# Міністерство освіти і науки України Національний технічний університет «Дніпровська політехніка»

Інститут електроенергетики

(інститут)

Факультет інформаційних технологій

(факультет)

Кафедра інформаційних технологій та комп’ютерної інженерії

(повна назва)

# ПОЯСНЮВАЛЬНА ЗАПИСКА кваліфікаційної роботи ступеня бакалавра

**студента** Гордієнко Данило Дмитрович

(П.І.Б.)

# академічної групи 121-20-1

(шифр)

**спеціальності** 121 Інженерія комп’ютерного забезпечення

(код і назва спеціальності)

# за освітньо-професійною програмою 121 Інженерія комп’ютерного забезпечення

(офіційна назва)

на тему Розробка  інтернет-магазину електронних пристроїв та аксесуарів з використанням  React, MobX, Sequelize.

(назва за наказом ректора)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Керівники** | **Прізвище, ініціали** | **Оцінка за шкалою** | | **Підпис** |
|  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Рецензент** |  |  |  |  |
|  |  |  | | |
| **Нормоконтролер** |  |  |  |  |

# Дніпро

## **2023**

**ЗАТВЕРДЖЕНО:**

інформаційних технологій та комп’ютерної інженерії

(повна назва)

(підпис) (прізвище, ініціали)

" " 2022 року.

# ЗАВДАННЯ на кваліфікаційну роботу ступеня бакалавр

**студента**  **академічної групи\_\_**

(прізвище, ініціали)

**спеціальності** 121 Інженерія комп’ютерного забезпечення

(код і назва спеціальності)

# за освітньо-професійною програмою 121 Інженерія комп’ютерного забезпечення

(офіційна назва)

**на тему** Розробка  інтернет-магазину електронних пристроїв та аксесуарів з використанням  React, MobX, Sequelize.

(назва за наказом ректора)

затверджена наказом ректора НТУ «Дніпровська політехніка» від 16.05.2023 р. № 350-с

|  |  |  |
| --- | --- | --- |
| **Розділ** | **Зміст завдання** | **Термін виконання** |
|  |  |  |
|  |  |  |
|  |  |  |

**Завдання видано**  .

(підпис керівника) (прізвище та ініціали)

# Дата видачі Дата подання до атестаційної комісії

**Прийнято до виконання**  Годієнко.Д.Д.

(підпис студента) (прізвище та ініціали

**РЕФЕРАТ**

Пояснювальна записка: \_\_\_ с., \_\_\_ рис., \_\_\_ табл., \_\_ дод., \_\_\_ джерел.

Об’єкт розробки : Інтернет-магазин електронної техніки та аксесуарів "Dom1store".

Мета даної роботи полягає у створенні сучасного інтернет-магазину, що спеціалізується на електроніці та аксесуарах, з використанням технологій React, MobX і Sequelize. Система була спроектована таким чином, щоб бути гнучкою та адаптованою до змін типу і набору функцій, а також до оновлень технологій і програмного забезпечення. Для перевірки функціональності інтернет-магазину було проведено серію тестів за ключовими аспектами його роботи.

Результати тестів були представлені у вигляді відповідних звітів. розроблений інтернет-магазин відповідає всім вимогам, що пред'являються до подібних систем. Він забезпечує зручне управління товарами, швидке оформлення замовлення і дружній інтерфейс. Структура і функціональність магазину були ретельно продумані, щоб забезпечити зручність управління і можливості для подальшого розширення.

**ЗМІСТ**

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ……………………………………………...

ВСТУП……………………………………………………………………………….

РОЗДІЛ 1. АНАЛІЗ ПРЕДМЕТНОЇ ГАЛУЗІ ТА ПОСТАНОВКА ЗАВДАННЯ .

* 1. Загальні відомості з предметної галузі…………………………………
  2. Призначення розробки та галузь застосування………………………..
  3. Підстава для розробки…………………………………………………..
  4. Постановка завдання……………………………………………………….
  5. Вимоги до програми або програмного виробу………………………….
     1. Вимоги до функціональних характеристик………………………………
     2. Вимоги до інформаційної безпеки…………………………………………
     3. Вимоги до складу та параметрів технічних засобів……………………….
     4. Вимоги до інформаційної та програмної сумісності ……………………

РОЗДІЛ 2. ПРОЄКТУВАННЯ ТА РОЗРОБКА ПРОГРАМНОГО ПРОДУКТУ..

* 1. Функціональне призначення програми……………………………………..
  2. Опис застосованих математичних методів………………………………
  3. Опис використаної архітектури та шаблонів проектування…………..
  4. Опис використаних технологій та мов програмування…………………
  5. Опис структури програми та алгоритмів її функціонування ………….
  6. Обґрунтування та організація вхідних та вихідних даних програми….
  7. Опис розробленого програмного продукту…….............…………………
     1. Використані технічні засоби…….………………………………………….
     2. Використані програмні засоби……………………………………………..
     3. Виклик та завантаження програми…………………………………………
     4. Опис інтерфейсу користувача………………………………………………

РОЗДІЛ 3. ЕКОНОМІЧНИЙ РОЗДІЛ......…………………………………………

* 1. Розрахунок трудомісткості та вартості розробки програмного продукту.
  2. Рахунок витрат на створення програми……………………………………

ВИСНОВКИ…………………………………………………………………………..

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ…………………………………………….

Додаток А. Код програми……………………………….…………………………..

Додаток Б. Відгук керівника економічного розділу……………………………….

Додаток В. Перелік файлів на диску……………………………………………….

**ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ,**

# СКОРОЧЕНЬ І ТЕРМІНІВ

IT - інформаційні технології

КС - комп'ютерна система

React - бібліотека JavaScript для розробки інтерфейсів користувача

MobX - бібліотека для управління станом додатків на JavaScript

Sequelize - ORM (Object-Relational Mapping) для Node.js, який підтримує різні бази даних

API - Application Programming Interface (Інтерфейс програмування додатків)

UI - User Interface (Інтерфейс користувача)

UX - User Experience (Взаємодія користувача)

HTTP - Hypertext Transfer Protocol (Протокол передачі гіпертексту)

URL - Uniform Resource Locator (Уніфікований локатор ресурсів)

CDN - Content Delivery Network (Мережа доставки контенту)

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

JWT - JSON Web Token (JSON-токен)

ORM - Object-Relational Mapping (Об'єктно-реляційне відображення)

SQL - Structured Query Language (Мова структурованих запитів)

API - Application Programming Interface (Інтерфейс програмування додатків)

# ВСТУП

Із розвитком технологій та зростанням цифрової активності, розробка інтернет-магазинів з використанням React, MobX та Sequelize стає невід'ємною частиною успішного сайту. Сьогодні споживачі все частіше віддають перевагу покупкам онлайн, а якісні та зручні інтернет-магазини відіграють важливу роль у залученні та утриманні цільових клієнтів.

На рахунок переваг React , та mobX .Перший дозволяє динамічно оновлювати вміст сторінки без перезавантаження, роблячи процес покупки зручнішим і швидшим для користувача. mobX відповідає за ефективне управління станом додатку, що важливо для інтерфейсів магазину. Використовуючи сучасні інструменти, він може вивчати поведінку користувачів, оптимізувати процеси продажів і реклами, швидко реагувати на зміни попиту і тенденції .

Інтернет-магазини, побудовані з використанням таких технологій, мають величезний потенціал зростання на швидкозмінному цифровому ринку. Впровадження передових технологій не тільки покращує клієнтський досвід, але й підвищує загальну ефективність магазину, роблячи його більш конкурентоспроможним та успішним у довгостроковій перспективі.

Такі нововведення допомагають не лише залучати нових клієнтів, але й утримувати старих покупців, які цінують своєчасність та зручність покупок онлайн . Також, вони забезпечують прозорий механізм отримання зворотного зв'язку від клієнтів, такі як чат та електронна пошта. Це не тільки дозволяє ефективно вирішувати питання, що виникають в процесі покупки, але й створює сприятливе враження про магазин у покупців. Важливим є й розуміння та використання тенденцій електронної комерції. Наприклад, розробники інтернет-магазинів можуть збільшити коефіцієнт конверсії та середній час оформлення замовлення, впровадивши технологію розпізнавання облич та системи рекомендацій, які адаптуються до індивідуальних потреб кожного користувача.

Загалом, розробка інтернет-магазинів на перерахованих вище технологіях дозволяє не тільки залучати та утримувати клієнтів у сучасному цифровому світі, але й дає можливість для подальшого розвитку та успіху компанії. Це дозволяє їй конкурувати та адаптуватися до швидкозмінних умов ринку, забезпечуючи при цьому високу якість обслуговування та задоволеність клієнтів. Така розробка також допомагає покращити клієнтську частину , впроваджуючи новий функціонал та привабливі елементи , які роблять процес покупок більш зручним і привабливим для клієнтів. А також допомагає поліпшити аналіз процесу продажів, забезпечуючи більш точне розуміння потреб і вподобань клієнтів та ефективне планування маркетингових кампаній і промоакцій.

Для самого розробника , важливо відзначити, що використання React, MobX і Sequelize дозволяє значно скоротити час, необхідний для розробки і впровадження нових функцій, завдяки архітектурі і гнучкості цих технологій.

Із переваг ще можу провести можливість легкого розширення в міру розширення магазинів або зміни бізнес-моделей, що дозволяє компаніям швидко реагувати на ринкові умови і потреби клієнтів. А також збільшення швидкості завантаження та продуктивності інтернет-магазинів, а використання передових технологій розробки може покращити користувацький досвід та зменшити кількість відмов від покупок.

Використання передових технологій дозволяє створювати більш гнучкі та адаптивні системи управління контентом, спрощує процес додавання та редагування товарів, а також покращує користувацький досвід для менеджерів магазинів. також може допомогти автоматизувати багато бізнес-процесів, таких як управління запасами, обробка замовлень і взаємодія з постачальниками ,умісність інтернет-магазинів з різними браузерами та пристроями, та можливість оптимізувати SEO та підвищити видимість в пошукових системах.

# 1 АНАЛІЗ ПРЕДМЕТНОЇ ГАЛУЗІ ТА ПОСТАНОВКА ЗАВДАННЯ

# 1.1 Загальні відомості з предметної галузі

Сучасні проблеми розвитку інтернет-магазинів електронних аксесуарів дуже складні, а їх ефективне вирішення вимагає комплексного підходу. Розвиток інтернет-технологій і стрімке зростання конкуренції на ринку електронної комерції зумовили необхідність створення інноваційних платформ і підвищення зручності електронних продажів. При цьому необхідно враховувати потреби користувачів у зручності та швидкості, а також постійно зростаючі вимоги до безпеки, адаптивності до змін та персоналізації взаємодії.

Експертний аналіз показує, що порівняно з існуючими лідерами ринку, яким може бути " Touch ", інші представники данної послуги мають обмежену адаптацію до інновацій, завантаження сторінок, та технічні проблеми з балансуванням функціональності та швидкості роботи платформи. Такі недоліки можуть погано впливати на продажі , та не цікавити клієнтів.

Зокрема, Touch, який відомий своїм багатим інтерфейсом, також може страждати від технічних проблем, таких як час завантаження сторінок і баланс між функціональністю і швидкістю. Крім того, можливості для оптимізації продажів і підвищення конверсії можуть бути втрачені, якщо не приділяти уваги інструментам аналітики та персоналізації.

Тому для досягнення успіху в цій сфері важливо розробити інтернет-магазин, який як мінімум буде не відставати від своїх конкурентів , та буде відповідати сучасним стандартам процес купівлі і продажу товарів і послуг через Інтернет і адаптується до змін і розширення відповідно до потреб клієнтів.

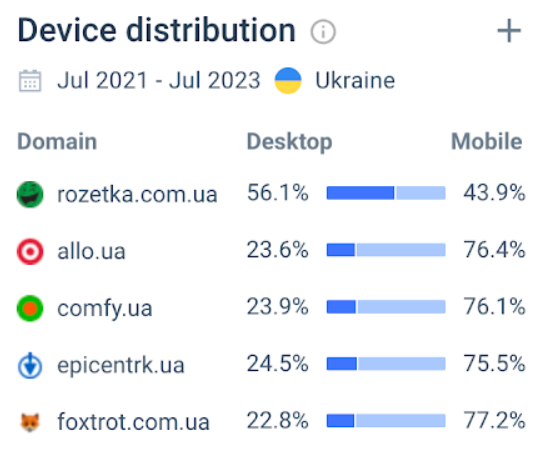


Рисунок 1.1 – Статистика найпоширеніших інтернет магазинів

Аналіз аналогів: Порівнюючи існуючі інтернет-магазини в галузі електроніки та аксесуарів, можна відзначити якісне і популярне рішення – інтернет-магазин "Touch". Цей магазин відомий своїм широким асортиментом, розширеним інтерфейсом і швидким оформленням замовлення. Однак навіть у такого популярного магазину можуть бути проблеми, такі як повільне завантаження сторінок і устарілий дизайн .

Ступінь вирішення проблеми: Інтернет-магазин "Touch" надає базовий функціонал для покупців і вирішує усі можливі завдання з управління товарами та замовленнями. Однак через постійні зміни в індустрії йому не вистачає гнучкості,та стилізаціі і він може виявитися не в змозі реагувати на нові вимоги ринку: "Touch" має проблеми з балансом між швидкістю і розширеною функціональністю сайту. Додавання нових функцій часто призводить до тривалого завантаження сторінок, що може негативно позначитися на користувацькому досвіді". Варто також зазначити, що у "Touch" дизайн є досить застарілим, та проблематичний, тобто не відповідає сучасним тенденціям. Покращення дизайну може зробити ваш сайт привабливішим і зручнішим у використанні, що може позитивно вплинути на задоволеність клієнтів.

Незадоволені вимоги до продукту: "Touch" , як інтернет мгазин відповідає багатьом вимогам, але не завжди відповідає сучасним тенденціям у сфері дизайну та безпеки. Поліпшення цих аспектів підвищить конкурентоспроможність магазину і забезпечить стабільне становище на ринку.

## **1.2** Призначення розробки та галузь застосування

У сучасному світі електронна комерція стала невід'ємною частиною повсякденного життя. Все більше покупців віддають перевагу зручності і доступності інтернет-магазинів. Тому виникає необхідність розробки інноваційних інтернет-магазинів, які не тільки пропонують зручне замовлення товарів, але і мають привабливий дизайн, який змушує покупців хотіти відвідати їх знову. У цьому розділі аналізуються цілі і необхідність розробки інтернет-магазину «Dom1store.» Він також описує сферу реалізації цієї системи.

Призначення

1. Зручне і швидке замовлення товару

Основною метою створення інтернет-магазину «Dom1store» є створення сайту, де покупці зможуть швидко і просто придбати улюблені електронні пристрої та аксесуари. За допомогою «Dom1store» клієнти можуть ефективно переглядати товар, шукати потрібний товар і додавати його в кошик. 1. Створення сайту, який дозволяє покупцям ефективно переглядати товари, шукати потрібні їм товари і додавати їх у кошик

2. Зміцнення конкурентоспроможності компанії

Використовуючи Dom1store, компанії можуть підвищити свою конкурентоспроможність за допомогою електронної комерції. Інтернет приваблює нових клієнтів, збільшує продажі і розширює сферу діяльності компанії.

3. Покращений користувальницький досвід

Однією з головних цілей при розробці «Dom1store» є створення інтуїтивно зрозумілого, привабливого і зручного інтерфейсу для покупця. Сучасні технології реактивного програмування, такі як React і MobX, забезпечують зручність користувача, створюють швидкі і адаптовані веб-інтерфейси

4. Забезпечення безпеки та надійності

Безпека та захист персональних даних є пріоритетом у розвитку Dom1store, а такі технології, як Sequelize, дозволяють не тільки аутентифікацію та авторизацію, але й захист від можливих кібератак.

Галузі застосування - «Dom1store» спеціалізується на продажу широкого спектру електронних пристроїв, включаючи смартфони, ноутбуки, планшети, комп'ютери та інші гаджети. Будь-який клієнт зможе зайти на сайт за допомогою комп’ютера , або телефона та знайти широкий асортимент товарів різних марок і моделей комп'ютерів і смартфонів за доступними цінами.

# 1.3 Підстави для розробки.

**Висновки**

Розроблене програмне забезпечення є ефективним і працездатним інструментом для автоматизації діяльності інтернет-магазину електронної техніки. Воно дозволяє здійснювати автоматизований збір, обробку і маніпулювання даними, спрощуючи процеси продажу, обліку та контролю за товарами в мережі магазинів.

Програмне забезпечення надає можливість виконувати різноманітні операції, такі як додавання, видалення і редагування інформації про товари, клієнтів, постачальників, продажі, покупців, а також перегляд інформації про них.

Актуальність розробленого програмного забезпечення обумовлена широким попитом на такі програмні продукти, що надають можливість електронного зберігання даних про товари, продажі, надходження, ведення бази даних, підбиття підсумків з продажу, отримання звітів і формування супутньої ділової документації.

Розроблене програмне забезпечення інформаційної системи призначене для застосування в будь-якій мережі інтернет-магазинів з подібним родом діяльності і схожими функціональними вимогами.

Створений додаток дозволить оптимізувати та спростити дії по веденню бази даних мережі магазинів; скоротити час на оформлення продажу; підвищити ефективність діяльності компанії шляхом електронного ведення документації з продажу і можливістю аналізу наявних даних і надання необхідних звітів і супутньої ділової документації.

Структура програми складається з клієнтського додатку, написаного на мові програмування високого рівня JavaScript (з використанням React та Node.js), що взаємодіє з базою даних, розробленою мовою SQL в системі управління базами даних.

В "Економічному розділі"

## **1.4** Постановка завдання

Мета: Техніко-економічна суть проблеми і причини необхідності її вирішення: створення інтернет-магазину у відповідь на зростаючий попит сучасного ринку обумовлено тим, що все більше користувачів віддають перевагу onlinePlatforms для покупки. Головна мета - кожен клієнт може легко і швидко отримати те, що йому потрібно через інтернет. З точки зору його технічної та економічної сутності, це повинно бути використання новітніх технологій, які забезпечують ефективність, безпеку і надійність системи.

Це список компонентів, які є нащадками розвиненої системи. Один з аспектів цієї угоди стосується продуктів і послуг, що надаються користувачами та їх особистою інформацією. Також за умови оформлення замовлення і оплати. Інформаційна система інтернет

Дані: продукт, категорія товару, користувач, замовлення. Змінні: ідентифікатор продукту, назва продукту, ціна продукту, кількість доступних одиниць.

Опис функції інформаційного ресурсу: Вихід системи інформуватиме користувачів про останні новини продукту, наявність на складі, ціни та умови доставки.

З точки зору збору та обробки вхідної інформації, для контролю якості, а також для внесення змін вимоги включають: Система повинна дозволяти:

Автоматизована система обробки замовлень і платежів. Адміністратор може змінювати каталог продукції, а також замовлення, відстежувати ціни та наявність на складі на льоту.

Автоматичне вирішення завдань може бути припинено при появі технічних збоїв, які порушують роботу системи та порушують захист даних користувачів.

Зв’язок між цією системою та іншими системами включає інтеграцію з системами оплати, доставки та обліку, щоб надавати користувачам найвищі послуги.

У різних ситуаціях вирішення системних завдань має бути налагоджений розподіл функцій між персоналом і технічними засобами: функції, які може запропонувати система, поділяються на ті, які може виконати людина, і інші, які необхідно виконати. машиною. Наприклад, механічна обробка замовлень і платежів входитиме в обов’язки технічного відділу, але адміністрування каталогу продукції буде здійснюватися персоналом.

## **1.5** Вимоги до програми або програмного виробу

## **1.5.1** Вимоги до функціональних характеристик

Функції, які користувач може застосувати:

Реєстрація та авторизація: щоб мати можливість увійти в систему, користувач повинен мати можливість створити обліковий запис, ввівши особисту інформацію, таку як ім’я та електронна адреса. Також необхідно, щоб система дозволяла входити користувачам, які вже мають облікові записи.

Для початку користувач повинен мати можливість розміщувати товари в кошику для покупок, видаляти їх або змінювати кількість товарів. Що стосується особистого облікового запису, користувачеві потрібна можливість мати власний обліковий запис, де він може переглядати таку інформацію, як дані про себе та історію своїх замовлень, а також змінювати особисті дані.

Функції адміністратора включають:

Для керування продуктами адміністратор матиме права додавати, оновлювати та видаляти продукти та контролювати їх категоризацію. Що стосується керування замовленнями, замовлення, розміщені користувачами, переглядаються та контролюються адміністратором, включаючи підтвердження, скасування або відправку замовлення.

Організація введення та виведення даних — це спосіб структурованого

зберігання даних для полегшення доступу, пошуку та маніпулювання даними. Користувач вводить інформацію в інтерфейс веб-сторінки за допомогою різних елементів керування, таких як текстові поля, кнопки та розкривні списки. Крім того, дані можна отримати із запитів до сервера. Результат запиту користувача, представлений на веб-сторінці, можна побачити в зручному для читання форматі, як каталог продуктів із переліком продуктів із детальним описом разом із їхніми зображеннями. Часові характеристики:

Продуктивність веб-браузера користувача залежить від швидкості, з якою обробляються ці запити; це означає, що сервер повинен обробляти запити за якомога менший час, а відповідь займає не більше 1-2 секунд для більш плавної та безперебійної взаємодії. Швидкість завантаження сторінки: веб-сторінки мають завантажуватися швидко, без помітних затримок, щоб забезпечити комфортну роботу користувача.

Організація та зберігання даних

Використовуючи Sequelize для ORM, дані про продукт, користувача та замовлення знаходяться в реляційній базі даних.

Існують різні критерії вибору формату представлення даних, серед яких: Дані вводяться кінцевим користувачем через веб-інтерфейс у форми, які можуть включати текстові поля, розкривні списки та перемикачі тощо. Система забезпечує вихідні дані, які можна легко прочитати, наприклад каталоги продуктів, дані вхідні аркуші, звіти або діаграми.

## **1.5.2** Вимоги до інформаційної безпеки

Гарантія надійної роботи

Машина повинна бути обладнана заходами для виявлення та усунення несправностей, які можуть виникнути, щоб вона могла продовжувати свою роботу гладко, без будь-яких перерв. Використання моніторингу та журналювання подій дозволяє знаходити проблеми та реагувати на них до того, як вони стануть серйозними.

Як інформація потрапляє в компанію та виходить з неї:

Перш ніж зберігати будь-які введені користувачем дані в базі даних, вони повинні пройти перевірку та обробку як запобіжний захід проти SQL-ін’єкції та атак XSS. Подібним чином дані, які надсилаються для передачі, потребують ретельного вивчення та належної перевірки, щоб забезпечити їх безпеку.

Час відновлення після розчарування має відрізнятися за тривалістю, і при визначенні тривалості часу відновлення доцільно враховувати тяжкість розчарування та його очікуваний вплив на реалізацію проекту.

Однією з важливих вимог, яку слід враховувати, є резервне копіювання даних і відновлення системи в разі можливих збоїв або втрати. Крім того, ми також повинні забезпечити захист від несанкціонованого доступу.

При контролі доступу користувачів до різних частин системи здійснюється за допомогою аутентифікації та авторизації. Шифрування мережевих даних і зберігання на сервері важливі, коли ви передаєте конфіденційні дані у своїй мережі.

Існують різні методи, які використовуються для запобігання несанкціонованому копіюванню програмного забезпечення, наприклад активація продукту, ключі та керування цифровими правами.

Застосування технологій у боротьбі з підробкою програмного забезпечення, що включає обфускацію та зворотний інжиніринг. Управління та захист системних даних.

Система аудиту та моніторингу є важливою системою, яка має бути на місці, щоб допомогти виявити несанкціонований доступ, а також втрату цілісності даних. Застосування методів контролю версій і журналювання допомагає відстежувати зміни, внесені до інформації, і визначати, чи відбуваються чи мали місце будь-які зловмисні дії.



Рисунок 1.2 – Найпоширеніші кібер-атаки

Забезпечення безпеки може включати такі заходи:

SQL-ін'єкція та XSS атаки:

Використання параметризованих запитів у базі даних.

Валідація та екранування введених користувачем даних.

Використання спеціальних фільтрів і бібліотек для очищення введених даних перед виведенням на веб-сторінки.

Розподілені деніал-оф-сервіс (DDoS) атаки:

Використання облачних або CDN рішень для розподілення трафіку.

Налаштування мережевих пристроїв для виявлення та блокування небезпечного трафіку.

Використання служби захисту DDoS для фільтрації атакувального трафіку перед його надходженням до серверів.

Фішинг та соціальна інженерія:

Проведення навчання співробітників щодо виявлення підозрілих електронних листів та посилань.

Використання антивірусного програмного забезпечення для виявлення та блокування шкідливих вкладень.

Малварні атаки:

Використання антивірусного програмного забезпечення з оновленням баз даних загроз.

Застосування файрвола для блокування небезпечного трафіку, що намагається з'єднатися зі зловмисними доменами чи IP-адресами.

Атаки на злам паролів:

Вимагання складних паролів від користувачів та їх періодична зміна.

Використання двофакторної аутентифікації для додаткового захисту облікових записів.

Атаки з викраденням даних:

Шифрування конфіденційних даних під час їх передачі та зберігання.

Використання моніторингу інформаційних потоків для виявлення незвичайних або підозрілих активностей.

## **1.5.3** Вимоги до складу та параметрів технічних засобів

Характеристики серверної інфраструктури:

Процесор: Мінімально потрібний процесор з достатньою швидкістю для обробки запитів та виконання операцій з базою даних. Оперативна пам'ять: Достатньо ОЗП для запуску Node.js сервера та обробки запитів від клієнтів. Диск: Достатньо простору для зберігання програмного коду, конфігураційних файлів та бази даних. Мережеве з'єднання: Висока швидкість та надійність мережевого з'єднання для обробки запитів від клієнтів.

Характеристики клієнтських пристроїв:

Процесор: Достатньо потужний для відтворення інтерфейсу магазину, обробки даних та виконання складних операцій. Оперативна пам'ять: Вистачає для швидкої роботи веб-браузера та відображення великої кількості товарів. Браузер: Підтримка сучасних версій браузерів для правильного відображення та функціонування інтерфейсу магазину. Програмне забезпечення: Node.js: Мінімальна підтримувана версія Node.js для запуску серверної частини додатку. React: Використання актуальної версії React для розробки клієнтського інтерфейсу. MobX: Наявність та підключення MobX для керування станом додатку на клієнтській стороні. Sequelize: Використання Sequelize для забезпечення доступу до бази даних та взаємодії з нею. База даних: Система управління базами даних (СУБД): Використання підтримуваної СУБД, яка підтримує Sequelize і відповідає потребам магазину. Характеристики СУБД: Достатньо продуктивна для обробки запитів та забезпечення швидкодії роботи магазину.

## **1.5.4** Вимоги до інформаційної та програмної сумісності

До інформаційних структур висуваються деякі вимоги, наприклад.

Для будь-якого інтернет-магазину при створенні бази даних важливо визначити потреби, які мають бути задоволені. Нормалізація таблиць забезпечить ефективне та ефективне зберігання та обробку даних. Виявлення залежностей між різними типами даних, такими як продукти, категорії, користувачі та замовлення через реляційні зв’язки. Щоб гарантувати як швидкість, так і надійність, масштабованість бази даних є одним із важливих факторів, який необхідно враховувати при визначенні її структури даних. Щоб представити дані про продукт у добре організований спосіб, необхідно враховувати характеристики, зображення, ціни тощо. Дані користувачів повинні зберігатися надійно та конфіденційно з урахуванням вимог захисту персональної інформації.

Необхідно враховувати вимоги до вихідного коду для інтерфейсу клієнта.

Щоб досягти ясності та простоти обслуговування, будь-який код має бути організований належним чином і належним чином розроблений. У той же час слід також застосовувати інноваційні методи, які сьогодні широко поширені в розробці, такі як компонентна архітектура та реактивне програмування.

Розробка бекенда включає забезпечення безпеки на стороні сервера шляхом запобігання потенційним загрозам безпеці, таким як атаки XSS.

Хороша структура та відповідна документація забезпечать легкість розуміння та підтримки базового коду. Процедури безпеки, які необхідно запровадити, включають хешування паролів, перевірку автентифікації та авторизації, перевірку обробки помилок і виняткові положення, які забезпечують надійність і безпеку програми.

Мови програмування та програмні засоби:

JavaScript: Одним із способів використання JavaScript є мова програмування як для клієнта, так і для сервера. Іншим можливим підходом є використання останніх версій JavaScript, щоб мати можливість використовувати всі можливості та функції, реалізовані в них.

Бібліотеки та фреймворки є основними компонентами будь-якого процесу розробки програмного забезпечення.

React.js обрано для розробки на стороні клієнта через його широке визнання та ефективність. Node.js разом із Express.js формує серверну частину, яка забезпечує як високу продуктивність, так і масштабованість.

Вимоги до інформаційних структур, вихідних кодів, мов програмування та програмних засобів обґрунтовано на основі усталеної практики розробки програмного забезпечення. Вимоги до безпеки, швидкості та масштабованості забезпечать надійну та якісну роботу системи. Застосування сучасних технологій і методологій дозволить виробляти продукт, який буде відповідати чинним стандартам, а також вимогам користувачів.

# 2 ПРОЕКТУВАННЯ ТА РОЗРОБКА ПРОГРАМНОГО ПРОДУКТУ

## **2.**1 Функціональне призначення програми

Функціональне призначення програми

Управління продуктами та каталогами: Однією з функцій системи керування продуктами є ведення каталогу шляхом додавання, видалення та редагування продуктів. Це також дозволяє класифікувати продукти на основі різних параметрів, таких як ціна, бренд і характеристики. Система здатна надати детальну інформацію про кожен продукт, включаючи зображення, описи та характеристики. Процес реєстрації та оформлення: коли ви додаєте товари до кошика для покупок і переходите до оформлення замовлення, ви можете оплатити своє замовлення через різні платіжні системи. Системи користувача: також важливо, щоб покупці мали адекватне та надійне відстеження своїх замовлень. Це означає, що система повинна мати можливість відстежувати замовлення та надсилати повідомлення про будь-які зміни їх статусу. Реєстрація та аутентифікація користувача. Можливість перегляду та зміни персональних даних. Управління замовленнями є одним із факторів. Сканери можна використовувати для моніторингу рівня запасів на складах. Коли кількість товару невелика, він надсилає сповіщення, щоб підтримувати рівень запасів. Відстежуйте кількість продуктів на складі та оновлюйте інформацію щоразу, коли розміщується замовлення.

Експлуатаційне призначення програми

Автоматизований процес: Зменште потребу в ручному втручанні в ручне керування продажами та обробку замовлень. Підвищте продуктивність співробітників, автоматизувавши рутинні завдання, щоб співробітники могли працювати ефективніше.

Друга ініціатива спрямована на збільшення різноманітності пропонованих продуктів. Це важлива стратегія, оскільки вона допомагає обслуговувати ширшу базу клієнтів і задовольняти їхні унікальні потреби. Завдяки платформі електронної комерції, яка підтримує різноманітні продукти, на полицях можна розмістити більше продуктів. Поліпшення обслуговування клієнтів: Забезпечити швидкий і зручний процес покупок

## **2.**2 Опис застосованих математичних методів.

Розробка інтернет-магазину не обов'язково потребує використання математичного апарату. Основні аспекти розробки такого магазину включають створення інтерфейсу користувача, роботу з базою даних для зберігання продуктів та інформації про замовлення, розробку функціоналу оплати та доставки. Головною метою є створення зручної та безпечної платформи для клієнтів, яка дозволяє їм придбати товари зручним способом. Тому важливіше зосередитися на програмуванні, дизайні та безпеці, ніж на математичних аспектах.

## **2.**3 Опис використаної архітектури та шаблонів проектування.

Додаток побудований на основі клієнт-серверної архітектури, що дозволяє нам ефективно розділити логіку між клієнтом і сервером.

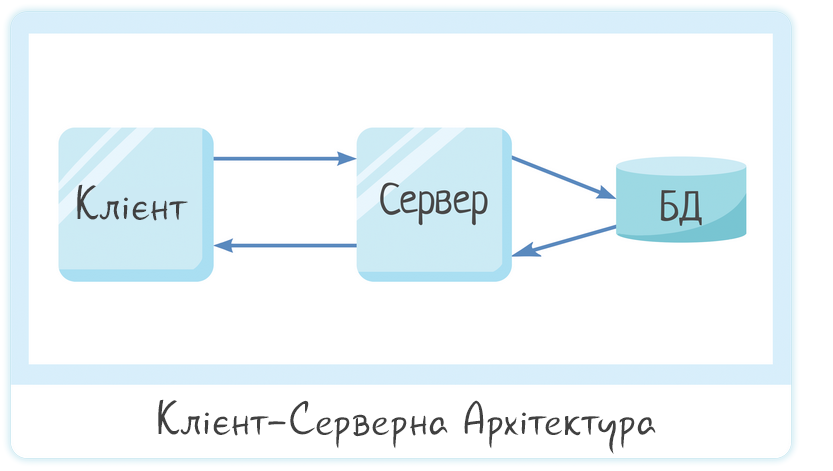


Рисунок 2.1 – Принцип роботи клієнт-серверної архітектури

Клієнт-серверна архітектура складається з двох основних елементів: клієнта і сервера. Клієнт - це комп'ютер користувача, який потребує інформації з сервера або надсилає інструкції серверу.

Сервер - це комп'ютер або пристрій, який є більш потужним, ніж зазвичай, і може виконувати певні дії, такі як запуск програмних програм, виконання завдань, орієнтованих на обслуговування за клієнтськими запитами, полегшення доступу користувачів до певних ресурсів та зберігання інформації та баз даних. Модель цієї системи полягає в тому, що клієнт відправляє запит на сервер, а сервер обробляє запит і відправляє кінцевий результат клієнту. Сервер може одночасно обслуговувати декілька клієнтів. Коли кілька запитів надходять одночасно, сервер черг і обробляє їх по одному.

Запити можуть бути пріоритетними, і може бути класифікація, яка спочатку приймає запит з найвищим пріоритетом.Модель такої системи полягає в тому, що клієнт відправляє запит на сервер, де він обробляється, і готовий результат відправляється клієнтові. Сервер може обслуговувати кілька клієнтів одночасно. Якщо одночасно приходить більше одного запиту, то вони встановлюються в чергу і виконуються сервером послідовно. Іноді запити можуть мати пріоритети. Запити з більш високими пріоритетами повинні виконуватися раніше.

Функції, які реалізуються на сервері:

* зберігання, доступ, захист і резервне копіювання даних;
* обробка клієнтського запиту;
* відправлення результату (відповіді) клієнту.
* Функції, які реалізуються на стороні клієнта:
* надання користувальницького інтерфейсу;
* формулювання запиту до сервера і його відправка;
* отримання результатів запиту і відправка додаткових команд (запитів на додавання, оновлення або видалення даних).

Функції, які реалізуються на стороні клієнта:

* надання користувальницького інтерфейсу;
* формулювання запиту до сервера і його відправка;
* отримання результатів запиту і відправка додаткових команд (запитів на додавання, оновлення або видалення даних).

Підхід до моделі життєвого циклу програмного забезпечення є гібридним. Agile, з одного боку, дає нам змогу задовольняти потреби клієнтів і швидко запускати нові функції відповідно до мінливих вимог ринку, тоді як Waterfall гарантує, що ми ретельно розробляємо та створюємо серверні рішення з увагою до деталей.

# 2.**4** Опис використаних технологій та мов програмування**.**

Мови програмування

Серед різних мов програмування, які ви можете вибрати в індустрії веб-розробки, JavaScript виділяється як дуже відома та популярна. Він зазвичай використовується для створення динамічних веб-сайтів і веб-додатків. Тепер давайте розглянемо деякі переваги використання JavaScript для розробки інтернет-магазину. Розробка клієнта та сервера: JavaScript може використовуватися як на стороні клієнта, так і на стороні сервера. Застосування JavaScript на стороні клієнта включає створення інтерактивного інтерфейсу користувача, а також зв’язок із серверами через AJAX. З іншого боку, JavaScript можна використовувати на Node.js для створення швидких і гнучких серверних програм. Асинхронність: Однією з найкорисніших функцій JavaScript є його здатність виконувати операції введення-виведення без переривання основного потоку, тобто в асинхронному режимі.[1]

Node.js — це середовище виконання JavaScript на стороні сервера, яке працює за допомогою механізму V8 Google Chrome. Ось деякі з переваг, які ви можете відчути від використання у розробці свого онлайн-магазину: Node.js використовує неблокуючу модель введення/виведення, яка відповідає за його швидкість і ефективність. Під час його запуску програми на стороні сервера можуть бути розроблені так, щоб вони були швидкими та винахідливими. Він забезпечує високу пропускну здатність і не страждає від багатьох одночасних з’єднань. Також він пропонує широкий спектр бібліотек і фреймворків, що полегшує розробку серверних програм. Наприклад, Express.js виділяється як кращий фреймворк для створення веб-серверів на основі Node.js. Одна мова, яка керує всіма: використання JavaScript як мови сценаріїв на стороні сервера та клієнта дає змогу розробнику створювати консолідовані програми, використовуючи лише одну мову програмування. Я вибрав JavaScript і Node.js для нашого проекту, оскільки обидва мають багатий набір функцій, високоефективні та швидкі в розробці веб-додатків, а також їх величезна популярність і активна спільнота розробників. Вони надають роздрібним торговцям фантастичні ресурси, якими вони можуть скористатися під час створення сучасних онлайн-магазинів, які добре функціонують, добре виглядають і водночас зручні для користувачів.[2]

Технології на стороні клієнта:

React.js: Популярна бібліотека JavaScript для побудови інтерфейсів користувача, відома своєю компонентною архітектурою, яка сприяє повторному використанню та підтримці коду. React спрощує процес створення інтерактивних інтерфейсів, розбиваючи додаток на менші, повторно використовувані компоненти. [3]

Redux: Прогнозований контейнер для стану JavaScript-додатків, який часто використовується з React для керування станом додатка. Redux забезпечує єдиний джерело правди для стану додатка, що полегшує його керування та налагодження в складних потоках даних. [4]

React Router DOM: Бібліотека маршрутизації для React-додатків, яка дозволяє декларативно налаштовувати маршрути. Вона дозволяє розробникам визначати динамічні маршрути та керувати навігацією у додатку, сприяючи створенню односторінкових додатків з кількома відображеннями.

Axios: Обіцяна HTTP-бібліотека для асинхронних запитів у JavaScript-додатках. Axios спрощує процес відправки HTTP-запитів та обробки відповідей, забезпечуючи такі функції, як перехоплення запитів та відповідей, автоматичний розбір JSON та обробка помилок. [5]

Formik: Бібліотека для створення форм у React-додатках, яка пропонує простий та інтуїтивно зрозумілий API для керування станом форми, її валідацією та поданням. Formik спрощує процес створення складних форм, обробляючи форму пов'язану логіку та надаючи інструменти для валідації та обробки помилок. [6]

Yup: JavaScript-бібліотека валідації схем, яка використовується з Formik для визначення та валідації схем форм. Yup дозволяє розробникам створювати міцні правила валідації для полів форми, забезпечуючи цілісність та однорідність даних. [7]

MobX: Бібліотека для керування станом JavaScript-додатків, особливо підходить для управління складним станом додатка з мінімальною кількістю коду. MobX використовує спостережувані структури даних та принципи реактивного програмування для автоматичного оновлення інтерфейсу користувача відповідно до змін стану. [8]

MobX React Lite: Легковажна бібліотека пов'язок React для MobX, яка пропонує оптимізовану інтеграцію з функціональними компонентами React. MobX React Lite спрощує процес підключення компонентів React до сховищ MobX, забезпечуючи ефективне управління станом у React-додатках.

React Select: Гнучкий та налаштовуваний компонент випадаючого списку для React-додатків, який поліпшує взаємодію користувача та доступність. React Select дозволяє розробникам створювати багатовибірні випадаючі списки, пошукові випадаючі списки та інші складні елементи інтерфейсу користувача з легкістю.

Swiper: Сучасна, мобільно-дружня бібліотека слайдера для побудови відгуків, слайдерів та галерей з відгуками у веб-додатках. Swiper підтримує жести на дотикових екранах, навігацію за допомогою клавіатури та плавні переходи, забезпечуючи безшовний досвід користувача на різних пристроях та розмірах екрану.

JWT Decode: Бібліотека для декодування JSON Web Tokens (JWT) в JavaScript-додатках. JWT Decode дозволяє розробникам витягувати та отримувати доступ до даних, закодованих у JWT, забезпечуючи аутентифікацію та авторизацію у клієнтському коді.

RC Slider: Компонент React для створення керованих ползунків із налаштованим стилем та поведінкою. RC Slider надає такі функції, як вибір діапазону, підказки та мітки, що робить його підходящим для реалізації ползунків у користувацькому інтерфейсі.

Web Vitals: Набір метрик для вимірювання та моніторингу продуктивності веб-сторінок, включаючи ключові показники продуктивності, такі як час завантаження сторінки, інтерактивність та візуальна стабільність. Web Vitals допомагає розробникам виявляти проблеми продуктивності та оптимізувати веб-додатки для поліпшення користувацького досвіду.

Технології на стороні сервера:

Express.js: Мінімалістичний веб-фреймворк для Node.js, що надає засоби для обробки HTTP-запитів, маршрутизації, інтеграції middleware та рендерингу шаблонів. Express.js спрощує розробку логіки на стороні сервера та створення RESTful API, роблячи його популярним вибором для побудови веб-серверів та API. [9]

Sequelize: ORM-бібліотека для Node.js, яка використовується для взаємодії з реляційними базами даних за допомогою JavaScript. Sequelize надає потужний набір функцій для визначення моделей, виконання запитів та керування транзакціями бази даних, покращуючи продуктивність та підтримку утримання взаємодії з базою даних. [10]

PostgreSQL: Могутня відкрита система управління реляційними базами даних (RDBMS), відома своєю надійністю, масштабованістю та рядом розширених функцій. PostgreSQL пропонує підтримку ACID-транзакцій, типи даних JSON та розширений індексування, що робить його підходящим для даних, інтенсивних за обсягом, застосунків, таких як інтернет-магазини. [11]

JWT (JSON Web Tokens): Стандарт для безпечної передачі інформації між сторонами у вигляді JSON-об'єктів, часто використовується для реалізації механізмів аутентифікації та авторизації у веб-додатках. JWT є компактними, URL-безпечними токенами, які можна легко генерувати, перевіряти та декодувати, що дозволяє стане-менше аутентифікацію у розподілених системах. [12]

bcrypt: Криптографічна хеш-функція, яка часто використовується для безпечного зберігання паролів у базах даних. bcrypt використовує адаптивні хеш-функції та методи соління, щоб захистити паролі від атак методом грубої сили та атак методом таблиць радуги, забезпечуючи безпеку користувальницьких облікових записів у веб-додатках. [13]

CORS (Cross-Origin Resource Sharing): Механізм, який дозволяє веб-серверам вказувати, які походження мають доступ до ресурсів на сервері. CORS є важливим для забезпечення безпеки веб-додатків та запобігання несанкціонованому доступу до чутливих даних шляхом обмеження запитів між різними доменами.

Express Fileupload: Проміжний програмний засіб для обробки завантаження файлів у додатках на основі Express.js, який надає засоби для розбору багатокомпонентних запитів та зберігання завантажених файлів на сервері. Express Fileupload спрощує процес обробки завантаження файлів та інтегрується з веб-сервером Express.js.

Dotenv: Модуль для завантаження змінних середовища з файлу .env у об'єкт process.env у додатках Node.js. Dotenv дозволяє розробникам налаштовувати параметри додатка, такі як облікові дані бази даних та ключі API, без жорсткого кодування їх у вихідний код, покращуючи безпеку та переносимість. [14]

UUID (Universally Unique Identifier): Стандарт для генерації унікальних ідентифікаторів, які є унікальними на різних системах та періодах часу. UUID часто використовуються як первинні ключі в базах даних та для унікальної ідентифікації ресурсів у розподілених системах, забезпечуючи цілісність даних та уникнення конфліктів у конкуруючих транзакціях.

Використання UUID (Унікальний Ідентифікатор Унікального) в веб-розробці має кілька переваг. Унікальність UUID гарантує, що ідентифікатори, створені за його допомогою, будуть унікальними навіть при створенні на різних серверах, в різних частинах світу, або в різні моменти часу. Це дозволяє уникнути конфліктів ідентифікаторів при створенні, наприклад, об'єктів бази даних чи сутностей в програмі.

Використання UUID є особливо важливим при роботі з базами даних, оскільки вони дозволяють генерувати ідентифікатори, які не будуть конфліктувати з уже існуючими записами. Це дозволяє ефективно розподіляти завдання між різними серверами або послугами, не переймаючись про можливі конфлікти ідентифікаторів.

Використання UUID також робить його дуже корисним для створення унікальних URL-адрес, токенів сесії, ідентифікаторів та ключів сесій, а також для створення унікальних ідентифікаторів користувачів. [15]

# 2.**5** Опис структури програми та алгоритмів її функціонування**.**

Логічна структура програми:

Express.js та маршрутизатори:

Express.js можливо без зусиль використовувати для створення внутрішньої програми та керування запитами HTTP. Маршрутизатори вказують, як програма має поводитися щодо певних URL-адрес і методів HTTP. Ілюстрація такого маршрутизатора призначена для отримання списку користувачів; він може мати URL як /users, а потім використовувати метод GET. Контролери діють як точки входу та несуть відповідальність за отримання даних із HTTP-запитів, виконання процесів баз даних і створення відповідей користувачам. В одному сценарії контролер, який відповідає за отримання відомостей про користувача, надсилає запит до бази даних, отримує інформацію про кожного користувача в списку, а потім надає її клієнту у своїй відповіді

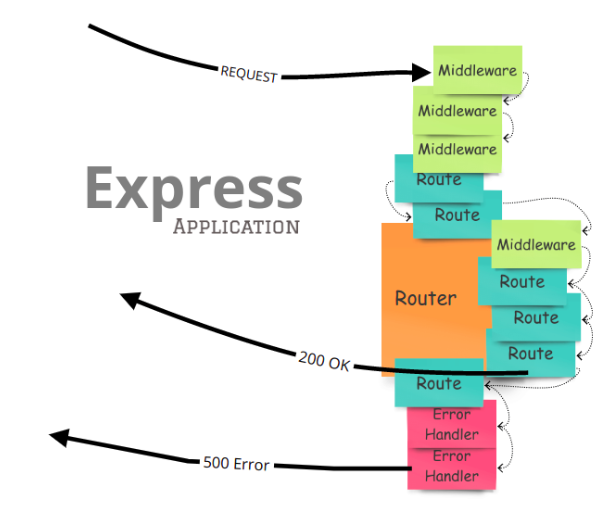


Рисунок 2.1 – Маршрутизатори у Express.js

Моделі бази даних Sequelize:

Моделі даних визначають організацію та зв’язки різних об’єктів у базі даних. Наприклад, особа може бути представлена ​​моделлю користувача, яка містить такі поля, як ім’я, адреса електронної пошти та пароль. Sequelize ORM забезпечує зручний спосіб роботи з базою даних за допомогою виразів JavaScript.

Переваги використання Sequelize ORM:

Простота використання: Sequelize ORM дозволяє вам використовувати мову JavaScript для взаємодії з базою даних, що робить процес розробки більш зручним і прозорим. Підтримка різних СУБД: Sequelize підтримує не тільки SQL бази даних, такі як PostgreSQL, MySQL, SQLite, MSSQL, але й NoSQL бази даних, що дозволяє вибирати найбільш підходящу базу даних для вашого проекту. Міграції: Sequelize забезпечує механізм міграцій, який спрощує управління структурою бази даних, дозволяючи легко оновлювати схему бази даних на різних етапах розробки. Асоціації: Sequelize надає зручний спосіб визначення асоціацій між моделями, такими як один-до-одного, один-до-багатьох та багато-до-багатьох. Це спрощує взаємодію з даними і дозволяє легко виконувати складні запити. Підтримка обіцянок (Promises) та async/await: Sequelize підтримує обіцянки та async/await, що дозволяє писати більш чистий та зрозумілий код без глибокого вкладення колбеків. Валідація: Sequelize надає можливість виконання валідації даних перед їх збереженням у базі даних, що допомагає запобігти некоректним даним. Підтримка транзакцій: Sequelize дозволяє виконувати операції з базою даних у межах транзакцій, що робить роботу з даними надійнішою.[16]

Контролери:

Логіка обробки запитів вбудована в контролери, які отримують дані з HTTP-запитів, виконують обов’язкові операції з базою даних, а потім обслуговують клієнтів із відповіддю. Контролер, який отримує список користувачів із бази даних, — це той, який має можливість надсилати запит до бази даних, отримувати відповідь, яка містить список користувачів, і надсилати його назад клієнту.

Алгоритм та функціонування програми:

Підключення до бази даних та створення моделей:

Алгоритм починається з підключення до бази даних за допомогою Sequelize і пошуку необхідних моделей. Моделі розроблено на основі схеми бази даних, а також зв’язків між таблицями.

Обробка HTTP-запитів:

HTTP-запит, і сервер отримає маршрут від URL-адреси до методу. Після цього викликається правильний контролер, який отримує дані з HTTP-запиту та передає їх до бази дани

Взаємодія з базою даних:

Операції з базою даних, такі як створення, читання, оновлення та видалення даних, є відповідальністю контролерів. Відповідно до викликів методів моделювання Sequelize також генеруватиме SQL-запити.

Опис структури бази даних програми за її наявності

Таблиця "User":

Ця таблиця зберігає інформацію про користувачів системи. Кожен користувач має унікальний ідентифікатор (id), електронну адресу (email), пароль (password), особисті дані, такі як ім'я (name), прізвище (surname), по батькові (patronymic), дату народження (date), номер телефону (phoneNumber), адресу (address) та роль (role) у системі.

Таблиця "Basket":

Ця таблиця представляє кошик користувача, в якому зберігаються товари. Кожен кошик має свій унікальний ідентифікатор (id).

Таблиця "BasketDevice":

У цій таблиці зберігаються товари, які знаходяться у кошику користувача. Кожен запис в таблиці містить ідентифікатор товару (id).

Таблиця "Order":

Ця таблиця містить інформацію про замовлення користувачів. Кожен запис містить унікальний ідентифікатор (id), дані замовлення, такі як електронна адреса (orderEmail), номер телефону (phoneNumber), спосіб доставки (delivery), ім'я (name), прізвище (surname), спосіб оплати (payment) та вартість (cost) замовлення.

Таблиця "OrderDevices":

Ця таблиця відображає товари, що містяться в кожному замовленні. Кожен запис містить ідентифікатор товару (id), його назву (name), категорію (category) та кількість (count) одиниць у замовленні.

Таблиця "Device":

У цій таблиці зберігається інформація про товари, які доступні для продажу. Кожен товар має унікальний ідентифікатор (id), назву (name), ціну (price), рейтинг (rating), зображення (img, img2, img3, img4), знижку (discount), категорію (category) та актуальність (relevance).

Таблиці "Type" та "Brand":

Ці таблиці містять дані про типи та бренди товарів відповідно. Кожен запис містить унікальний ідентифікатор (id) та назву (name).

Таблиця "Rating":

У цій таблиці зберігаються рейтинги користувачів для кожного товару. Кожен запис містить унікальний ідентифікатор (id) та оцінку (rate).

Таблиця "DeviceInfo":

Ця таблиця містить додаткову інформацію про кожен товар. Кожен запис містить унікальний ідентифікатор (id), заголовок (title) та опис (description).

Таблиця "TypeBrand":

Ця проміжна таблиця відображає зв'язок між типами та брендами товарів.

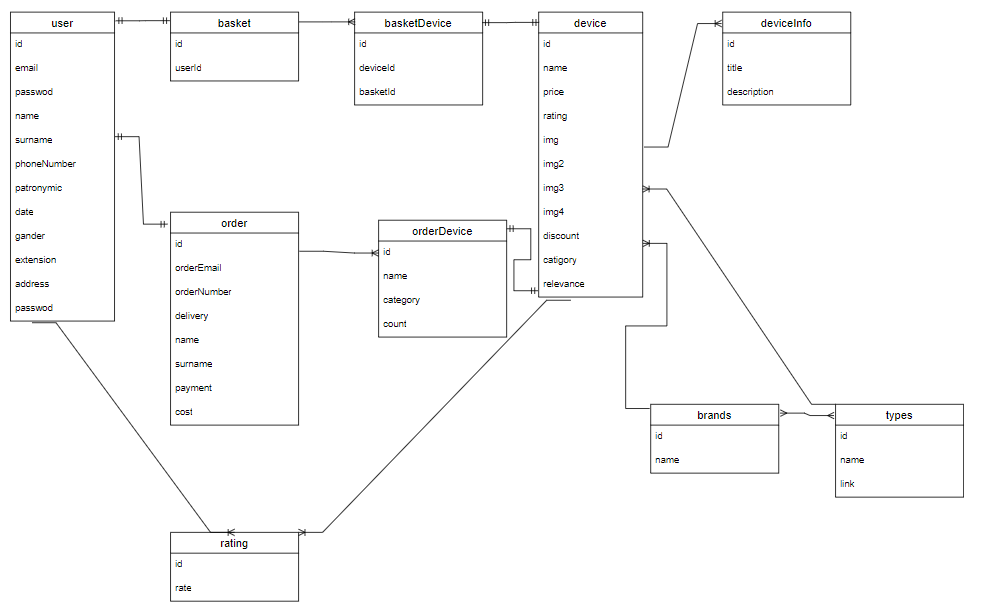


Рисунок 2.3 – Діаграма бази данних Dom1store

Основна структура бази даних програми містить наступні таблиці:

User (Користувач):

id: Цілочисельний ідентифікатор (первинний ключ, автоматично збільшується).

email: Строка, унікальний електронний адрес користувача.

password: Строка, пароль користувача.

name: Строка, ім'я користувача.

surname: Строка, прізвище користувача.

phoneNumber: Строка, номер телефону користувача.

patronymic: Строка, по батькові користувача.

date: Строка, дата народження користувача.

gender: Строка, стать користувача.

extension: Строка, розширення (можливо, номер картки або щось подібне).

address: Строка, адреса користувача.

role: Строка, роль користувача в системі (за замовчуванням "USER").

Basket (Кошик):

id: Цілочисельний ідентифікатор (первинний ключ, автоматично збільшується).

BasketDevice (Продукт у кошику):

id: Цілочисельний ідентифікатор (первинний ключ, автоматично збільшується).

Order (Замовлення):

id: Цілочисельний ідентифікатор (первинний ключ, автоматично збільшується).

orderEmail: Строка, електронна адреса для замовлення.

phoneNumber: Строка, номер телефону для замовлення.

delivery: Строка, спосіб доставки замовлення.

name: Строка, ім'я клієнта, що замовляє.

surname: Строка, прізвище клієнта, що замовляє.

payment: Строка, спосіб оплати замовлення.

cost: Ціле число, вартість замовлення.

OrderDevices (Продукт у замовленні):

id: Цілочисельний ідентифікатор (первинний ключ, автоматично збільшується).

name: Строка, унікальна назва продукту.

category: Строка, категорія продукту.

count: Строка, кількість одиниць продукту у замовленні.

Device (Пристрій):

id: Цілочисельний ідентифікатор (первинний ключ, автоматично збільшується).

name: Строка, унікальна назва пристрою.

price: Строка, ціна пристрою.

rating: Ціле число, рейтинг пристрою.

img: Строка, шлях до зображення пристрою.

img2, img3, img4: Строки, додаткові зображення пристрою.

discount: Ціле число, знижка на пристрій.

category: Строка, категорія пристрою.

relevance: Строка, актуальність пристрою.

Type (Тип):

id: Цілочисельний ідентифікатор (первинний ключ, автоматично збільшується).

name: Строка, унікальна назва типу.

link: Строка, унікальне посилання на тип.

Brand (Бренд):

id: Цілочисельний ідентифікатор (первинний ключ, автоматично збільшується).

name: Строка, унікальна назва бренду.

Rating (Рейтинг):

id: Цілочисельний ідентифікатор (первинний ключ, автоматично збільшується).

rate: Ціле число, рейтинг.

DeviceInfo (Інформація про пристрій):

id: Цілочисельний ідентифікатор (первинний ключ, автоматично збільшується).

title: Строка, заголовок інформації.

description: Строка, опис інформації.

# 2.**6** Обґрунтування та організація вхідних та вихідних даних програми

Характер, організація і попередня підготовка вхідних даних:

Користувачські дані: Вхідні дані у формі замовлень, реєстрації користувачів, оновлення профілю що надходять через фронтенд за допомогою Formik, які збирають інформацію та відправляють її на сервер за допомогою HTTP запитів з використанням Axios. Попередня валідація здійснюется за допомогою Yup.

Дані каталогу товарів: Вони надходять з бази даних через API, що забезпечується сервером за допомогою Sequelize та Express. Ці дані вже структуровані та готові для відображення на фронтенді.

Авторизація та аутентифікація: Вхідні дані для авторизації користувача введені через форму, або передані у вигляді токенів, що декодуються з допомогою бібліотеки jwt-decode.

Характер і організація вихідних даних:

Відповіді на запити: Сервер надсилае вихідні дані у форматі JSON через API за допомогою Express. Ці дані містять інформацію про користувача (якщо він авторизований), товари, категорії, тощо.

Сповіщення та підтвердження: У випадку успішної реєстрації, оформлення замовлення або інших подій, сервер надсилає підтвердження через формат JSON або відображати повідомлення на фронтенді.

Формат, опис і спосіб кодування вхідних та вихідних даних:

Формат даних: Для вхідних даних форматом є JSON або форма введення на стороні клієнта, яка передає дані у форматі JSON через HTTP запити. Для вихідних даних також використовується JSON для передачі даних від сервера до клієнта.

Опис даних: Дані описані у вигляді схеми (наприклад, використовуючи Sequelize моделі для бази даних), або типів даних).

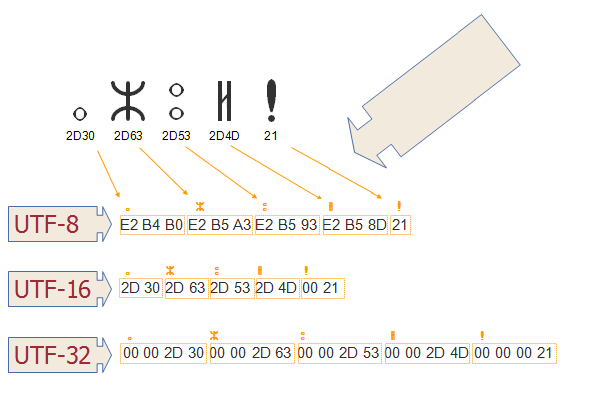
Спосіб кодування: Для передачі даних через мережу використовувется кодування у форматі UTF-8 для забезпечення сумісності та надійності передачі. ****

Рисунок 2.3 – Перелік utf кодувань

# 2.**7** Опис розробленого програмного продукту

# 2.**7.1** Використані технічні засоби

Під час розробки мого інтернет-магазину я активно використовуємо передові технології та інструменти.

Для контролю версій і управління кодом я вибрав Git. Це дає змогу ефективно відстежувати зміни та забезпечувати цілісність кодової бази.

Як основну мову програмування для серверної розробки використав JavaScript, а також Node.js, який забезпечує високу продуктивність і масштабованість нашого серверного коду, що дозволяє нам ефективно обробляти високі навантаження в магазині.

Для створення інтерфейсу клієнтської частини магазину я обрав фреймворк React, який забезпечує високу продуктивність і зручність в оновленні користувацького інтерфейсу , а також надає безліч готових компонентів і бібліотек для швидкого розроблення користувацького інтерфейсу, що спрощує нашу роботу.

Як базу даних PostgreSQL, надійну і потужну реляційну базу даних, що забезпечує надійне зберігання і обробку даних ,та й забезпечує дотримання ACID-властивостей, забезпечуючи нам цілісність і надійність даних у магазині.

Для проектування макетів користувацького інтерфейсу і створення прототипів був використаний інструмент Figma. Він допомагає створювати привабливі та функціональні макети для поліпшення користувацького досвіду, а ще дає нам змогу створювати спільні макети й отримувати зворотний зв'язок від команди для поліпшення дизайну інтерфейсу.

Також я активно використовував розгалуження і злиття в Git для ефективної співпраці та управління змінами в коді.

# 2.**7.2** Використані програмні засоби

Операційна система, на якій базується наш проект, - Windows. Операційна система Windows, на якій базується наш проект, має кілька ключових переваг, які сприяють успішному функціонуванню нашої системи. Широкий розповсюджений: Windows - одна з найбільш поширених операційних систем у світі. Це означає, що більшість користувачів вже знайомі з цією платформою, що полегшує прийняття та використання нашої системи. Різноманітність програмного забезпечення: Windows має великий вибір програмного забезпечення, що підтримується на цій платформі. Це означає, що ми можемо використовувати широкий спектр інструментів і програм для розробки, тестування та підтримки нашого проекту. Користувацький інтерфейс: Windows відомий своїм інтуїтивно зрозумілим та легким у використанні користувацьким інтерфейсом. Це робить нашу систему більш доступною та зручною для користувачів, що сприяє популярності та прийняттю серед аудиторії. Сумісність з апаратним забезпеченням: Windows підтримує широкий спектр апаратного забезпечення, що дозволяє нам працювати на різних пристроях та конфігураціях без значних обмежень. Підтримка: Microsoft забезпечує активну підтримку та регулярні оновлення для своєї операційної системи, що дозволяє нам тримати нашу систему в актуальному та безпечному стані. Загалом, операційна система Windows надає нам необхідні інструменти та середовище для успішної розробки, впровадження та підтримки нашого проекту, забезпечуючи надійність, доступність та зручність для користувачів. [17]

Для написання коду та розвитку функціоналу нашого проекту використовується різноманітні програмні інструменти, серед яких Visual Studio Code. Цей текстовий редактор є одним з найпопулярніших та потужних текстових редакторів для розробки програмного забезпечення. Він має кілька ключових переваг, які роблять його ідеальним вибором для написання коду та розвитку функціоналу нашого проекту: Легка вага та швидкодія: Visual Studio Code - це легкий текстовий редактор, який запускається швидко і має мінімальний вплив на продуктивність системи. Він не навантажує систему зайвими ресурсами, що дозволяє швидко приступити до роботи з проектом. Розширюваність: VS Code має широкий вибір розширень, які дозволяють розширити його функціональність згідно з потребами проекту. Завдяки цьому, ми можемо налаштувати середовище розробки так, щоб воно відповідало нашим потребам та стандартам. Інтеграція з Git: VS Code має вбудовану підтримку для системи контролю версій Git. Це дозволяє нам легко виконувати коміти, переглядати відмінені файли та керувати гілками прямо з редактора, що полегшує колаборацію та керування кодовою базою проекту. Підтримка різних мов програмування: Visual Studio Code підтримує багато мов програмування та фреймворків, включаючи JavaScript, Python, Java, C#, і багато інших. Це робить його універсальним інструментом для розробки незалежно від мови програмування, яку ми використовуємо у нашому проекті. Активна спільнота та підтримка: VS Code користується великою спільнотою розробників та має активну підтримку від Microsoft. Це означає, що ми можемо легко знайти відповіді на свої питання, отримати поради та рекомендації щодо роботи з редактором. [18]

Для тестування API та взаємодії з сервером використовується інструмент Postman. Postman - це інструмент для тестування API, який дозволяє розробникам створювати, тестувати та відлагоджувати HTTP запити та відповіді. Ось декілька ключових причин, чому я використовував Postman: Створення запитів: Postman надає зручний інтерфейс для створення різних типів HTTP запитів, таких як GET, POST, PUT, DELETE тощо. Це дозволяє швидко та легко перевіряти різні ендпоінти API. Налаштування параметрів: Інструмент дозволяє налаштовувати різні параметри запиту, такі як заголовки, тіло запиту, параметри запиту та багато іншого. Це дозволяє виконувати різні сценарії тестування та перевіряти різні варіанти взаємодії з API. Організація колекцій: Postman дозволяє організувати набір запитів у колекції, що спрощує управління та навігацію між ними. Це особливо корисно, коли потрібно тестувати різні частини або функціональність API.

Тестування автоматизованих сценаріїв: Postman дозволяє створювати автоматизовані тести для перевірки відповідей API на певні запити. Це дозволяє автоматизувати процес тестування та забезпечити стабільність API. Підтримка середовищ: Інструмент має підтримку робочих середовищ, що дозволяє легко переключатися між різними наборами налаштувань для різних середовищ (наприклад, розробка, тестування, продакшн). [19]

Управління версіями коду здійснюється за допомогою системи управління версіями Nginx Nginx — це високоефективний веб-сервер, що зазвичай використовується як проксі-сервер, а також для збалансування навантаження, обробки запитів HTTPS, обслуговування веб-сайтів з високим навантаженням, а також для доставки контенту через протоколи HTTP, HTTPS, SMTP, POP3 і IMAP.

Основні переваги Nginx:

Висока продуктивність: Nginx розроблений з фокусом на високу продуктивність і оптимізований для великої кількості одночасних з'єднань. Його архітектура створена таким чином, що він може ефективно обробляти тисячі одночасних запитів. Низьке споживання ресурсів: Використання пам'яті і центрального процесора у Nginx дуже ефективне, що робить його ідеальним вибором для серверів з обмеженими ресурсами. Висока масштабованість: Nginx легко масштабується горизонтально за допомогою різних методів, включаючи вирішення проблеми навантаження на сервери, балансування навантаження і кешування. Проста конфігурація: Конфігурація Nginx здійснюється за допомогою текстових файлів, що робить процес конфігурування веб-сервера досить простим і зрозумілим. Проксі-сервер і балансувальник навантаження: Nginx часто використовується як проксі-сервер, який може передавати запити на різні служби, віддалені сервери або додатки. Він також здатний балансувати навантаження між різними серверами, що дозволяє підвищити надійність та ефективність роботи інфраструктури. [20]

Загальна використання цих програмних засобів допомагає нам забезпечити ефективну та надійну роботу нашої системи, забезпечуючи високу якість та зручність для користувачів.

# 2.**7.3** Виклик та завантаження програми

Ось як викликати та завантажити програму інтернет-магазину Dom1store

Щоб розпочати роботу з моїм інтернет-магазином, спочатку потрібно клонувати репозиторій з відповідного носія даних. Для цього використовуйте команду: git clone [URL репозиторію]

Переконайтеся, що встановлено Git на вашому пристрої перед виконанням цієї команди.

Після завершення клонування перейдіть у каталог, де ви хочете мати копію репозиторію. Це можна зробити, відкривши термінал та використовуючи команду: cd [назва папки проекту] Тепер, для завантаження необхідних залежностей для React, використовуйте команду: npm install

Після завершення встановлення залежностей перейдіть до папки "client" за допомогою команди: cd client .Використайте команду "npm install" ще раз, щоб встановити залежності для клієнтської частини проекту. npm install Після завершення цього кроку перейдіть до кореневої папки проекту за допомогою команди: cd ..

Далі, перейдіть до папки "server" за допомогою команди: cd server Використовуйте команду "npm install" для встановлення залежностей для серверної частини проекту. Перед цим переконайтеся, що Node.js встановлено на вашому пристрої. npm install Після завершення встановлення залежностей, перейдіть до кореневої папки проекту . Тепер, щоб запустити проект, переконайтеся, що ви знаходитесь в кореневій папці (server). Запустіть сервер, використовуючи команду: npm run dev

Після успішного запуску сервера відкрийте нове вікно терміналу. Перейдіть до папки "client" за допомогою команди: cd client Запустіть клієнт, використовуючи команду: npm start Якщо все пройшло успішно, програма буде доступна за адресою "<http://localhost:3000/>".

Перевірте консоль, щоб переконатися, що немає помилок. Якщо у вас виникли проблеми з запуском програми, переконайтеся, що всі залежності встановлені правильно. Додайте необхідні файли конфігурації, якщо це потрібно, і переконайтеся, що всі шляхи налаштовані на вашому пристрої. Якщо у вас є додаткові налаштування, переконайтеся, що вони збережені і вірно відображені. Переконайтеся, що у вас достатньо оперативної пам'яті для запуску програми та зверніть увагу на обсяг програми, переконайтеся, що ваш пристрій відповідає вимогам програми. Переконайтеся, що назва папки проекту не містить пробілів або спеціальних символів. Переконайтеся, що URL репозиторію вказаний правильно при клонуванні проекту. Перевірте, чи не містить ваша програма жодних застарілих частин коду. Забезпечте, щоб всі зміни були збережені та залиті в репозиторій перед завантаженням програми. Перевірте логи, щоб впевнитися, що програма працює коректно після запуску.

# 2.**7.4** Виклик та завантаження програми

Головна сторінка:

Опис: Головна сторінка інтернет-магазину містить важливу інформацію про акції, нові товари, а також найпопулярніші товари. На головній сторінці розміщений слайдер новинок, що дозволяє користувачам ознайомитися з останніми доданими товарами. Також присутній свайпер популярних товарів, який дозволяє швидко ознайомитися з найбільш відомими та популярними продуктами. Клікабельні посилання на головні категорії товарів, акції та рекламні банери дозволяють користувачам легко здійснити перехід до відповідних сторінок.

Управління:

Слайдер новинок: Клікабельні об'єкти, які дозволяють переглядати нові товари.

Свайпер популярних товарів: Клікабельні об'єкти, що дозволяють швидко ознайомитися з популярними товарами.

Головні категорії товарів: Клікабельні посилання для переходу до відповідних категорій товарів.

Акції: Клікабельні об'єкти для перегляду акцій та спеціальних пропозицій.

Рекламні банери: Клікабельні посилання, які привертають увагу користувача та надають доступ до різних акцій та пропозицій.

Ілюстрація:

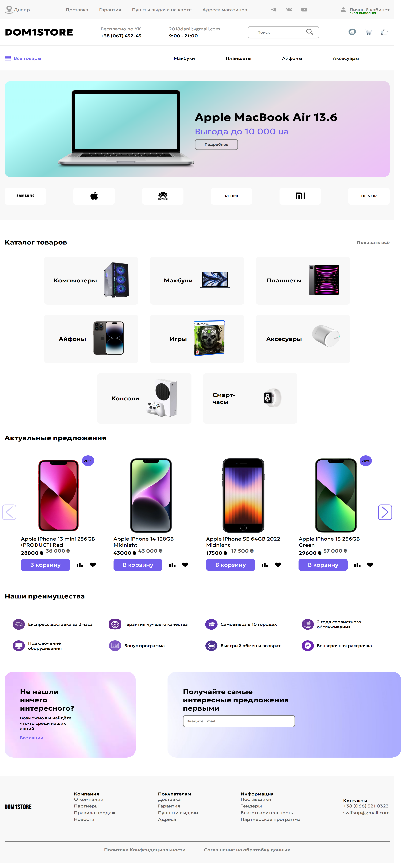


Рисунок 2.4 – Cкриншот головної сторінки Dom1store

Категорії товарів:

Опис: Список категорій товарів, що дозволяє користувачам швидко перейти до потрібної категорії товарів.

Управління:

Категорії товарів: Клікабельні посилання для перегляду підкатегорій та товарів в цій категорії.

Підкатегорії: При кліці на кожну категорію відображаються підкатегорії.

Товари: При кліці на кожну підкатегорію відображаються товари в цій категорії.

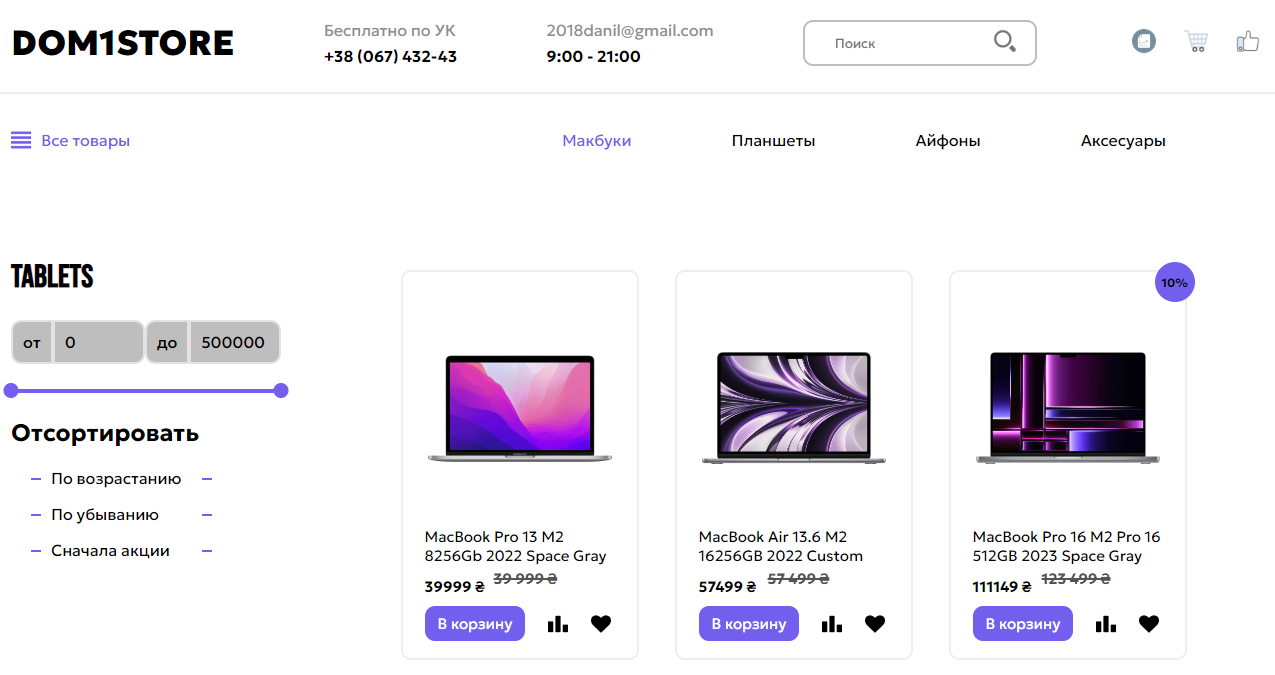


Рисунок 2.5 – Cкриншот списку категорій Dom1store

Пошук:

Опис: Можливість здійснення пошуку товарів за назвою, категорією тощо.

Управління:

Введення тексту для пошуку та вибір параметрів пошуку.

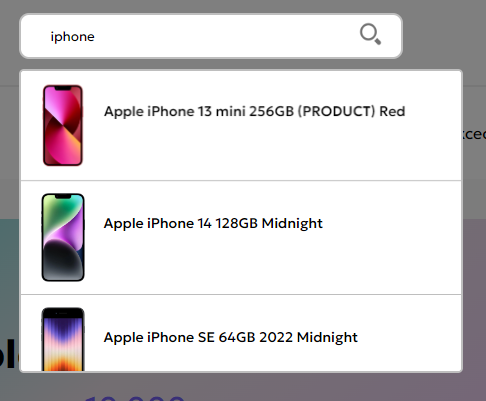


Рисунок 2.6 – Cкриншот управління пошуком Dom1store

Кошик:

Опис: Перегляд товарів, які додані до кошика, обчислення загальної вартості та оформлення замовлення.

Управління:

Додавання товарів: Кнопка "Додати до кошика" для додавання товару в кошик.

Видалення товарів: Можливість видалення товарів з кошика.

Зміна кількості товарів: Можливість змінити кількість одиниць товару для замовлення.

Ілюстрація:

Рисунок 2.4 – Cкриншот головної сторінки Dom1store

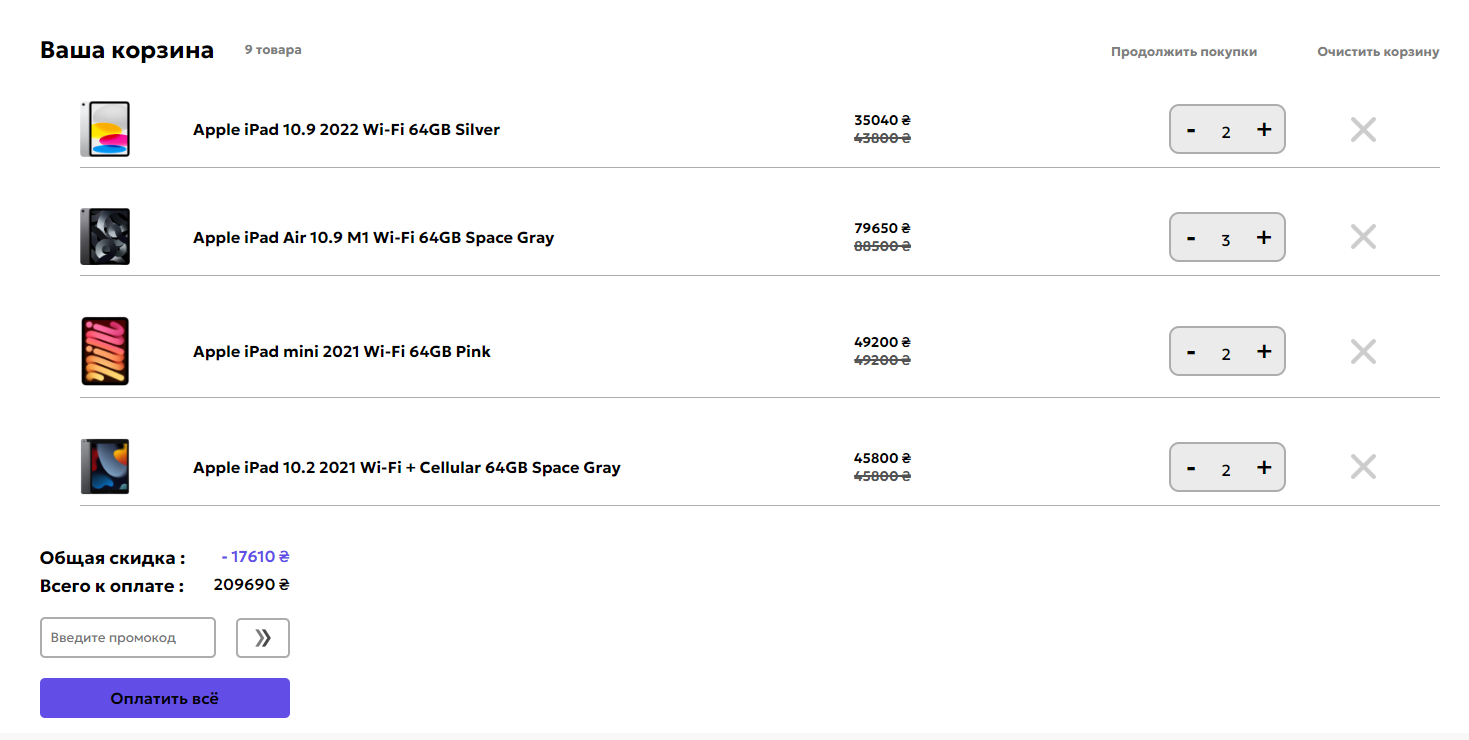


Рисунок 2.7 – Cкриншот кошик Dom1store

Особистий кабінет:

Опис: Сторінка для авторизованих користувачів, де вони можуть переглядати свої замовлення, редагувати профіль тощо.

Управління:

Перегляд особистих даних: Користувач може переглядати інформацію, яка стосується його особистих даних, таку як ім'я, адреса електронної пошти, контактний номер телефону тощо.

Редагування профілю: Користувач може змінювати свої особисті дані, такі як ім'я, адреса електронної пошти, контактний номер телефону тощо.

Зміна адреси доставки: Користувач може змінювати адресу доставки для майбутніх замовлень.

Перегляд історії замовлень: Користувач може переглядати історію своїх замовлень, включаючи статус доставки та іншу важливу інформацію.

Перегляд доставлених товарів: Користувач може переглядати і підтверджувати доставку отриманих товарів.

Ілюстрація:

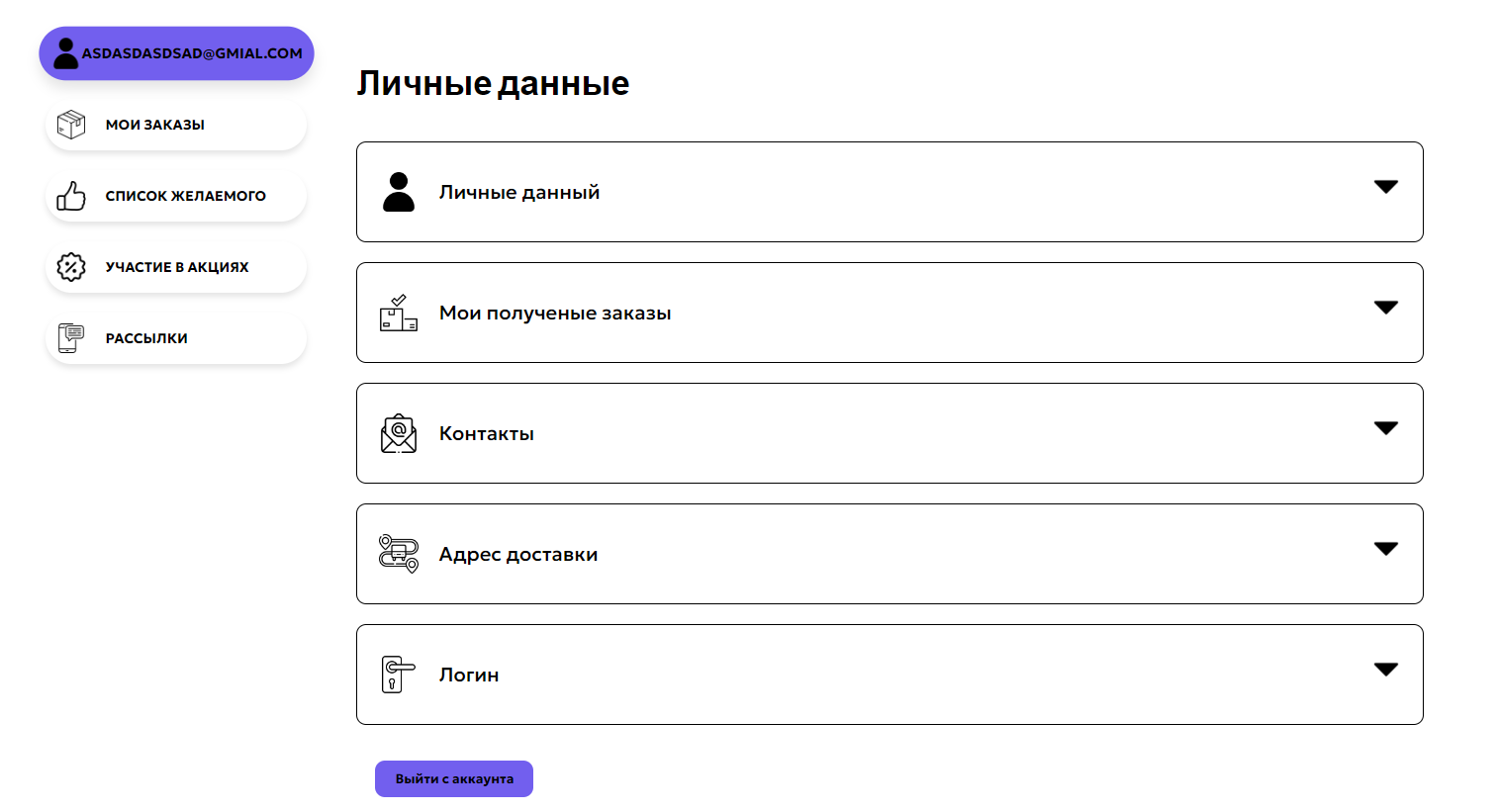


Рисунок 2.7 – Cкриншот особистого кабінету Dom1store

# 3Економічний розділ

**Висновки**

Розроблене програмне забезпечення є ефективним і працездатним інструментом для автоматизації діяльності інтернет-магазину електронної техніки. Воно дозволяє здійснювати автоматизований збір, обробку і маніпулювання даними, спрощуючи процеси продажу, обліку та контролю за товарами в мережі магазинів.

Програмне забезпечення надає можливість виконувати різноманітні операції, такі як додавання, видалення і редагування інформації про товари, клієнтів, постачальників, продажі, покупців, а також перегляд інформації про них.

Актуальність розробленого програмного забезпечення обумовлена широким попитом на такі програмні продукти, що надають можливість електронного зберігання даних про товари, продажі, надходження, ведення бази даних, підбиття підсумків з продажу, отримання звітів і формування супутньої ділової документації.

Розроблене програмне забезпечення інформаційної системи призначене для застосування в будь-якій мережі інтернет-магазинів з подібним родом діяльності і схожими функціональними вимогами.

Створений додаток дозволить оптимізувати та спростити дії по веденню бази даних мережі магазинів; скоротити час на оформлення продажу; підвищити ефективність діяльності компанії шляхом електронного ведення документації з продажу і можливістю аналізу наявних даних і надання необхідних звітів і супутньої ділової документації.

Структура програми складається з клієнтського додатку, написаного на мові програмування високого рівня JavaScript (з використанням React та Node.js), що взаємодіє з базою даних, розробленою мовою SQL в системі управління базами даних.

В "Економічному розділі"

**СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ**

1. Що таке JavaScript URL: https://cases.media/en/article/sho-take-javascript (дата звернення: 11.04.2024).
2. Що таке NodeJs URL: https://devzone.org.ua/post/shcho-take-nodejs-osnovy-servernoyi-rozrobky-na-javascript (дата звернення: 11.04.2024).
3. Початок роботи з React URL: https://developer.mozilla.org/ru/docs/Learn/Tools\_and\_testing/Client-side\_JavaScript\_frameworks/React\_getting\_started (дата звернення: 11.04.2024).
4. Redux Керівництво

URL: https://highload.today/redux-react/ (дата звернення: 11.04.2024).

1. HTTP клієнт для браузера і node.js на основі Promise URL: https://axios-http.com/ru/docs/intro (дата звернення: 11.04.2024).
2. React Formik + styled-components URL: https://medium.com/flyparakeet/react-formik-styled-components-add78b37971f (дата звернення: 11.04.2024).
3. JS: Предметно-орієнтований дизайн URL: https://ru.hexlet.io/courses/js-ddd/lessons/validation/theory\_unit (дата звернення: 11.04.2024).
4. Просте масштабоване управління станом. URL: https://www.jscamp.app/ru/docs/state-management/reactnativestate00 (дата звернення: 11.04.2024).
5. Веб-фреймворк Express (Node.js/JavaScript)

URL: https://www.jscamp.app/ru/docs/state-management/reactnativestate00

(дата звернення: 11.04.2024).

1. Як Sequelize допомагає з базами даних в Node.js URL: https://foxminded.ua/ru/sequelize-chto-eto/ (дата звернення: 11.04.2024).
2. PostgreSQL що це таке і чому це важливо?URL: https://foxminded.ua/ru/postgresql-chto-eto/#:~:text=Это%20свободная%20и%20открытая%20система,СУБД)%20с%20открытым%20исходным%20кодом. 0 (дата звернення: 11.04.2024).
3. Структура JWT (JSON Web Token) . URL: https://www.linkedin.com/pulse/10-питань-про-jwt-які-вас-можуть-спитати-сергей-s--ihxee (дата звернення: 11.04.2024).
4. Bcrypt – довідка. URL: https://reporter.zp.ua/bcrypt-dovidka.html (дата звернення: 11.04.2024).
5. Використання змінних середовища в Node.js URL: https://habr.com/ru/companies/ruvds/articles/351254/ (дата звернення: 11.04.2024).
6. Глибоке занурення в унікальні ідентифікатори . URL: https://www.tempmail.us.com/uk/javascript/Створення-унікальних-ідентифікаторів-у-javascript-посібник-із-uuid-та-guid (дата звернення: 11.04.2024).
7. Введение в Sequelize ORM для Node.js . URL: https://coursehunter.net/course/vvedenie-v-sequelize-orm-dlya-node-js (дата звернення: 11.04.2024).
8. ЯКУ ОС КРАЩЕ ОБРАТИ? URL: https://www.nspace.ua/info/windows-linux-ta-macos-chim-vidriznyayutsya-ta-yaku-os-obrati (дата звернення: 11.04.2024).
9. Використання Visual Studio Code URL: https://learn.microsoft.com/uk-ua/power-pages/configure/vs-code-extension (дата звернення: 11.04.2024).
10. Postman - must have для QA URL: https://qagroup.com.ua/publications/znajomtes-postman-must-have-dlia-qa/ (дата звернення: 11.04.2024).
11. Nginx — вебсерверURL: https://brander.ua/technologies/nginx (дата звернення: 11.04.2024).