Міністерство освіти і науки України

Луцький національний технічний університет

Факультет комп'ютерних та інформаційних технологій

Кафедра інженерії програмного забезпечення

КВАЛІФІКАЦІЙНА РОБОТА ЗА СТУПЕНЕМ ВИЩОЇ ОСВІТИ «БАКАЛАВР»

СТВОРЕННЯ ПРОГРАМНОГО ЗАСТОСУНКУ ДЛЯ РЕАЛІЗАЦІЇ ПРОЦЕСІВ ЕЛЕКТРОННОЇ КОМЕРЦІЇ НА БАЗІ NODE.JS

CREATION OF A SOFTWARE APPLICATION FOR THE IMPLEMENTATION OF E-COMMERCE PROCESSES BASED ON NODE.JS

спеціальність 121 Інженерія програмного забезпечення
(шифр і назва спеціальності)

освітня програма «Інженерія програмного забезпечення»
(назва освітньої програми)

Виконав. здобувач вищої освіти
Групи ІПЗс-21
Лис Ярослав Петрович
_____
(підпис)

Керівник.
к.т.н., доцент
Ящук Андрій Анатолійович
_____
(підпис)

Кваліфікаційну роботу
допущено до захисту
«8» 06 2024 р.
к.т.н., доцент
Гарант освітньої програми.
Ліщина Наталія Миколаївна
_____
(підпис)

Луцьк – 2024 року

ЛУЦЬКИЙ НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ

Факультет *комп'ютерних та інформаційних технологій*
Кафедра *інженерії програмного забезпечення*
Ступінь вищої освіти. *бакалавр*
Галузь знань. *12 Інформаційні технології*
Спеціальність. *121 Інженерія програмного забезпечення*
Освітня програма. *«Інженерія програмного забезпечення»*

ЗАТВЕРДЖУЮ
Завідувач кафедри
________ Н. Мирна
«2» __02__ 2024 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ ЗДОБУВАЧУ ВИЩОЇ ОСВІТИ

Лису Ярославу Петровичу
(прізвище, ім'я, по батькові)

1. Тема кваліфікаційної роботи: *Створення програмного застосунку для реалізації процесів електронної комерції на базі Node.js*

Керівник роботи: *к.т.н., доцент, Ящук Андрій Анатолійович*
затверджені наказом закладу вищої освіти від «*30*» *грудня 2023 р.* № *477/01-02*
2. Строк подання здобувачем вищої освіти кваліфікаційної роботи *«5» червня 2024 р.*
3. Вихідні дані до роботи: *технічне та програмне забезпечення ЕОМ, вимоги до розробки програмного забезпечення, ергономічні вимоги до функціонування програмного засобу.*

4. Зміст розрахунково-пояснювальної записки (перелік питань, що потрібно розробити): *Аналіз сучасного стану проблеми, існуючих методів і засобів її розв'язання, аналіз, визначення вимог до розроблюваного програмного забезпечення та проектування програмного забезпечення, опис функціонального наповнення об'єкта проектування, практична реалізація об'єкта проектування.*

5. Перелік графічного матеріалу: *17 рисунків*

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис	
		завдання видав	завдання прийняв
Аналіз предметної області	*Ящук А. А.*		
Специфікації вимог до розробленої системи	*Ящук А. А.*		
Розробка об'єкта проєктування	*Ящук А. А.*		
Нормоконтроль	*Повстяна Ю. С.*		
Гарант ОП	*Ліщина Н. М.*		
Показник запозичень тексту	3,9 %		
Академічна доброчесність	*Ящук А. А.*		

7. Дата видачі завдання *«2» лютого 2024 р.*

КАЛЕНДАРНИЙ ПЛАН

№ З/п	Назва етапів кваліфікаційної роботи бакалавра	Строк виконання етапів роботи	Примітка
1	*Огляд літературних джерел по темі кваліфікаційної роботи бакалавра*	*07.02.2024 р.*	вик.
2	*Аналіз проблеми розробки та впровадження об'єкту проектування*	*27.02.2024 р.*	вик.
3	*Обґрунтування вибору шляхів, технологій (алгоритмів) і засобів вирішення поставленого завдання*	*12.03.2024 р.*	вик.
4	*Розробка функціонально-структурної схеми роботи об'єкта проектування та проектування бази даних*	*17.04.2024 р.*	вик.
5	*Практична реалізація об'єкта проектування та розробка бази даних*	*18.05.2024 р.*	вик.
6	*Тестування та налагодження об'єкта проектування*	*01.06.2024 р.*	вик.
7	*Здача чистового варіанту кваліфікаційної роботи бакалавра на кафедру*	*05.06.2024 р.*	вик.

Здобувач вищої освіти _____ (підпис) *Лис Я. П.* (прізвище, ініціали)

Керівник кваліфікаційної роботи _____ (підпис) *Ящук А. А.* (прізвище, ініціали)

Міністерство освіти і науки України
Луцький національний технічний університет

Рецензія
на кваліфікаційну роботу

Здобувач вищої освіти: *Лис Ярослав Петрович*

Тема: *«Створення програмного застосунку для реалізації процесів електронної комерції на базі Node.js».*

Коротка характеристика кваліфікаційної роботи та прийнятих рішень: *Кваліфікаційна робота бакалавра складається з трьох розділів, в яких проаналізовані проблематика та поставлені завдання за темою роботи, здійснено теоретичне дослідження та практичну реалізацію для реалізації процесів електронної комерції, здійснено тестування розробленої системи*

Самостійні розробки і пропозицій автора: *Автор самостійно розробив програмний застосунок для реалізації процесів електронної комерції на базі Node.js.*

Практичне значення роботи *полягає в тому, що розробка приносить суттєві вигоди для бізнесів, споживачів та економіки в цілому, а саме автоматизація процесів, таких як управління замовленнями, обробка платежів, управління запасами та логістикою, дозволяє скоротити час та ресурси, необхідні для виконання рутинних завдань.*

Недоліки: *Істотних недоліків в роботі не виявлено.*

Загальний висновок: *У кваліфікаційній роботі бакалавра здійснено аналіз предметної області, вибір методів і засобів для розробки, розроблено програмний продукт, здійснено його тестування. Усі поставлені завдання виконано в повному обсязі.*

Кваліфікаційна робота здобувача освіти Лиса Ярослава Петровича рекомендується до захисту екзаменаційній комісії та заслуговує позитивної оцінки.

Рецензент кваліфікаційної роботи: Саварин Павло Вікторович, к.п.н., доцент кафедри ЦОТ

(підпис)

«07» червня 2024 р.

Міністерство освіти і науки України
Луцький національний технічний університет
Факультет комп'ютерних та інформаційних технологій
Кафедра інженерії програмного забезпечення

ВІДГУК
КЕРІВНИКА НА КВАЛІФІКАЦІЙНУ РОБОТУ

Здобувач вищої освіти *Лис Ярослав Петрович*
(прізвище, ім'я, по батькові)
група *ІПЗс-21*

Тема кваліфікаційної роботи:
Створення програмного застосунку для реалізації процессів електронної комерції на базі Node.js
Керівник *Ящук Андрій Анатолійович*

Актуальність теми
Електронна комерція продовжує демонструвати стійке зростання у всьому світі. Збільшення кількості інтернет-користувачів та зростання популярності онлайн-шопінгу створюють значний попит на ефективні та зручні програмні рішення. Програмні застосунки, що надають користувачам можливість легко здійснювати покупки, отримувати рекомендації та користуватися іншими функціями, стають необхідними для задоволення цих вимог.
Об'єкт дослідження
Розробка програмного застосунку, призначеного для автоматизації та оптимізації процесів електронної комерції
Характеристика теоретичного рівня, наявності самостійних розробок і практичної значимості роботи:
Автор аналізує різні методи та засоби для розробки функціонального Web-додатку. Визначає найкращі з них і обґрунтовує своє рішення. Зміст роботи відповідає обраній темі.

Зауваження та недоліки: *Суттєвих недоліків в роботі не виявлено.*

Загальний висновок:
Кваліфікаційна робота бакалавра виконана на високому рівні. Робота виконана відповідно до вимог, логічно й послідовно описує хід виконання поставленого завдання. Пояснювальна записка відповідає стандартам до її оформлення. Робота може бути оцінена на високу оцінку

Керівник _____ **(*Ящук А.А.*)**
(підпис) (прізвище, ім'я, по батькові)

«07» 06 2024 р.

АНОТАЦІЯ

Лис Я. П. Програмний застосунок для реалізації процесів електронної комерції на базі Node.js. Рукопис.

Кваліфікаційна робота бакалавра ОП «Інженерія програмного забезпечення». Луцький національний технічний університет. Луцьк, 2024.

Кваліфікаційна робота бакалавра складається зі вступу, трьох розділів, висновків, списку використаних джерел та додатків. У першому розділі роботи аналізується предметна область. У другому розділі кваліфікаційної роботи представлена специфікація вимог до розроблюваної системи. У третьому розділі кваліфікаційної роботи описано процес створення програмного застосунку. У висновках узагальнено інформацію, відображену у попередніх частинах. Результати розробок можуть бути застосовані в практичній діяльності.

Ключові слова: серверна частина, база даних, інструментарій для інтеграції з різноманітними обліковими системами.

ABSTRACT

Lys Y. P. A Software Application for the Implementation of E-commerce Processes based on Node.js. Manuscript.

Bachelor's qualifying thesis of the OP "Software Engineering". Lutsk National Technical University. Lutsk, 2024.

The first section of the work analyzes the subject area. In another section of the qualification work, the specification of the requirements for the developed system is presented. The third section of the qualification work describes the process of creating software. The conclusions summarize the information presented in the previous parts. Development results can be used in practical activities.

Keywords: server part, database, tools for integration with various accounting systems.

ЗМІСТ

ВСТУП

Актуальність теми. Електронна комерція продовжує демонструвати стійке зростання у всьому світі. Збільшення кількості інтернет-користувачів та зростання популярності онлайн-шопінгу створюють значний попит на ефективні та зручні програмні рішення. Сучасні споживачі віддають перевагу зручним, швидким та персоналізованим покупкам. Програмні застосунки, що надають користувачам можливість легко здійснювати покупки, отримувати рекомендації та користуватися іншими функціями, стають необхідними для задоволення цих вимог. Бізнеси, які використовують передові технології та пропонують своїм клієнтам зручні та функціональні застосунки, мають конкурентну перевагу. Розробка інноваційних рішень для електронної комерції дозволяє компаніям виділитися на фоні конкурентів і залучати більше клієнтів. Швидкий розвиток технологій, таких як штучний інтелект, машинне навчання, великі дані та інтернет речей (IoT), відкриває нові можливості для вдосконалення процесів електронної комерції. Дослідження та впровадження цих технологій у програмні застосунки дозволяє покращувати користувацький досвід і підвищувати ефективність бізнесу. Можливість продавати товари та послуги на міжнародному рівні є однією з ключових переваг електронної комерції. Програмні застосунки, що підтримують багатомовність, мультивалютність та інтеграцію з різними платіжними системами, є необхідними для успішного виходу на глобальний ринок.

Дослідження програмних застосунків для реалізації процесів електронної комерції є актуальним та важливим з точки зору розвитку бізнесу, задоволення споживчих потреб і використання нових технологій. Це дозволяє створювати інноваційні рішення, які сприяють зростанню та ефективності компаній у сучасному динамічному ринковому середовищі.

Мета роботи – обробка та впровадження інноваційного програмного застосунку, який забезпечує ефективне управління процесами електронної комерції, підвищує зручність для користувачів та сприяє зростанню бізнесу.

Завдання кваліфікаційної роботи:

– проаналізувати основні бізнес-процеси електронної комерції, такі як управління замовленнями, обробка платежів, управління складом та логістикою, щоб знизити витрати та підвищити ефективність операцій;

– розробити інтуїтивний та зручний інтерфейс для користувачів, що забезпечить простоту навігації, швидкість завантаження та доступ до персоналізованих рекомендацій і пропозицій;

– забезпечити інтеграцію з популярними платіжними системами та методами оплати для спрощення процесу оплати та підвищення безпеки транзакцій;

– оптимізувати програмний застосунок для роботи на мобільних пристроях, забезпечуючи безперебійний доступ та зручність користування незалежно від платформи;

– створити модульну архітектуру застосунку, що дозволить легко додавати нові функції та масштабувати систему відповідно до зростання бізнесу та зміни його потреб.

Об’єктом дослідження є програмний застосунок, призначений для автоматизації та оптимізації процесів електронної комерції, включаючи управління замовленнями, обробку платежів, управління складом, логістику, аналітику та забезпечення безпеки даних.

Предметом дослідження є методи, технології, алгоритми та процеси, які використовуються для розробки, впровадження та оптимізації програмного застосунку, що підтримує всі аспекти електронної комерції.

Розробка програмного застосунку для реалізації процесів електронної комерції має значне практичне значення, оскільки вона приносить суттєві вигоди для бізнесів, споживачів та економіки в цілому, а саме автоматизація процесів, таких як управління замовленнями, обробка платежів, управління запасами та логістикою, дозволяє скоротити час та ресурси, необхідні для виконання рутинних завдань, оптимізація бізнес-процесів та зниження витрат на операційні процеси через автоматизацію і більш ефективне управління ресурсам,

можливість виходу на глобальний ринок завдяки підтримці багатомовності та мультивалютності, що дозволяє залучати клієнтів з різних країн.

РОЗДІЛ 1
АНАЛІЗ ТА ПОСТАНОВКА ЗАВДАНЬ ТЕОРЕТИЧНИХ ОСНОВ ПРОГРАМНОГО ДОДАТКУ

1.1 Аналіз сучасного стану процесів електронної комерції

Аналіз змісту процесів електронної комерції може охоплювати різні аспекти цього промислу, включаючи економічні, технологічні, соціальні та правові аспекти. Ось декілька ключових пунктів, які можуть бути включені в такий аналіз:

– огляд основних платформ електронної комерції, таких як Amazon, eBay, Alibaba, Shopify тощо. Аналіз їх функцій, можливостей та технологічних інновацій, таких як машинне навчання для персоналізованої рекомендації товарів, розширені реальності для онлайн-покупок тощо;

– вивчення поведінки споживачів в онлайн-середовищі, включаючи їх вподобання, звички покупок, а також вплив соціальних мереж на процеси прийняття рішень щодо покупки;

– аналіз процесів доставки та управління запасами, включаючи проблеми останньої милі, оптимізацію маршрутів та використання технологій Інтернету речей (IoT) для відстеження вантажів;

– розгляд заходів забезпечення безпеки, включаючи захист від шахрайства, кібератак та витоків даних. Аналіз впровадження методів аутентифікації та шифрування для забезпечення конфіденційності та цілісності даних покупців;

– оцінка впливу електронної комерції на традиційну роздрібну торгівлю, аналіз моделей бізнесу електронної комерції та їх прибутковості, а також розгляд питань оподаткування та митних тарифів;

– вивчення правових аспектів електронної комерції, таких як захист споживачів, регулювання онлайн-торгівлі, а також проблеми з інтелектуальною власністю та авторськими правами;

– огляд новітніх технологій та інновацій у сфері електронної комерції, таких як використання штучного інтелекту для автоматизації процесів, розробка блокчейн-технологій для підвищення безпеки тощо.

Цей аналіз допоможе зрозуміти динаміку та ключові фактори успіху в електронній комерції і визначити можливості для розвитку бізнесу в цій галузі. Аналіз змісту процесів електронної комерції полягає в розгляді всіх етапів, які включаються в цей вид бізнесу, від початку до кінця. Ось деякі ключові аспекти, які можна включити в такий аналіз:

– створення веб-сайту або платформи. Це включає в себе розробку веб-сайту або мобільної програми для представлення товарів чи послуг для потенційних покупців;

– управління контентом. Забезпечення актуального та привабливого контенту для відвідувачів, такого як фотографії продуктів, описи, ціни та інша інформація;

– організація каталогу продуктів. Створення структурованого каталогу, який дозволяє покупцям швидко знаходити потрібні товари;

– обробка замовлень. Забезпечення ефективного процесу приймання, обробки та виконання замовлень;

– управління складом. Контроль за наявністю товарів на складі, їх відстеження та перезамовлення, щоб уникнути нестачі або надмірного запасу;

– обробка платежів. Забезпечення безпеки та зручності платіжних операцій для покупців, включаючи обробку кредитних карт, електронних грошей тощо;

– доставка та логістика. Організація ефективного процесу доставки товарів від продавця до покупця, включаючи вибір оптимальних способів доставки та відстеження посилок;

– обслуговування клієнтів. Надання підтримки клієнтам через різні канали зв’язку, такі як телефон, електронна пошта, чат тощо, і вирішення їх запитів та проблем;

– маркетинг та реклама. Розробка та реалізація стратегій маркетингу та реклами для залучення нових клієнтів та збільшення продажів;

– аналітика та звітність. Збір, аналіз та використання даних про продажі та поведінку покупців для покращення бізнесу та прийняття стратегічних рішень. Аналіз цих процесів допомагає бізнесам ідентифікувати сильні та слабкі сторони їхньої електронної комерції та здійснювати вдосконалення для підвищення ефективності та конкурентоспроможності.

Існує безліч рішень і платформ для реалізації процесів електронної комерції, і вони можуть різнитися від маленьких до великих, відкритих джерел до пропрієтарних. Ось кілька відомих рішень.

Shopify – це одна з найпопулярніших платформ для створення та управління онлайн-магазинами. Вона пропонує широкий набір інструментів для створення магазинів, обробки замовлень, управління інвентарем, а також аналітики та звітності [1].

Magento є однією з найбільш потужних відкритих платформ електронної комерції. Вона надає розширені можливості налаштування та розширення, що робить її відмінним вибором для великих та середніх підприємств. Це відкрита платформа, яка надає розширені можливості для розробки та налаштування магазинів та дозволяє створювати різноманітні інтернет-магазини, від невеликих до корпоративних рішень [2].

WooCommerce – це безкоштовний плагін для WordPress, який дозволяє перетворити ваш веб-сайт на повноцінний онлайн-магазин. Він має велику спільноту користувачів і широкий вибір розширень [3].

BigCommerce – це платформа електронної комерції, яка пропонує широкий спектр функцій, включаючи інструменти маркетингу, аналітику, управління складом та інші [4].

Wix Stores – це інструмент для створення – онлайн-магазинів, який інтегрований з платформою створення веб-сайтів Wix. Він пропонує простий у використанні інтерфейс та широкий вибір дизайнерских рішень [5].

Squarespace – це інша платформа для створення веб-сайтів, яка також має можливості електронної комерції. Вона відома своїми красивими дизайнами та легкістю використання [6].

Amazon Webstore – це платформа для створення та управління онлайн-магазинами. Вона дозволяє підприємствам створювати магазини на базі наявної інфраструктури та використовувати всі переваги екосистеми [7].

Кожна з цих платформ має свої переваги та недоліки, і вибір конкретного рішення залежить від потреб вашого бізнесу, рівня технічної експертизи та бюджету (рис. 1.1).

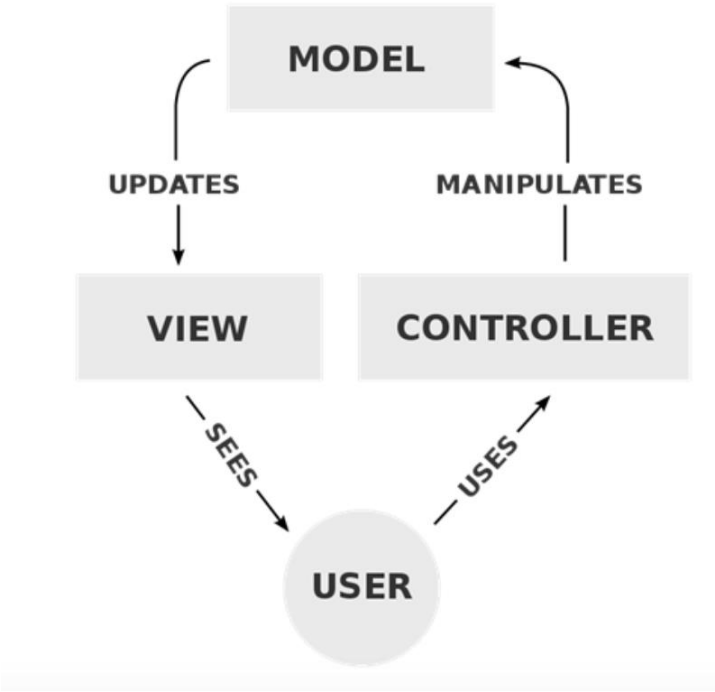

Рисунок 1.1 – Структура інтернет-магазину

Вибір платформи повинен залежати від конкретних потреб вашого бізнесу, бюджету, ресурсів та інших факторів. Статистика розповсюдженості приведена на рисунку 1.2.

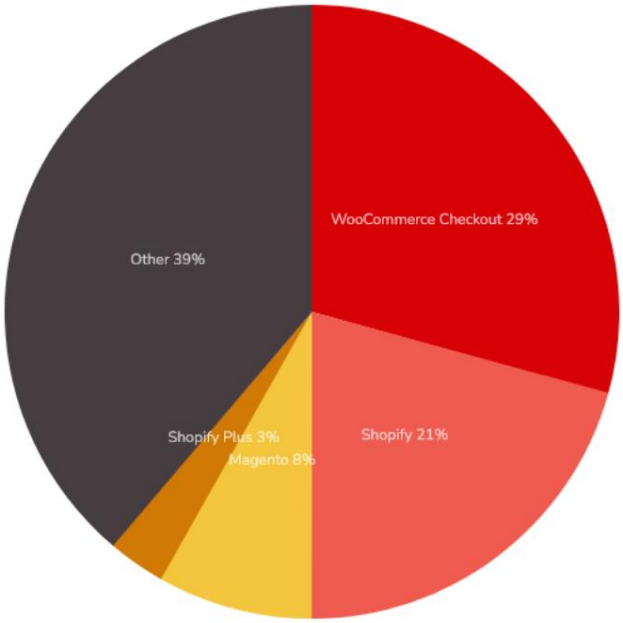

Рисунок 1.2 – Аналіз застосунків [8]

Ці рішення мають велику популярність та широкий функціонал, який дозволяє бізнесам створювати й управляти успішними інтернет-магазинами. Однак вибір конкретного рішення може залежати від потреб вашого бізнесу, бюджету та технічних можливостей (рис. 1.3).

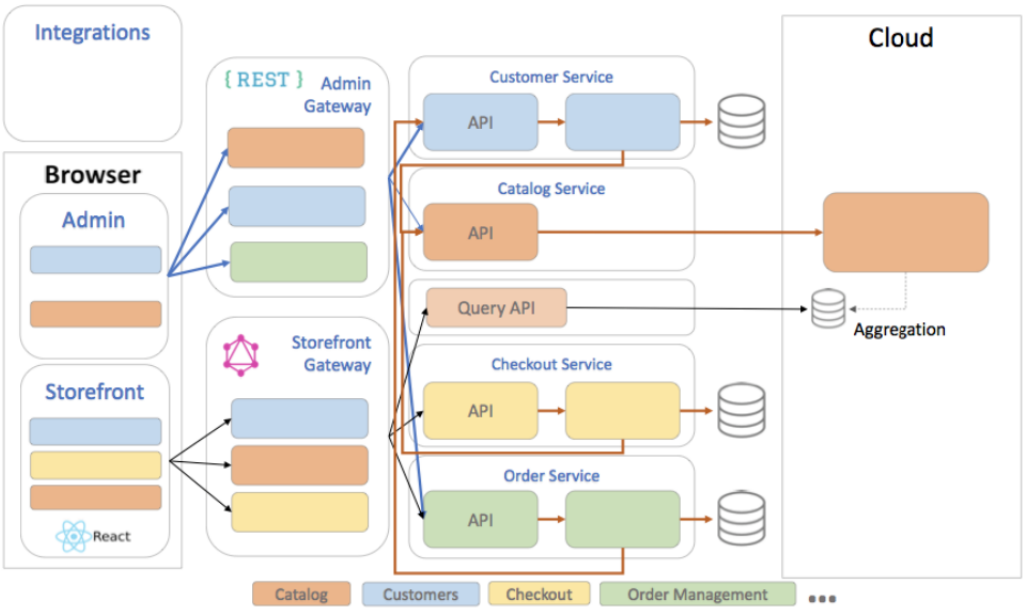

Рисунок 1.3 – Схема технічних можливостей

1.2 Постановка завдання на кваліфікаційну роботу

Завдання на кваліфікаційну роботу можна розділити на наступні пункти:

– проаналізувати основні бізнес-процеси електронної комерції, такі як управління замовленнями, обробка платежів, управління складом та логістикою, щоб знизити витрати та підвищити ефективність операцій;

– розробити інтуїтивний та зручний інтерфейс для користувачів, що забезпечить простоту навігації, швидкість завантаження та доступ до персоналізованих рекомендацій і пропозицій;

– забезпечити інтеграцію з популярними платіжними системами та методами оплати для спрощення процесу оплати та підвищення безпеки транзакцій;

– оптимізувати програмний застосунок для роботи на мобільних пристроях, забезпечуючи безперебійний доступ та зручність користування незалежно від платформи;

– створити модульну архітектуру застосунку, що дозволить легко додавати нові функції та масштабувати систему відповідно до зростання бізнесу та зміни його потреб.

Це лише загальна постановка задачі та часткових завдань, які можна розширити та уточнити в залежності від конкретної області дослідження та методів, що використовуються.

За результатами аналізу поточного стану процесів та відомих рішень для реалізації проекту було вирішено використовувати CMS при розробці. Одною з найкращих технологій для досягнення мети є фреймворк KeystoneJS на основі технології NodeJS.

KeystoneJS – це фреймворк з відкритим кодом, який орієнтований на малий бізнес та стартапи. Варто зазначити, що він є надзвичайно простим в процесі розвитку, а також є можливість інтеграції з іншими ІС за допомогою GraphQl API. Основне завдання розділили на часткові завдання для подальшого рішення [9].

РОЗДІЛ 2
СПЕЦИФІКАЦІЯ ВИМОГ ДО РОЗРОБКИ ПРОГРАМНОГО ДОДАТКУ

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

Розробка архітектури програмного застосунку для електронної комерції включає в себе кілька етапів, одним із яких є інженерія вимог. Тут описано загальний процес інженерії вимог для розробки такого програмного застосунку. Спілкування з клієнтом або стейкхолдерами для збору вимог щодо функціональності, властивостей та обмежень програмного застосунку. Це може включати вимоги до сторінок, функцій пошуку, оформлення замовлення, системи оплати, управління запасами ,підтвердження та уточнення зібраних вимог, ідентифікація протиріччя і невиправленість.

Створення документу, який містить усі вимоги до програмного застосунку може бути у вигляді текстового опису, діаграм, прототипів, визначення можливих ризиків, що можуть вплинути на виконання вимог, і розробка плану їх управління. Організація процесу підтвердження вимог з клієнтом, щоб переконатися, що вони правильно розуміють і погоджуються з усіма вимогами. Встановлення процесу управління змінами, який дозволить ефективно вносити зміни в вимоги протягом життєвого циклу проекту. Підтвердження, що розроблений програмний застосунок відповідає усім визначеним вимогам. Збереження усіх документів, пов’язаних з інженерією вимог, для подальшого використання, аналізу та звітування. Цей процес інженерії вимог допомагає забезпечити якість та повноту вимог до програмного застосунку, а також уникнути непорозумінь між розробниками та клієнтами.

Отже, завдання кваліфікаційної роботи полягають в тому, щоб:

– автоматизувати основні бізнес-процеси електронної комерції, такі як управління замовленнями, обробка платежів, управління складом та логістикою, щоб знизити витрати та підвищити ефективність операцій;

– розробити інтуїтивний та зручний інтерфейс для користувачів, що забезпечить простоту навігації, швидкість завантаження та доступ до персоналізованих рекомендацій і пропозицій;

– забезпечити високий рівень безпеки для захисту персональних даних користувачів та запобігання шахрайству, відповідно до сучасних стандартів та регуляцій;

– оптимізувати програмний застосунок для роботи на мобільних пристроях, забезпечуючи безперебійний доступ та зручність користування незалежно від платформи;

– створити модульну архітектуру застосунку, що дозволить легко добавляти нові функції та розширяти систему відповідно до зростання бізнесу та зміни його потреб;

– провести тестування застосунку, задля уникнення можливих проблем в експлуатації.

2.2 Методи та алгоритми вирішення поставленого завдання

Архітектурне проектування програмного застосунку – це процес розробки високорівневої структури програмного застосунку, яка визначає його компоненти, їх взаємодію та організацію. Основні кроки у процесі архітектурного проектування включають визначення архітектурних, функціональних та нефункціональних вимог до програмного застосунку і визначення вимог до архітектури, таких як масштабованість, надійність, продуктивність тощо. Ось декілька прикладів основних кроків:

– вибір базового архітектурного стилю, такого як клієнт-сервер, мікросервіси, багатошарова архітектура тощо, який найкраще відповідає потребам програмного застосунку;

– визначення основних компонентів програмного застосунку та їх взаємодії. Це може включати розробку діаграм класів, діаграм компонентів, діаграм пакетів тощо. Розробка моделей даних та стратегій управління даними в

програмному застосунку, таких як вибір бази даних, моделювання схеми бази даних, розробка схеми міграції тощо;

– розробка інтерфейсу користувача, а саме визначення способів взаємодії користувача з програмним застосунком та розробка відповідного інтерфейсу, включаючи діаграми прототипів, діаграми потоку користувача тощо;

– вибір технологій та інструментів, які будуть використовуватися для реалізації програмного застосунку, а також розробка стратегій інтеграції зовнішніх сервісів та компонентів;

– виконання тестів на архітектурному рівні для перевірки правильності та ефективності архітектури.

Архітектурне проектування є ключовим етапом у розробці програмного застосунку, оскільки від нього залежить якість, ефективність та розширюваність програмного продукту (рис. 2.1).

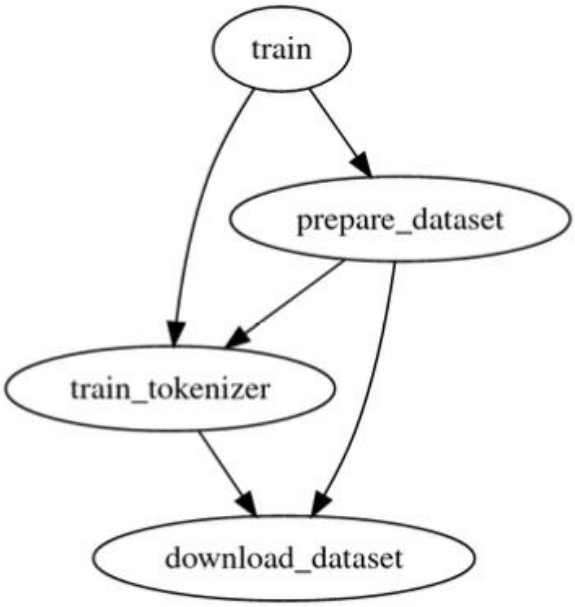

Рисунок 2.1 – Архітектурне проектування

На рисунку 2.2 показано схематичне зображення додатку.

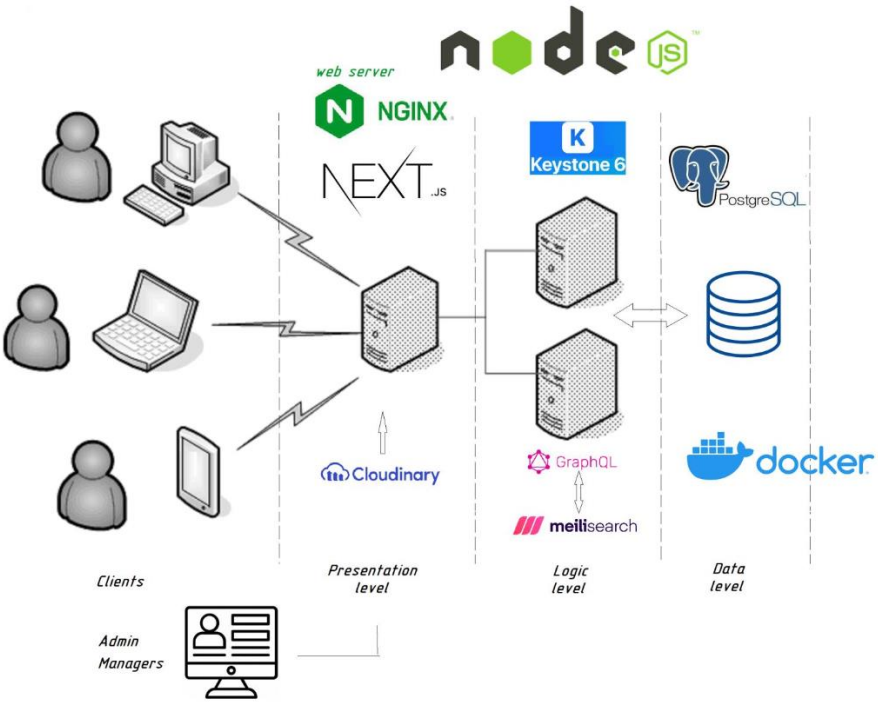

Рисунок 2.2 – Схематичне зображення додатку

Архітектурне проектування програмного застосунку в електронній комерції включає в себе розробку високорівневої структури системи, яка визначає загальну організацію компонентів та їх взаємодію. Ось кроки, які можна виконати під час архітектурного проектування програмного застосунку для електронної комерції:

– встановлення вимог щодо архітектури програмного застосунку, таких як масштабованість, надійність, продуктивність та безпека;

– вибір архітектурних паттернів, які найкраще відповідають потребам системи. Наприклад, MVC (Model-View-Controller), Microservices, архітектура зі шаровими компонентами тощо;

– визначення моделей даних та їх взаємозв’язків для зберігання інформації про товари, замовлення, користувачів тощо;

– визначення компонентів програмного застосунку та їхніх взаємозв’язків. Це може включати серверну та клієнтську частини, а також компоненти для управління користувачами, оплатою, маркетингом тощо;

– розробка API для взаємодії між різними компонентами програмного застосунку, а також вибір протоколів взаємодії, таких як REST, GraphQL тощо;

– розробка стратегій безпеки для захисту від несанкціонованого доступу до даних користувачів, платіжних інформації та інших конфіденційних даних;

– визначення можливих проблем з продуктивністю та шляхи їх вирішення шляхом оптимізації запитів до бази даних, кешування тощо [10].

Ці кроки допоможуть забезпечити ефективну та масштабовану архітектуру програмного застосунку для електронної комерції, що відповідає потребам бізнесу та користувачів рисунок 2.3.

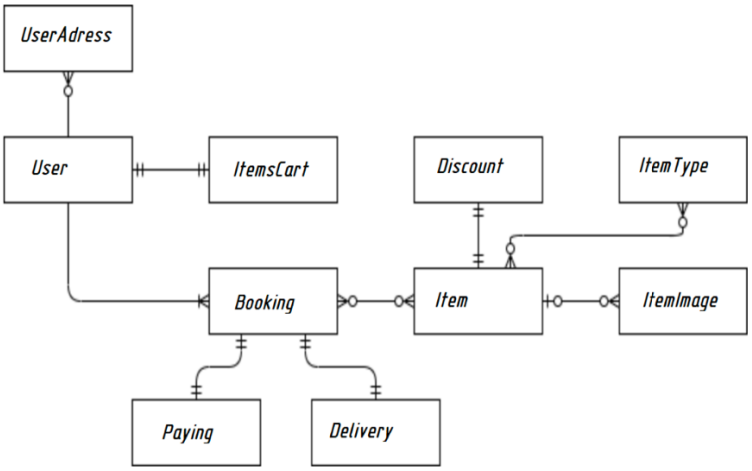

Рисунок 2.3 – Структура бази даних

Встановлення необхідної програмної інфраструктури для розробки та функціонування застосунку електронної комерції – це важливий етап. Node.js та Docker – це два популярних інструменти, які можна використовувати для цього.

Node.js є платформою з відкритим кодом для виконання JavaScript на сервері. Він часто використовується для розробки веб-серверів та веб-застосунків. Для реалізації процесів електронної комерції, ви можете використовувати Node.js для створення серверної частини вашого застосунку, обробки запитів користувачів, взаємодії з базою даних та іншими завданнями [11].

Docker – це платформа для розробки, доставки та запуску програмного забезпечення за допомогою контейнеризації. Використовується для створення контейнерів, які містять ваше програмне середовище, включаючи оперативну

систему, залежності, конфігурацію тощо [12]. Це дозволяє вам створювати однорідне середовище для розгортання вашого застосунку на будь-якому сервері. Таким чином, ви можете почати з встановлення Node.js для розробки серверної частини вашого застосунку електронної комерції, а потім сконфігурувати і використати Docker для створення контейнерів з вашим програмним середовищем, яке буде функціонувати на сервері рисунок 2.4.

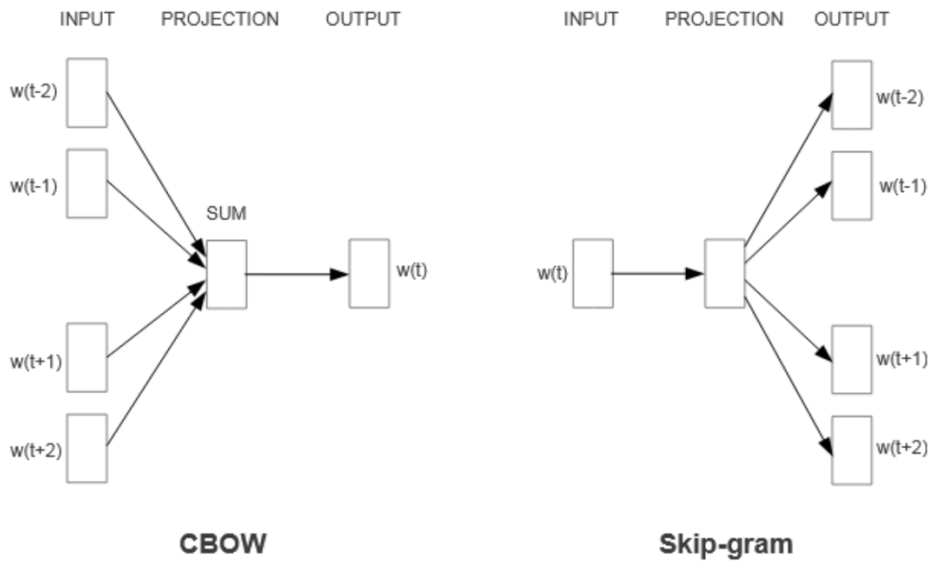

Рисунок 2.4 – Принцип роботи додатку

Для керування залежностями проекту буде використано технологію Yarn. Вона дозволить встановити необхідні пакети та компоненти інфраструктури, і створить файл package.json для опису проекту.

Для зберігання та керування версіями вашого коду буде використано GitHub.

Keystone – це CMS (система управління контентом), яка дозволяє легко створювати та управляти back-end компонентами застосунку електронної комерції, адміністративною панеллю тощо.

Next.js – це інструментарій для створення front-end компонентів застосунку на основі React. Данна технологія використовується для створення

веб-інтерфейсу вашого застосунку та забезпечення користувачам швидкого і зрозумілого графічного інтерфейсу.

PostgreSQL – це СУБД (систему управління базами даних) для зберігання та організації даних вашого проекту [13].

Для зберігання та обробки зображень застосунку буде використано CDN Cloudinary. Вона забезпечить швидку та ефективну доставки зображень користувачам застосунку.

Інтегруючи ці інструменти, буде створено повноцінний застосунок електронної комерції з якісною back-end та front-end частинами, а також забезпечене ефективне управління даними та зображеннями.

На рисунку 2.5 зображено функціональну схему діаграми класів.

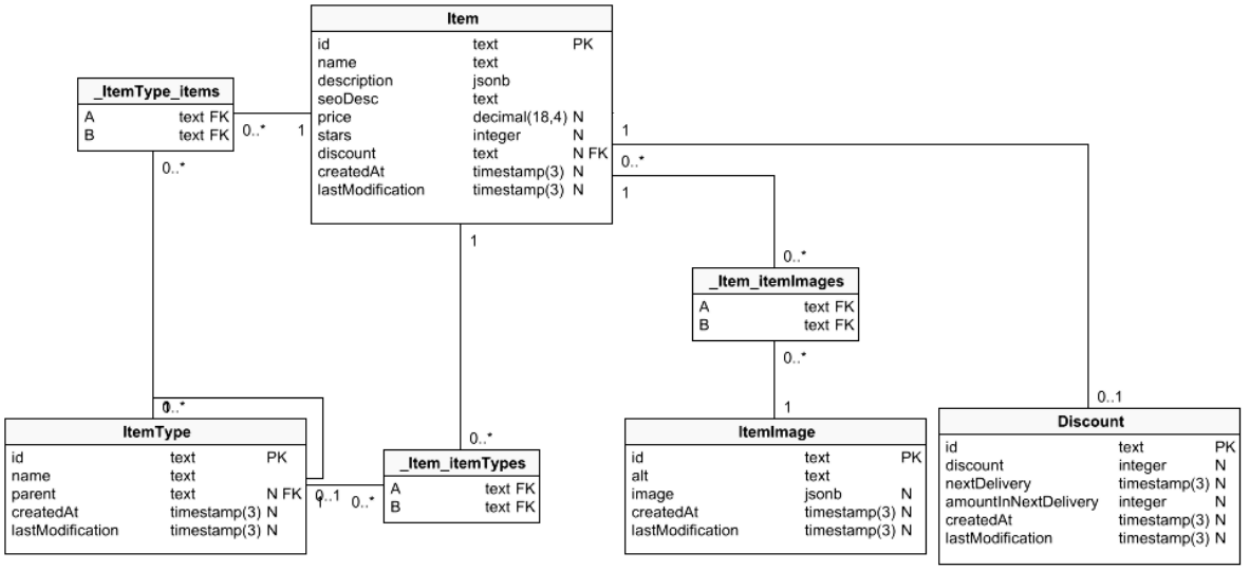

Рисунок 2.5 – Uml-діаграма класів програмного застосунку

РОЗДІЛ 3
РОЗРОБКА ПРОГРАМНОГО ЗАСТОСУНКУ ДЛЯ РЕАЛІЗАЦІЇ ПРОЦЕСІВ ЕЛЕКТРОННОЇ КОМЕРЦІЇ НА БАЗІ NODE.JS

3.1 Практична реалізація об’єкта проектування

Програмна розробка програмного застосунку для реалізації процесів електронної комерції – це комплексний процес, що включає розробку back-end, front-end та адміністративної частини системи. Було ретельно проаналізовано вимоги до застосунку електронної комерції, включаючи функціональність, інтерфейс користувача, вимоги до безпеки та інші. Платформу для back-end розробки було вибрано Keystone.js на базі Node.js. Перед створенням програмного продукту була розроблена серверна логіка для обробки замовлень, керування товарами, обліку користувачів, оплати та інших аспектів електронної комерції і підключено базу даних для зберігання даних про замовлення, товари, користувачів тощо. Після підготовки почалася реалізація моделі заказу. В лістингу 3.1 продемонстровано створення списку «Bookink» та його взаємодія з базою даних.

Лістинг 3.1 – Демонстрація моделі заказу

```
export const Booking = list ({
  fields: {
    myUser: relationship({ ref: 'MainUser'}),
    myItems: json(),
    myPayment : relationship({ ref: 'MyPaying'}),
    myDelivery: relationship({ ref: 'MyDelivery'}),
    myEmployee: relationship({ ref: 'MyUser'}),
    status: select({ type: 'enum', options: BookingStatusOptions}),
    myCreatedAt: timestamp({
      defaultValue: {kind: 'now'},}),
  access: {
    operation: {
      create: ({session }) => !!session,
      update: ({session }) => !!session,
      delete: ({session }) => !!session,}}
```

Кінець лістингу 3.1

Реалізована реєстрація облікового запису користувача для його подальшої взаємодії з додатком. Під час розробки було використано можливість роботи в асинхронному режимі, для економії часу очікування користувача та ресурсів сервера. В лістингу 3.2 зображено реалізацію моделі замовника.

Лістинг 3.2 – Демонстрація моделі замовника

```
//buyer model
export const User = list({
      fields: {
            //data validation
            myName: text({ validation: (isRequired: true } }),
            myEmail: text ({
                  validation: isRequired: true },
                  isIndexed: 'unique',
                  isFilterable: true,
      }),
      //pass validation
      myPassword: password({ validation: isRequired: true } }),
      myRole: select({
            type: 'enum',
            mainoptions: UsersRolesValues,
            defValue: UsersRoles. Customer,
            }),
      //User address validation
      keyUseraddress: relationship({
            ref: 'UserAddress', many: true }),
      access: {
            operation: {
                  item: {
                  update: (session, inputData }) =>
                  filterCustomerAccessCreate (session, inputData),
                  },
      }),
```

Кінець лістингу 3.2

Перед відправленням замовлення, необхідно перевірити вхідні дані та у разі некоректного вводу інформації попередити замовника, валідацію даних зображено в лістингу 3.3.

Лістинг 3.3 – Демонстрація валідації даних

```
//validation
export const UserAddress = list({
      fields: (
            userAddressName: text(),
            name: text({ validation: (isRequired: true } }),
            userStreetAddress: text({ validation: (isRequired: true } }),
            userStreetAddress2: text(),
            userCity: text({ validation: (isRequired: true } }),
            userPostCode: text ({ validation: (isRequired: true } }),
            userCountry: text ({ validation: { isRequired: true } }),
            telNo: text({ validation: isRequired: true } }),

            //user relations
            user: relationship({
                  ref: 'User',
                  ui: {
                        hideCreate: true,
                  },
            }),
            createdAt: timestamp ({
                  defaultValue: { kind: 'now' },
            }),
            lastModification: timestamp ({
                  defaultValue: kind: 'now' },
                  db: {
                        updatedAt: true,
            },}),},
```

Кінець лістингу 3.3

Завдяки функціоналу Keystone.js добавлено основні характеристики товару. Додатково реалізовано можливість добавлення короткого опису, та картинки, завдяки чому у користувача буде більше інструментів для редагування товару на сторінці, а завдяки алгоритмам по оптимізації зображень, час загрузки сайту залишиться майже незмінним. Також у користувача буде можливість редагувати вхідні дані в майбутньому через панель адміністратора з графічним інтерфейсом, реалізацію моделі товару зображено в лістингу 3.4.

Лістинг 3.4 – Реалізація моделі товару

```
export const Item = list({
      fields: {
            pdcName: text({ validation: { isRequired: true } }),
            pdcDescription: document (),
            seoDesc: text ({ ui: { displayMode: 'textarea' } }),
            pdcItemTypes: relationship({ ref: 'ItemType', many: true }),
            pdcItemImages: relationship({ ref: 'ItemImage', many: true }),
            pdcPrice: decimal(),
            pdcStars: integer (),
            pdcDiscount: relationship({ ref: 'Discount' }),
            createdAt: timestamp ({
                  defaultValue: { kind: 'now' },
            }),
            lastModification: timestamp ({ defaultValue: ( kind: 'now' },
            db: {
                  updatedAt: true,
            },}),
                  },}),
```

Кінець лістингу 3.4

Не менш важливою функцією є реалізація корзини замовлень. Була реалізована можливість добавлення, видалення та зміна кількості товару в кошику та перевірка на наявність в базі данних (ліст. 3.5).

Лістинг 3.5 – Реалізація корзини замовлень

```
export const ItemsCart = list({
      fields: {
            user: relationship({
                  ref: 'MainUser',
                  ui: {
                        hideCreate: true,
                  },
            }),
            items: relationship({
                  ref: 'CartItem',
                  many: true,
                  db: {
                        foreignKey: true,
                  },
```

```
            }),
            lastModified: timestamp ({
                  defaultValue: { kind: 'now' },
                  db: {
                        updatedAt: true,
                  },}),
            sum: decimal({ validation: { isRequired: true },}),},
```

Кінець лістингу 3.5

Важливо грамотно організувати роботу корзини з базою даних, та можливість роботи в асинхроному режимі (ліст. 3.6).

Лістинг 3.6 – Взаємодія корзини з базою даних

```
hooks: {
      resolveInput: async ( resolvedData, context }) => {
            if (resolvedData?.items.connect.length) {
                  const mainItems = await
context.query.ItemsCartItem.findMany ({
                        where: {
                        id: {
                              in: resolvedData?.items.connect.map(
                                    (el: id: string }) => el.id
                  ),},},
                  query: amount item price }',
                  });
                  const sum items.reduce((prev: number, current) => {
                        const currentSum= current.item.price
                  return prev + currentSum;
            }, 0);
                  resolvedData.sum = sum;
            return resolvedData;},
                  },}),
```

Кінець лістингу 3.6

У випадку, коли користувач готовий зробити замовлення, необхідно реалізувати можливість доставки товару. Реалізація даної моделі зображена в лістингу 3.7.

Лістинг 3.7 – Модель доставки товару

```
export const Delivery = list({
      fields: {
            status: select({ type: 'enum', options: DeliveryStatusOptions }),
externalId: text(),
            booking: relationship({ ref: 'Booking' }),
            worker: relationship({ ref: 'User' }),
            createdAt: timestamp ({
                  defaultValue: kind: 'now' },
            }),
            lastModification: timestamp ({
                  defaultValue: kind: 'now' },
                  db: {
                        updatedAt: true,
            },

            }),

},
```

Кінець лістингу 3.7

Після підтвердження замовлення, користувач зможе оплатити його через інтегровані платіжні системи (ліст. 3.8).

Лістинг 3.8 – Інтеграція платіжних систем

```
import {PayingStatusOptions } from '../consts/paying-status-options.const';
import {filterCustomerAccess, filterCustomerAccessCreate } from '../shared';
export const Paying = list({
      fields: {
            booking: relationship({ ref: 'Booking' }),
            sumAll: decimal(),
            currency: text(),
            externalId: text(),
            status: select({ type: 'enum', options: PayingStatusOptions }),
            createdAt: timestamp ({
                  defaultValue: }),
```

Кінець лістингу 3.8

Для постійних клієнтів було реалізовано систему знижок, лістинг 3.9.

Лістинг 3.9 – Модель книги знижок

```
export const Discount = list({
      fields: {
            discount: integer (),
            nextDelivery: timestamp (),
            amount InNext Delivery: integer(),
            createdAt: timestamp ({
                  defaultValue: { kind: 'now' },
            }),
      lastModification: timestamp ({
            defaultValue: { kind: 'now' },
            db: {
                  updatedAt: true,
      },}),},
```

Кінець лістингу 3.9

Після створення основного функціоналу можна виконати запуск та перевірку правильного налаштування бази даних, рисунок 3.1.

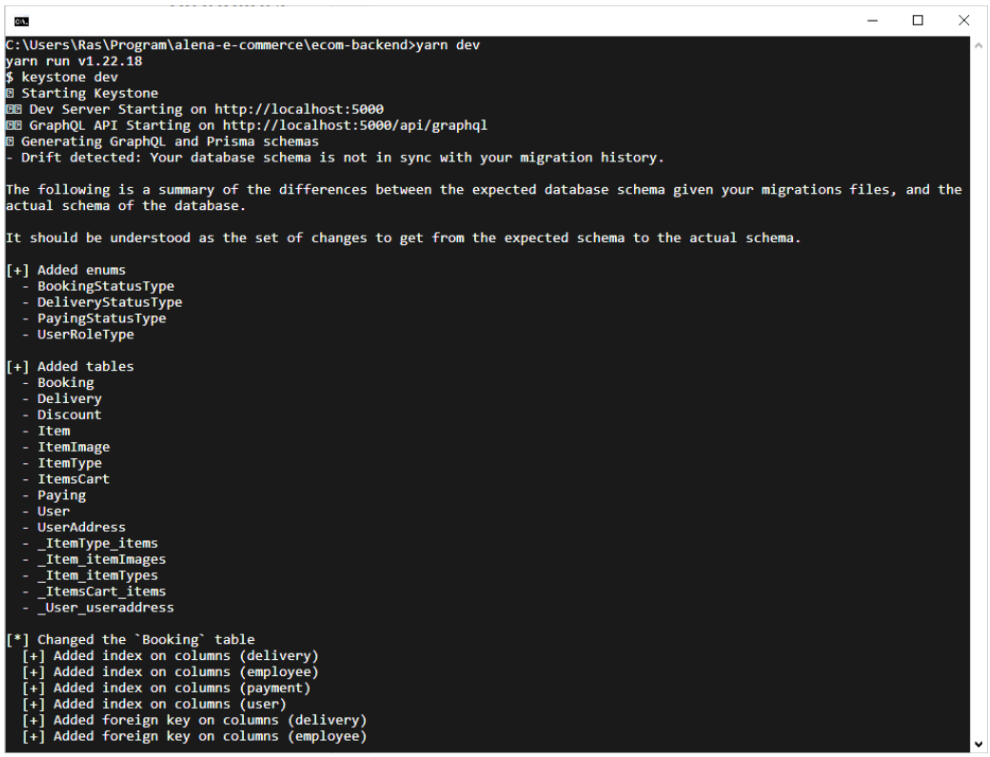

Рисунок 3.1 – Структура бази даних

В результаті було розроблено інтуїтивний та зручний інтерфейс для користувачів, що забезпечить простоту навігації, швидкість завантаження та доступ до персоналізованих рекомендацій і пропозицій.

Реалізовано основні функції продукту. Зручний інтерфейс та функціональні можливості застосунку підвищують задоволеність користувачів, забезпечуючи легкість навігації, персоналізовані рекомендації та швидкий доступ до потрібних товарів і послуг. Це сприяє підвищенню лояльності клієнтів та збільшенню обсягу продажів.

3.2 Тестування та налагодження продукту

Є кілька причин і необхідних аспектів тестування програмного забезпечення. Помилки програмного забезпечення можуть призвести до некоректної роботи програми, збоїв, втрати даних і навіть порушення безпеки. Ще до випуску програми воно може допомогти виявити помилки та переконатися, що програмне забезпечення відповідає вимогам. Користувачі хочуть бути впевненими, що програмне забезпечення, яке вони використовують, працює.

Тестування допомагає переконатися, що програмне забезпечення актуальне та переконатися в якості, надійності та відповідності програмного продукту стандартам і вимогам, підтвердити надійність і забезпечити позитивний досвід користувача адже помилки на початку розробки можна виправити на ранній стадії, уникаючи дорогих виправлень і змін пізніше.

Враховуючи масштаб проекту, було прийняте рішення про ручне тестування додатку, адже в першу чергу необхідно перевірити основну функціональність та переконатись, що графічний інтерфейс користувача не містить помилок, а також це дозволить ефективніше дослідити застосунок від лиця клієнтів.

Також перед початком тестування необхідно налаштувати програмне середовище, та встановити необхідні бібліотеки для коректної роботи програмного забезпечення.

Для початку роботи з додатком, нам необхідно запустити локальний сервер за допомогою функціоналу Keystone, після чого в нас автоматично загрузить API та базу даних. Приклад запуску локального сервера зображений на рисунку 3.2.

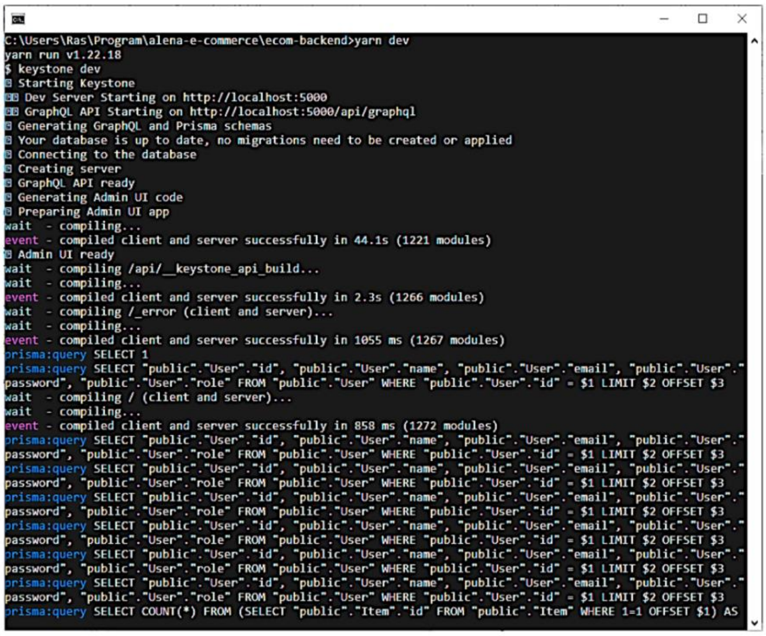

Рисунок 3.2 – Запуск локального сервера

Для початку протестуємо основні функції нашого додатку. Перед використанням продукту необхідно пройти реєстрацію, графічний інтерфейс користувача створено в сучасному стилі, інтуїтивно зрозумілий та адаптований під всі типи пристроїв, що дозволить користувачам всіх пристроїв користуватися нашим додатком.

Реалізовано основні принципи валідації вхідних даних від користувача, що значно зменшить кількість помилок в роботі продукту та збільшить рівень захисту користувачів. Вікно реєстрації зображено на рисунку 3.3.

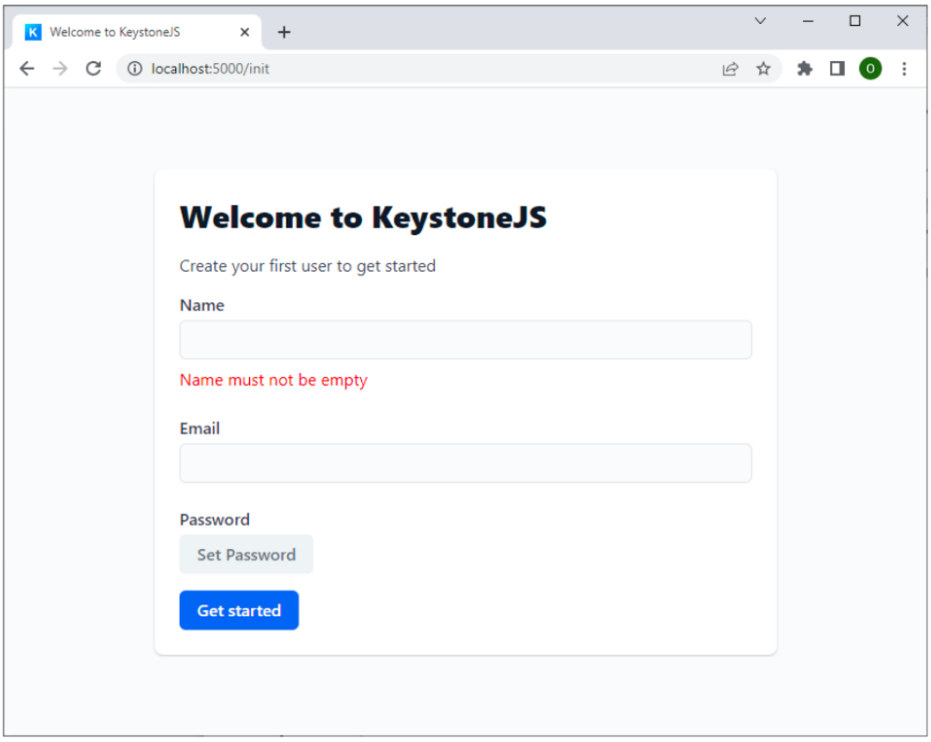

Рисунок 3.3 – Вікно реєстрації користувача

Завдяки функціоналу Keystone користувач може взаємодіяти з базою даних через зручний графічний інтерфейс, що робить використання додатку легким, зручним та швидким (рис. 3.4).

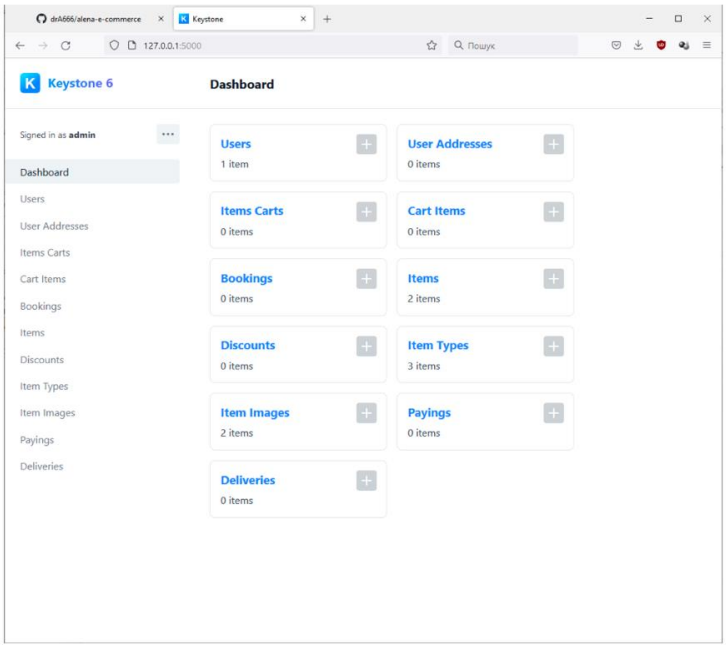

Рисунок 3.4 – База даних

Приклад створення підкатегорій для наших майбутніх товарів зображено на рисунку 3.5.

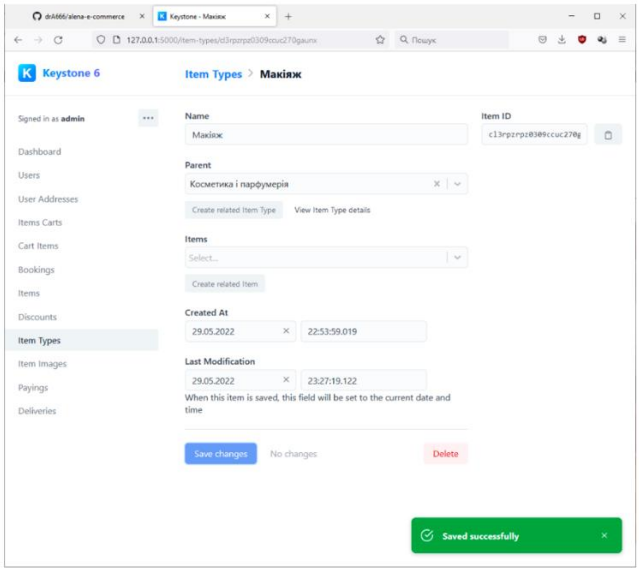

Рисунок 3.5 – Схема зображення продукції

Спробуємо створити обліковий запис для гостя (рис. 3.6).

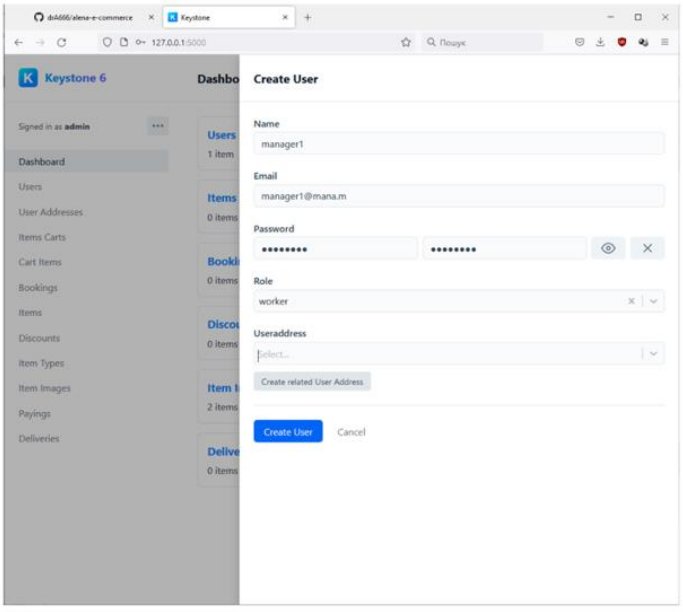

Рисунок 3.6 – Створення облікового запису гостя

При створені облікового запису, попробуємо ввести некоректний пароль, довжиною до 8 символів. Так як ми передбачили це виключення, в інтерфейсі користувача появилось попередження про помилку (рис. 3.7).

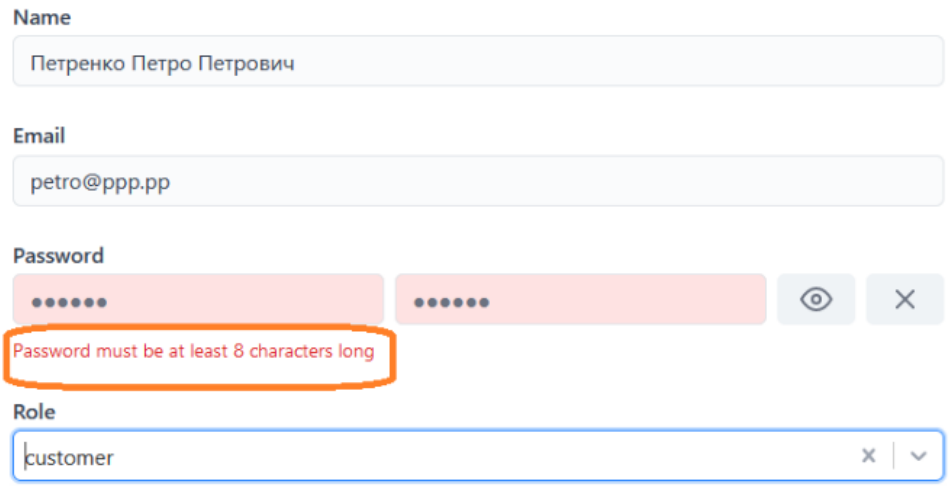

Рисунок 3.7 – Некоректний пароль

У випадку, якщо одне з обов'язкових полів буде незаповнене, на графічному інтерфейсі з'явиться відповідне повідомлення (рис. 3.8).

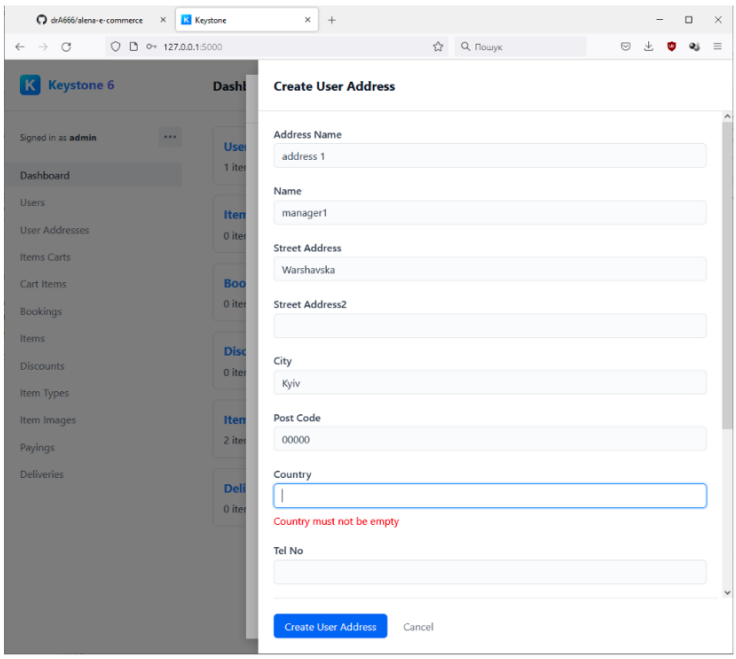

Рисунок 3.8 – Помилка пустого поля

Попробуємо добавити товар в наш каталог, для цього заповнимо необхідні поля та за допомогою кнопки «Create item» підтвердимо дію. В кутку екрана можемо побачити повідомлення про успішне створення товару (рис. 3.9).

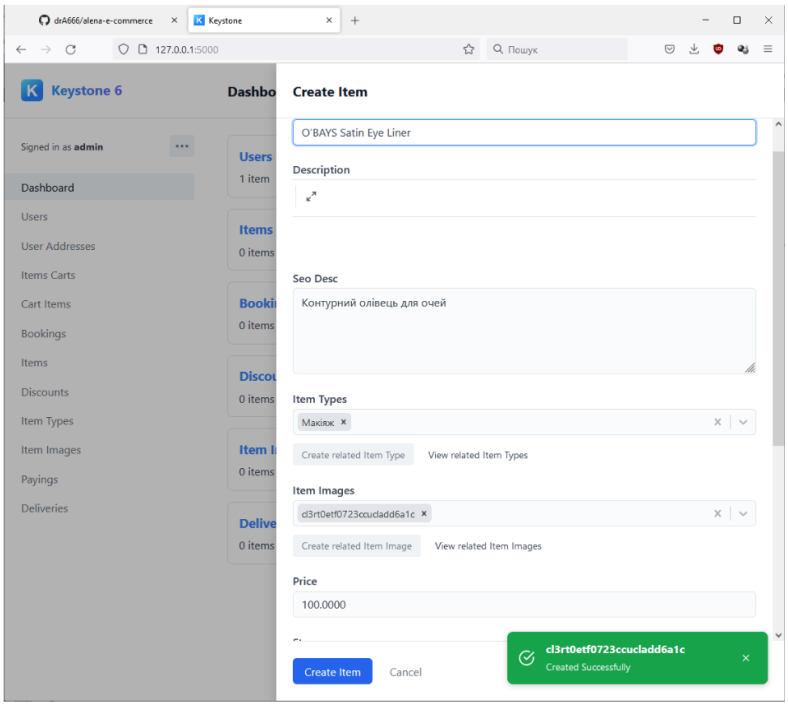

Рисунок 3.9 – Створення товару

Під час тестування додатку було перевірено основний функціонал. Всі функції працюють відповідно до поставленого завдання. Основні недоліки які були виявлено – усунуті, тому можна зробити висновок, що програмний продукт готовий до використання.

3.3 Розробка документації для програмного продукту

Програмна документація на програмний застосунок є важливим інструментом для розробників, тестувальників і користувачів, оскільки вона надає докладну інформацію про функціональність, архітектуру та інші аспекти програмного застосунку. На рисунку 3.10 зображено API до нашого продукту.

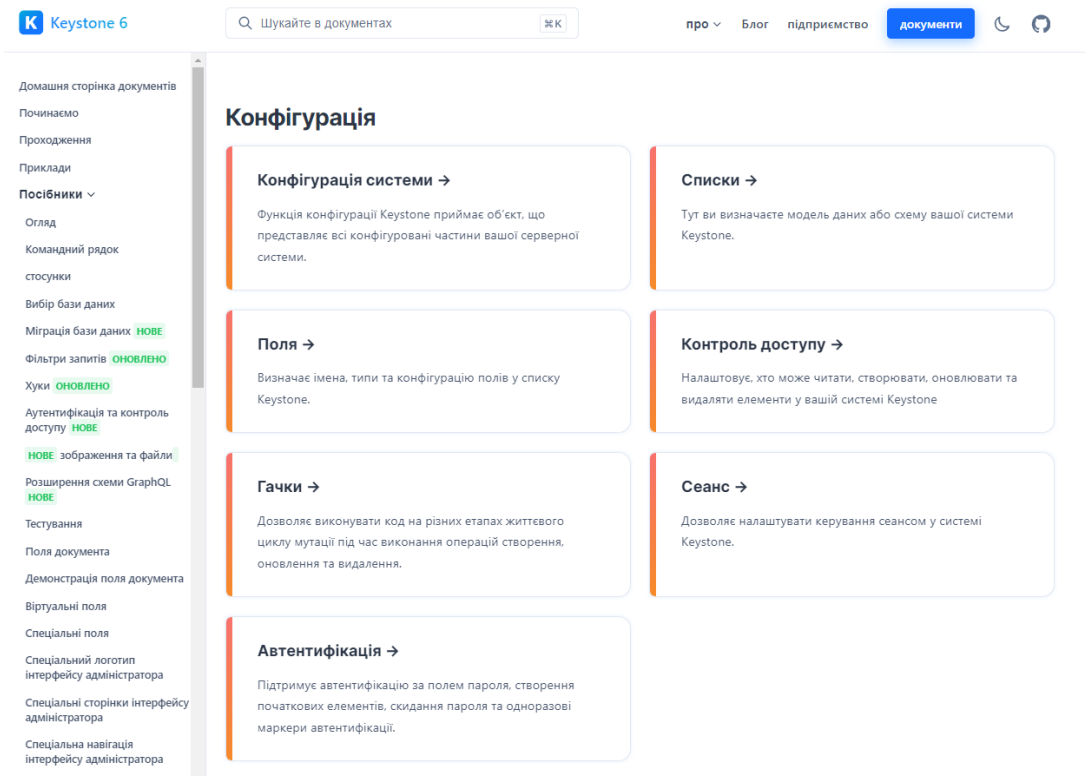

Рисунок 3.10 – API документація Keystone [15]

Завдяки широкому функціоналу Keystone ми маємо змогу використовувати API з офіційною документацією, що робить використання додатку зручним та легким. Завдяки графічному інтерфейсу користувача, всі операції є інтуїтивно зрозумілими.

Для створення проекту необхідно створити та ініціалізувати нові директорії. Для цього виконаємо декілька команд (ліст. 3.10).

Лістинг 3.9 – Команди для старту Keystone

```
mkdir keystone-learning
cd keystone-learning
yarn init
yarn add @keystone-6/core
export default {};
yarn add typescript
```

Кінець лістингу 3.10

Для того, щоб добавити базу даних, та добавити першого користувача необхідно зробити імпорт з модуля, та створити перший список (ліст. 3.11).

Лістинг 3.11 – Створення бази даних та користувача

```
import { config, list } from '@keystone-6/core';
import { allowAll } from '@keystone-6/core/access';
import { text } from '@keystone-6/core/fields';

export default config({
  db: {
    provider: 'sqlite',
    url: 'file:./keystone.db',
  },
  lists: {
    User: list({
      access: allowAll,
      fields: {
        name: text({ validation: { isRequired: true } }),
        email: text({ validation: { isRequired: true }, isIndexed:
'unique' }),
      },
    }),
  },});
```

Кінець лістингу 3.11

ВИСНОВКИ

Розробка програмного застосунку для електронної комерції є важливим кроком до цифрової трансформації бізнесу. Такий застосунок не лише підвищує ефективність внутрішніх процесів, але й покращує взаємодію з клієнтами, забезпечує безпеку даних та сприяє зростанню бізнесу. Інтеграція з сучасними технологіями та гнучкість архітектури дозволяють бізнесам швидко адаптуватися до змінних умов ринку та вимог клієнтів, забезпечуючи конкурентні переваги та стійкий розвиток.

В результаті виконаної роботи було проаналізовано сучасний стан процесів електронної комерції та досліджено популярні методи в створені програмних додатків а також методи для вирішення поставленого завдання та вибрано найкращі алгоритми для реалізації програмного продукту

Розроблено back-end, front-end та адміністративної частини системи. В результаті роботи вдалося створити зручний та інтуїтивно зрозумілий графічний інтерфейс користувача, що забезпечує легке користування додатком.

Було інтегровано основні платіжні системи для спрощення процесу оплати та підвищення безпеки транзакцій.

Завдяки технологіям Keystone вдалося оптимізувати програмний застосунок для роботи на мобільних пристроях, забезпечуючи безперебійний доступ та зручність користування незалежно від платформи.

Також було реалізовано модульну архітектуру застосунку, що дозволить легко додавати нові функції та масштабувати систему відповідно до зростання бізнесу та зміни його потреб.

Після тестування програмного продукту, виправлено основні недоліки, що дозволить уникнути помилок в роботі застосунку в майбутньому, а також забезпечить стабільну функціональність.

.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Try Shopify free and start a business or grow an existing one. URL: https://wallnut.digital/shops/shopify/review/ (дата звернення: 15.04.2024).

2. An enterprise platform designed for endless growth. URL: https://business.adobe.com/products/magento/magento-commerce.html (дата звернення: 15.04.2024).

3. WooCommece is the best platform to grow with. URL: https://woocommerce.com/learn/ (дата звернення: 16.04.2024).

4. Replatforming Guide: Roadmap Migrating Your Ecommerce Store. URL: https://www.bigcommerce.com/resources/guides/ (дата звернення: 17.04.2024).

5. Professional eCommerce platform to sell online. URL: https://www.wix.com/app-market/wix-stores (дата звернення: 20.04.2024).

6. Create a website or online store with an all-in-one solution from Squarespace. URL: https://www.squarespace.com/circle (дата звернення: 20.04.2024).

7. Amazon Webstore is a software as a service e-commerce platform. URL: https://aws.amazon.com/solutions/retail/web-store/ (дата звернення: 20.04.2024).

8. E-commerce Defined: Types, History, and Examples. URL: https://www.investopedia.com/terms/e/ecommerce.asp (дата звернення: 20.04.2024).

9. The superpowered CMS for developers. URL: https://keystonejs.com/docs/getting-started (дата звернення: 20.04.2024).

10. Architectury is a toolchain aimed to ease developing multiplatform mods. URL: https://docs.architectury.dev/start (дата звернення: 20.04.2024).

11. Node.js is a free, open-source, cross-platform JavaScript run-time environment. URL: https://nodejs.org/en/learn/getting-started/introduction-to-nodejs (дата звернення: 20.04.2024).

12. Docker is a platform designed for helping developers build, share, and run container applications. URL: https://docs.docker.com/ (дата звернення: 20.04.2024).

13. PostgreSQL, the world's advanced open source database. URL: https://www.postgresql.org/ (дата звернення: 02.05.2024).

14. Take advantage of Cloudinary's capabilities in your environments and technologies. URL: https://cloudinary.com/documentation (дата звернення: 03.05.2024).

15. Keystone's GraphQL API. URL: https://keystonejs.com/docs/config/overview (дата звернення: 05.05.2024).