МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Чорноморський національний університет
імені Петра Могили
Факультет комп'ютерних наук
Кафедра інтелектуальних інформаційних систем

ДОПУЩЕНО ДО ЗАХИСТУ
Завідувач кафедри інтелектуальних
інформаційних систем, д-р техн. наук, проф.
_____Ю. П. Кондратенко
«____»_____2024 р.

КВАЛІФІКАЦІЙНА РОБОТО БАКАЛАВРА

СИСТЕМА ПРОДАЖУ ТОВАРІВ З РЕАЛІЗАЦІЄЮ АЛГОРИТМУ ПОШУКУ

Спеціальність 122 «Комп'ютерні науки»

122 – КРБ – 402.2010221

Виконав студент 4-го курсу, групи 402
_____ О. О. Савчук
«17» червня 2024 р.
Керівник: старший викладач кафедри ІПЗ
_____М.В. Фаленкова
«17» червня 2024 р

Миколаїв – 2024

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Чорноморський національний університет ім. Петра Могили

Факультет комп'ютерних наук

Кафедра інтелектуальних інформаційних систем

Рівень вищої освіти **бакалавр**
Спеціальність **122 «Комп'ютерні науки»**
(шифр і назва)
Галузь знань **12 «Інформаційні технології»**
(шифр і назва)

ЗАТВЕРДЖУЮ

Завідувач кафедри інтелектуальних
інформаційних систем, д-р техн. наук, проф.
_____ Ю. П. Кондратенко

«____» _____2024 р.

З А В Д А Н Н Я
на виконання кваліфікаційної роботи

Видано студенту групи 402 факультету комп'ютерних наук Савчуку Олексію Олександровичу.

1. Тема кваліфікаційної роботи «Система продажу товарів з реалізацією алгоритму пошуку».

Керівник роботи Фаленкова Марина Володимирівна, старший викладач кафедри ІПЗ

Затв. наказом Ректора ЧНУ ім. Петра Могили від «28» грудня 2023 р. № 271

2. Строк представлення кваліфікаційної роботи студентом «17» червня 2024 р.

3. Вхідні (початкові) дані до роботи:

– інформація про товарні позиції, їх характеристики та наявність на складі;

– алгоритми пошуку та фільтрації товарів за різними критеріями.

Очікуваний результат: система продажів товару, що дозволяє ефективно здійснювати пошук товарних позицій за визначеними критеріями.

4. Перелік питань, що підлягають розробці (зміст пояснювальної записки):

– огляд існуючих рішень та систем електронної комерції;

– визначення основних проблем та викликів у процесі продажів та пошуку товарів;

– порівняльний аналіз ефективності різних методів пошуку;

– тестування алгоритмів на різних наборах даних;

– порівняння результатів за критеріями ефективності та зручності використання.

5. Перелік графічного матеріалу: сторінок – 60, таблиць – 8, рисунків – 18, посилань – 10 та презентація.

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис
Спеціальна частина з охорони праці	Алексєєва А. О. доцент кафедри екології	

Керівник роботи старший викладач кафедри ІПЗ, Фаленкова М. В.

(наук. ступінь, вчене звання, прізвище та ініціали)

_____

(підпис)

Завдання прийнято до виконання Савчук О.О.

(прізвище та ініціали)

_____

(підпис)

Дата видачі завдання «14 » січня 2024 р.

КАЛЕНДАРНИЙ ПЛАН
виконання кваліфікаційної роботи

Тема: Система продажу товару з реалізацією алгоритму пошуку

№	Найменування роботи	Початок	Закінчення	Примітки
1	Подання заяви на затвердження теми та керівників КРБ	10.11.2023	15.11.2023	Виконано
2	Отримання завдання на виконання КРБ	10.01.2024	15.01.2024	Виконано
3	Складання календарного плану роботи на весь період виконання КРБ	16.01.2024	30.01.2024	Виконано
4	Отримання завдання на переддипломну практику	15.04.2024	29.04.2024	Виконано
5	Проходження переддипломної практики, збір та аналіз матеріалів до КРБ	29.04.2024	11.05.2024	Виконано
6	Розробка звіту з переддипломної практики	12.05.2024	15.05.2024	Виконано
7	Виконання КРБ: аналіз аналогічних веб-ресурсів та реалізація системи продажу товарів	13.05.2024	22.06.2024	Виконано
8	Перший попередній захист КРБ на засіданні комісії кафедри	27.05.2024	27.05.2024	Виконано
9	Доробка та остаточне оформлення КРБ	28.05.2024	09.06.2024	Виконано
10	Другий попередній захист КРБ на засіданні комісії кафедри	10.06.2024	10.06.2024	Виконано
11	Подання КРБ рецензенту	13.06.2024	13.06.2024	Виконано
11	Подання КРБ, її електронної копії та інших документів (відгуку, рецензії) до захисту	17.06.2024	21.06.2024	Виконано
12	Захист КРБ перед екзаменаційною комісією (ЕК)	24.06.2024	28.06.2024	Виконано

Розробив студент Савчук О.О. _____

(прізвище та ініціали) (*підпис*)

Керівник роботи старший викладач кафедри ІПЗ, Фаленкова М. В.

_____

(наук. ступінь, вчене звання, прізвище та ініціали) *(підпис)*

«29» 01 2024 р.

АНОТАЦІЯ

кваліфікаційної роботи студента групи 402 ЧНУ ім. Петра Могили

Савчука Олексія Олександровича

Тема: «Система продажу товарів з реалізацією алгоритму пошуку»

Актуальність: електронна комерція є однією з найбільш швидкорозвиваючих галузей, що надає можливість споживачам купувати товари та послуги онлайн з максимальним комфортом. Розробка системи продажів товару з ефективним алгоритмом пошуку є актуальною, оскільки забезпечує підвищення задоволеності користувачів, зручність та швидкість знаходження потрібних товарів.

Об'єкт роботи – процес електронної комерції.

Предмет роботи – технології розробки веб-застосунку для продажу товарів з реалізацією алгоритму пошуку.

Метою кваліфікаційної роботи є розробка ефективної системи продажів товарів з використанням сучасних технологій та алгоритмів пошуку, що забезпечить користувачам зручний та швидкий доступ до необхідних товарів.

Пояснювальна записка складається зі вступу, трьох розділів та висновків

Структура роботи:

перший розділ: аналіз предметної області електронної комерції, огляд аналогічних застосунків.

другий розділ: аналіз і планування функціоналу проекту, включаючи аналіз необхідних функцій на прикладі аналогічних веб-ресурсів.

третій розділ: опис основних програмних компонентів проекту, включаючи підключення до бази даних MongoDB з використанням Mongoose, створення моделей даних для користувачів, продуктів і замовлень.

Ключові слова: електронна комерція, алгоритм пошуку, веб-застосунок, Node.js, база даних, кошик, замовлення, обліковий запис користувача.

ABSTRACT

Qualification work of the student of group 402 of Petro Mohyla Black Sea National University

Oleksii Savchuk

Topic: "Product Sales System with Search Algorithm Implementation"

Relevance: E-commerce is one of the fastest-growing industries, providing consumers with the ability to purchase goods and services online with maximum convenience. The development of a sales system with an efficient search algorithm is relevant as it ensures increased user satisfaction, convenience, and speed in finding the desired products.

Object of work: the process of e-commerce.

Subject of work: technologies for developing a web application for selling goods with the implementation of a search algorithm.

The aim of the qualification work is to develop an effective sales system using modern technologies and search algorithms, providing users with convenient and quick access to the necessary goods.

The explanatory note consists of an introduction, three chapters, and conclusions.

Structure of work:

first chapter: analysis of the e-commerce domain, review of similar applications.

second chapter: analysis and planning of the project functionality, including the analysis of necessary functions based on similar web resources.

third chapter: description of the main software components of the project, including connecting to the MongoDB database using Mongoose, creating data models for users, products, and orders.

Keywords: E-commerce, search algorithm, web application, Node.js, database, shopping cart, order, user account, order payment.

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ

БД – База Даних

API – Application Programming Interface (Інтерфейс програмування застосунків)

CRUD – Create, Read, Update, Delete (Створення, Читання, Оновлення, Видалення)

CSS – Cascading Style Sheets (Каскадні таблиці стилів)

CVV – card verification code

DB – Data Base

DB – Database (База даних)

E-commerce– Electronic Commerce (Електронна комерція)

HTTP – HyperText Transfer Protocol (Протокол передачі гіпертексту)

HTTPS – HyperText Transfer Protocol Secure (Захищений протокол передачі гіпертексту)

JS – JavaScript

JS – JavaScript (Мова програмування JavaScript)

JSON – JavaScript Object Notation (Нотація об'єктів JavaScript)

JWT – JSON Web Token (Веб-токен на основі JSON)

MERN – MongoDB, Express.js, React.js, Node.js (Технологічний стек MongoDB, Express.js, React.js, Node.js)

MVC – Model-View-Controller (Модель-Вид-Контролер)

NoSQL – Not Only SQL (Не тільки SQL)

PHP – Hypertext Preprocessor

REST – Representational State Transfer (Передача репрезентативного стану)

SEO – Search Engine Optimization (Оптимізація для пошукових систем)

SQL – Structured Query Language (Мова структурованих запитів)

UI – User Interface (Інтерфейс користувача)

URL – Uniform Resource Locator (Уніфікований вказівник ресурсів)

UX – User Experience (Досвід користувача)

ВСТУП

Актуальність теми

На сьогоднішній день електронна комерція стала невід'ємною частиною сучасного бізнесу та споживання. Інтернет-магазини надають можливість покупцям обирати та купувати товари з будь-якого місця та у будь-який час, що значно спрощує процес купівлі та робить його більш зручним. Основними перевагами систем продажів товарів в інтернеті є:

– доступність цілодобово;
– широкий вибір товарів;
– можливість порівняння цін та характеристик;
– зручні методи оплати та доставки;
– економія часу.

З розвитком технологій та збільшенням конкуренції на ринку електронної комерції, важливим стає впровадження ефективних алгоритмів пошуку товарів, що дозволяють швидко знаходити потрібні товари серед великого асортименту. Таким чином, створення системи продажів товарів з реалізацією алгоритму пошуку є актуальним завданням, яке дозволить підвищити ефективність роботи інтернет-магазину та задовольнити потреби користувачів.

Об'єктом роботи є процес розробки системи продажів товарів з реалізацією алгоритму пошуку.

Предметом роботи є алгоритм створення системи продажів товарів, який забезпечує ефективний пошук та вибір товарів користувачами.

Метою кваліфікаційної роботи є розробка системи продажів товарів з реалізацією алгоритму пошуку, яка підвищить ефективність пошуку та задоволеність користувачів інтернет-магазину.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ ЕКЕКТРОННОЇ КОМЕРЦІЇ

1.1 Огляд застосунків-аналогів

Одним з важливих етапів аналізу вимог до програмного забезпечення є аналіз наявних аналогів. Це дозволяє визначити, чи існують на ринку такі програмні засоби, які можуть задовольнити потреби користувачів, або які можуть бути використані в якості основи для створення нового ПЗ. Також це допомагає визначитись у критеріях функціональності, інтерфейсу користувача, швидкодії, масштабованості та інших.

Для виявлення основних функцій, недоліків та переваг, які потрібно буде врахувати при розробці системи продажів товарів з реалізацією алгоритму пошуку, були розглянуті наступні аналоги: Amazon, eBay та Shopify(табл. 1.1-1.3).

Таблиця 1.1 – Опис вебзастосунку Amazon

Назва	**Amazon**
Архітектура	Web application;
Виробник	Amazon;
Мова реалізації	Java, JavaScript;
Функції	-пошук товарів за ключовими словами;
	- фільтрування та сортування товарів;
	- додавання товарів до кошика;
	- оформлення замовлення;
	- оплата замовлень різними способами;
	- відстеження замовлень;
	- відгуки та рейтинги товарів;
Переваги	- широкий асортимент товарів;

Закінчення таблиці 1.1

	- зручний та інтуїтивний інтерфейс;
	- надійна система оплати та доставки;
	- велика кількість відгуків користувачів;
	- висока швидкодія та масштабованість;
Недоліки	- складність для новачків;
	- високі комісії для продавців;
	- інколи складна навігація;
Посилання	https://www.amazon.com

Таблиця 1.2 – Опис веб-застосунку eBay

Назва	**eBay**
Архітектура	Web application;
Виробник	eBay;
Мова реалізації	Java, JavaScript;
Функції	- пошук товарів за ключовими словами;
	- фільтрування та сортування товарів;
	- додавання товарів до кошика;
	- оформлення замовлення;
	- оплата замовлень різними способами;
	- відстеження замовлень;
	- відгуки та рейтинги товарів;
	- можливість участі в аукціонах;
Переваги	- широкий асортимент товарів;
	- можливість купівлі за фіксованою ціною

Закінчення таблиці 1.2

	або через аукціон
	- зручний та інтуїтивний інтерфейс;
	- надійна система оплати та доставки;
	- велика кількість відгуків користувачів;
Недоліки	- складність для новачків;
	- високі комісії для продавців;
	- інколи складна навігація;
	- наявність шахраїв серед продавців;
Посилання	https://www.ebay.com

Таблиця 1.3 – Опис веб-застосунку Shopify

Назва	**Shopify**
Архітектура	Web application;
Виробник	Shopify Inc.;
Мова реалізації	Ruby on Rails;
Функції	- створення та налаштування інтернет-магазину;
	- пошук товарів за ключовими словами;
	- фільтрування та сортування товарів;
	- додавання товарів до кошика;
	- оформлення замовлення;
	- оплата замовлень різними способами;
	- відстеження замовлень;
	- відгуки та рейтинги товарів;
	- інтеграція з різними платіжними системами

Закінчення таблиці 1.3

	та службами доставки;
Переваги	- простота використання;
	- широкий спектр інтеграцій;
	- гнучкість та масштабованість;
	- зручний та інтуїтивний інтерфейс;
	- надійна система оплати та доставки;
Недоліки	- висока вартість підписки та додаткових
	послуг;
	- обмежена функціональність для великих
	магазинів без додаткових плагінів;
	- інколи складна навігація;
Посилання	https://www.shopify.com

Висновки

Аналіз аналогів показав, що існуючі системи продажів товарів мають широкий спектр функцій, які значно полегшують процес купівлі для користувачів. Основними критеріями(табл. 1.4), на які було звернуто особливу увагу, є:

Таблиця 1.4 – Критерії аналізу

Критерій	Опис
Доступність	доступність для користувачів без базових знань програмування;
Мультимовність	доступність для широкої аудиторії;
Актуальність інформації	забезпечення актуальної інформації на сайті;
Простота використання	інтуїтивний інтерфейс;
Платформна доступність	доступність на будь-якій платформі;

1.2 Визначення мети дослідження

Метою даної роботи є розробка та реалізація інтернет-магазину з ефективним алгоритмом пошуку, який забезпечить зручний та швидкий доступ до товарів для користувачів. Основні завдання, що стоять перед цією роботою,

1) розробка алгоритму пошуку: спроектувати та реалізувати алгоритм пошуку, який забезпечить точне та швидке знаходження товарів за різними критеріями, такими як ключові слова, параметри, категорії тощо. Цей алгоритм повинен бути оптимізований для швидкої роботи навіть при великому обсязі даних;

2) створення користувацького інтерфейсу: розробити зручний та інтуїтивно зрозумілий інтерфейс для користувачів, що дозволить їм легко використовувати алгоритм пошуку та здійснювати покупки без зайвих труднощів;

3) оптимізація продуктивності: приділити увагу оптимізації продуктивності системи, зокрема, швидкості відповіді на запити користувачів та масштабованості для обробки одночасних запитів;

4) аналіз та оцінка результатів: провести аналіз ефективності розробленого інтернет-магазину з алгоритмом пошуку, порівняти його зі схожими системами на ринку, оцінити задоволеність користувачів та виявити можливі напрямки подальшого вдосконалення.

1.3 Аналіз існуючих проблем електронної комерції

У сучасному світі інтернет-торгівлі, де доступ до товарів широкий та конкуренція велика, важливою задачею для будь-якого інтернет-магазину є забезпечення ефективного та зручного пошуку для користувачів. Проте існують деякі основні проблеми, які можуть виникнути при розробці алгоритму пошуку для інтернет-магазину:

– недостатня точність результатів пошуку: у намаганні забезпечити швидкий пошук, деякі інтернет-магазини можуть стикатися з проблемою

недостатньої точності результатів. Це може виникати через недостатню обробку та індексацію даних або через недосконаліть алгоритму пошуку;

– повільна швидкість пошуку при великому обсязі даних: зі зростанням кількості товарів в магазині може виникати проблема повільної швидкості пошуку. Це може бути наслідком неефективної обробки даних або недостатньої оптимізації алгоритму пошуку;

– складність управління фільтрами та параметрами пошуку: інтернет-магазини можуть мати велику кількість параметрів та фільтрів для пошуку товарів. Управління цими фільтрами та параметрами може бути складним для користувачів та вимагати ефективного інтерфейсу та алгоритму обробки;

– необхідність масштабування системи: при зростанні обсягів даних та кількості користувачів, інтернет-магазин повинен бути готовий масштабуватися для забезпечення стабільної роботи системи та швидкого відгуку на запити користувачів;

– конкурентний тиск та вимоги користувачів: у сфері інтернет-торгівлі конкуренція велика, і користувачі мають високі очікування щодо зручності та швидкості роботи магазинів. Недоліки у системі пошуку можуть призвести до втрати клієнтів та зниження прибутковості бізнесу.

Розуміння цих проблем дозволить нам ефективно розробити та реалізувати алгоритм пошуку, який враховуватиме потреби та очікування користувачів, а також допоможе підтримувати конкурентоспроможність на ринку.

1.4 Призначення проекту та його межі

Метою даного проекту є створення системи продажів товарів з реалізацією алгоритму пошуку. Ця система повинна забезпечити зручну та ефективну платформу для продавців і покупців, що дозволить здійснювати купівлю та продаж товарів з мінімальними зусиллями та високою швидкістю. Основними функціями системи є:

– пошук товарів за ключовими словами та фільтрами;

– реалізація кошика покупок;
– оформлення та оплата замовлень;
– управління обліковими записами користувачів;
– адміністративна панель для управління товарами та користувачами;
– відгуки та рейтинги товарів.

Межі проекту(табл. 1.5-1.8) визначають, що буде реалізовано в рамках даної системи, а що залишиться поза її межами. Основні межі проекту включають:

Таблиця 1.5 – Межі проекту

Компонент	**Опис**
Пошук товарів	Реалізація алгоритму пошуку, який дозволяє знаходити товари за ключовими словами, категоріями та фільтрами.
Реалізація кошика покупок	Можливість додавання товарів до кошика, перегляд кошика, редагування кількості товарів та видалення товарів з кошика.
Оформлення та оплата замовлень	Процес оформлення замовлення, включаючи введення даних про доставку, вибір способу оплати та підтвердження замовлення.
Управління обліковими записами	Реєстрація нових користувачів, авторизація, відновлення паролю, редагування профілю та перегляд історії замовлень.
Адміністративна панель	Панель адміністратора для управління товарами, категоріями, користувачами та замовленнями.
Відгуки та рейтинги товарів	Можливість користувачів залишати відгуки та оцінювати товари, переглядати рейтинги та коментарі інших користувачів.

Таблиця 1.6 – Функціональні межі

Функціональність	**Опис**
Алгоритм пошуку	Реалізація алгоритму пошуку з можливістю швидкої індексації нових товарів та обробки запитів користувачів у режимі реального часу.
Кошик покупок	Збереження стану кошика для анонімних та зареєстрованих користувачів, можливість подальшого редагування кошика.
Процес оплати	Підтримка різних способів оплати, включаючи кредитні картки, електронні гаманці та банківські перекази.
Облікові записи користувачів	Забезпечення безпеки та конфіденційності даних користувачів, можливість зміни персональних даних.
Адміністративні функції	Інтерфейс для адміністраторів з можливістю додавання, редагування та видалення товарів, управління категоріями та користувачами.
Система відгуків	Механізм модерації відгуків, забезпечення захисту від спаму та несанкціонованих коментарів.

Таблиця 1.7 – Технічні межі

Технічний аспект	**Опис**
Платформи	Веб-версія системи, адаптована для використання на ПК, планшетах та смартфонах.
Інтеграції	Інтеграція з платіжними системами та службами доставки.
Безпека	Використання сучасних методів шифрування для захисту даних користувачів та транзакцій.
Масштабованість	Система повинна підтримувати можливість розширення функціональності та збільшення навантаження без втрати продуктивності.

Таблиця 1.8 – Обмеження проекту

Обмеження	**Опис**
Термін реалізації	Проект повинен бути завершений протягом одного року.
Бюджет	Обмежений бюджет на розробку та впровадження системи.

Закінчення таблиці 1.8

Технічні ресурси	Використання доступних ресурсів для розробки та тестування системи.
Користувацька підтримка	Обмежені ресурси для надання підтримки користувачам у період запуску системи.

Висновок

Призначення проекту визначає створення зручної та ефективної платформи для продажів товарів з алгоритмом пошуку, що забезпечить високий рівень задоволеності користувачів. Межі проекту включають визначення функціональних, технічних та обмежуючих аспектів, які допоможуть сфокусуватися на досягненні основних цілей та забезпечити успішну реалізацію системи.

1.5 Вибір мови програмування для проекту

Вибір Node.js для реалізації інтернет-магазину є обґрунтованим з кількох причин, що відповідають конкретним потребам та вимогам проекту:

– швидкодія та асинхронність: Node.js відомий своєю асинхронною моделлю програмування, що дозволяє ефективно обробляти багатопотокові запити без блокування основного потоку виконання. Це особливо важливо для інтернет-магазинів, де велика кількість одночасних запитів від користувачів може призвести до затримок у відповіді на запити, якщо не використовувати асинхронний підхід;

– екосистема npm: Node.js має велику екосистему пакетів npm, що значно спрощує розробку та розширення функціональності проекту. Готові модулі, доступні через npm, дозволяють швидко впроваджувати різноманітні функції, включаючи алгоритм пошуку, оптимізацію та інші;

– швидкість виконання JavaScript-коду: використання движка V8 Chrome дозволяє Node.js виконувати JavaScript-код з високою швидкодією. Це забезпечує ефективну роботу магазину навіть при великій кількості одночасних запитів від користувачів[2];

– масштабованість: Node.js гнучкий у масштабуванні, що дозволяє розширювати функціональність інтернет-магазину з легкістю. При правильному проектуванні та налагодженні, Node.js може працювати добре як для невеликих стартапів, так і для великих підприємств з великим обсягом трафіку;

– огляд обмежень: незважаючи на всі переваги, важливо враховувати обмеження Node.js, такі як однопоточність, яка може вплинути на продуктивність в разі великої кількості обчислювально-інтенсивних операцій. В таких випадках може бути необхідно розглянути альтернативні рішення або оптимізувати код для використання паралельних процесів або кластеризації[1].

1.6 Обґрунтування вибору Node.js для реалізації проекту

Після порівняння Node.js з іншими мовами програмування та середовищами, можна побачити, що Node.js має деякі унікальні переваги:

– асинхронність та швидкодія: Node.js дозволяє ефективно обробляти багатопотокові запити, що робить його оптимальним вибором для високонавантажених систем, таких як інтернет-магазини;

– JavaScript на фронтенді та бекенді: використання JavaScript як на клієнтській, так і на серверній стороні дозволяє уникнути дублювання коду та спростити управління проектом[2];

– екосистема npm: Node.js має велику екосистему модулів npm, що спрощує розробку та розширення функціональності проекту;

– підтримка асинхронного програмування: Node.js має потужну підтримку асинхронного програмування, що дозволяє легко створювати ефективні асинхронні додатки[1].

2 АНАЛІЗ ТА ПЛАНУВАННЯ РЕАЛІЗАЦІЇ ФУНКЦІОНАЛУ ПРОЕКТУ

2.1 Аналіз необхідного функціоналу на прикладі аналогічних веб-ресурсів

Функціонал оплати замовлення є ключовим елементом будь-якого інтернет-магазину. Він забезпечує завершення покупки, дозволяючи користувачам оплатити обрані товари або послуги. Розглянемо, як цей процес реалізовано на декількох популярних платформах електронної комерції.

1. **Amazon**

Процес оплати:

– вибір товарів: користувач додає товари до кошика;

– перехід до кошика: після завершення вибору товарів користувач переходить до кошика, де може переглянути список товарів, їх кількість та підсумкову вартість;

– оформлення замовлення: користувач натискає кнопку "Proceed to Checkout" для переходу до оформлення замовлення;

– вхід/реєстрація: якщо користувач не увійшов у свій обліковий запис, його просять увійти або зареєструватися;

– введення інформації: користувач вводить або підтверджує свою адресу доставки, вибирає спосіб доставки та спосіб оплати (кредитна/дебетова картка, Amazon Pay, PayPal);

– перевірка замовлення: перед остаточним підтвердженням користувач переглядає підсумкову вартість замовлення з урахуванням доставки та податків;

– підтвердження: після підтвердження оплати користувач отримує електронний лист з підтвердженням замовлення.

Особливості:

– підтримка багатьох способів оплати, включаючи кредитні картки, дебетові картки та цифрові гаманці;

– можливість зберігання платіжних даних для майбутніх покупок;

– інтеграція з Amazon Pay для швидшої та зручнішої оплати.

2. eBay

Процес оплати:

– вибір товарів: користувач додає товари до кошика або здійснює покупку через аукціон;

– перехід до кошика: користувач переглядає кошик і натискає "Go to checkout" для переходу до оплати;

– вхід/реєстрація: вхід у обліковий запис або реєстрація;

– введення інформації: введення адреси доставки та вибір способу доставки;

– вибір способу оплати: eBay підтримує оплату через PayPal, кредитні та дебетові картки, Apple Pay, Google Pay;

– перевірка та підтвердження: підтвердження замовлення та оплата.

Особливості:

– можливість оплати через PayPal, що забезпечує додаткову безпеку для користувачів;

– підтримка мобільних платіжних систем, таких як Apple Pay і Google Pay;

– захист покупців через систему гарантії eBay Money Back Guarantee.

3. Alibaba

Процес оплати:

– вибір товарів: додавання товарів до кошика;

– перехід до кошика: перегляд кошика та натискання "Checkout" для переходу до оплати;

– вхід/реєстрація: вхід у обліковий запис або реєстрація;

– введення інформації: Введення або підтвердження адреси доставки та вибір способу доставки;

– вибір способу оплати: Alibaba пропонує різноманітні способи оплати, включаючи кредитні картки, банківські перекази, Alipay;

– перевірка та підтвердження: підтвердження деталей замовлення та оплата.

Особливості:

– інтеграція з Alipay, яка забезпечує швидку та безпечну оплату;

– підтримка міжнародних платежів через банківські перекази та кредитні картки;

– захист транзакцій через систему Escrow, яка гарантує безпечність платежів.

Висновки

Аналіз функціоналу оплати замовлень на популярних інтернет-ресурсах показує, що всі ці платформи пропонують користувачам зручний і безпечний процес оплати, який включає:

– широкий вибір способів оплати;

– можливість збереження платіжних даних для майбутніх покупок;

– захист транзакцій через інтеграцію з платіжними системами (PayPal, Alipay).

Для розробки власної системи продажів товарів з реалізацією алгоритму пошуку, важливо врахувати досвід цих платформ, забезпечивши користувачам зручний і безпечний процес оплати, а також інтеграцію з популярними платіжними системами.

Аналіз функціоналу "Створення аккаунту для користувачів" на аналогічних веб-ресурсах

Функціонал створення облікового запису є важливою частиною будь-якого інтернет-ресурсу, оскільки він дозволяє користувачам зберігати свої дані, відстежувати замовлення, зберігати улюблені товари та користуватися іншими персоналізованими послугами. Розглянемо, як цей процес реалізовано на декількох популярних платформах електронної комерції.

1. Amazon

Процес створення аккаунту:

– перехід до реєстрації: на головній сторінці або при спробі здійснити покупку, користувач може натиснути на "Sign in" і вибрати "Create your Amazon account";

– введення особистої інформації: користувач вводить ім'я, електронну пошту та пароль;

– верифікація: Amazon може надіслати верифікаційний код на електронну пошту користувача для підтвердження адреси;

– завершення реєстрації: після введення верифікаційного коду, аккаунт створено, і користувач може почати користуватися всіма функціями.

Особливості:

– простий і швидкий процес реєстрації;

– верифікація електронної пошти для підвищення безпеки;

– можливість додавання додаткової інформації після реєстрації (адреса, платіжні дані тощо)

2. eBay

Процес створення аккаунту:

– перехід до реєстрації: користувач натискає "Register" на головній сторінці;

– вибір типу облікового запису: eBay пропонує створити особистий або бізнес-аккаунт;

– введення особистої інформації: користувач вводить ім'я, електронну пошту, пароль, а також номер телефону для додаткової верифікації;

– верифікація: верифікація електронної пошти та/або номеру телефону;

– завершення реєстрації: після верифікації користувач може почати користуватися аккаунтом.

Особливості:

– можливість створення як особистого, так і бізнес-аккаунту;

– додаткова верифікація через номер телефону для підвищення безпеки;

– інтеграція з PayPal для швидшого створення облікового запису.

3. Alibaba

Процес створення аккаунту:

– перехід до реєстрації: на головній сторінці або при спробі здійснити покупку, користувач натискає "Join Free";

– введення особистої інформації: користувач вводить електронну пошту, пароль, країну та тип аккаунту (покупець або постачальник);

– верифікація: Alibaba надсилає верифікаційний код на електронну пошту або телефон для підтвердження;

– завершення реєстрації: після введення верифікаційного коду аккаунт створено, і користувач може почати користуватися всіма функціями платформи.

Особливості:

– вибір типу аккаунту (покупець або постачальник) під час реєстрації;

– верифікація через електронну пошту або телефон для підвищення безпеки;

– можливість додавання додаткової інформації та налаштувань аккаунту після реєстрації.

Висновки

Аналіз функціоналу створення облікових записів на популярних інтернет-ресурсах показує, що всі ці платформи надають користувачам простий, швидкий та безпечний процес реєстрації, який включає:

– введення основної особистої інформації (ім'я, електронна пошта, пароль);

– верифікація електронної пошти та/або номеру телефону для підвищення безпеки;

– можливість вибору типу облікового запису (особистий, бізнес або покупець, постачальник).

Для розробки власної системи продажів товарів, важливо забезпечити зручний процес створення облікового запису, враховуючи досвід цих платформ. Це

допоможе залучити більше користувачів та підвищити їх задоволеність від користування вашою системою.

Аналіз функціоналу "кошика" на аналогічних веб-ресурсах

Функціонал кошика є одним з найважливіших елементів інтернет-магазину, оскільки він дозволяє користувачам додавати товари, переглядати їх, змінювати кількість, видаляти непотрібні позиції та переходити до оформлення замовлення. Розглянемо, як цей процес реалізовано на декількох популярних платформах електронної комерції.

1. Amazon

Процес використання кошика:

– додавання товарів: користувач може додати товар до кошика натисканням кнопки "Add to Cart" на сторінці товару;

– перегляд кошика: для перегляду кошика користувач натискає іконку кошика у верхньому правому куті сторінки або вибирає опцію "Cart" у меню;

– редагування: у кошику користувач може змінити кількість товарів, видалити товар, зберегти його для майбутніх покупок (опція "Save for later"), або перейти до оформлення замовлення ("Proceed to Checkout");

– рекомендації: під кошиком Amazon показує рекомендації товарів на основі вмісту кошика;

– збереження стану кошика: якщо користувач не завершує покупку, товари залишаються у кошику під час наступних сеансів.

Особливості:

– автоматичне збереження стану кошика для зареєстрованих користувачів;

– можливість додавання товарів до списку бажань або збереження для майбутніх покупок;

– рекомендації товарів на основі вмісту кошика.

2. eBay

Процес використання кошика:

– додавання товарів: товари додаються до кошика натисканням кнопки "Add to cart" на сторінці товару;

– перегляд кошика: користувач може переглянути кошик, натиснувши на іконку кошика у верхньому правому куті сторінки або вибравши "Cart" у меню;

– редагування: користувач може змінити кількість товарів, видалити товари, або перейти до оформлення замовлення ("Go to checkout");

– об'єднання покупок: eBay дозволяє об'єднувати покупки від різних продавців в одному кошику;

– збереження стану кошика: Товари зберігаються у кошику, навіть якщо користувач виходить з аккаунту.

Особливості:

– підтримка об'єднання покупок від різних продавців;

– автоматичне збереження стану кошика для зареєстрованих користувачів;

– можливість перегляду додаткової інформації про продавця прямо з кошика.

3. Alibaba

Процес використання кошика:

– додавання товарів: користувач додає товари до кошика натисканням кнопки "Add to Cart" на сторінці товару;

– перегляд кошика: кошик можна переглянути, натиснувши на іконку кошика у верхньому правому куті сторінки або вибравши "My Cart" у меню;

– редагування: користувач може змінити кількість товарів, видалити товари, або перейти до оформлення замовлення ("Proceed to Checkout");

– об'єднання покупок: Alibaba дозволяє об'єднувати покупки від різних постачальників в одному кошику;

– збереження стану кошика: Товари залишаються у кошику до завершення покупки або видалення користувачем.

Особливості:

– підтримка об'єднання покупок від різних постачальників;

– автоматичне збереження стану кошика;

– можливість спілкування з постачальниками безпосередньо з кошика для уточнення деталей замовлення.

Висновки

Аналіз функціоналу кошика на популярних інтернет-ресурсах показує, що всі ці платформи надають користувачам зручний і багатофункціональний інтерфейс кошика, який включає:

– можливість додавання, видалення та редагування товарів у кошику;

– автоматичне збереження стану кошика для зареєстрованих користувачів;

– Об'єднання покупок від різних продавців або постачальників в одному кошику;

– додаткові функції, такі як збереження товарів для майбутніх покупок, рекомендації товарів, та інтеграція з іншими сервісами (список бажань, спілкування з продавцями).

Для розробки власної системи продажів товарів, важливо забезпечити зручний та інтуїтивно зрозумілий функціонал кошика, враховуючи досвід цих платформ. Це підвищить зручність користування та задоволеність клієнтів від взаємодії з вашою системою.

Аналіз функціоналу "ревізії від користувачів" на аналогічних веб-ресурсах

Функціонал ревізій (відгуків) від користувачів є важливою частиною будь-якого інтернет-магазину, оскільки він дозволяє покупцям ділитися своїм досвідом, а також допомагає іншим користувачам приймати обґрунтовані рішення щодо покупок. Розглянемо, як цей процес реалізовано на декількох популярних платформах електронної комерції.

1. Amazon

Процес створення та перегляду відгуків:

– додавання відгуку: користувач може залишити відгук про товар, який він придбав, натиснувши кнопку "Write a customer review" на сторінці товару;

– оцінка товару: користувач вибирає рейтинг від 1 до 5 зірок.;

– текстовий відгук: користувач пише текстовий відгук, в якому описує свій досвід використання товару. Додатково, можна завантажити фотографії або відео товару;

– публікація: після написання відгуку, користувач натискає кнопку "Submit" для публікації;

– перегляд відгуків: всі відгуки відображаються на сторінці товару, сортуються за корисністю, датою або оцінкою.

Особливості:

– можливість оцінки корисності відгуків іншими користувачами (кнопки "Helpful" та "Not Helpful");

– фільтрація відгуків за рейтингом (наприклад, показати лише відгуки з 5 зірками);

– відгуки можуть містити фотографії та відео, що допомагає іншим користувачам краще зрозуміти товар.

2. eBay

Процес створення та перегляду відгуків:

– додавання відгуку: після покупки користувач може залишити відгук про продавця і товар, натиснувши "Leave feedback" у своєму акаунті;

– оцінка товару: користувач вибирає рейтинг від 1 до 5 зірок для товару та продавця;

– текстовий відгук: користувач пише текстовий відгук, в якому описує свій досвід покупки;

– публікація: після написання відгуку, користувач натискає кнопку "Submit" для публікації;

– перегляд відгуків: всі відгуки відображаються на сторінці товару та на сторінці профілю продавця.

Особливості:

– оцінка окремо товару та продавця;

– можливість відповіді продавця на відгуки покупців;

– система зворотного зв'язку з покупцями через приватні повідомлення для вирішення можливих проблем.

3. Alibaba

Процес створення та перегляду відгуків:

– додавання відгуку: користувач може залишити відгук про товар після завершення угоди, натиснувши "Leave Feedback" у своєму акаунті;

– оцінка товару: користувач вибирає рейтинг від 1 до 5 зірок;

– текстовий відгук: користувач пише текстовий відгук про товар та роботу продавця;

– публікація: після написання відгуку, користувач натискає кнопку "Submit" для публікації;

– перегляд відгуків: всі відгуки відображаються на сторінці товару, а також у профілі продавця.

Особливості:

– можливість додавання фотографій та відео до відгуків;

– відгуки можуть бути фільтровані за оцінками та датою;

– система рейтингу продавців на основі відгуків покупців.

Висновки

Аналіз функціоналу ревізій (відгуків) на популярних інтернет-ресурсах показує, що всі ці платформи надають користувачам зручний інтерфейс для створення та перегляду відгуків, який включає:

– можливість залишати текстові відгуки та додавати фотографії або відео;

– оцінку товару за допомогою рейтингу (від 1 до 5 зірок);

– публікацію відгуків, які можуть бути переглянуті іншими користувачами;

– сортування та фільтрація відгуків за різними критеріями (корисність, дата, оцінка).

Для розробки власної системи продажів товарів, важливо забезпечити зручний та інтуїтивно зрозумілий функціонал відгуків, враховуючи досвід цих платформ. Це підвищить довіру до вашого інтернет-магазину та допоможе іншим користувачам приймати обґрунтовані рішення щодо покупок.

Аналіз функціоналу "додавання нових товарів" на аналогічних веб-ресурсах

Функціонал додавання нових товарів є важливою частиною будь-якої платформи електронної комерції, оскільки він дозволяє продавцям оновлювати асортимент та забезпечувати покупців актуальною інформацією про товари. Розглянемо, як цей процес реалізовано на декількох популярних платформах електронної комерції.

1. Amazon

Процес додавання нових товарів:

– вхід до облікового запису продавця: продавець входить у свій Seller Central обліковий запис;

– перехід до інтерфейсу додавання товарів: продавець вибирає опцію "Add a Product" у меню;

– вибір категорії товару: продавець вибирає відповідну категорію для товару;

– заповнення інформації про товар: продавець вводить інформацію про товар, включаючи назву, опис, ціну, SKU, виробника, ключові характеристики, та додає зображення;

– налаштування інвентарю: продавець вводить кількість товарів на складі;

– доставка та опції: Продавець вказує методи доставки та опції;

– перевірка та публікація: Продавець переглядає всю введену інформацію і натискає "Submit" для публікації товару на платформі.

Особливості:

– можливість масового додавання товарів за допомогою імпорту CSV файлів;

– інтеграція з FBA (Fulfillment by Amazon) для автоматизованого управління інвентарем та доставкою;

– інструменти для оптимізації списку товарів, такі як рекомендації щодо ключових слів.

2. eBay

Процес додавання нових товарів:

– вхід до облікового запису продавця: продавець входить у свій eBay Seller Hub обліковий запис;

– перехід до інтерфейсу додавання товарів: продавець натискає "List an item" у меню;

– вибір категорії товару: продавець вводить назву товару, і eBay автоматично пропонує відповідні категорії;

– заповнення інформації про товар: продавець вводить детальну інформацію про товар, включаючи назву, опис, ціну, стан, та додає зображення;

– налаштування інвентарю та доставки: продавець вводить кількість товарів на складі, вказує методи доставки та встановлює вартість доставки;

– перевірка та публікація: продавець переглядає всю введену інформацію і натискає "List item" для публікації товару на платформі.

Особливості:

– можливість налаштування аукціону або фіксованої ціни;

– інструменти для просування товарів, такі як спонсоровані списки;

– зручний інтерфейс для управління інвентарем та відстеження продажів.

3. Alibaba

Процес додавання нових товарів:

– вхід до облікового запису продавця: продавець входить у свій Alibaba Seller Central обліковий запис;

– перехід до інтерфейсу додавання товарів: продавець натискає "Add New Product" у меню;

– вибір категорії товару: продавець вибирає відповідну категорію для товару;

– заповнення інформації про товар: продавець вводить детальну інформацію про товар, включаючи назву, опис, специфікації, ціну, мінімальне замовлення, та додає зображення;

– налаштування інвентарю та доставки: продавець вводить кількість товарів на складі, вказує методи доставки та встановлює вартість доставки;

– перевірка та публікація: продавець переглядає всю введену інформацію і натискає "Submit" для публікації товару на платформі.

Особливості:

– можливість налаштування MOQ (мінімального обсягу замовлення) для оптових продажів;

– інструменти для управління інвентарем та відстеження виконання замовлень;

– можливість додавання відео до опису товару для кращого представлення.

Висновки

Аналіз функціоналу додавання нових товарів на популярних інтернет-ресурсах показує, що всі ці платформи надають продавцям зручний інтерфейс для додавання та управління товарами, який включає:

– інтуїтивно зрозумілий процес вибору категорії товару;

– детальне заповнення інформації про товар, включаючи назву, опис, ціну, зображення та інші характеристики;

– налаштування інвентарю та методів доставки;

– перевірку та публікацію товарів на платформі.

Для розробки власної системи продажів товарів, важливо забезпечити зручний та інтуїтивно зрозумілий функціонал додавання нових товарів, враховуючи досвід цих платформ. Це допоможе продавцям легко оновлювати асортимент і підтримувати актуальність інформації про товари на вашій платформі.

Аналіз функціоналу "Реалізація сторінки замовлення" на аналогічних веб-ресурсах

Функціонал сторінки замовлення є ключовим елементом процесу покупки в інтернет-магазині. Він забезпечує користувачів можливістю переглянути своє замовлення, вибрати методи доставки та оплати, а також підтвердити покупку. Розглянемо, як цей процес реалізовано на декількох популярних платформах електронної комерції.

1. Amazon

Процес оформлення замовлення:

– перегляд кошика: користувач натискає на іконку кошика та вибирає "Proceed to Checkout";

– інформація про доставку: користувач вводить або підтверджує адресу доставки;

– методи оплати: користувач вибирає метод оплати (кредитна/дебетова картка, банківський рахунок, подарункова карта тощо);

– огляд замовлення: користувач переглядає своє замовлення, включаючи товари, кількість, ціни, податки, та вартість доставки;

– підтвердження замовлення: користувач натискає кнопку "Place your order" для підтвердження замовлення.

Особливості:

– можливість зберігати декілька адрес доставки та методів оплати для швидшого оформлення замовлення;

– опції для швидкої доставки (Prime) та відстеження замовлення;

– рекомендації додаткових товарів на сторінці підтвердження замовлення.

2. eBay

Процес оформлення замовлення:

– перегляд кошика: користувач натискає на іконку кошика та вибирає "Go to checkout";

– інформація про доставку: користувач вводить або підтверджує адресу доставки;

– методи оплати: Користувач вибирає метод оплати (PayPal, кредитна/дебетова картка тощо);

– огляд замовлення: користувач переглядає своє замовлення, включаючи товари, кількість, ціни, податки, та вартість доставки;

– підтвердження замовлення: Користувач натискає кнопку "Confirm and pay" для підтвердження замовлення.

Особливості:

– інтеграція з PayPal для швидшого та безпечнішого оформлення замовлення;

– можливість додавати примітки до замовлення для продавця;

– опції для відстеження замовлення та контакт з продавцем через платформу.

3. Alibaba

Процес оформлення замовлення:

– перегляд кошика: користувач натискає на іконку кошика та вибирає "Check out";

– інформація про доставку: користувач вводить або підтверджує адресу доставки;

– методи оплати: користувач вибирає метод оплати (кредитна/дебетова картка, банківський переказ тощо);

– огляд замовлення: користувач переглядає своє замовлення, включаючи товари, кількість, ціни, податки, та вартість доставки;

– підтвердження замовлення: Користувач натискає кнопку "Place order" для підтвердження замовлення.

Особливості:

– можливість узгодження умов доставки та оплати безпосередньо з продавцем;

– інструменти для управління оптовими замовленнями, включаючи мінімальні обсяги замовлень (MOQ);

– відстеження замовлення та зворотний зв'язок з постачальником через платформу.

Висновки

Аналіз функціоналу сторінки замовлення на популярних інтернет-ресурсах показує, що всі ці платформи надають користувачам зручний інтерфейс для оформлення замовлення, який включає:

– можливість перегляду та редагування кошика перед оформленням замовлення;

– введення або підтвердження адреси доставки;

– вибір з декількох методів оплати;

– огляд замовлення з детальним розрахунком вартості товарів, податків та доставки;

– підтвердження замовлення за допомогою натискання однієї кнопки.

Для розробки власної системи продажів товарів, важливо забезпечити зручний та інтуїтивно зрозумілий функціонал сторінки замовлення, враховуючи досвід цих платформ. Це допоможе користувачам легко та безпечно оформлювати свої замовлення, підвищуючи їх задоволеність від користування вашою системою.

Аналіз функціоналу "Адміністративна панель" на аналогічних веб-ресурсах

Адміністративна панель є ключовим інструментом для управління інтернет-магазином, оскільки вона дозволяє адміністраторам і продавцям керувати товарами, замовленнями, користувачами, аналітикою та іншими аспектами роботи платформи. Розглянемо, як цей процес реалізовано на декількох популярних

платформах електронної комерції.

1. Shopify

Функціонал адміністративної панелі:

– панель керування (Dashboard): відображає ключові метрики, такі як продажі, відвідуваність сайту, замовлення, а також повідомлення про нові дії;

– управління товарами (Products): додавання, редагування та видалення товарів, управління варіаціями товарів, імпорт та експорт товарів через CSV файли;

– управління замовленнями (Orders): перегляд, обробка та статус замовлень, створення замовлень вручну, друк етикеток для відправки;

– клієнти (Customers): управління інформацією про клієнтів, перегляд історії покупок, сегментація клієнтів;

– аналітика (Analytics): деталізовані звіти про продажі, поведінку клієнтів, трафік, та інші ключові показники;

– маркетинг (Marketing): управління маркетинговими кампаніями, налаштування знижок та промокодів, інтеграція з маркетинговими інструментами;

– налаштування магазину (Settings): управління налаштуваннями магазину, платіжними методами, доставкою, податками, мовами, та дизайном сайту.

Особливості:

– інтеграція з численними додатками для розширення функціоналу;

– зручний та інтуїтивно зрозумілий інтерфейс;

– система сповіщень та автоматизованих дій (workflows).

2. WooCommerce (WordPress)

Функціонал адміністративної панелі:

– панель керування (Dashboard): відображає основні показники магазину, такі як продажі, замовлення, відвідуваність;

– управління товарами (Products): додавання, редагування та видалення товарів, управління категоріями та тегами, налаштування атрибутів товарів;

– управління замовленнями (Orders): перегляд і обробка замовлень, управління статусами замовлень, генерація рахунків-фактур;

– клієнти (Customers): перегляд і управління інформацією про клієнтів, історія замовлень;

– звіти (Reports): детальні звіти про продажі, клієнтів, товари, купони;

– налаштування (Settings): управління налаштуваннями магазину, платежами, доставкою, податками, інтеграціями з іншими сервісами.

Особливості:

– велика кількість плагінів для розширення функціоналу;

– гнучкість і налаштовуваність завдяки платформі WordPress;

– можливість створення індивідуальних дизайнів і функцій.

3. Magento

Функціонал адміністративної панелі:

– панель керування (Dashboard): відображає основні метрики магазину, такі як продажі, замовлення, відвідування, середній чек;

– управління товарами (Catalog): додавання, редагування та видалення товарів, управління атрибутами та категоріями товарів, імпорт та експорт товарів;

– управління замовленнями (Sales): перегляд і обробка замовлень, управління рахунками, доставкою, кредитними нотами;

– клієнти (Customers): управління інформацією про клієнтів, групування клієнтів, управління підписками на розсилки;

– маркетинг (Marketing): налаштування промокодів, знижок, управління кампаніями, сегментація клієнтів;

– звіти та аналітика (Reports): детальні звіти про продажі, замовлення, товари, податки;

– налаштування (Configuration): управління загальними налаштуваннями магазину, інтеграціями, платіжними методами, доставкою, безпекою.

Особливості:

– високий рівень налаштовуваності та масштабованості;

– підтримка багатомовності та багатовалютності;

– інтеграція з ERP та CRM системами.

Висновки

Аналіз функціоналу адміністративної панелі на популярних інтернет-ресурсах показує, що всі ці платформи надають адміністраторам і продавцям потужні інструменти для управління магазинами, які включають:

– зручний інтерфейс панелі керування з основними метриками;

– управління товарами, замовленнями, клієнтами та аналітикою;

– інтеграція з маркетинговими інструментами та додатками для розширення функціоналу;

– налаштування магазину, включаючи платежі, доставку, податки та дизайн.

Для розробки власної системи продажів товарів, важливо забезпечити зручний та інтуїтивно зрозумілий функціонал адміністративної панелі, враховуючи досвід цих платформ. Це дозволить адміністраторам і продавцям ефективно керувати своїм інтернет-магазином та оптимізувати процеси для покращення обслуговування клієнтів.

Аналіз функціоналу "база даних" на аналогічних веб-ресурсах

База даних є ключовим компонентом будь-якої платформи електронної комерції, оскільки вона зберігає і керує всією інформацією про товари, замовлення, користувачів та інші аспекти роботи магазину. Розглянемо, як цей функціонал реалізовано на прикладі популярних платформ електронної комерції.

1. Shopify

Архітектура бази даних:

– модель даних: Shopify використовує реляційну базу даних для зберігання інформації про товари, замовлення, клієнтів та інші сутності;

– таблиці: Основні таблиці включають products, orders, customers, inventory, transactions, та інші;

– зв'язки між таблицями: Встановлюються через зовнішні ключі, наприклад, order_id у таблиці order_items для зв'язку з таблицею orders.

Особливості:

– масштабованість: використання шардінгу та реплікації для масштабування бази даних;

– безпека: зашифроване зберігання конфіденційної інформації, регулярні резервні копії;

– доступність: використання хмарних сервісів для високої доступності та надійності.

2. WooCommerce (WordPress)

Архітектура бази даних:

– модель даних: WooCommerce зберігає дані у базі даних WordPress, яка є реляційною базою даних (MySQL);

– таблиці: основні таблиці включають wp_posts (для товарів), wp_postmeta (метадані,товарів),wp_woocommerce_order_items, wp_woocommerce_order_itemmeta, wp_users, wp_usermeta, та інші;

– зв'язки між таблицями: встановлюються через зовнішні ключі, наприклад, post_id у таблиці wp_postmeta для зв'язку з таблицею wp_posts.

Особливості:

– гнучкість: легке розширення структури бази даних за допомогою плагінів;

– інтеграція: підтримка інтеграції з багатьма іншими плагінами та сервісами через API;

– безпека: використання стандартних засобів безпеки WordPress, регулярні резервні копії.

3. Magento

Архітектура бази даних:

– модель даних: Magento використовує реляційну базу даних (MySQL) для зберігання великого обсягу інформації про товари, замовлення, клієнтів та інші сутності;

– таблиці: основні таблиці включають catalog_product_entity, sales_order, customer_entity, eav_attribute, inventory_stock, та інші;

– зв'язки між таблицями: використовуються зовнішні ключі для зв'язків, наприклад, entity_id у таблиці catalog_product_entity для зв'язку з іншими таблицями.

Особливості:

– масштабованість: підтримка шардінгу та реплікації для масштабування бази даних;

– гнучкість: складна модель даних з використанням EAV (Entity-Attribute-Value) для зберігання атрибутів товарів;

– безпека: використання сучасних методів шифрування, регулярні резервні копії.

Аналіз функціоналу бази даних на популярних платформах електронної комерції показує, що всі вони використовують реляційні бази даних для зберігання ключових даних про товари, замовлення та клієнтів. Основні аспекти включають:

– масштабованість: використання шардінгу та реплікації для обробки великої кількості даних і запитів;

– безпека: зашифроване зберігання конфіденційної інформації, регулярні резервні копії;

– гнучкість: можливість розширення моделі даних та інтеграції з іншими сервісами.

Для розробки власної системи продажів товарів, важливо забезпечити надійну і масштабовану архітектуру бази даних, яка дозволить ефективно керувати всіма аспектами магазину та забезпечувати високу продуктивність і безпеку даних.

2.2 Вибір середовища бази даних для проекту

У цьому розділі ми розглянемо всі аспекти реалізації бази даних та адміністративної панелі для нашого інтернет-магазину. Перш ніж приступати до розробки, важливо визначити потреби вашого додатку та вибрати оптимальний тип бази даних. Розглянемо такі пункти:

– аналіз вимог до бази даних: визначимо, яка інформація повинна бути збережена у вашій базі даних. Це можуть бути дані про товари, користувачів, замовлення, категорії товарів тощо;

– вибір типу бази даних: розглянемо різні типи баз даних, такі як реляційні (наприклад, MySQL, PostgreSQL) та нереляційні (наприклад, MongoDB). Оберемо той, що найбільше підходить для потреб вашого проекту;

– проектування схеми бази даних: розробимо схему бази даних, включаючи таблиці, поля, зв'язки між таблицями та індексацію;

– реалізація бази даних: розглянемо процес створення та налаштування бази даних, створення таблиць, введення початкових даних та встановлення прав доступу.

Адміністративна панель дозволяє вам керувати та адмініструвати ваш магазин, додавати нові товари, відстежувати замовлення, керувати користувачами тощо. Розглянемо такі аспекти:

– функціональні вимоги: проаналізуємо всі функції, які повинні бути доступні в адміністративній панелі. Це може включати додавання, редагування та видалення товарів, керування замовленнями, управління користувачами та багато іншого;

– дизайн інтерфейсу користувача: розробимо зручний та інтуїтивно зрозумілий інтерфейс адміністративної панелі, який дозволить легко керувати вашим магазином;

– реалізація функціональності: програмно реалізуємо всі функції адміністративної панелі, забезпечуючи зручний та безпечний доступ до функцій управління магазином.

2.3 Сторінка замовлення та додавання нових товарів

У цьому розділі ми розглянемо створення сторінки замовлення та додавання нових товарів за допомогою адміністративної панелі для вашого інтернет-магазину, яка дозволить користувачам здійснювати покупки. Розглянемо такі аспекти:

– функціональні вимоги до сторінки замовлення: визначимо всі етапи процесу замовлення, включаючи вибір товарів, введення контактної інформації, вибір методу доставки та оплати, підтвердження замовлення;

– інтерфейс користувача: розробимо зручний та привабливий інтерфейс сторінки замовлення, який буде забезпечувати зручний та інтуїтивно зрозумілий процес покупки;

– реалізація функціональності: програмно реалізуємо всі етапи процесу замовлення, забезпечуючи збереження даних замовлення в базі даних та взаємодію з системами оплати та доставки.

При додаванні нових товарів важливо забезпечити зручний та ефективний інтерфейс для адміністраторів магазину.

Розглянемо такі аспекти:

– форма додавання товарів: розробимо форму додавання товарів, яка буде містити поля для введення основних характеристик товару, таких як назва, опис, ціна, зображення, категорія тощо;

– мультимедійний контент: врахуємо можливість завантаження зображень та відео товарів, а також опис товарів у візуальному редакторі, який дозволить адміністраторам зручно форматувати текст;

– валідація даних: забезпечимо валідацію введених даних, щоб уникнути помилок та забезпечити консистентність даних в базі;

– збереження даних: реалізуємо механізми для збереження введених даних про нові товари в базі даних магазину та їх подальше відображення на сайті.

2.4 Ревізії від користувачів та кошик з товарами

У цьому підрозділі ми розглянемо процес ревізій від користувачів та кошик для користувачів, які дозволять користувачам залишати відгуки, оцінки для товарів у нашому інтернет-магазині та перевірити весь перелік обраних товарів для замовлення. Ревізії від користувачів є важливою частиною магазину, оскільки вони допомагають іншим користувачам приймати ухвалення щодо покупки. Розглянемо такі аспекти:

– форма ревізій: розробимо форму, за допомогою якої користувачі можуть залишати свої відгуки та оцінки для товарів. Форма може містити поля для текстового коментаря, вибору оцінки за п'ятибальною шкалою тощо;

– модерація ревізій: розглянемо можливість модерації ревізій, щоб перевіряти коментарі користувачів перед їх публікацією на сайті. Це дозволить уникнути публікації неприйнятних чи образливих коментарів;

– відображення ревізій: забезпечимо механізми для відображення ревізій користувачів на сторінці товару, щоб інші користувачі могли бачити відгуки та оцінки товарів перед покупкою.

Реалізація кошика є важливою складовою будь-якого інтернет-магазину, оскільки вона дозволяє користувачам зберігати товари, які вони бажають придбати, перед оформленням замовлення. Нижче наведено детальний опис процесу реалізації кошика:

– додавання товарів до кошика: користувач має можливість додавати товари до свого кошика, натискаючи на відповідну кнопку «Додати до кошика» на сторінці продукту. Ця дія запускає процес додавання товару до списку товарів у кошику;

– відображення вмісту кошика: після додавання товарів користувач має можливість переглянути вміст свого кошика на окремій сторінці. Тут він може перевірити список доданих товарів, їх кількість та загальну вартість;

– зміна кількості товарів: користувач має можливість змінювати кількість товарів у кошику, встановлюючи бажану кількість для кожного товару;

– видалення товарів з кошика: користувач може видалити товар зі свого кошика, якщо він вирішить відмовитися від покупки цього товару;

– оформлення замовлення: після того, як користувач завершить додавання товарів до кошика та перевірки їх, він може перейти до процесу оформлення замовлення, де він вказує адресу доставки, спосіб оплати та інші деталі.

2.5 Створення аккаунту для користувачів та оплата замовлення

Створення облікового запису для користувача є важливим етапом для забезпечення персоналізованого досвіду використання інтернет-магазину. Нижче наведено кроки, необхідні для реалізації цього функціоналу:

– реєстрація нового користувача: користувач має можливість зареєструвати новий обліковий запис, вказавши свою електронну адресу та пароль;

– авторизація користувача: після успішної реєстрації користувач може увійти до свого облікового запису, використовуючи введену раніше електронну адресу та пароль;

– персоналізований досвід: користувачеві надається персоналізований досвід використання магазину, включаючи можливість зберігати адреси доставки, переглядати історію замовлень та налаштовувати інші параметри облікового запису.

Процес оплати замовлення є критично важливою частиною функціоналу інтернет-магазину, яка повинна бути надійною та безпечною. Нижче наведено основні кроки для реалізації цього процесу:

– вибір способу оплати: користувач має можливість вибрати бажаний спосіб оплати замовлення, такий як оплата банківською картою, платіжний сервіс, оплата при отриманні тощо;

– введення платіжних даних: користувач повинен ввести необхідні платіжні дані, такі як номер кредитної карти, термін дії, CVV-код та інші відомості, в залежності від обраного способу оплати;

– підтвердження оплати: після введення платіжних даних система здійснює оплату замовлення та повертає користувачеві підтвердження про успішну оплату;

– оформлення замовлення: після підтвердження оплати замовлення, користувач може завершити процес оформлення замовлення, отримавши підтвердження про успішне прийняття його замовлення.

3 РЕАЛІЗАЦІЯ СИСТЕМИ ПОШУКУ З АЛГОРИТМОМ ПОШУКУ

3.1 Реалізація основних функцій управління користувачами

У даному підрозділі розглянемо код для управління користувачами в системі продажу товарів, який включає функції аутентифікації, реєстрації, оновлення профілю, отримання списку користувачів, видалення користувачів та отримання інформації про конкретного користувача. Для цього використовуються наступні технології та бібліотеки: express-async-handler(рис. 3.1) для обробки асинхронних функцій, mongoose для роботи з MongoDB, а також власні утиліти для генерації токенів.[5]

```
import asyncHandler from "express-async-handler";
import User from "../models/userModel.js";
import generateToken from "../utils/generateToken.js";
```

Рисунок 3.1 - Допоміжні бібліотеки «e-a-h» та «mongoose»

3.1.1 Аутентифікація користувача

Нижче на малюнку(рис 3.2) можна побачити функцію яка аутентифікує користувача за допомогою email та паролю

```
export const authUser = asyncHandler(async (req, res) => {
  const { email, password } = req.body;
  const user = await User.findOne({ email });

  if (user && (await user.matchPassword(password))) {
    res.status(201).json({
      success: true,
      user: {
        id: user._id,
        name: user.name,
        email: user.email,
        isAdmin: user.isAdmin,
        token: generateToken(user._id),
      },
    });
  } else {
    res.status(401);
    throw new Error("Invalid email or password");
  }
});
```

Рисунок 3.2 - Функція аутентифікації

Як можна побачити якщо всі дані вірні, то функція генерує токен та успішно авторизує користувача, в інакшому випадку ми отримаємо error 401(«Invalid email or password»)

3.1.2 Отримання профілю користувача

Після того як юзер успішно авторизується він отримує свої дані(рис. 3.3), якщо вони існують.

```
export const getUserProfile = asyncHandler(async (req, res) => {
  const user = await User.findById(req.user.id);

  if (user) {
    res.status(201).json({
      success: true,
      user: {
        id: user._id,
        name: user.name,
        email: user.email,
        isAdmin: user.isAdmin,
      },
    });
  } else {
    res.status(401);
    throw new Error("User not found");
  }
});
```

Рисинук 3.3 - Отримання даних користувачем

В інакшому випадку, якщо користувача не буде знайдено, буде помилка 401(«User not found»)

3.1.3 Реєстрація нового користувача

Якщо користувач у перший раз в інтернет магазині, то він зможе

зараєструватися.

Функція яка за це відповідає наведена нижче на малюнку(рис 3.4).

```
export const registerUser = asyncHandler(async (req, res) => {
  const { name, email, password } = req.body;
  const userExist = await User.findOne({ email });

  if (userExist) {
    res.status(401);
    throw new Error("User already exist");
  }

  const user = await User.create({ name, email, password });

  res.status(201).json({
    success: true,
    user: {
      id: user._id,
      name: user.name,
      email: user.email,
      isAdmin: user.isAdmin,
      token: generateToken(user._id),
    },
  });
});
```

Рисунок 3.4 - Функція реєстрації

Як ми бачимо, після вводу даних, якщо такого користувача не знайдено його буде успішно зареєстровано в інакшому випадку юзер отримає помилку, яка сповістить його, що такий юзер вже існує.

3.2 Реалізація основних функцій управління замовленнями

У даному розділі розглянемо код для управління замовленнями в системі електронної комерції. Він включає функції додавання замовлень, отримання деталей замовлення, оновлення статусу оплати та доставки, а також отримання

списку замовлень. Для реалізації цих функції ми використовуємо ті ж самі технології і бібліотеки, що наведені на рисунку 3.1

3.2.1 Додавання замовлення

Ця функція(рисунок 3.5) додає нове замовлення до бази даних. Вона перевіряє наявність елементів замовлення, створює нове замовлення та зберігає його. У разі успіху, повертається відповідь з кодом 201 та деталями замовлення.

```
export const addOrderItems = asyncHandler(async (req, res) => {
  const {
    orderItems,
    shippingAddress,
    paymentMethod,
    taxPrice,
    shippingPrice,
    totalPrice,
  } = req.body;

  if (orderItems && orderItems.length === 0) {
    res.status(401);
    throw new Error("No order items");
  }

  const createdOrder = new Order({
    user: req.user._id,
    orderItems,
    shippingAddress,
    paymentMethod,
    taxPrice,
    shippingPrice,
    totalPrice,
  });

  await createdOrder.save();

  res.status(201).json({ success: true, order: createdOrder });
});
```

Рисунок 3.5 - Функція додавання замовлення

3.2.2 Оновлення статусу оплати замовлення

Функція оновлює статус оплати замовлення(рис. 3.6). Вона встановлює поля isPaid, paidAt та paymentResult, після чого зберігає зміни в базу даних. У разі успіху повертається відповідь з кодом 201 та оновленими даними замовлення.

```
export const updateOrderToPaid = asyncHandler(async (req, res) => {
  let order = await Order.findById(req.params.id);

  if (!order) {
    res.status(401);
    throw new Error("Order not found");
  }

  order.isPaid = true;
  order.paidAt = Date.now();
  order.paymentResult = {
    id: req.body.id,
    status: req.body.status,
    update_time: req.body.update_time,
    email_address: req.body.payer.email_address,
  };

  const updatedOrder = await order.save();

  res.status(201).json({ success: true, order: updatedOrder });
});
```

Рисунок 3.6 - Функція оновлення статусу оплати замовлення

3.3 Реалізація функціоналу кошика та ревізій від користувачів

В цьому розділі ми розглянемо код та реалізацію кошику та ревізій на товар від користувачів.

Кошик дозволяє користувачам додавати товари, видаляти їх, зберігати адресу доставки та метод оплати. Для цього використовуються бібліотеки axios(рис. 3.7) для HTTP-запитів та redux для управління станом.

```
import axios from "axios";
import * as actions from "../constants/cartConstants";
```

Рисунок 3.7 - Бібліотеки і технології для реалізації кошику

3.3.1 Додавання товарів до кошику

Функція addToCart(рис. 3.8) додає товар до кошика. Вона виконує HTTP-запит до API для отримання деталей товару за його ідентифікатором, а потім відправляє відповідну дію до Redux-стану.

```
export const addToCart = (id, qty) => async (dispatch, getState) => {
  const { data } = await axios.get(`/api/products/${id}`);

  dispatch({
    type: actions.CART_ADD_ITEM,
    payload: {
      product: data.product._id,
      name: data.product.name,
      image: data.product.image,
      price: data.product.price,
      countInStock: data.product.countInStock,
      qty,
    },
  });

  localStorage.setItem("cartItems", JSON.stringify(getState().cart.cartItems));
};
```

Рисунок 3.8 - Реалізація функції addToCart

У цій функції:

– виконується запит до API для отримання інформації про товар за його id;

– викликається дія CART_ADD_ITEM з деталями товару, який додається до кошика;

– зберігаються елементи кошика в localStorage для збереження стану кошика між сесіями.

3.3.2 Видалення товарів з кошику

Функція removeFromCart(рис. 3.9) видаляє товар з кошика. Вона відправляє відповідну дію до Redux-стану та оновлює localStorage.

```
export const removeFromCart = (id) => async (dispatch, getState) => {
  dispatch({ type: actions.CART_REMOVE_ITEM, payload: id });

  localStorage.setItem("cartItems", JSON.stringify(getState().cart.cartItems));
};
```

Рисунок 3.9 - Реалізація функції removeFromCart

У цій функції:

– викликається дія CART_REMOVE_ITEM з ідентифікатором товару, який необхідно видалити з кошика;

– оновлюються елементи кошика в localStorage.

3.3.3 Створення відгуку про товар

Функція createProductReview(рис. 3.10 створює новий відгук про товар.

Рисунок 3.10 - Функція createProductReview

У цій функції:

– відправляється дія PRODUCT_CREATE_REVIEW_REQUEST перед початком запиту;

– отримується токен користувача для авторизації запиту;

– виконується запит до API для створення нового відгуку про товар за його id.

– у разі успіху, відправляється дія PRODUCT_CREATE_REVIEW_SUCCESS;

– у разі помилки, якщо помилка пов'язана з авторизацією, відбувається вихід користувача з системи, та відправляється дія PRODUCT_CREATE_REVIEW_FAIL з повідомленням про помилку.

3.4 Управління товарами у системі

Тут ми розглянемо реалізацію та код, який забезпечує управління товарами у системі електронної комерції. Основні функції включають отримання списку товарів, отримання деталей товару, створення, оновлення та видалення товарів, створення відгуків і отримання топових товарів. Для реалізації HTTP-запитів використовується бібліотека axios, а для управління станом — бібліотека redux.

```
import * as actions from "../constants/productConstants";
import axios from "axios";
import { logout } from "./userActions";
```

Рисунок 3.11 - Допоміжні бібліотеки для реалізації системи управління товарами

3.4.1 Створення нового товару

Нижче буде відображена реалізація функції(рис. 3.12) яка буде додавати наш товар.

```
export const createProduct = (dataProduct) => async (dispatch, getState) => {
  try {
    dispatch({ type: actions.PRODUCT_CREATE_REQUEST });

    const { userLogin: { userInfo } } = getState();

    const config = {
      headers: {
        "Content-Type": "application/json",
        Authorization: `Bearer ${userInfo.token}`,
      },
    };

    await axios.post(`/api/products`, dataProduct, config);

    dispatch({ type: actions.PRODUCT_CREATE_SUCCESS });
  } catch (error) {
    const message = error.response && error.response.data.message ? error.response.data.me
    if (message === "not authorized, no token") {
      dispatch(logout());
    }
    dispatch({
      type: actions.PRODUCT_CREATE_FAIL,
      payload: message,
    });
  }
};
```

Рисунок 3.12 - Реалізація функції createProduct

Ця функція робить наступне:

– відправляється дія PRODUCT_CREATE_REQUEST перед початком запиту;

– отримується токен користувача для авторизації запиту;

– виконується запит до API для створення нового товару;

– у разі успіху, відправляється дія PRODUCT_CREATE_SUCCESS;

– у разі помилки, якщо помилка пов'язана з авторизацією, відбувається вихід користувача з системи, та відправляється дія PRODUCT_CREATE_FAIL з повідомленням про помилку.

3.4.2 Видалення вже створеного товару

Функція deleteProduct(рис. 3.13) видаляє товар за його ідентифікатором.

```
export const deleteProduct = (id) => async (dispatch, getState) => {
  try {
    dispatch({ type: actions.PRODUCT_DELETE_REQUEST });

    const { userLogin: { userInfo } } = getState();

    const config = {
      headers: {
        "Content-Type": "application/json",
        Authorization: `Bearer ${userInfo.token}`,
      },
    };

    await axios.delete(`/api/products/${id}`, config);

    dispatch({ type: actions.PRODUCT_DELETE_SUCCESS });
  } catch (error) {
    const message = error.response && error.response.data.message ? error.response.data.me
    if (message === "not authorized, no token") {
      dispatch(logout());
    }
    dispatch({
      type: actions.PRODUCT_DELETE_FAIL,
      payload: message,
    });
  }
};
```

Рисунок 3.13 - Код функції deleteProduct

Під час виконання функції відбувається наступне:

– відправляється дія PRODUCT_DELETE_REQUEST перед початком запиту;

– отримується токен користувача для авторизації запиту;

– виконується запит до API для видалення товару за його id;

– у разі успіху, відправляється дія PRODUCT_DELETE_SUCCESS;

– у разі помилки, якщо помилка пов'язана з авторизацією, відбувається вихід користувача з системи, та відправляється дія PRODUCT_DELETE_FAIL з повідомленням про помилку.

3.4 Створення та встановлення MongoDB до проекту

Останнім кроком нашого проекту є база даних, проаналізувавши усі можливі варіант, я зупинився на хмарному середовищі MongoDB Atlas, адже це дозволяє

створити кластер бази даних у хмарі, що забезпечує високу доступність і простоту управління.

Мною було створено аккаунт та кластер, після чого я отримав URI для підключення який буде виглядати приблизно так: mongodb+srv://<username>:<password>@cluster0.mongodb.net/<dbname>?retryWrites=true&w=majority.

Далі я реалізував функцію(рис. 3.14) для підключення яку можна побачити нижче:

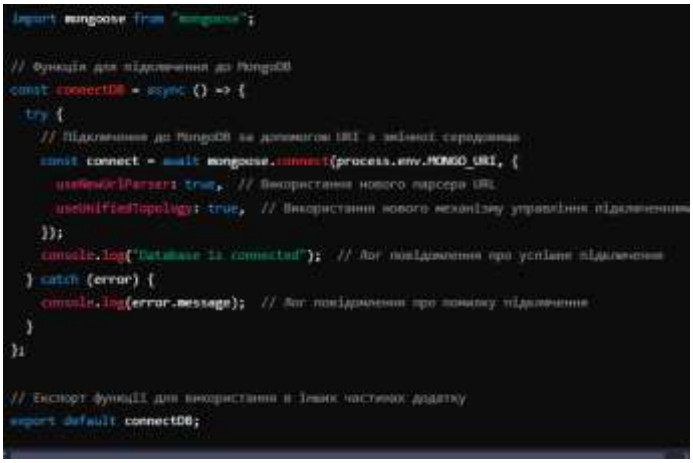

Рисунок 3.14 - Функція connectDB

Ця функція робить наступне:

– імпорт Mongoose: Імпортується бібліотека Mongoose для роботи з MongoDB.[5]

– функція connectDB: Це асинхронна функція, яка намагається підключитися до бази даних MongoDB, використовуючи URI з змінної середовища MONGO_URI;

– конфігурація: Параметри useNewUrlParser і useUnifiedTopology використовуються для уникнення старих механізмів MongoDB та забезпечення стабільного підключення;

– логування: Якщо підключення успішне, в консоль виводиться повідомлення "Database is connected". Якщо сталася помилка, виводиться повідомлення з текстом помилки;

– експорт функції: Функція експортується для використання в інших частинах додатку.

Для використання цієї функції, імпортував її у основний файл додатку (наприклад, server.js) і викликав перед запуском сервера(рис 3.15):

```
import express from 'express';
import connectDB from './config/db.js';
import dotenv from 'dotenv';

dotenv.config();

const app = express();

// Підключення до бази даних
connectDB();

const PORT = process.env.PORT || 5000;

app.listen(PORT, console.log(`Server running on port ${PORT}`));
```

Рисунок 3.15 - Імпорт функції у основний файл

Це забезпечує підключення до бази даних перед тим, як сервер почне слухати запити.

Результат(рис. 3.16) який ми отримали після підключення БД, на рисунку зображені дані продуктів які на даний момент існують.

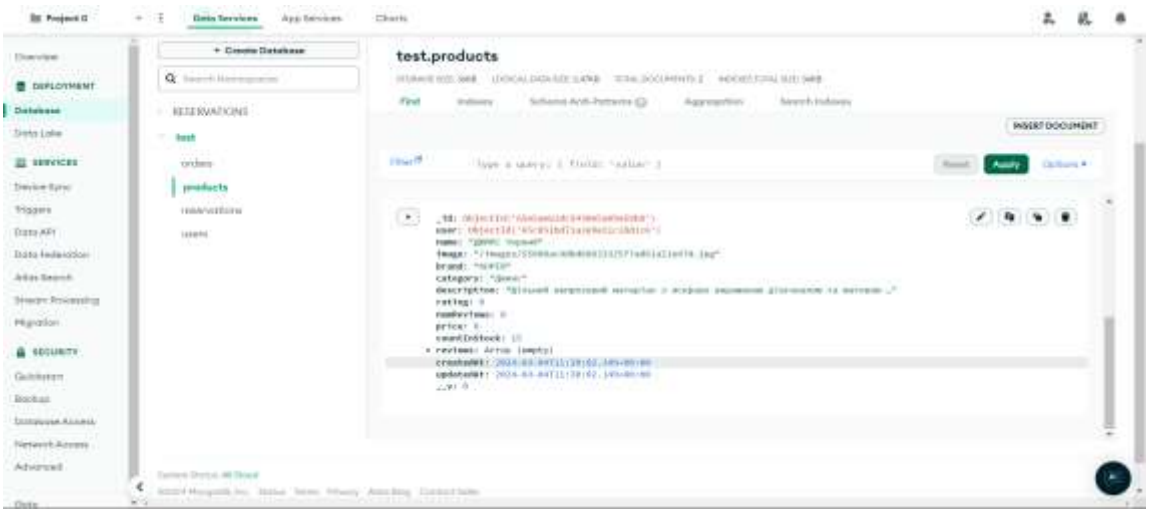

Рисунок 3.16 - Функціонування бази даних

3.5 Реалізація алгоритму пошуку

Реалізація алгоритму пошуку(рис. 3.17) продуктів у цьому проекті забезпечує користувачам можливість знаходити продукти на основі ключових слів, фільтрації за категоріями, а також підтримує пагінацію для зручного перегляду результатів.

Рисунок 3.17 - Реалізація алгоритму пошуку продуктів

Як ми бачимо функція працює наступним чином:

Параметри пагінації та пошуку:

– pageSize визначає кількість продуктів, що відображаються на одній сторінці;

– page визначає номер сторінки, яка запитується;

– keyword визначає умову пошуку, яка базується на ключовому слові, введеному користувачем. Якщо ключове слово задано, формується об'єкт для пошуку з використанням регулярного виразу ($regex) та опції і для нечутливого до регістру пошуку.

Отримання кількості документів.

Count зберігає кількість продуктів, які відповідають умовам пошуку.

Запит до бази даних.

Products зберігає масив продуктів, що відповідають умовам пошуку, обмежених параметрами пагінації (limit і skip).

Відповідь сервера.

Сервер відправляє відповідь з масивом продуктів, поточною сторінкою та загальною кількістю сторінок(рис. 3.18).

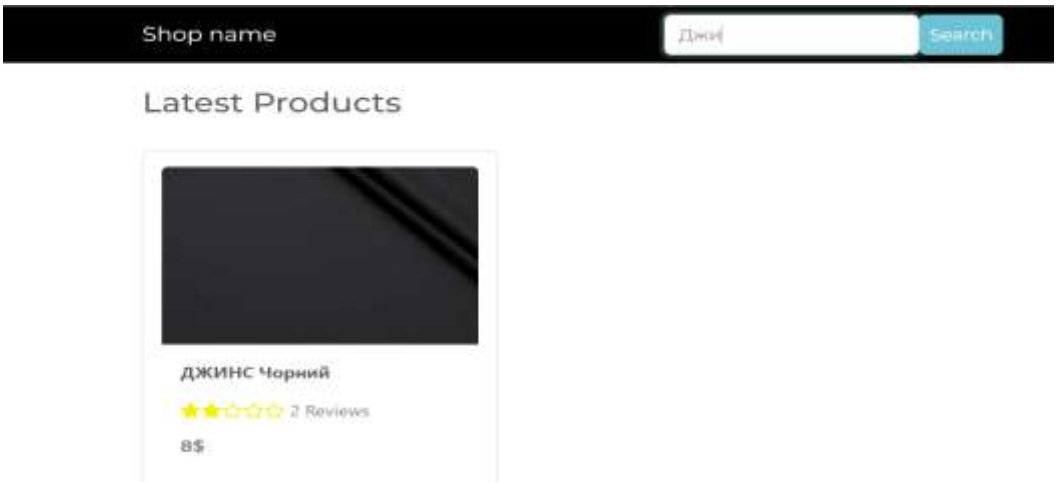

Рисунок 3.18 - Функціонал алгоритму пошуку продуктів

ВИСНОВКИ

Під час виконання кваліфікаційної роботи було докладено зусиль для розробки інтернет-магазину з використанням Node.js для серверної частини та JavaScript для клієнтської. Проект включав реалізацію алгоритму пошуку, адміністративної панелі, сторінки замовлення, додавання нових товарів та ревізій від користувачів.

У процесі розробки було вивчено та застосовано кращі практики веб-розробки, зокрема використання Node.js та його екосистеми, яка включає в себе npm для управління пакетами. Використання однієї мови програмування на клієнтській та серверній стороні (JavaScript) спростило розробку та підтримку проекту.

Реалізовано адміністративну панель, що дозволяє керувати товарами, замовленнями та користувачами, забезпечуючи зручний та ефективний інтерфейс для адміністраторів магазину. Також було створено сторінку замовлення, яка дозволяє користувачам здійснювати покупки зручно та швидко.

Додавання нових товарів та ревізій від користувачів було успішно реалізовано з урахуванням зручності та безпеки введення даних.

У результаті цієї роботи був створений функціональний та ефективний інтернет-магазин, який може використовуватися для торгівлі та надання послуг в Інтернеті.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Савінова А. В., Шамшурін Д. О. Розробка веб-додатків з використанням Node.js та Express.js. Київ, 2020.

2. Дубінін А. В., Іванова О. П. Основи програмування на JavaScript. Харків, 2019.

3. Максимов С. Повний курс по React і Redux. Київ, 2018.

4. Флэнаган Д. JavaScript. Підручник по сучасному JavaScript-програмуванню. Київ, 2017.

5. Макфарланд Д. MongoDB в дії. Харків, 2019.

6. Кейсі Т. Node.js в дії. Львів, 2016.

7. Волкін П. Програмування на Node.js: Розробка під веб-сервери, настільні та мобільні платформи. Харків, 2018.

8. Мур Д. Node.js, MongoDB та AngularJS веб-розробка. Київ, 2017.

9. Ричардсон Л. React.js Essentials. Львів, 2018.

10. Вільямс М. Express.js Повний курс. Від початку до кінця. Київ, 2020.

11. Крайнєв Д. В., Павленко О. І. Основи веб-розробки. Одеса, 2020.

12. Браун Е. Професійний Node.js. Практичний посібник для розробників. Дніпро, 2019.

13. Гриньов І. О., Коваль С. П. Сучасні підходи до розробки веб-застосунків. Львів, 2018.

14. Сміт Дж. Angular для початківців: покрокове керівництво. Київ, 2019.

15. Фішер Д. Розширене програмування на JavaScript. Харків, 2017.

16. Харрінгтон Б. Архітектура програмних систем на основі Node.js. Львів, 2020.

17. Іванова А. Веб-розробка з використанням JavaScript та Node.js. Київ, 2018.

18. Робінсон П. Розробка масштабованих веб-застосунків на Express.js. Харків, 2019.

19. Міллер К. Поглиблене вивчення MongoDB. Львів, 2018.

20. Блейк Дж. Повний гід по Full-Stack розробці. Київ, 2020.

21. Поляков О. А., Кравченко І. В. Node.js: Поглиблений курс. Дніпро, 2019.

22. Гарднер М. Проектування веб-застосунків з використанням React та Redux. Харків, 2018.

23. Шелдон А. Основи кібербезпеки для веб-розробників. Львів, 2019.

24. Жуков С. В. Практичне керівництво з MongoDB та Node.js. Київ, 2017.

25. Тихонов П. А., Ковальчук О. І. Інтеграція AngularJS з Node.js: Практичні поради. Одеса, 2020.