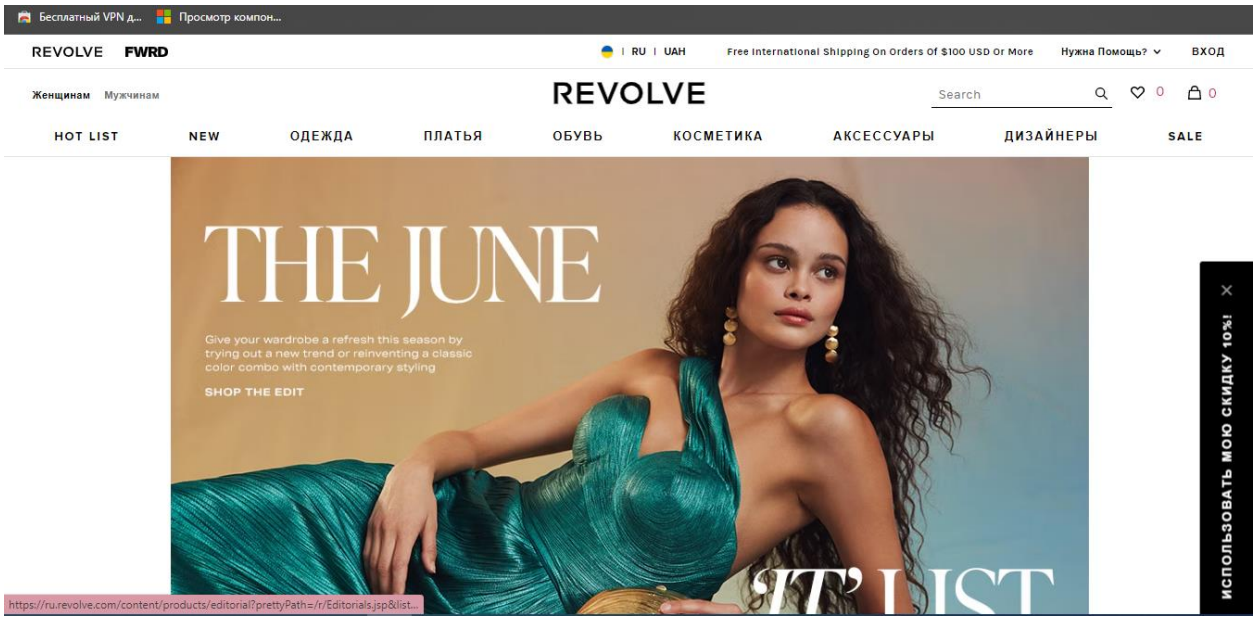

Рисунок 1.2 – головна сторінка магазину Revolve

Назва: Shein.com

Розробник (дистриб'ютор): Shein Group Limited

Архітектура: 3-tier web application

Мова реалізації: PHP, JavaScript

Переваги:

1. Широкий вибір модних трендів за доступними цінами.
2. Багато варіантів розмірів та вибір замість для більшої гнучкості.
3. Міжнародна доставка та безкоштовна доставка при певному замовленні.
4. Акції, знижки та програма лояльності.

Недоліки:

1. Якість товарів може варіюватися, особливо для дешевих позицій.
2. Деякі клієнти зазначають проблеми з поверненням товарів із-за високих витрат та складнощів у процесі.
3. Не завжди відповідає очікуванням щодо якості та відповідності розмірам.

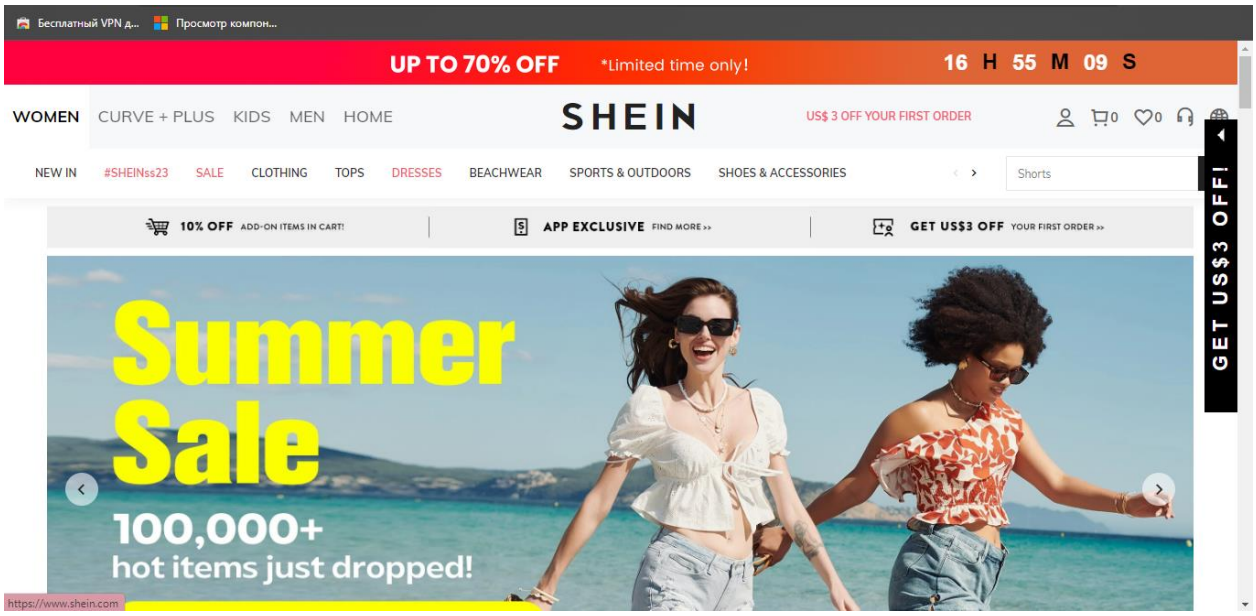

Рисунок 1.3 – головна сторінка магазину Shein

1.2 Структурні та функціональні особливості

Клієнтська частина застосунку - це графічний інтерфейс, який відображається в браузері користувача та взаємодіє з серверною частиною застосунку через API.

Основними елементами клієнтської частини вебзастосунку будуть:

1. Головна сторінка: на ній будуть відображатись основні розділи вашого інтернет-магазину, такі як каталог товарів, розділ зі знижками, новинки та інші.

2. Каталог товарів: на цій сторінці користувачі зможуть переглядати список товарів з можливістю сортування та фільтрації за різними параметрами, такими як ціна, розмір, колір та інші. Кожен товар буде відображатись зі своїм зображенням, назвою та ціною.

3. Сторінка товару: на цій сторінці буде відображена детальна інформація про обраний товар, така як: опис, характеристики, розмірні таблиці, зображення та інші. Також на цій сторінці користувач зможе додати товар до кошика та здійснити замовлення.

4. Кошик: на цій сторінці користувач може переглянути список товарів, які він додав до свого кошика та здійснити замовлення, обравши спосіб оплати та доставки.

5. Реєстрація та авторизація користувачів: на сторінках реєстрації та авторизації користувачів будуть поля для введення електронної пошти та пароля, а також кнопки для відправлення даних до сервера та відновлення пароля.

Як було зазначено раніше, функціональні вимоги до системи, що розробляється, представлені чітко визначеним списком функцій, а саме:

1. Реєстрація та авторизація користувачів.
2. Пошук та перегляд товарів за категоріями.
3. Додавання товарів в кошик та оформлення замовлення.
4. Оплата замовлення.

Для розробки веб-застосунку на базі Vue.js та Vuex [8] існує велика кількість різноманітних методів та засобів, що можуть бути використані для вирішення поставлених завдань. Розглянемо деякі з них.

Vue Router - це офіційна бібліотека для роботи з маршрутизацією в Vue.js. Вона дозволяє організовувати навігацію між сторінками за допомогою URL-адрес. З її допомогою можна визначити маршрути, налаштувати перехоплювання запитів, редиректити користувачів на інші сторінки, і багато іншого. Використання Vue Router значно полегшує роботу зі створенням SPA, оскільки дозволяє відобразити на одній сторінці різні компоненти залежно від поточного URL.

Axios - це бібліотека для виконання запитів на сервер за допомогою HTTP протоколу. Axios дозволяє виконувати запити на сервер, які повертають дані в різних форматах (JSON, XML, HTML, і т.д.), та здійснювати інші операції з HTTP, такі як налаштування заголовків запиту, обробка помилок, таймаут, і т.д. Axios є дуже популярним засобом для виконання запитів на сервер, особливо в SPA.

Vuex - це бібліотека стану, яка дозволяє керувати станом додатку в Vue.js. Vuex [8] зберігає всі дані, що потрібні додатку, в одному централізованому місці, що дозволяє уникнути дублювання даних та покращити їх управління. В Vuex є декілька ключових понять, таких як Store, Mutations, Actions, та Getters, які дозволяють працювати зі станом додатку, модифікувати його та отримувати дані зі сторінки.

Одним з основних методів для роботи з формами є бібліотека Vuelidate. Вона дозволяє валідувати дані, які вводять користувачі, і відображати повідомлення про помилки. Vuelidate підтримує валідацію різних типів даних, таких як текстові поля, числа, дати, електронні адреси, інші поля форм, а також може працювати з валідацією вкладених об'єктів та масивів.

Ще одним методом є використання бібліотеки axios для взаємодії з сервером. Axios надає легкий і зручний інтерфейс для відправки запитів на сервер та отримання відповідей. Вона підтримує різні методи запитів, такі як GET, POST, PUT, DELETE, а також може відправляти дані в різних форматах, таких як JSON, FormData, URL encoded, інші.

Для роботи з кошиком товарів можна використовувати Vuex, який є становим менеджером для Vue.js. Він дозволяє зберігати стан додатку та здійснювати мутації над ним, що дозволяє зберігати інформацію про товари, які додані в кошик, а також кількість кожного з товарів, їх ціну та інші важливі дані. За допомогою Vuex [8] можна здійснювати асинхронні операції, такі як додавання товару до кошика, оновлення його кількості, видалення з кошика та оформлення замовлення.

1.3 Аналіз існуючих методів і засобів вирішення поставлених завдань

Для того щоб у повній мірі виконати поставлені завдання та досягти мети роботи не існує інших методів та шляхів, крім розробки сучасного, адаптивного вебзастосунку.

Для реалізації інтернет-магазину для продажу одягу можна використовувати різноманітні програмні засоби та фреймворки. Розробка сучасного, адаптивного веб-застосунку є важливим завданням для створення інтернет-магазину продажу одягу. Використання фреймворків може спростити та прискорити процес розробки. Розглянемо декілька популярних фреймворків і визначимо, чому Vue.js є одним з найкращих виборів.

Розробка веб-застосунків за допомогою React.js та Angular є популярними підходами у сучасному програмуванні. Обидва фреймворки мають свої особливості, переваги та недоліки, які варто враховувати при виборі для створення інтернет-магазину продажу одягу. Розглянемо кожен фреймворк детальніше:

React.js: React.js є одним з найпопулярніших JavaScript-фреймворків для розробки веб-застосунків. Він розроблений командою Facebook та має велику спільноту розробників, що сприяє активному розвитку та підтримці. Головною концепцією React.js є розділення користувацького інтерфейсу на невеликі, повторно використовувані компоненти. Це дозволяє побудувати динамічні та інтерактивні інтерфейси з мінімальними зусиллями.

Одна з ключових особливостей React.js - використання віртуального DOM (Document Object Model). Він є відображенням реального DOM і дозволяє React.js ефективно виконувати оновлення і зміни елементів інтерфейсу. Завдяки віртуальному DOM, React.js забезпечує високу швидкодію та ефективну роботу з компонентами.

React.js також надає широкий спектр додаткових бібліотек та інструментів, таких як Redux, React Router, Jest, і багато інших, що полегшують розробку та тестування веб-застосунків. Однак, важливо відзначити, що React.js має дещо крутіший навчальний криву порівняно з іншими фреймворками, включаючи Vue.js. Це означає, що для початківців може знадобитися більше часу та зусиль для вивчення та засвоєння концепцій та підходів React.js.

Angular: Angular є повноцінним фреймворком для розробки веб-застосунків, розробленим компанією Google. Він має широкий набір функцій та інструментів, що дозволяє розробникам будувати великі та складні проєкти. Однак, поріг входження в Angular вважається вищим, а розробка вимагає більшої кількості коду порівняно з іншими фреймворками, зокрема з Vue.js.

Основна концепція Angular - це структурований підхід до розробки веб-застосунків, який використовує TypeScript, об'єктно-орієнтовану мову програмування, для створення компонентів, сервісів, директив та інших елементів додатку. Angular надає широкі можливості для управління станом додатку, валідації даних, маршрутизації та багато інших функцій, що часто потрібні для розробки великих та складних проєктів.

Angular також має вбудований механізм тестування та інструменти для розробки, такі як Angular CLI, які спрощують рутинні завдання. Однак, через свою складність, Angular може бути складнішим для вивчення та використання, особливо для новачків у веб-розробці.

При виборі між React.js та Angular для розробки інтернет-магазину продажу одягу, важливо враховувати потреби та рівень навичок команди розробників, час, який можна витратити на вивчення та розробку, а також масштаб та складність проєкту.

Один з популярних варіантів - використання фреймворку Vue.js.

Vue.js є JavaScript-фреймворком, що використовується для створення веб-інтерфейсів та SPA (Single Page Applications). Основна мета Vue.js полягає у спрощенні процесу розробки та підтримці коду, завдяки чому розробники можуть швидко створювати високоякісні веб-застосунки.

Vuex є бібліотекою для Vue.js, яка дозволяє керувати станом застосунок та забезпечує швидкий та простий доступ до даних з будь-якої точки застосунку. Ця бібліотека зберігає дані в одному місці (store), що дозволяє розробникам зосередитись на розробці функціоналу та інтерфейсу, а не на

керуванні даними. Завдяки компонентній архітектурі Vue.js, розробники можуть перевикористовувати код та підтримувати його.

Одним з основних переваг Vue.js є його простота та легкість вивчення, що дозволяє швидко навчитися розробляти веб-застосунки на цій технології. Крім того, Vue.js має широку спільноту розробників, яка постійно поповнює базу знань та створює нові розширення та плагіни для розширення функціоналу.

Однією з ключових особливостей Vue.js є реактивність. Це означає, що зміни в стані додатку автоматично відображаються на сторінці без необхідності перезавантаження сторінки. Це забезпечує користувачам більш приємний та безперервний досвід використання веб-застосунків.

Окрім того, Vuex дозволяє використовувати middleware, які забезпечують додаткові функціональність та допомагають розробникам підтримувати код. Наприклад, middleware може використовуватися для логування даних, валідації вхідних даних, збереження даних до локального сховища та багатьох інших завдань.

Однією з найбільших переваг Vue.js та Vuex є їх гнучкість та можливість легко налаштувати їх під потреби проєкту. Розробники можуть використовувати різні плагіни та компоненти для додавання нового функціоналу до свого проєкту та легко налаштовувати їх для досягнення бажаного результату.

Загалом, Vue.js та Vuex є потужними інструментами для розробки веб-застосунків з високою якістю та складною логікою. Вони дозволяють швидко створювати веб-інтерфейси, керувати станом додатку та забезпечувати реактивність та гнучкість. Ці інструменти мають велику спільноту розробників та велику кількість документації, що дозволяє розробникам швидко знайти відповіді на свої питання та розв'язувати проблеми.

Vue.js та Vuex також мають вбудовану підтримку компонентної архітектури, що дозволяє розробникам розділити додаток на окремі

компоненти, що полегшує розробку та зберігання коду. Це також дозволяє забезпечити повторне використання коду та зменшення залежності між компонентами, що робить проєкт більш гнучким та легко розширюваним.

Використовується шаблонізатор Pug для полегшення розробки та зменшення кількості написаного коду. Використання Pug дозволяє створювати код з меншою кількістю рядків, що забезпечує більшу читабельність та зручність у роботі.

Крім того, для стилізації було використано препроцесор SCSS. Використання SCSS дозволяє створювати більш складні та зручні для розробки стилі, що забезпечує швидше та ефективніше написання CSS коду.

Нарешті, для зберігання та управління станом додатку, використовується бібліотека Vuex. Використання Vuex дозволяє забезпечити консистентність та зручність управління станом мого додатку, що дуже важливо для досягнення високої продуктивності та ефективності у роботі мого вебзастосунку для продажу одягу.

1.4 Специфікація вимог до ПЗ

Даний проєкт передбачає розробку веб-застосунку для продажу одягу на основі Vue.

1. Призначення:
    - Призначенням застосунку є вдосконалення процесу продаж одягу.
2. Загальний опис:
    - Сфера застосування: застосунок можна використовувати для продажу одягу.
    - Характеристики користувачів: користувач повинен мати смартфон, планшет або ПК з доступом до мережі Інтернет.
3. Функції системи:

Основні ролі користувачів:

– гість – користувач який має можливість обрати товар, подивитися і знайти товар, додати обраний товар у кошик і оформити замовлення.

– адміністратор – має можливість додавати товар до каталогу.

Авторизація та реєстрація користувачів:

a. Реєстрація користувачів за допомогою форми з валідацією введених даних.
b. Авторизація користувачів за допомогою електронної пошти та пароля.

1) Пошук та перегляд товарів:
   a. Пошук товарів за назвою, категорією та фільтрами.
   b. Відображення списку товарів з пагінацією та сортуванням.
   c. Можливість переглядати деталі товару, його фотографії та відгуки про товар.
2) Кошик та замовлення:
   a. Додавання товарів у кошик та відображення його вмісту.
   b. Оформлення замовлення з заповненням форми з контактними даними та адресою доставки.

4. Функціональні вимоги:

- Веб-застосунок має забезпечувати можливість користувачам здійснювати покупки в магазині, додавати товари до кошика, оформлювати замовлення, виконувати пошук тощо.

- Додатково можуть бути реалізовані такі функції, як розрахунок вартості доставки, оплата онлайн, фільтрація товарів за різними параметрами (розмір, колір, стиль тощо).

5. Вимоги до технічного забезпечення:

- Веб-застосунок повинен бути доступний користувачам з будь-якого пристрою з доступом до Інтернету.

- Веб-застосунок має бути забезпечений достатньою швидкістю завантаження сторінок і відповідним рівнем безпеки.

Налагоджена система збереження інформації про замовлення і клієнтів.

6. Вимоги до програмного забезпечення:

- Веб-застосунок повинен бути побудований на основі Vue.js та використовувати Vuex для управління станом даних в програмі.

- Для стилізації веб-сторінок використовується SCSS.

- Мова програмування - JavaScript.

- Для забезпечення надійності та безпеки можуть бути використані такі технології, як SSL-шифрування, обробка валідації вхідних даних на стороні сервера, застосування антифрод-заходів тощо.

Висновки до розділу 1

В ході досліджень описаних в поточному розділі було проведено детальний аналіз предметної галузі, а саме існуючих застосунків. Виявлено їх переваги та недоліки, виявлено основні технологічні особливості та обмеження.

Виходячи із проведеного аналізу застосунків та ринку вебтехнологій було проаналізовано основні засоби для вирішення поставлених задач програмним шляхом і досягнення основної мети роботи.

Враховуючи всі проаналізовані фактори, можна зробити висновок, що Vue.js є одним з найкращих фреймворків для розробки інтернет-магазину продажу одягу. Він надає простоту, ефективність, гнучкість та широкі можливості для розширення.

Тема роботи вебзастосунок продажу одягу є актуальною.

Згідно з створено специфікацію вимог до ПЗ дотримуючись зазначеної структури. У специфікації вимог зазначено короткий опис ПЗ.

2 МОДЕЛЮВАННЯ ВЕБЗАСТОСУНКУ ПРОДАЖУ ОДЯГУ НА ОСНОВІ VUE

2.1 Розробка сценаріїв використання та діаграм послідовності

Діаграма послідовності «Оформлення замовлення на сайті магазину» демонструє етапи оформлення замовлення. При виконанні цього варіанту використання користувач додає товари до кошика і відкриває кошик. Після цьогосистема відображає форму для заповлнення інформації для замовлення.

Користувач заповнює форму і натискає кнопку "Підтвердити замовлення". Система перевіряє правильність заповнення форми.

Рисунок 2.1 – Діаграма послідовності «Оформлення замовлення на сайті магазину»

Діаграма послідовності «Розрахунок вартості замовлення» демонструє етапи розрахунку вартості замовлення(рис. 2.2). При виконанні цього варіанту

використання користувач переглядає список товарів у кошику та обирає спосіб доставки. Система розраховує вартість замовлення, враховуючи вартість товарів та вартість доставки. В самому кінці користувач переглядає вартість замовлення та підтверджує його оформлення.

Рисунок 2.2 – Діаграма послідовності «Розрахунок вартості замовлення»

Діаграма послідовності «Пошук товару в системі» демонструє етапи пошуку товару(рис. 2.3). При виконанні цього варіанту використання користувач вводить назву товару у поле пошуку. Система відображає список товарів, що відповідають запиту користувача. Користувач вибирає потрібний товар та переглядає інформацію про нього.[1]

Рисунок 2.3 – Діаграма «Пошук товару в системі»

Діаграма варіантів використання системи Інтернет-магазин, яка була створена за допомогою UML в інструментальному середовищі UML online [2][(рис. 2.4).

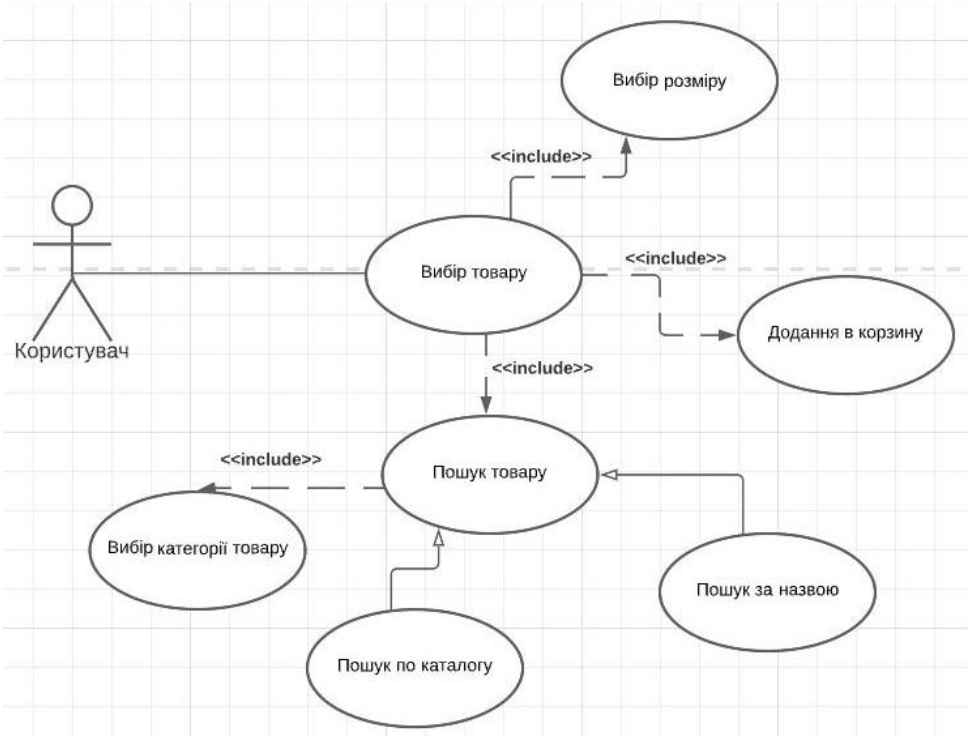

Рисунок 2.4 – Діаграма варіантів використання системи вебзастосуноку

Розроблена діаграма використання включає наступних акторів:

- **Користувач:** Користувач може оформити замовлення на сайті магазину.

Таблиця 2.1 – Сценарій оформлення замовлення на сайті магазину.

Діючі особи	Користувач
Мета	Оформлення замовлення на сайті магазину.
1) Користувач додає товари до кошика. 2) Користувач відкриває кошик і перевіряє товари. 3) Користувач натискає кнопку "Оформити замовлення". 4) Система відображає форму для заповнення інформації про замовлення (ім'я, адреса доставки, спосіб оплати і т.д.). 5) Користувач заповнює форму і натискає кнопку "Підтвердити замовлення". 6) Система перевіряє правильність заповнення форми.	

Таблиця 2.2 – Сценарій розрахунку вартості замовлення

Діючі особи	Користувач
Мета	Користувач може дізнатися вартість замовлення, вибравши товари та обравши спосіб доставки.
1) Користувач переглядає список товарів у кошику та обирає спосіб доставки. 2) Система розраховує вартість замовлення, враховуючи вартість товарів та вартість доставки. 3) Користувач переглядає вартість замовлення та підтверджує його оформлення.	

Таблиця 2.5 – Сценарій пошуку товару

Діючі особи	Користувач
Мета	Користувач може шукати товари у системі
1) Користувач вводить назву товару у поле пошуку. 2) Система відображає список товарів, що відповідають запиту користувача. 3) Користувач вибирає потрібний товар та переглядає інформацію про нього.	

2.2 Проєктування системи

При розробці будь-якої складної системи, будь то програмне забезпечення або апаратна конструкція, ключовим етапом є проєктування. Це процес, в ході якого структуруються вимоги та визначається функціональний дизайн системи. Для успішного виконання цього етапу важливо мати ефективні інструменти, які допоможуть визначити та уточнити вимоги, а також зобразити функціональну структуру системи зрозумілим способом. У цьому розділі ми детально розглянемо два такі інструменти - блок-схеми та DFD.

Розробка будь-якої системи починається з уточнення вимог - функціональних, нефункціональних та технічних. Вимоги є основою проєкту та визначають, яким чином система має працювати і які цілі мають бути досягнуті.

У процесі визначення вимог застосовуються різні техніки, такі як спільна робота, інтерв'ю зі зацікавленими сторонами, аналіз вимог і т.д. Результатом цього процесу є визначені і документовані вимоги до системи, які

включають функціональність, продуктивність, надійність, безпеку, сумісність та інші аспекти.

Функціональний дизайн, у свою чергу, визначає, як система буде реалізовувати вимоги. Він включає в себе детальний опис функціональних блоків системи, зв'язків між ними, а також вхідних і вихідних параметрів кожного блоку. Функціональний дизайн може бути зображений у вигляді блок-схем, діаграм потоку даних, дерев рішень тощо. Це допомагає зрозуміти логіку роботи системи та забезпечити її ефективність.

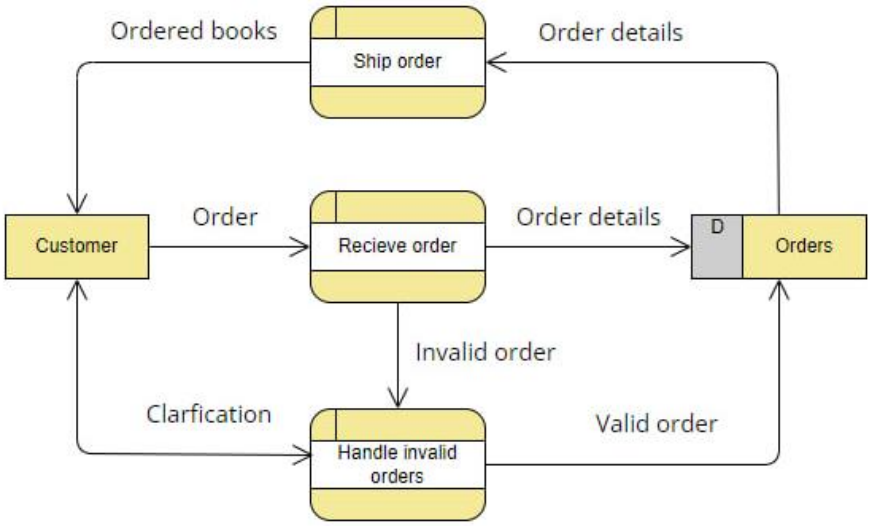

Рисунок 2.5 – зображення DFD

Блок-схеми - це графічний інструмент, що використовується для візуалізації логіки алгоритмів та послідовності дій в програмному коді. Вони дозволяють розбити великі задачі на більш маленькі підзадачі та візуально представити залежності між ними.

Основними елементами блок-схем є:

Процеси: Вони представлені у вигляді прямокутників і вказують на конкретні дії або операції, які виконуються в програмі. Це можуть бути обчислення, умовні оператори, цикли, виклики функцій тощо.

Рішення (уточнення): Вони представлені у вигляді ромбів і використовуються для прийняття рішень в залежності від певних умов. Вони мають різні гілки, що вказують на різні варіанти розвитку подій.

Введення/виведення даних: Це символи, що представляють зчитування або виведення даних з програми. Вони можуть включати введення користувача, читання файлів, виведення результатів тощо.

З'єднувачі: Вони представлені у вигляді стрілок або ліній, які показують потік виконання програми. Вони використовуються для з'єднання елементів блок-схеми та вказівки на наступний крок.

Розробка блок-схем для коду відбувається на етапі проектування програмного забезпечення і має наступні переваги:

Візуалізація логіки: Блок-схеми дозволяють програмістам візуалізувати логіку своїх алгоритмів та структур даних, що полегшує розуміння програми як для них самих, так і для інших членів команди.

Виявлення помилок: Блок-схеми дозволяють виявляти потенційні помилки та некоректні поведінки програми ще до написання фактичного коду. Це дозволяє зекономити час і ресурси на налагодження та виправлення помилок після впровадження.

Розбиття на підзадачі: Блок-схеми допомагають розбити великі задачі на більш маленькі підзадачі, що полегшує розподіл роботи у команді та організацію програмного процесу.

Документація: Блок-схеми можуть використовуватись як документація для програмного коду, де вони послужать поясненням логіки програми, залежностей між компонентами та вхідних/вихідних даних.

Загалом, блок-схеми є потужним інструментом при проектуванні програмного забезпечення. Вони допомагають програмістам краще розуміти та організувати свій код, спрощують процес розробки та полегшують комунікацію в команді розробників.

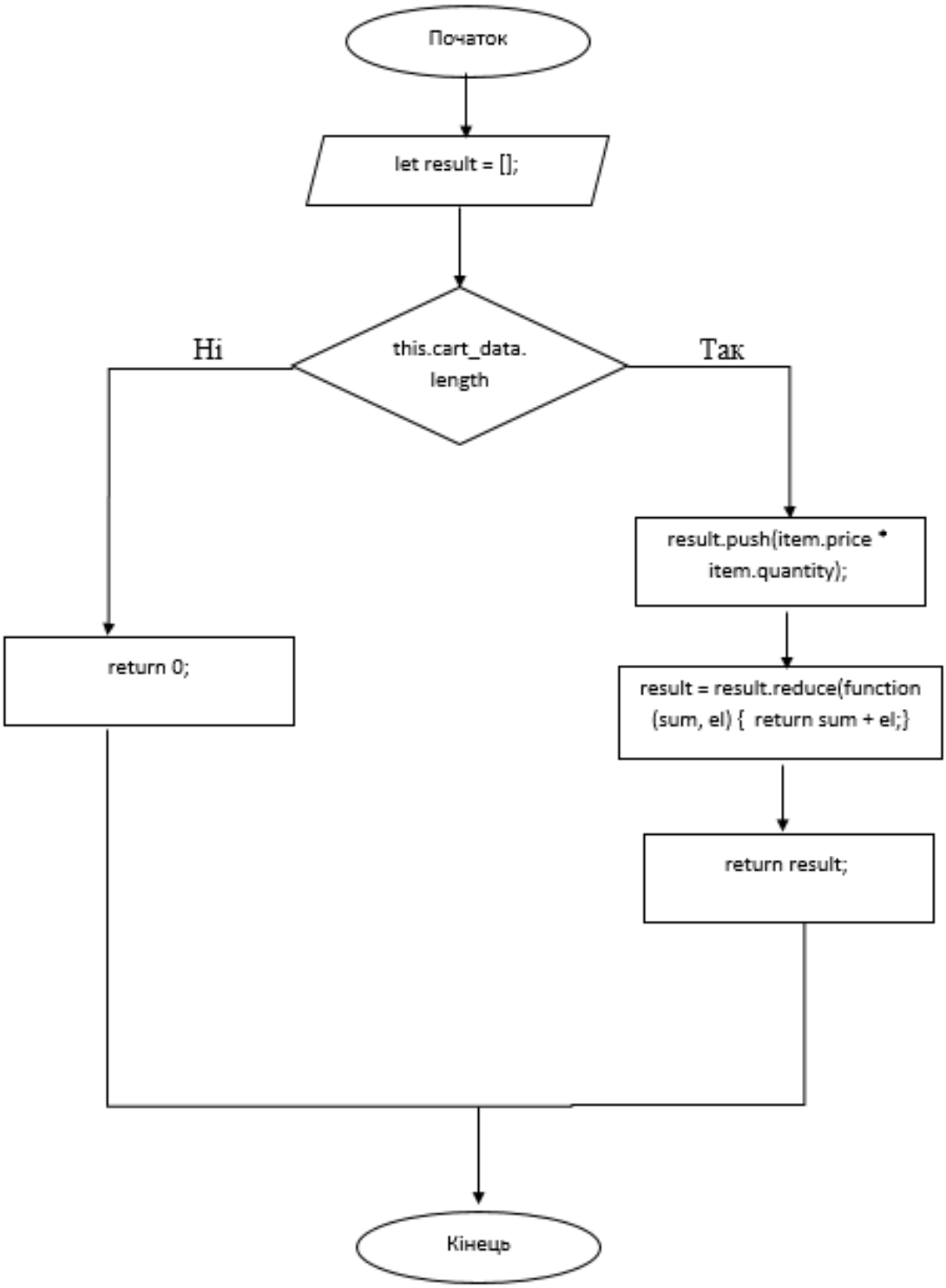

Рисунок 2.6 – блок-схема підрахунку загальної вартості покупки

Цей код представляє функцію cartTotalCost(), яка обчислює загальну вартість товарів у кошику.

Код починається зі створення змінної result, яка буде використовуватися для зберігання проміжних результатів обчислень. Далі, за допомогою

умовного оператора if, перевіряється, чи кошик cart_data має хоча б один елемент. Якщо так, виконується цикл for, який проходиться по кожному елементу кошика.

У циклі кожному елементу присвоюється вартість item.price * item.quantity, яка обчислюється шляхом множення ціни товару на його кількість. Отримані результати додаються до масиву result за допомогою методу push().

Після циклу за допомогою методу reduce() всі елементи масиву result підсумовуються, що дає загальну вартість товарів у кошику. Підсумований результат присвоюється змінній result.

У кінці функції, якщо кошик порожній (не має жодного елемента), повертається значення 0. В іншому випадку, повертається обчислена загальна вартість товарів.

Ця функція дозволяє зручно обчислити вартість товарів у кошику, враховуючи їх ціни та кількість.

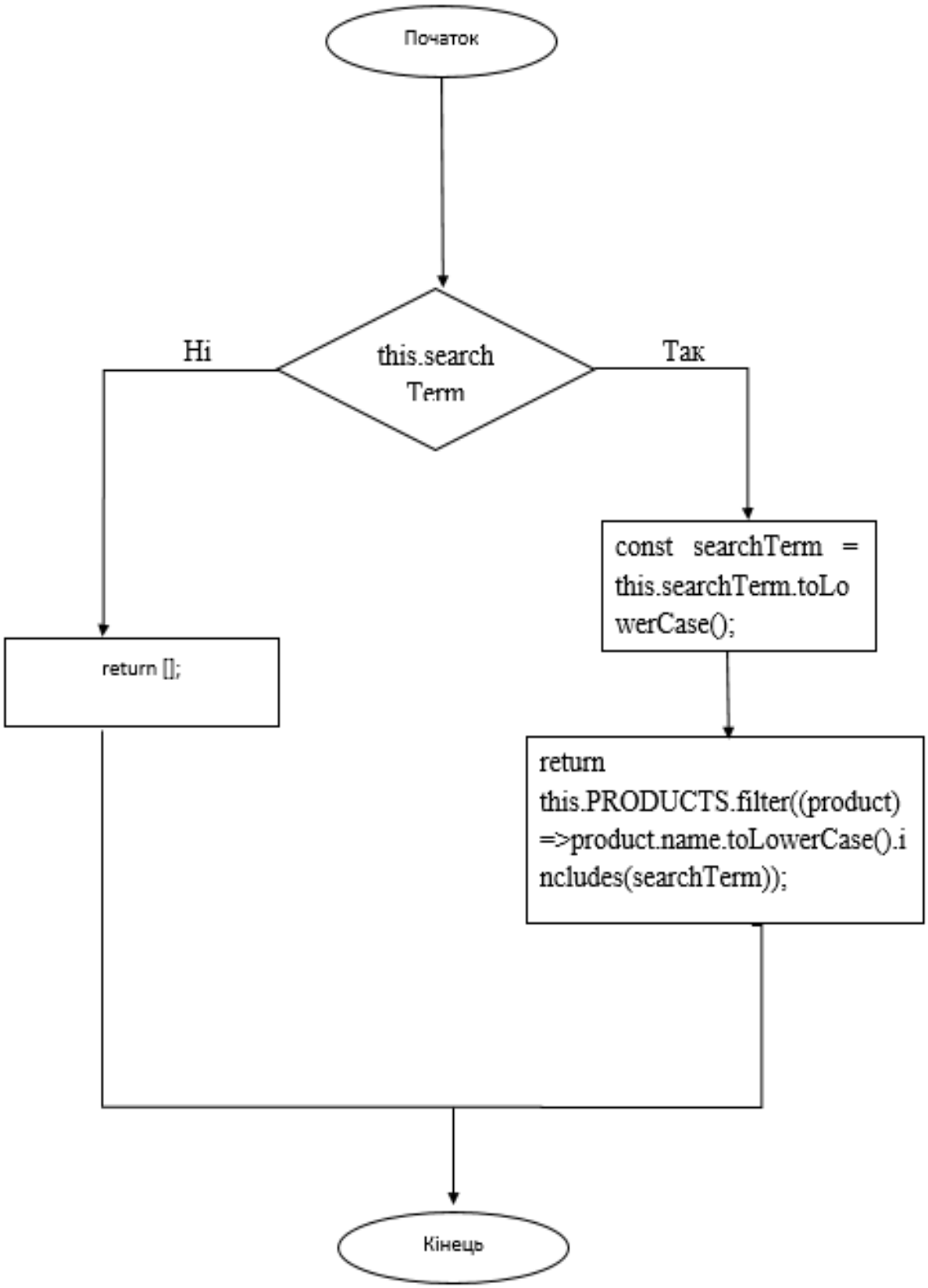

Рисунок 2.7 – блок-схема отримання історії пошуку

Функція розпочинається з перевірки, чи встановлено значення для searchTerm. Це поле використовується для введення пошукового запиту. Якщо searchTerm має значення, код продовжує виконання.

Далі, створюється локальна змінна searchTerm, до якої присвоюється значення this.searchTerm.toLowerCase(). Це значення перетворюється у нижній регістр за допомогою методу toLowerCase(). Це робиться для того, щоб забезпечити регістронезалежний пошук.

Далі, використовується метод filter(), щоб фільтрувати елементи з масиву this.PRODUCTS (припустимо, що this.PRODUCTS містить список продуктів). Кожен елемент масиву this.PRODUCTS перевіряється на умову product.name.toLowerCase().includes(searchTerm). Ця умова перевіряє, чи входить searchTerm в назву продукту product.name в нижньому регістрі. Якщо так, продукт додається до результату.

На кінці функції, якщо searchTerm не було встановлено (немає значення), функція повертає порожній масив [], оскільки немає жодних схожих полів пошуку для виконання.

Отже, ця функція шукає схожі поля пошуку в списку продуктів на основі введеного пошукового запиту searchTerm і повертає відповідні результати у вигляді масиву продуктів.

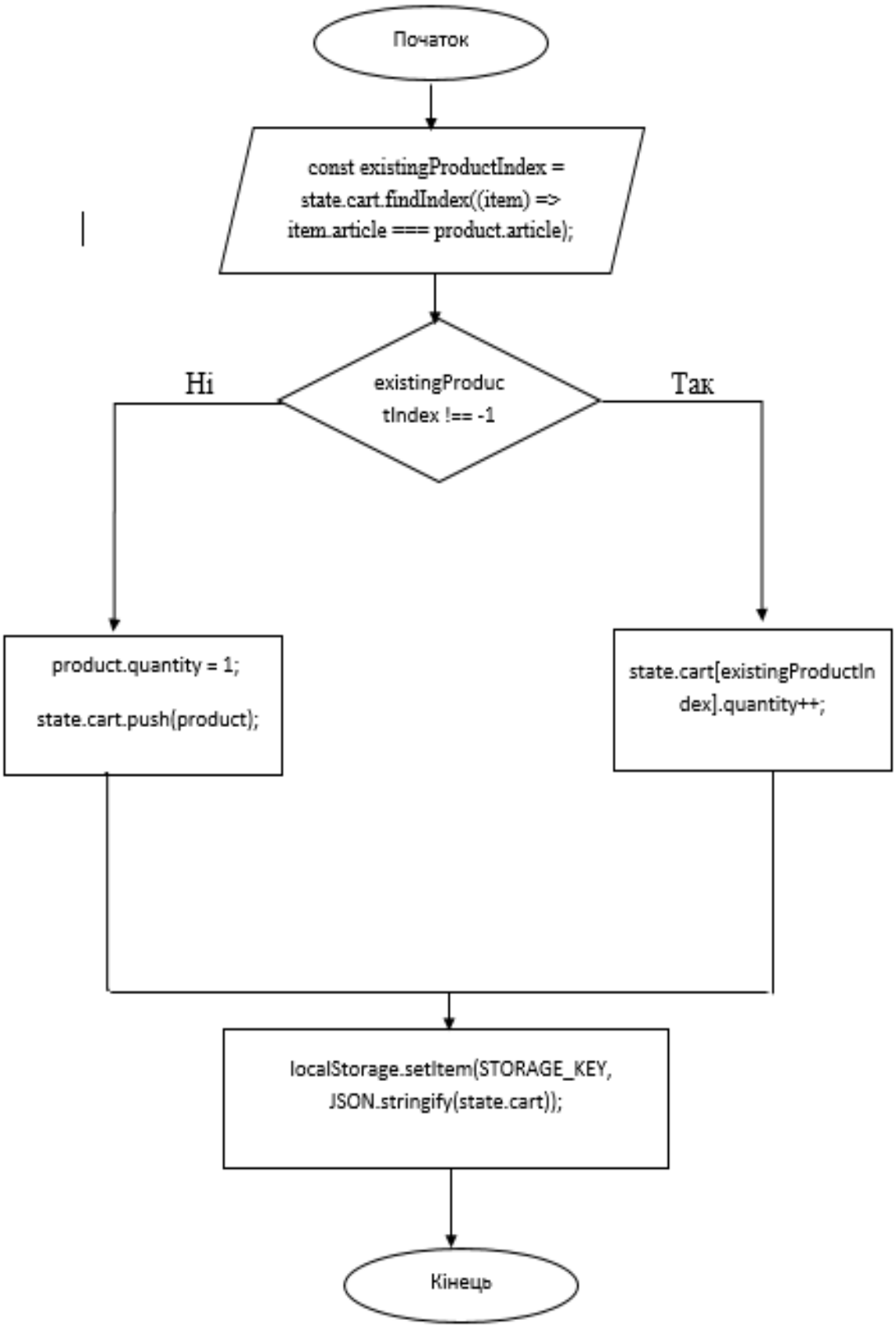

Рисунок 2.8 – блок-схема мутації для зберігання товарів у кошик

Ця мутація виконує наступні дії:

Спочатку, використовуючи метод findIndex(), шукаємо індекс вже існуючого продукту в state.cart, який має такий самий артикул (article) як переданий продукт product. Метод findIndex() шукає перший елемент, який задовольняє умову, передану у функцію зворотного виклику. У нашому випадку, умовою є item.article === product.article. Якщо знайдено вже існуючий продукт, індекс цього продукту зберігається в змінній existingProductIndex. Якщо не знайдено, existingProductIndex буде мати значення -1.

Далі, перевіряється, чи був знайдений вже існуючий продукт (тобто existingProductIndex не дорівнює -1). Якщо так, збільшується кількість продукту в кошику за допомогою оператора ++. Це означає, що кількість продукту збільшується на 1.

Якщо не знайдено вже існуючий продукт, створюється поле quantity для переданого продукту product і присвоюється значення 1. Після цього, цей продукт додається до state.cart за допомогою методу push().

Нарешті, оновлений state.cart зберігається в локальному сховищі (localStorage) за допомогою методу setItem(). Для зберігання використовується ключ STORAGE_KEY, і дані конвертуються у формат JSON за допомогою JSON.stringify().

Отже, ця мутація додає або збільшує кількість товару в кошику, залежно від того, чи вже існує в кошику товар з таким самим артикулом.

2.3 Проєктування інтерфейсів користувача

Мати макет сайту – це дуже важливо для будь-якої веб розробки, оскільки він є основою для розробки та виконання проєкту. Макет сайту дозволяє дизайнеру візуалізувати ідеї та концепції веб сайту та передати їх розробникам у вигляді готового проєкту.

Наявність макету сприяє ефективній комунікації між командами дизайнерів, розробників та клієнтів, що дозволяє уникнути непорозумінь та помилок у процесі розробки. Також, макет сайту є важливим етапом для оцінки обсягу робіт та затрат на розробку проєкту.

Крім того, макет сайту дозволяє оцінити зовнішній вигляд та функціональність веб сайту ще до того, як розпочнеться розробка. Це дозволяє виявити та виправити можливі проблеми та недоліки ще на ранніх етапах проєкту, що зменшує ризик помилок та забезпечує успішне завершення проєкту.

Вебзастосунок як правило має певні незмінні елементи розмітки, такі як верхній навігаційний бар (Рисунок 2.9) та підвал (Рисунок 2.10).

Рисунок 2.9 – Дизайн верхнього навігаційного бару

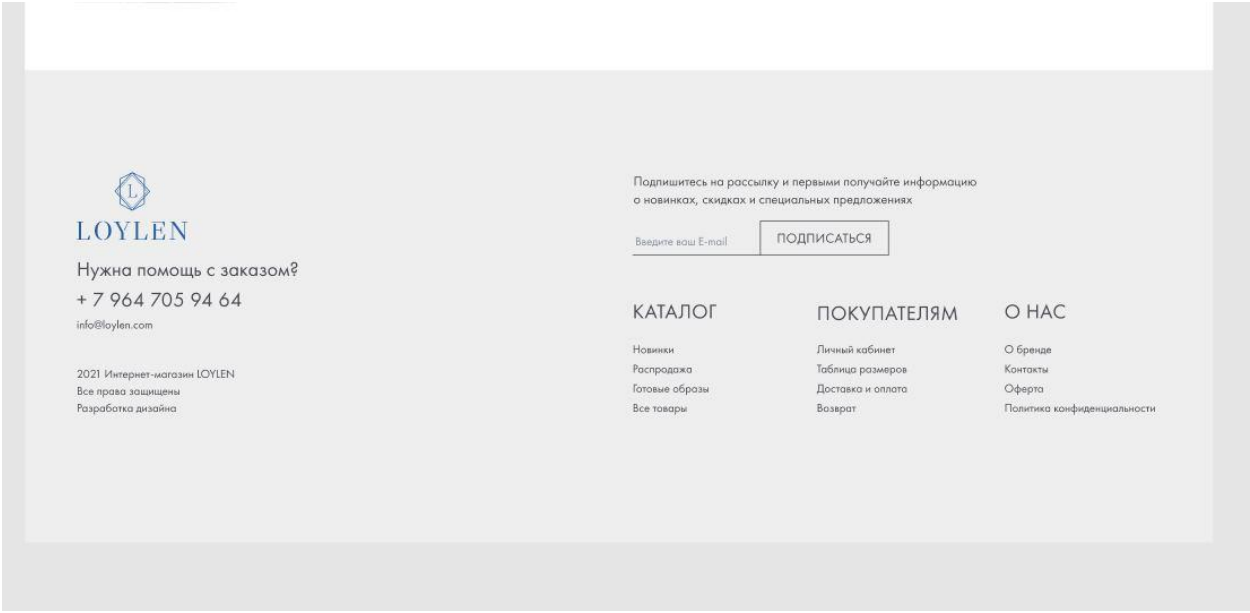

Рисунок 2.10 – Дизайн підвалу

Головна сторінка має зручний інтерфейс, користувач може переглянути частину каталогу та завдяки зручному навігаційному бару може перейти на інші сторінки.

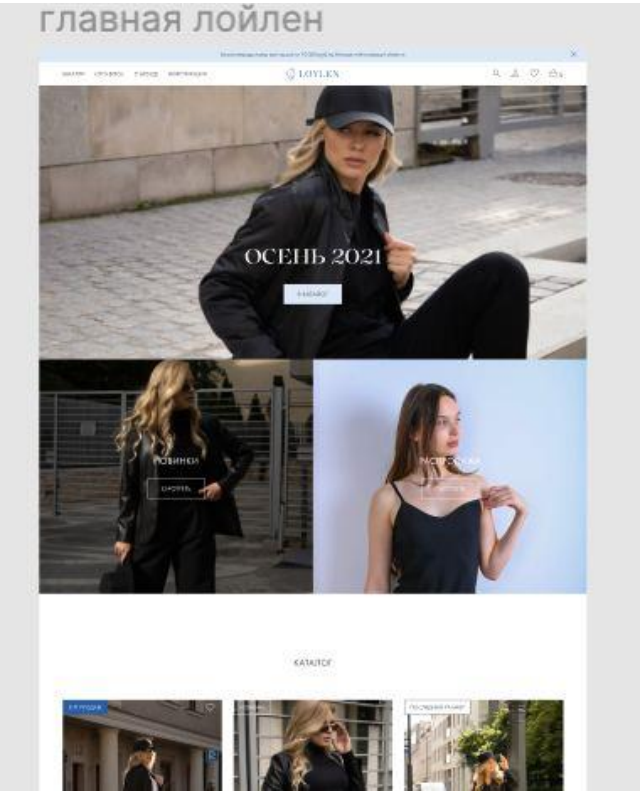

Рисунок 2.11 – Дизайн домашньої сторінки

Знизу розміщена секція з кращими продуктами, яка включає зображення товарів, назву та ціну, щоб забезпечити легкий доступ до найбільш популярних товарів.

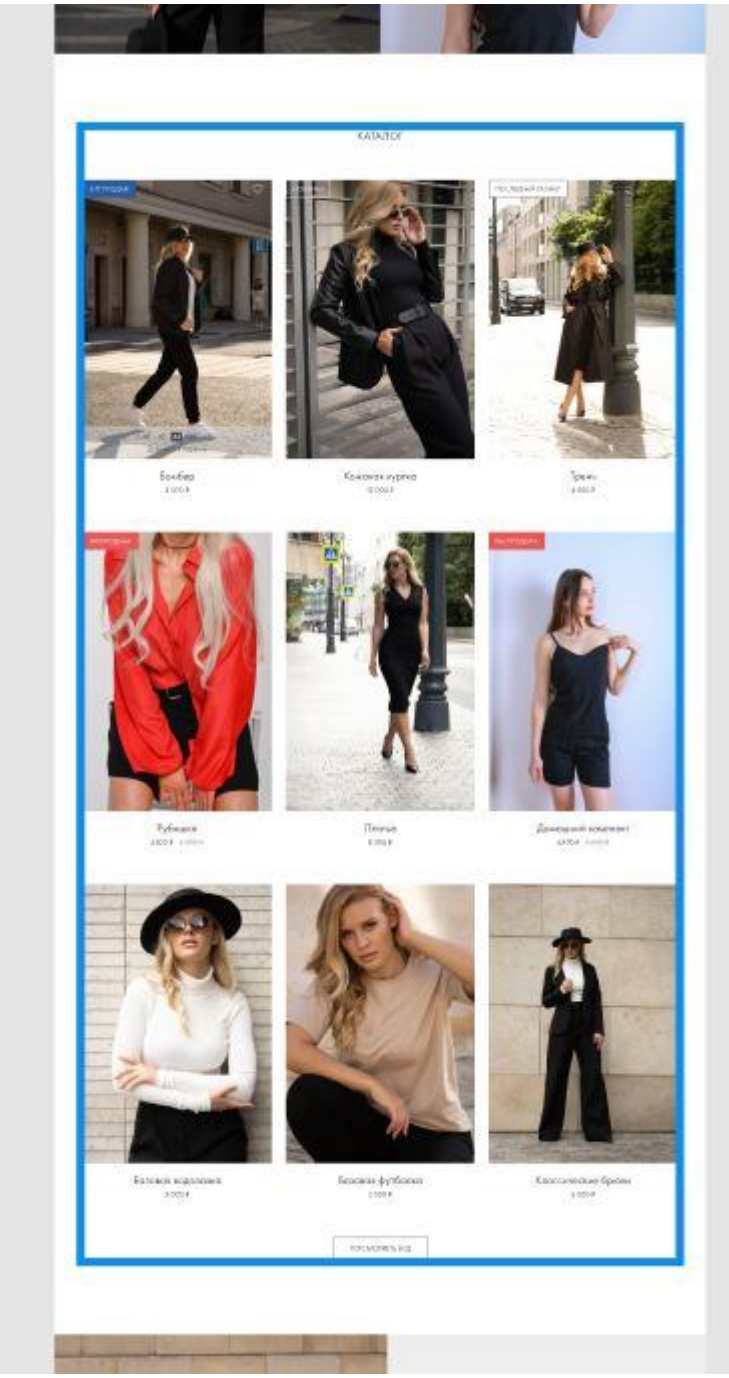

Рисунок 2.12 – Дизайн головної сторінки

Даний дизайн головної сторінки відображається у вигляді макету в Figma. Сторінка містить елементи, які зазвичай зустрічаються на головних сторінках інтернет-магазинів брендової одягу. На верхній частині розміщений логотип магазину та основне меню. Далі, на сторінці відображаються банери з пропозиціями та акціями магазину, які мають яскравий дизайн та привабливий вигляд для користувачів.

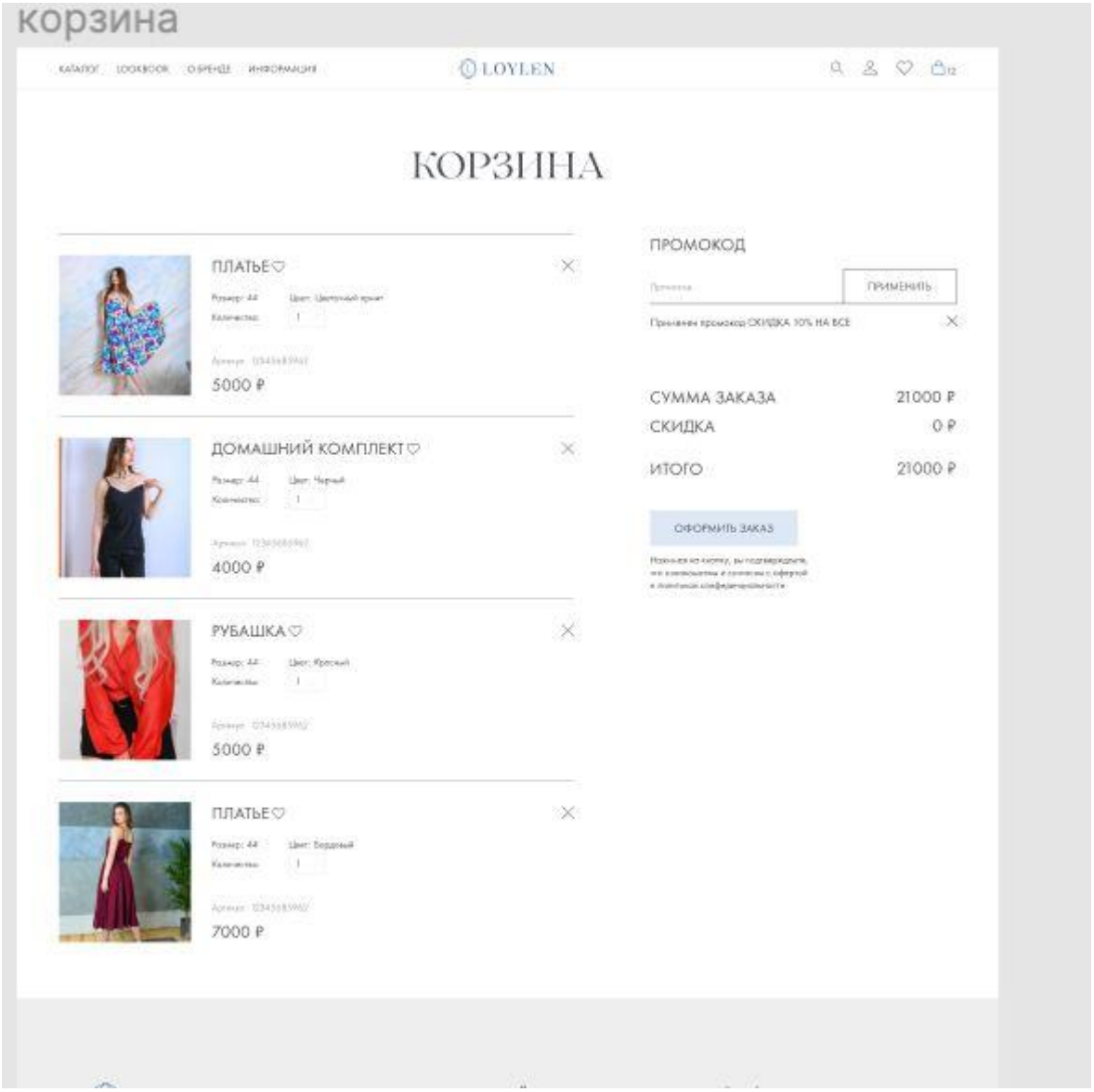

Рисунок 2.13 – Дизайн корзини

Дизайн корзини відображає вибрані товари та інформацію про них, таку як загальна сума замовлення та кількість товарів. У верхній частині корзини розташована назва "Корзина" і кнопка "Оформити замовлення". Під цією інформацією знаходиться список замовлених товарів, кожен з яких представлений зображенням, назвою, кількістю та ціною. При наведенні курсору на елемент замовлення з'являється кнопка видалення товару з корзини. У нижній частині корзини розташована загальна сума замовлення та кнопка "Оформити замовлення".

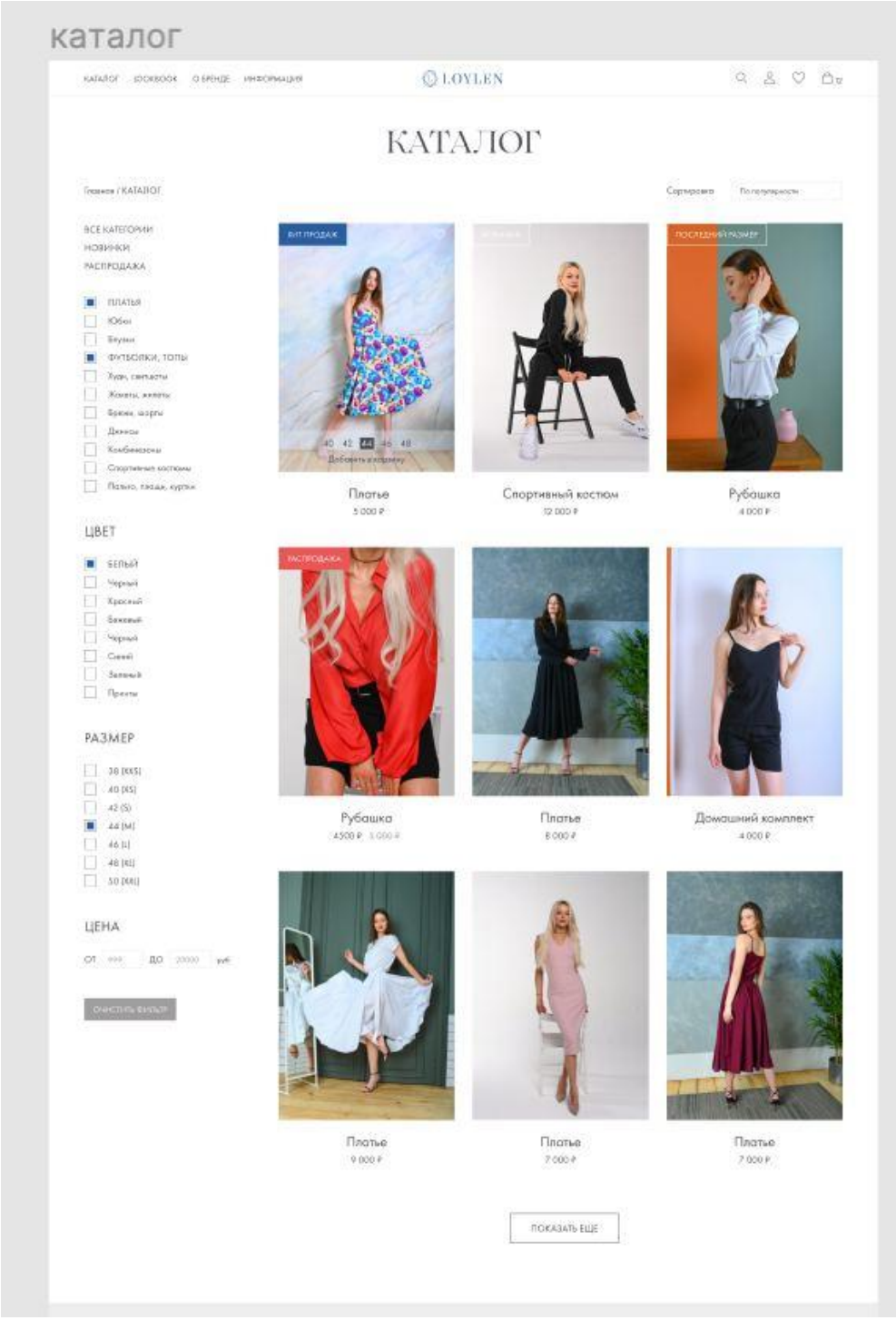

Рисунок 2.14 – Дизайн каталогу

Дизайн каталогу містить список товарів з їх зображеннями, назвами та цінами. В головній частині сторінки знаходиться сітка з кількома категоріями товарів, які можна відобразити на сторінці. При виборі категорії, з'являються відповідні товари. При наведенні курсору на товар з'являється кнопка "Додати в корзину". У верхній частині сторінки розташована навігаційна

панель зі списком категорій товарів, кнопкою "Корзина" та полем пошуку товарів.

Висновки до розділу 2

В межах розділу було проаналізовано можливі методики розробки певних проєктних рішень для забезпечення повного та успішного виконання функціональних та нефункціональних вимог до ПЗ

Також були проаналізовані наявні на ринку технології і прийнято рішення про те, які саме з них необхідно використовувати для успішної та безпроблемної реалізації поставленої задачі.

На основі вимог до ПЗ сконструйовано діаграми та сценарії до системи, що розробляється, а саме сценарії використання у різних формах, діаграму варіантів використання, діаграму класів та діаграму послідовності.

Виходячи з вимог до ПЗ та структури застосунку було представлено макети інтерфейсів публічної частини вебзастосунку, а саме домашньої сторінки, сторінки каталогу, сторінки корзини.

3 ПРОЄКТУВАННЯ ВЕБЗАСТОСУНКУ ПРОДАЖУ ОДЯГУ НА ОСНОВІ VUE

3.1 Ієрархія класів системи

В наш час багато компаній використовують інтернет-магазини для продажу своїх товарів та послуг. У процесі розробки таких систем використовуються класи для реалізації контролерів та моделей. В цій кваліфікаційній роботі розроблено інтернет-магазин для продажу одягу, в якому також використовуються класи. Наприклад, клас Customer представляє покупця, клас Product - товари, які продаються на сайті, а клас Order - замовлення, яке складається зі списку елементів замовлення та інформації про покупця та спосіб оплати. Класи є одним із важливих елементів при розробці великих програмних проєктів, тому розуміння їхньої ролі та призначення є важливим для успішної розробки програмного забезпечення.

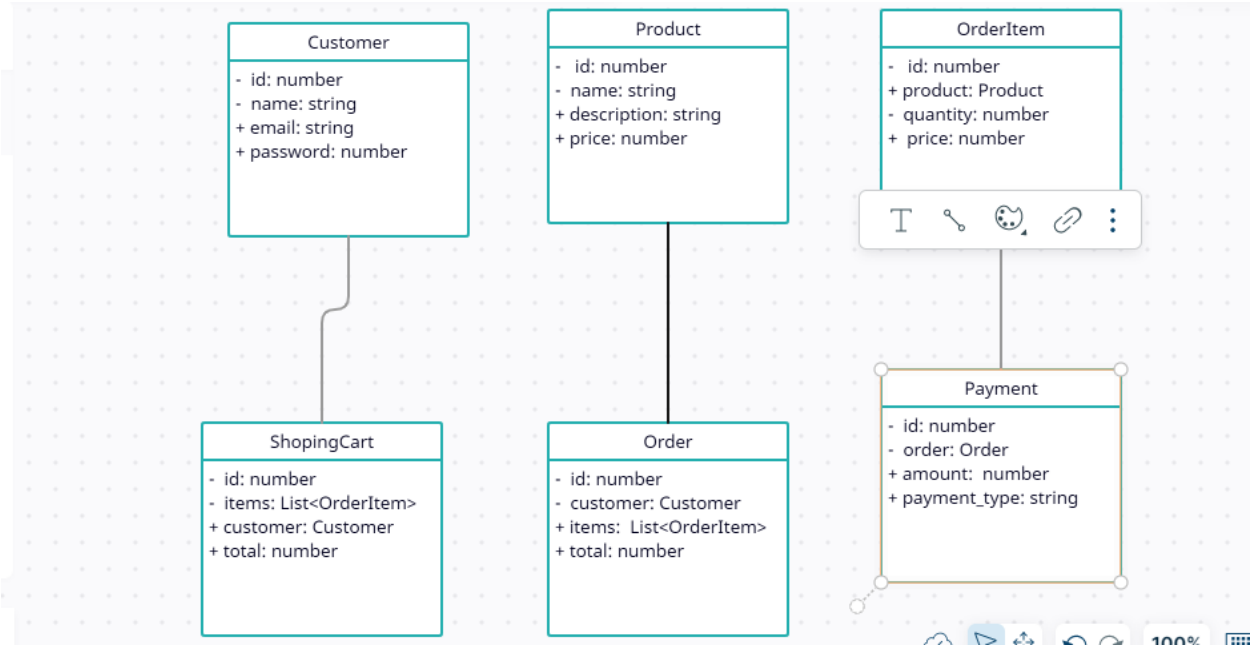

Рисунок 3.1 – Діаграма класів

В системі використовуються класи для реалізації контролерів та моделей.

Таблиця 2.1 – Опис класів системи

Назва класу	Призначення класу
Customer	Customer представляє покупця, який здійснює покупки на сайті магазину. У класі містяться властивості, такі як ім'я, пошту, айди та пароль.
Product	Product редставляє товари, які продаються на сайті магазину. У класі містяться властивості, такі як назва, опис, ціна.
OrderItem	OrderItem представляє один елемент замовлення, який містить інформацію про товар і кількість, що була замовлена. У класі містяться властивості, такі як товар, кількість, ціна.
ShopingCart	ShopingCart представляє кошик покупок, який містить список елементів замовлення.
Order	Order представляє замовлення, яке складається зі списку елементів замовлення та інформації про покупця та спосіб оплати.
Payment	Payment містить інформацію про сплату за замовлення, таку як ідентифікатор платежу, суму оплати, дату оплати та статус оплати.

3.2 Вибір технологій та мов програмування

В процесі розробки веб-додатку було необхідно вибрати набір технологій та мов програмування, які найбільше підходять для вирішення задач проєкту та забезпечують ефективність та швидкість розробки.

Одним із найбільш важливих елементів веб-додатку є стан додатку, який повинен бути збереженим та доступним на різних рівнях додатку. Для вирішення цієї задачі було вирішено використовувати Vuex - бібліотеку управління станом додатку. Вона дозволяє зберігати стан додатку в централізованому місці, що забезпечує зручний доступ до стану додатку з будь-якого компонента.

Vue.js було обрано як основний фреймворк для розробки веб-додатку. Це дозволяє забезпечити швидку та ефективну розробку завдяки легкій вивченості та зручному синтаксису. Також, Vue.js має велику та активну спільноту, що забезпечує швидке вирішення проблем та підтримку фреймворку.

Для розмітки веб сторінок було вирішено використовувати Pug - шаблонізатор HTML. Pug дозволяє скоротити кількість коду та зробити його більш зрозумілим та читабельним. Крім того, Pug підтримує наслідування шаблонів, що дозволяє уникнути дублювання коду та зробити розмітку більш організованою та структурованою.

Щодо стилізації веб сторінок, було вирішено використовувати SCSS - мову CSS з препроцесором. SCSS забезпечує зручний та ефективний спосіб стилізації веб сторінок за допомогою функцій, змінних, міксінів та інших конструкцій, що значно полегшує розробку та підтримку коду. Крім того, SCSS дозволяє створювати багатошарові стилі, що забезпечує більшу гнучкість та повторне використання коду.

3.3 Написання usecases

У розділі "Написання Use Cases" проводиться детальна розробка варіантів використання системи. Use Cases (варіанти використання) є інструментом для опису поведінки системи з точки зору користувача або іншого актора. Вони описують взаємодію між акторами і системою, визначають ціль використання та послідовність кроків, необхідних для досягнення цієї цілі.

Основна мета написання Use Cases полягає у виявленні та документуванні функціональних вимог до системи. Вони дозволяють зрозуміти, як система буде використовуватись різними акторами, що вони очікують від системи та як система повинна взаємодіяти з ними.

Під час написання Use Cases використовується стандартна нотація, яка включає заголовок варіанту використання, опис акторів, опис сценарію використання та приклади вхідних даних та результатів. Крім того, варіанти використання можуть мати додаткові розділи, які деталізують додаткові аспекти поведінки системи.

Написання Use Cases дозволяє уточнити функціональні вимоги до системи, зрозуміти потреби та очікування користувачів та інших акторів, а також виявити потенційні проблеми та невідповідності у вимогах. Вони є важливим інструментом комунікації між розробниками, замовниками та іншими зацікавленими сторонами, що допомагає уникнути непорозумінь та забезпечує взаєморозуміння щодо очікувань щодо функціональності системи.

Результатом розробки Use Cases є документ, який може використовуватись як основа для подальшого проектування системи, розробки тестових сценаріїв та імплементації програмного забезпечення.

Use Case: Оформлення замовлення

Опис: Користувач може оформити замовлення на сайті магазину.

Актори:

Користувач: Основний користувач системи, який бажає зробити покупку.

Передумови: користувач додав товари до свого кошика.

Головний успішний сценарій:

1. Користувач відкриває свій кошик.

2. Система відображає список товарів, які користувач додав до кошика, разом з їх кількістю та ціною.Користувач переглядає товари у кошику та перевіряє їх.

3. Користувач натискає кнопку "Оформити замовлення".

4. Система перевіряє, чи всі товари в кошику є в наявності.

5. Система відображає форму для заповнення інформації про замовлення (ім'я, адреса доставки, спосіб оплати тощо).

6. Користувач заповнює форму, вводить необхідні дані та натискає кнопку "Підтвердити замовлення".

7. Система перевіряє коректність введеної інформації та наявність обов'язкових полів.

8. Користувач отримує підтвердження про успішне оформлення замовлення.

Альтернативні сценарії:

1. Якщо товари в кошику відсутні:

Система відображає повідомлення користувачу про відсутність товарів у кошику та пропонує додати товари до кошика.

2. Якщо користувач не заповнив обов'язкові поля у формі:

Система відображає повідомлення про неправильно заповнені поля та нагадує користувачу про необхідність їх заповнення.

Use Case: Пошук товару

Опис: Користувач може шукати товари у системі за допомогою пошукової функції.

Актори:

Користувач: Основний користувач системи, який шукає певний товар.

Передумови:

Користувач знаходиться на сторінці каталогу інтернет-магазину.

Головний успішний сценарій:

1. Користувач вводить назву товару або його характеристики у поле пошуку.
2. Система аналізує введений запит та відображає результати пошуку - список товарів, що відповідають запиту.
3. Користувач переглядає список знайдених товарів та їх характеристики.

Альтернативні сценарії:

1. Якщо товар не знайдений:

Система повідомляє користувача про відсутність результатів пошуку та пропонує спробувати знайти інший товар або використати розширений пошук.

1. Якщо введено неправильну назву товару або характеристику:

Система пропонує перевірити правильність введення та надає рекомендації щодо введення коректних даних.

Use Case: Фільтрація товарів

Опис: Користувач може фільтрувати товари за різними параметрами для зручності пошуку та вибору.

Актори:

Користувач: Основний користувач системи, який хоче вибрати товари за певними критеріями.

Передумови:

Користувач знаходиться на сторінці каталогу або результатів пошуку товарів.

Головний успішний сценарій:

1. Користувач використовує доступні фільтри (наприклад, ціновий діапазон, категорія) для обмеження вибору товарів.

2. Система аплікує вибрані фільтри до списку товарів та відображає відфільтрований результат.
3. Користувач переглядає список товарів, які задовольняють вибрані фільтри.

Альтернативні сценарії:

1. Якщо результати фільтрації не знайдено:

Система повідомляє користувача про відсутність товарів, які задовольняють вибрані фільтри, та пропонує змінити параметри фільтрації.

2. Користувач змінює вибрані фільтри:
3. Система оновлює список товарів з урахуванням нових фільтрів та відображає змінений результат.

Use Case: Реєстрація

Опис: Користувач може зареєструвати свій обліковий запис в системі.

Актори:

Користувач: Основний користувач системи, який бажає створити обліковий запис.

Передумови:

1. Користувач знаходиться на сторінці реєстрації.
2. Головний успішний сценарій:
3. Користувач вводить свої особисті дані (ім'я, прізвище, електронну пошту, пароль тощо) у відповідні поля реєстраційної форми.
4. Користувач натискає кнопку "Зареєструватися" для відправки заповненої форми.

Альтернативні сценарії:

1. Якщо введені дані некоректні або не відповідають вимогам:

Система повідомляє користувача про помилки та пропонує виправити неправильно введені дані.

Висновки до розділу 3

В межах розділу завдяки проведеним дослідженням та розробці, було досягнуто успіху в досягненні поставлених цілей. Були розроблені детальні сценарії використання, що дозволяють користувачам взаємодіяти з системою та виконувати необхідні дії. Також були створені діаграми послідовності, які візуалізують взаємодію між об'єктами та компонентами системи.

Написання usecases дозволило чітко визначити функціональні вимоги до системи та обов'язки користувачів. Цей етап розробки є важливим для забезпечення зрозумілості та відповідності системи потребам користувачів.

Отримані результати роботи мають значний вплив на область дослідження систем. Відповідно до цього, було здійснено покращення щодо продуктивності та ефективності роботи системи, а також поліпшено взаємодію з користувачами.

4 КОДУВАННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

Кодування ПЗ є ключовим етапом у процесі розробки програмних продуктів. Цей етап вимагає перетворення вимог до програмного продукту в код, який може бути розуміним та виконуваним комп'ютером. Кодування включає написання програмного коду, структурування даних, розміщення функцій та модулів, а також реалізацію алгоритмів.

Під час кодування програмного забезпечення розробники використовують різні мови програмування, фреймворки, бібліотеки та інструменти, щоб створити працездатну та ефективну програму. Кодування вимагає не лише технічних навичок, але й творчого мислення, аналітичного мислення та здатності до проблемного мислення.

Основна мета кодування полягає в перетворенні концепцій та вимог в реалізований програмний код. Це включає розробку архітектури програми, вибір підходів до реалізації функціональності, написання ефективного та якісного коду, використання засобів для тестування та відлагодження програмного продукту.

Кодування програмного забезпечення вимагає дотримання кращих практик програмування, таких як читабельність коду, модульність, повторне використання коду, використання коментарів та документації для зрозумілості та підтримки коду в майбутньому.

У даному контексті кодування програмного забезпечення є важливим етапом у створенні веб-застосунків для продажу одягу. Використання відповідних веб-технологій та фреймворків, таких як Vue.js, дозволяє розробникам ефективно та швидко реалізувати бажану функціональність та створити зручний та привабливий інтерфейс для користувачів.

У цьому дослідженні ми детально розглянемо процес кодування програмного забезпечення для веб-застосунків продажу одягу та розглянемо основні аспекти використання Vue.js для досягнення цієї мети. Ми розглянемо

переваги та можливості Vue.js, а також важливі аспекти розробки веб-застосунків з використанням цього фреймворку.

4.1 Робота зі зовнішнім JSON-файлом як джерелом даних

У цій роботі використовується Vue та Vuex для реалізації функціональності зберігання товарів, каталогування в кошику та обраному. Був використаний JSON-файл, який містить дані про товари, як джерело даних для веб-додатку. Це дозволяє працювати з фіксованими даними без необхідності використання бази даних або серверної частини.

Структура JSON-файлу

JSON-файл містить products: масив об'єктів, що представляють товари. Кожен об'єкт містить різні властивості, такі як article, quantity та інші, які відображають характеристики товарів.

Завантаження даних з JSON-файлу:

У коді використовується пакет axios для отримання даних з JSON-файлу. Проводиться запит до ресурсу http://localhost:4000/products, щоб отримати дані про товари. За допомогою мутації Vuex SET_PRODUCTS_TO_STATE зберігаються отримані дані в стані додатку.

Збереження та оновлення даних

Були використані мутації Vuex для зміни стану додатку. Наприклад, мутація SET_CART додає товар до кошика, а мутація SET_FAVORITE додає товар до обраного. LocalStorage використовується для збереження змінених даних у JSON-форматі, щоб вони були доступні після перезавантаження сторінки.

Використання даних у компонентах

Для отримання даних зі стану додатку використовуються геттери Vuex. Наприклад, був використаний this.$store.getters.PRODUCTS для отримання

списку товарів. Це дозволяє легко отримувати дані зі стану та використовувати їх у компонентах для відображення і взаємодії з користувачем.

Загальний результат

Було успішно реалізовано роботу зі зовнішнім JSON-файлом як джерелом даних у вебдодатку, використовуючи Vue та Vuex. Це дає можливість працювати з фіксованими даними та зберігати зміни між сеансами користувача. Реалізація демонструє ефективне використання Vuex для управління станом додатку та забезпечення зручного інтерфейсу для користувача.

4.2 Програмна реалізація

Програмна реалізація вебдодатку, який використовує Vue та Vuex, включає різні компоненти та функціональність, що дозволяють користувачам взаємодіяти з додатком та отримувати необхідну інформацію. Нижче наведено опис основних аспектів програмної реалізації вебдодатку:

1. Структура компонентів

Використовуються компоненти Vue для організації різних частин додатку, таких як головна сторінка, каталог товарів, кошик, обране тощо. Кожен компонент відповідає за свою власну частину інтерфейсу та взаємодії з користувачем.

2. Використання Vuex для управління станом

Використання Vuex для зберігання стану додатку. Стан включає дані про товари, кошик та обрані елементи. За допомогою мутацій змінюється стан додатку, додаючи товари до кошика, видаляючи їх з кошика, додаючи до обраного тощо. Використання Vuex спрощує керування станом додатку та забезпечує односторонню потокову модель даних.

3. Взаємодія з сервером через Axios

Для отримання даних про товари було використано бібліотеку Axios для здійснення HTTP-запитів до сервера. Запити включають отримання списку

товарів з сервера, додавання товарів до кошика та обраного, видалення товарів тощо. Це дозволяє оновлювати дані тдодатку в реальному часі та забезпечує синхронізацію з сервером.

4. Збереження даних у локальному сховищі

Використання localStorage для збереження змінених даних (кошик, обране) між сеансами користувача. При завантаженні додатку перевіряється наявність збережених даних у localStorage та відновлення їх у відповідні змінні стану.

5. Диспетчеризація дій за допомогою actions

Actions було використано для спрощення виклику мутацій та взаємодії з сервером. Дії дозволяють викликати мутації та здійснювати асинхронні запити до сервера, такі як отримання списку товарів або додавання товару до кошика.

6. Використання геттерів для отримання даних

Геттери наявні для отримання даних зі стану додатку. Вони дозволяють отримати актуальні дані про товари, кошик та обрані елементи та передати їх у відповідні компоненти для відображення.

4.3 Керівництво адміністратора

Процес завантаження даних проходить за допомогою JSON-сервера під час виконання команди json-server --watch db.json --port 4000. JSON-сервер - це простий сервер, який надає можливість створювати RESTful API для роботи з файлами у форматі JSON.

Команда json-server --watch db.json --port 4000 запускає JSON-сервер та налаштовує його на використання файлу db.json як джерела даних. Сервер слухатиме на порту 4000.

Процес завантаження даних за допомогою JSON-сервера включає наступні кроки:

1. Завантаження та встановлення Node.js:

Перед початком роботи треба переконатися, що на комп'ютері встановлено Node.js. Можна завантажити його з офіційного веб-сайту Node.js та встановити на операційну систему.

2. Встановлення JSON-сервера:

З використанням пакетного менеджера npm (який входить до складу Node.js), встановлюємо JSON-сервер, виконавши команду npm install -g json-server в командному рядку. Це дозволить глобально використовувати JSON-сервер з будь-якого місця в системі.

3. Підготовка файлу з даними (db.json):

Перед запуском сервера підготовлено файл db.json, в якому міститимуться дані, які будуть завантажені. Формат файлу повинен відповідати специфікації JSON.

Запуск JSON-сервера:

Виконавши команду json-server --watch db.json --port 4000 в командному рядку. Ця команда запустить сервер та підключить файл db.json як джерело даних. Сервер буде слухати на порту 4000.

4. Перевірка роботи сервера:

Після запуску сервера було перевірено його роботу, відкривши веб-браузер і перейшовши за адресою http://localhost:4000.

Цей процес дозволяє завантажити дані з файлу db.json за допомогою JSON-сервера і надає можливість працювати з ними за допомогою RESTful API.

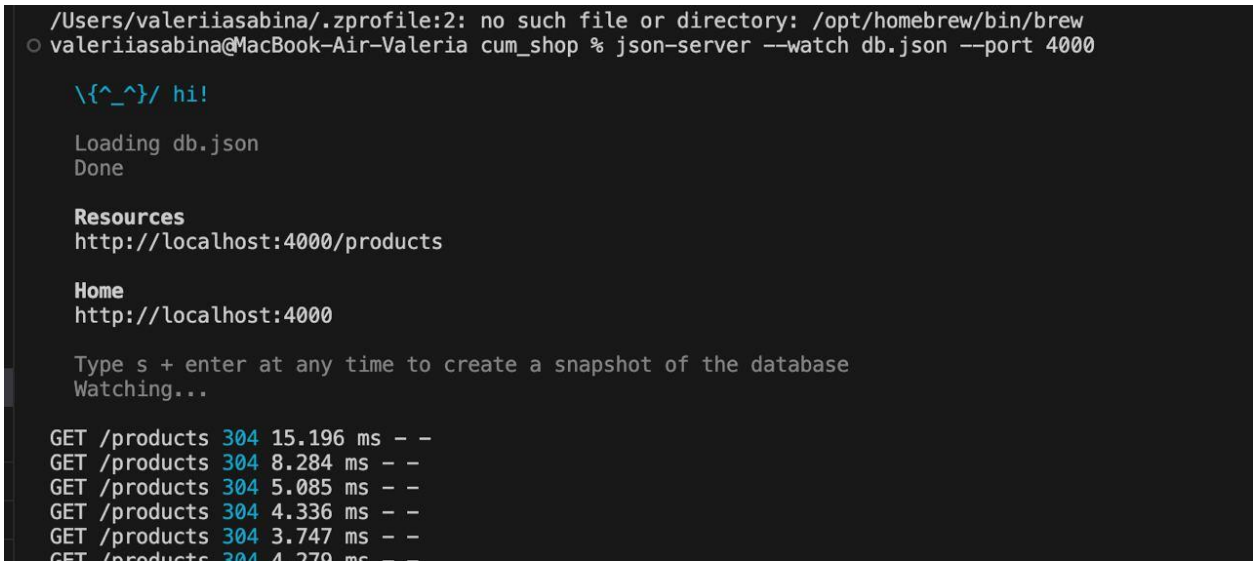

Рисунок 4.1 – Запус команди json-server --watch db.json --port 4000

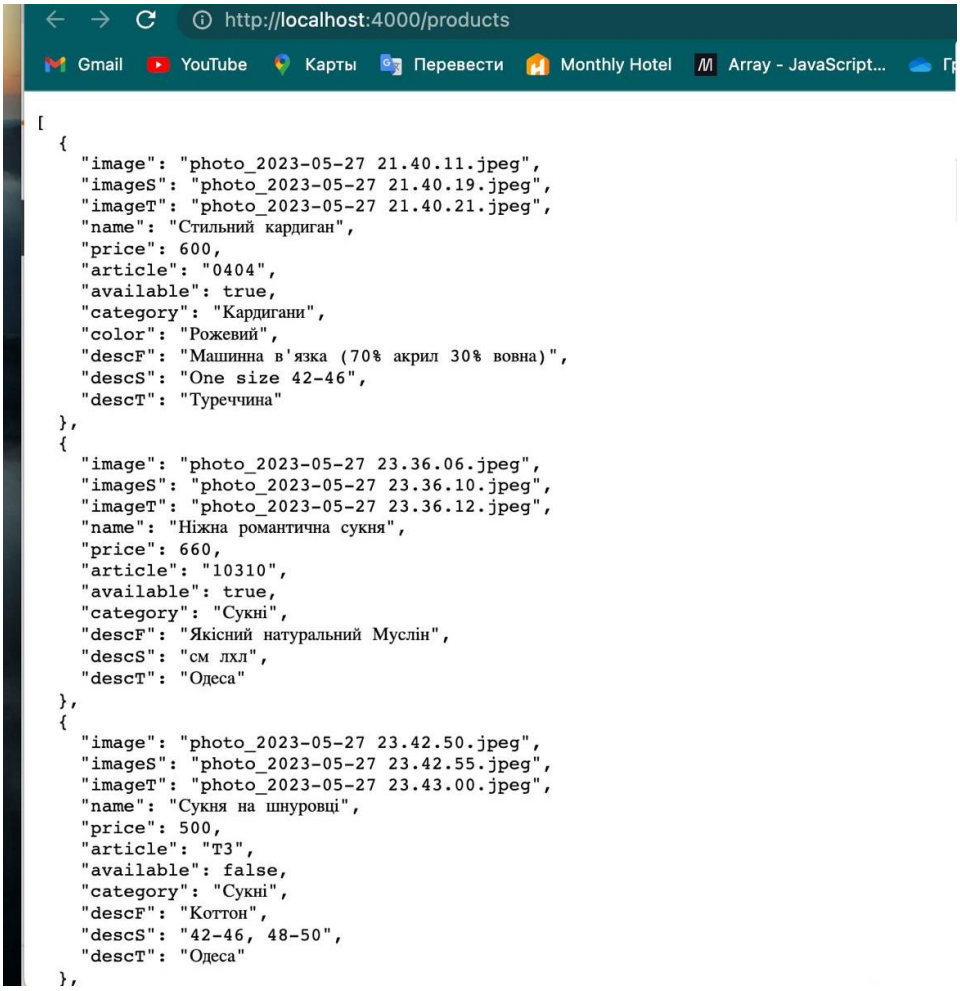

Рисунок 4.2 – Перевірка наявності даних

4.4 Керівництво користувача

Після успішного завершення процесу завантаження даних, можна вже використовувати вебзастосунок в повному обсязі. При першому візиту вебзастосунку, показується головна сторінка (Рисунок 4.3), де користувач може переглянути деякі товари з каталогу та додати їх до кошика.

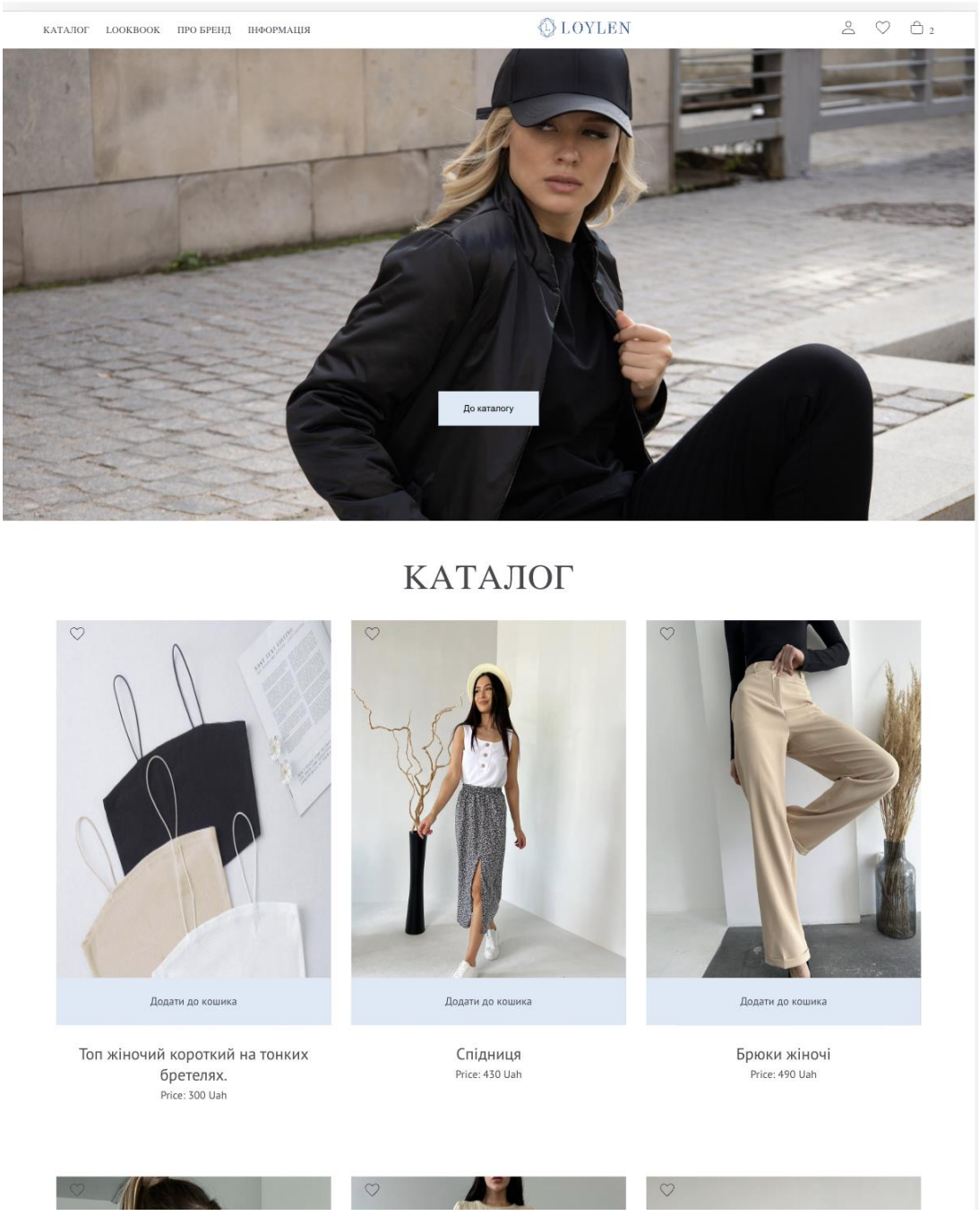

Рисунок 4.3 – Інтерфейс головної сторінки

Також у вебзастосунку мається сторінка про бренд (Рисунок 4.4). Користувач має можливість скористатись фільтрами, та вивести товари по категоріям, вивести товар по мінімальній та максимальній вартості та відсортувати від максимальної вартості до мінімальної та від мінімальної до максимальної(Рисунок 4.5).

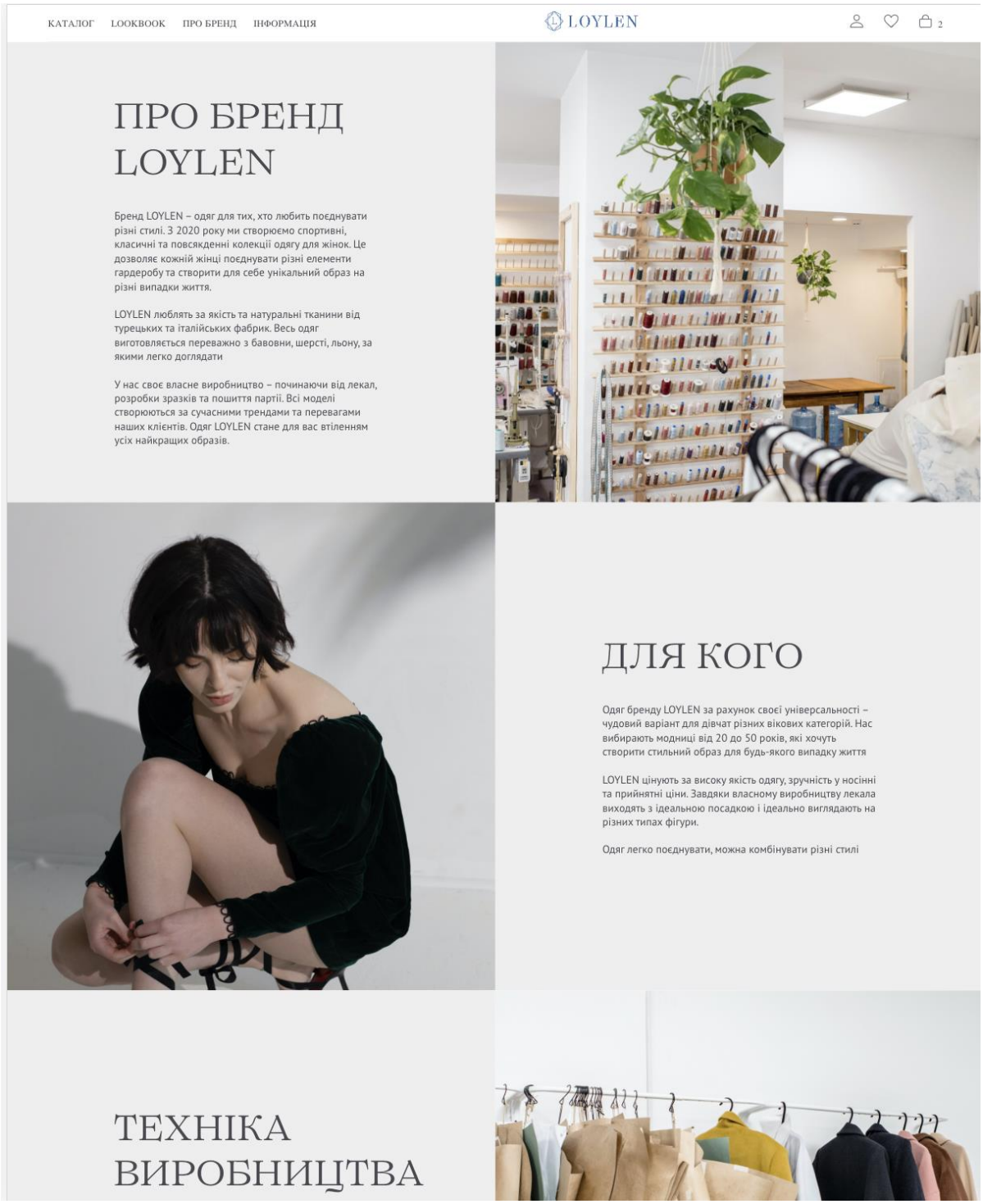

Рисунок 4.4 – Інтерфейс сторінки про бренд

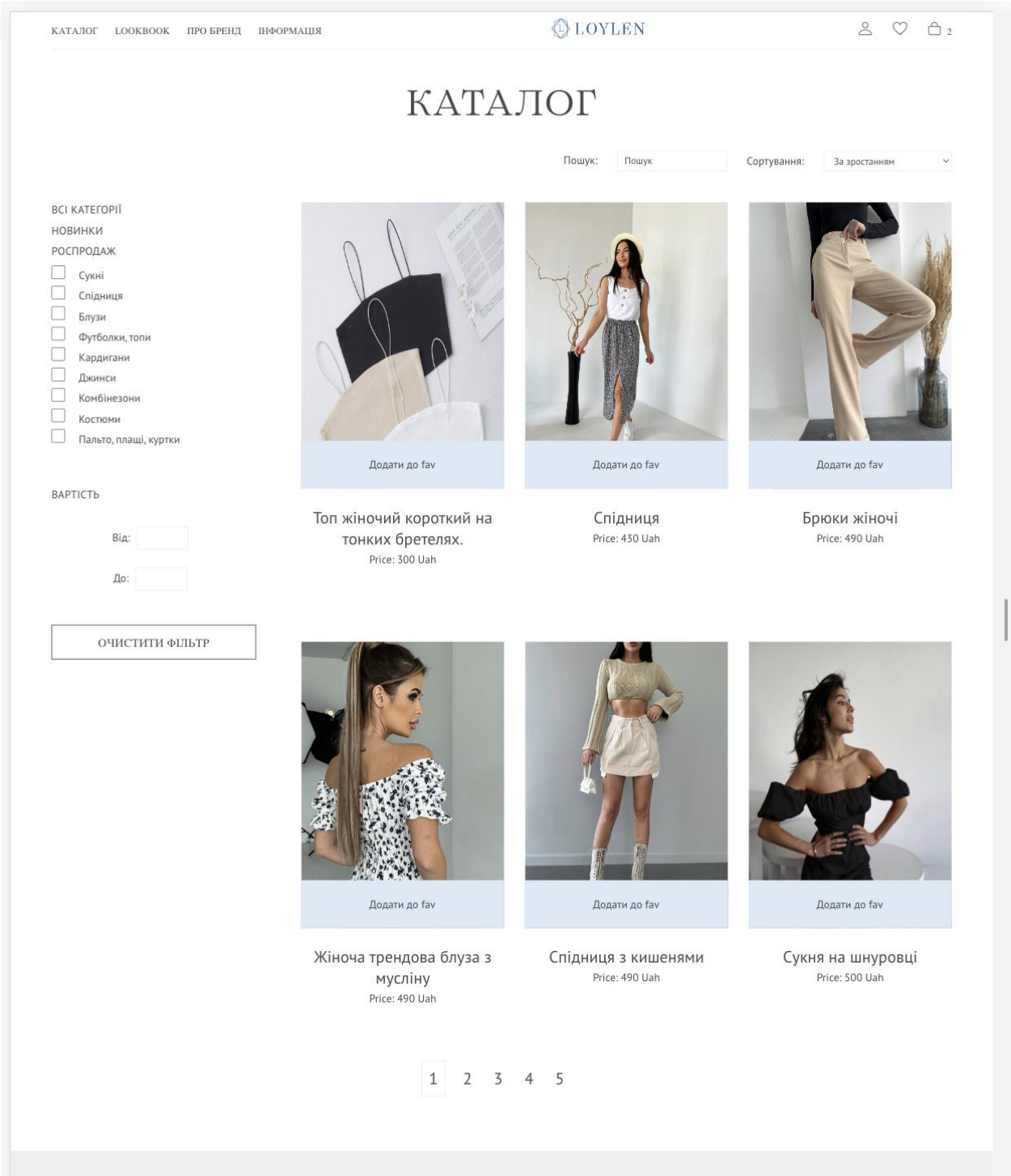

Рисунок 4.5 – Інтерфейс сторінки з товарами з використанням фільтрів

На сторінці «Корзини» можна оформити замовлення, переглянути товари додані до корзини та видалити їх (Рисунок 4.6).

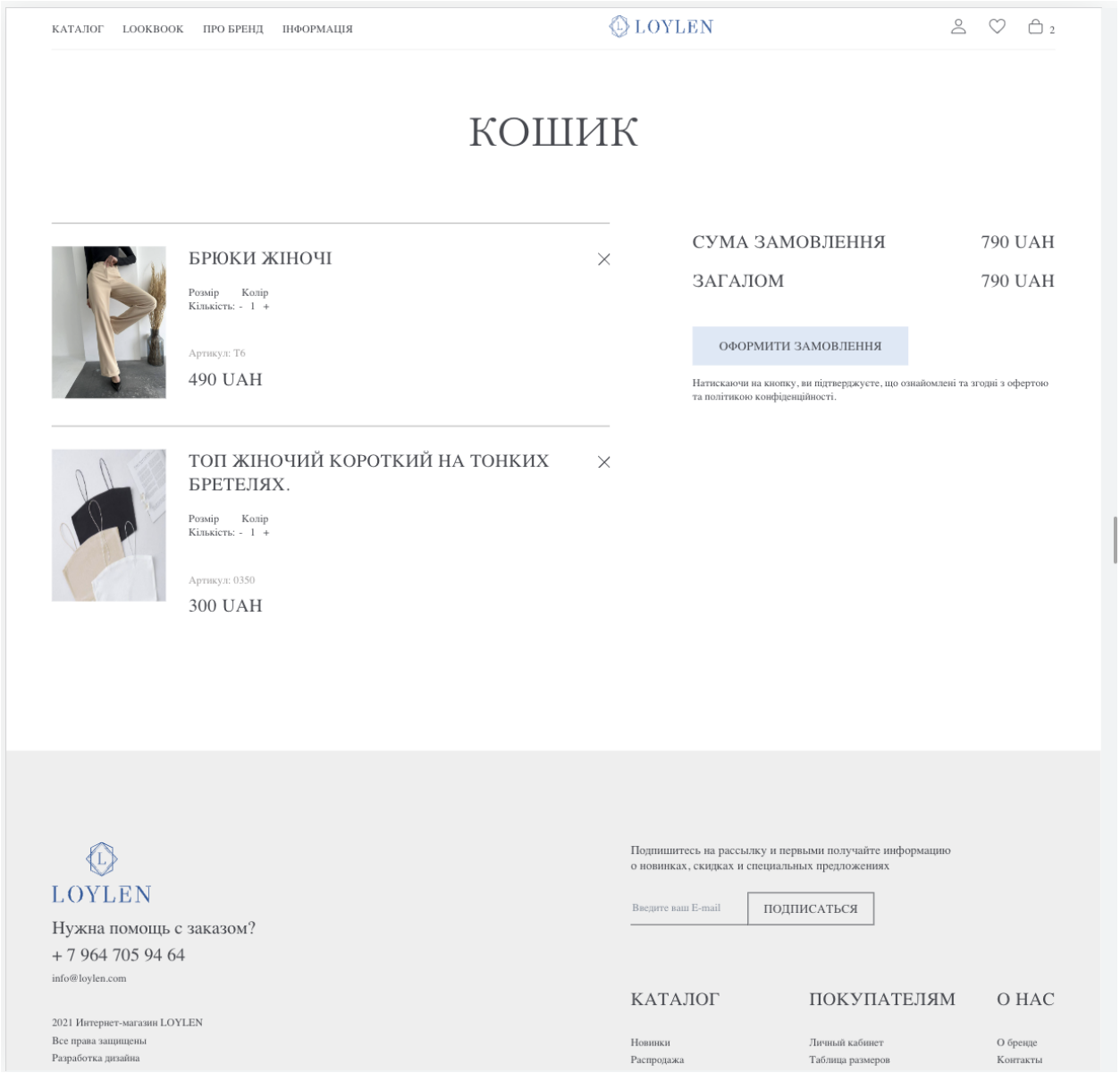

Рисунок 4.6 – Інтерфейс сторінки Корзини

Користувач може перейти до оформлення замовлення (Рисунок 4.7) і вказавши всі поля буде його переправлено на сторінку (Рисунок 4.8) де буде його номер замовлення.

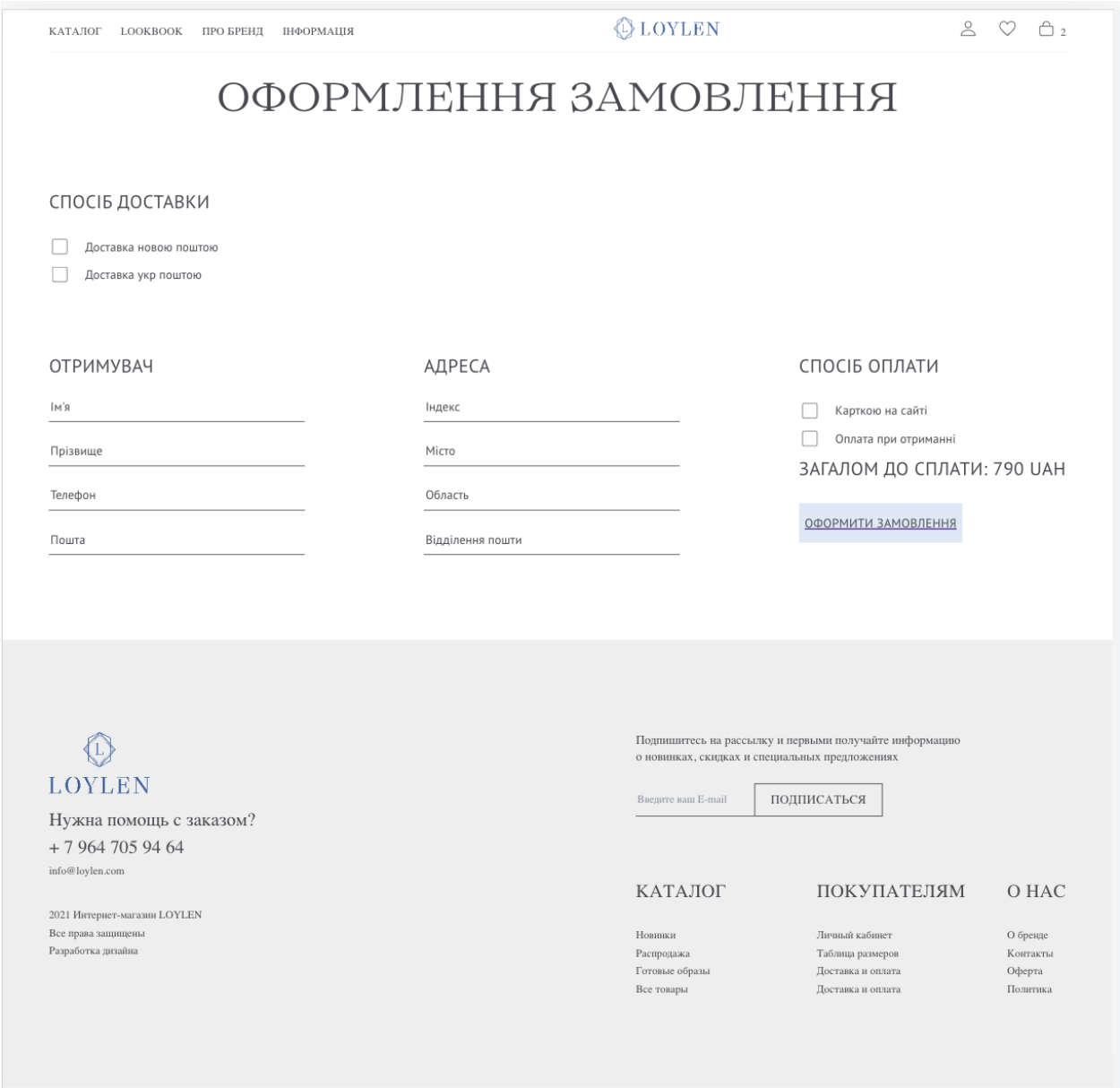

Рисунок 4.7 – Сторінка оформлення замовлення

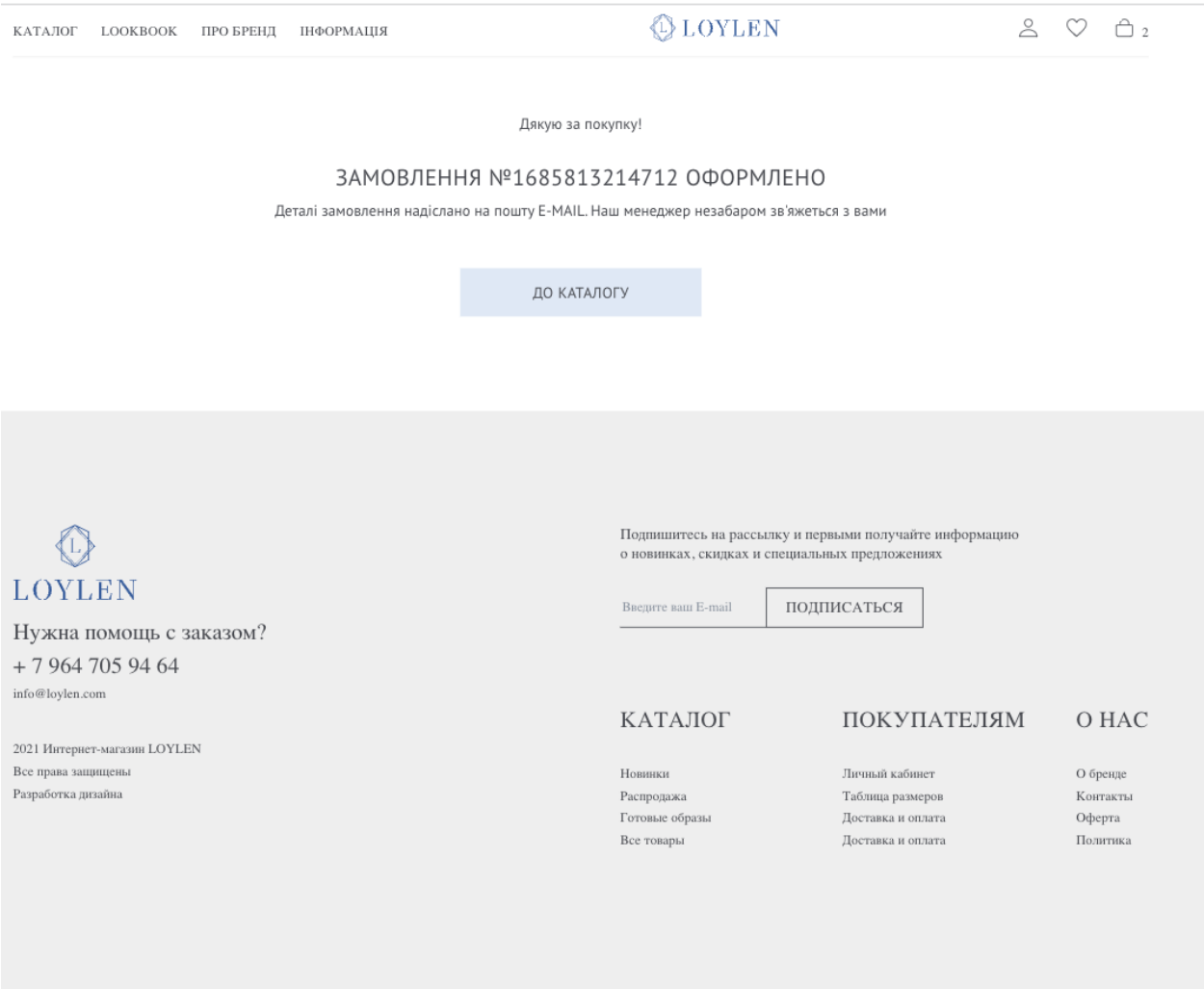

Рисунок 4.8 – Успішне оформлене замовлення

Висновки до розділу 4

В межах розділу з використанням обраних мов програмування та технологій було реалізовано вебзастосунок дотримуючись вимог до ПЗ, заздалегідь розроблених моделей, проєктних рішенньь та створених інтерфейсів ПЗ.

Використовуючи функціонал вебзастосунку, було проведено його тестування. За результатами тестування можна зробити висновок, що створена система повністю задовольняє функціональні та нефункціональні вимоги, які були поставлені на початку роботи, а вебзастосунок повністю готовий до використання.

ВИСНОВКИ

У рамках кваліфікаційної роботи бакалавра було проведено детальне дослідження предметної галузі вебзастосунків для продажу одягу. Аналізуючи ринок вебтехнологій та враховуючи вимоги до програмного забезпечення, було прийняте рішення використовувати Vue та Vuex як основні технології для розробки вебзастосунку.

Проведений аналіз показав, що Vue має значні переваги у розробці веб-інтерфейсів, забезпечує швидку реактивну відповідь та зручний спосіб організації компонентів. Використання Vuex дозволяє ефективно керувати станом даних у вебзастосунку та спрощує управління станом між компонентами.

Розроблений вебзастосунок на основі Vue та Vuex задовольняє всі вимоги, включаючи структуру та функціональність. Макети інтерфейсів публічної частини вебзастосунку були успішно створені, що дозволяє користувачам зручно переглядати та придбати товари.

Отримані результати аналізу та розробки вебзастосунку можуть бути корисними для подальшого розвитку в сфері e-commerce та використання вебтехнологій. Враховуючи широкий спектр функцій та переваг Vue та Vuex, ці технології можуть бути використані для створення потужних та зручних вебзастосунків для різних галузей.

Завершуючи кваліфікаційну роботу, можна зазначити, що успішна реалізація проекту доводить важливість вивчення та використання сучасних вебтехнологій, таких як Vue та Vuex, у розробці вебзастосунків. Подальші дослідження в цій галузі можуть сприяти вдосконаленню та розширенню можливостей веб-розробки та покращенню користувацького досвіду.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Diagrams online. URL: https://app.diagrams.net/. (дата звернення: 10.01.2023).

2. UML Diagrams Full Course. URL: https://www.youtube.com/watch?v=WnMQ8HlmeXc. (дата звернення: 03.02.2023).

3. UML. URL: https://evergreens.com.ua/ru/articles/uml-diagrams.html. (дата звернення: 04.02.2023).

4. Офіційна документація Vue.js. . URL - https://vuejs.org/v2/guide/. (дата звернення: 19.02.2023).

5. Відеоуроки по Vue.js на сайті Vue Mastery. URL - https://www.vuemastery.com/courses/. (дата звернення: 01.03.2021).

6. Розробка веб-додатків на Vue.js: від початків до продукту - книга автора Мішель Ларель (Michelle Ichinco). Підручники і посібники. URL - https://www.ozon.ru/context/detail/id/164364092/. (дата звернення: 04.03.2023).

7. Building Applications with Spring 5 and Vue.js 2 - книга авторів Ігоря Лондаря та Александра Колесніченка. Packt Publishing . URL - https://www.amazon.com/Building-Applications-Spring-Vue-js-2-ebook/dp/B07G2Q9X6D. (дата звернення: 15.03.2023).

8. "Vue.js 2 and Vuex: Build a Professional Vue App with Vuex and Vue Router" author Anthony Gore, Packt Publishing 30-35p.

9. "Vue.js: Up and Running: Building Accessible and Performant Web Apps" authors Callum Macrae, O'Reilly Media 45-47p.

10. "Vue.js in Action" authors Erik Hanchett та Benjamin Listwon, Manning Publications ,15-21 p.

11. "Vue.js 2 Web Development Projects" author Guillaume Chau, Packt Publishing, 41-49p.

ДОДАТОК А

КОД КОНФІГУРАЦІЇ МУТАЦІЇ VUEX-СТОРУ

Мутації (mutations) виконують зміни в стані додатка. Кожна мутація приймає state (стан) та додаткові параметри, такі як product або index, і змінює відповідні частини стану.

Наприклад, SET_CART додає продукт до кошика. Якщо продукт вже присутній у кошику, збільшується його кількість, в іншому випадку продукт додається з початковою кількістю 1. Стан кошика зберігається в локальному сховищі за допомогою localStorage.setItem.

Аналогічно, інші мутації, такі як SET_FAVORITE, REMOVE_FROM_CART, REMOVE_FROM_FAVORITE, INCREMENT, DECREMENT, виконують зміни в стані та зберігають оновлений стан у локальному сховищі.

Мутація LOAD_STATE_FROM_LOCAL_STORAGE використовується для завантаження стану кошика та обраного з локального сховища при запуску додатка.

Цей код встановлює основну структуру Vuex-стору та мутації для роботи з продуктами, кошиком та обраними елементами. Однак, він не містить дієвих дій (actions) або геттерів (getters), які можуть бути присутніми у повній реалізації додатка з використанням Vuex.

```
mutations: {
  SET_PRODUCTS_TO_STATE: (state, products) => {
    state.products = products;
  },
  SET_CART: (state, product) => {
    const existingProductIndex = state.cart.findIndex(
      (item) => item.article === product.article
    );
    if (existingProductIndex !== -1) {
      state.cart[existingProductIndex].quantity++;
```

```
  } else {
    product.quantity = 1;
    state.cart.push(product);
  }
  localStorage.setItem(STORAGE_KEY, JSON.stringify(state.cart));
},
SET_FAVORITE: (state, product) => {
  const existingProductIndex = state.favorite.findIndex(
    (item) => item.article === product.article
  );
  if (existingProductIndex !== -1) {
    state.favorite[existingProductIndex].quantity++;
  } else {
    product.quantity = 1;
    state.favorite.push(product);
  }
  localStorage.setItem(STORAGE_KEY, JSON.stringify(state.favorite));},
REMOVE_FROM_CART: (state, index) => {
  state.cart.splice(index, 1);
  localStorage.setItem(STORAGE_KEY, JSON.stringify(state.cart)); },
REMOVE_FROM_FAVORITE: (state, index) => {
  state.favorite.splice(index, 1);
  localStorage.setItem(STORAGE_KEY, JSON.stringify(state.favorite));},
INCREMENT: (state, index) => {
  state.cart[index].quantity++;
  localStorage.setItem(STORAGE_KEY, JSON.stringify(state.cart));},
LOAD_STATE_FROM_LOCAL_STORAGE: (state) => {
  const savedCart = localStorage.getItem(STORAGE_KEY);
  if (savedCart) {
    state.cart = JSON.parse(savedCart);
  }
  const savedFavorite = localStorage.getItem(STORAGE_KEY);
  if (savedFavorite) {
    state.favorite = JSON.parse(savedFavorite);} },},
```

ДОДАТОК Б

КОД КОНІФІГУРАЦІЇ VUEX-СТОРУ

Даний код описує геттери та ініціалізацію Vuex Store.

В розділі getters визначено три функції-геттери: PRODUCTS, CART і FAVORITE. Геттери використовуються для отримання даних зі стану додатка. Наприклад, геттер PRODUCTS повертає список продуктів зі стану state.products, геттер CART повертає кошик зі стану state.cart, а геттер FAVORITE повертає список обраних продуктів зі стану state.favorite.

Далі, за допомогою store.dispatch("LOAD_STATE_FROM_LOCAL_STORAGE"), ініціалізується дія LOAD_STATE_FROM_LOCAL_STORAGE. Ця дія викликає відповідну мутацію, яка завантажує дані з локального сховища і оновлює стан додатка відповідно.

Нарешті, export default store експортує створений екземпляр Vuex Store, щоб його можна було використовувати в інших частинах програми.

```
  getters: {
   PRODUCTS(state) {
     return state.products;
   },
   CART(state) {
     return state.cart;
   },
   FAVORITE(state) {
     return state.favorite;
   },
  },
});
store.dispatch("LOAD_STATE_FROM_LOCAL_STORAGE");
export default store;
```

ДОДАТОК В

КОД КОНФІГУРАЦІЇ МЕТОДУ СОРТУВАННЯ

Метод sortedProducts() виконує фільтрацію та сортування продуктів згідно з введеними умовами, такими як пошуковий термін, обрані категорії, мінімальна і максимальна ціни. Результатом є відфільтрований і відсортований масив продуктів, який повертається з методу.

```
sortedProducts() {
    let products = [...this.PRODUCTS];
    if (this.searchTerm) {
      const searchTerm = this.searchTerm.toLowerCase();
      products = products.filter((product) =>
        product.name.toLowerCase().includes(searchTerm)
      ); }
    if (this.selectedCategories.length > 0) {
      products = products.filter((product) =>
        this.selectedCategories.includes(product.category)
      );}
    if (this.minPrice) {
      products = products.filter(
        (product) => product.price >= Number(this.minPrice)); }
    if (this.maxPrice) {
      products = products.filter(
        (product) => product.price <= Number(this.maxPrice)); }
    if (this.sortType === "price") {
      products.sort((a, b) =>
        this.sortDirection === "asc" ? a.price - b.price : b.price - a.price);
    } else {
      products.sort((a, b) =>
        this.sortDirection === "asc"
          ? a.name.localeCompare(b.name)
          : b.name.localeCompare(a.name) );}
    return products; },
```