Вона допомагає організаціям ефективніше управляти інформацією,
покращувати продуктивність та приймати обґрунтовані рішення

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ЗАПОРІЗЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

МАТЕМАТИЧНИЙ ФАКУЛЬТЕТ
Кафедра програмної інженерії

КВАЛІФІКАЦІЙНА РОБОТА БАКАЛАВРА

на тему: **«РОЗРОБКА ІНФОРМАЦІЙНОЇ СИСТЕМИ ЗАСОБАМИ DJANGO ТА REACT»**

Виконав: студент 4 курсу, групи 6.1219-2пі
спеціальності 121 інженерія програмного забезпечення
(шифр і назва спеціальності)
освітньої програми програмна інженерія
(назва освітньої програми)
А.М. Ахінько
(ініціали та прізвище)

Керівник доцент кафедри програмної інженерії, Phd Столярова А.В.
(посада, вчене звання, науковий ступінь, прізвище та ініціали)

Рецензент завідувач кафедри фундаментальної та прикладної математики, професор, д.т.н. Гребенюк С.М.
(посада, вчене звання, науковий ступінь, прізвище та ініціали)

Запоріжжя – 2023

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ЗАПОРІЗЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

Факультет математичний

Кафедра програмної інженерії

Рівень вищої освіти бакалавр

Спеціальність 121 інженерія програмного забезпечення
(шифр і назва)

Освітня програма програмна інженерія

ЗАТВЕРДЖУЮ
Завідувач кафедри програмної інженерії, к.ф.-м.н., доцент

_____ Лісняк А.О.
(підпис)

“ 07 ” 02 2023 р.

З А В Д А Н Н Я

НА КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТОВІ

Ахіньку Андрію Максимовичу
(прізвище, ім’я та по-батькові)

1. Тема роботи Розробка інформаційної системи засобами Django та React

керівник роботи Столярова Анастасія Валеріївна, Phd
(прізвище, ім’я та по-батькові, науковий ступінь, вчене звання)

затверджені наказом ЗНУ від « 26 » січня 2023 року № 102-с

2. Строк подання студентом роботи 07.06.2023 р.

3. Вихідні дані до роботи 1. Постановка задачі.
2. Перелік літератури.

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити)
1. Постановка задачі.
2. Основні теоретичні відомості.
3. Розробка інформаційної системи засобами Django та React.

5. Перелік графічного матеріалу (з точним зазначенням обов’язкових креслень)
презентація за темою докладу

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
1	Кудін О.В., доцент кафедри програмної інженерії		
2	Кудін О.В., доцент кафедри програмної інженерії		
3	Кудін О.В., доцент кафедри програмної інженерії		

7. Дата видачі завдання 07.02.2023 р.

К А Л Е Н Д А Р Н И Й П Л А Н

№	Назва етапів кваліфікаційної роботи	Строк виконання етапів роботи	Примітка
1.	Розробка плану роботи.	08.02.2023	
2.	Збір вихідних даних.	22.02.2023	
3.	Обробка методичних та теоретичних	15.03.2023	
	джерел.		
4.	Розробка першого та другого розділу.	12.04.2023	
5.	Розробка третього розділу.	15.05.2023	
6.	Оформлення та нормоконтроль	01.06.2023	
	кваліфікаційної роботи бакалавра.		
7.	Захист кваліфікаційної роботи.	22.06.2023	

Студент _____ А.М. Ахінько
(підпис) (ініціали та прізвище)

Керівник роботи _____ А.В. Столярова
(підпис) (ініціали та прізвище)

Нормоконтроль пройдено

Нормоконтролер _____ А.В. Столярова
(підпис) (ініціали та прізвище)

РЕФЕРАТ

Кваліфікаційна робота бакалавра «Розробка інформаційної системи засобами Django та React»: 45 с., 14 рис., 8 джерел, 1 додаток.

ВЕБ-РОЗРОБКА, ФРЕЙМВОРК, DJANGO, JS, REACT.

Об’єкт дослідження – фреймворк React бібліотеки для роботи з ним.

Мета роботи: розробка інтернет магазину засобами Django та React.

Метод дослідження – аналітичний, порівняльний.

У кваліфікаційній роботі розглянуто предметну область, розроблено вебсайт інтернет-магазину. Розглянуто мови програмування Python та JavaScript, середовища розробки Django та React.

Розробка була поділена на клієнтську, з використанням React, та серверну, з використанням Django.

SUMMARY

Bachelor’s qualifying paper «Development of an Information System Using Django and React»: 45 pages, 14 figures, 8 references, 1 supplement.

WEB DEVELOPMENT, FRAMWORK, DJANGO, JS, REACT.

Object of the study is React framework library to work with it.

Aim of the study is development of an online store using Django and React.

The methods of research are analytical, comparative.

In the qualification work, the subject area was considered, the website of the online store was developed. Python and JavaScript programming languages, Django and React development environments are considered.

Development was divided into client-side, using React, and server-side, using Django.

ЗМІСТ

ВСТУП

У сучасному цифровому світі інтернет став невід'ємною складовою нашого повсякденного життя. Розробка веб-сайтів є ключовим елементом успішного інтернет-присутності, незалежно від того, чи мовиться про особистий блог, електронний магазин або корпоративний портал.

Ця кваліфікаційна робота присвячена розробці веб-сайту з використанням двох потужних та популярних інструментів – Django та React. Django є високопродуктивним фреймворком розробки веб-додатків, побудованим на мові програмування Python. Він надає зручний інтерфейс та широкий набір функціональності для розробки швидких, масштабованих та безпечних веб-додатків.

React, з свого боку, є потужною бібліотекою JavaScript для розробки інтерфейсів користувача. Він дозволяє створювати ефективні та інтерактивні веб-сторінки, забезпечуючи швидку відповідь та динамічне оновлення контенту без перезавантаження сторінки.

У цій роботі буде розглянуто процес розробки веб-сайту з використанням Django та React. Будуть розглянуті основні принципи цих інструментів, їх інтеграція та взаємодія, а також найкращі практики розробки веб-сайтів. Будуть розглянуті ключові етапи проєктування, розробки та тестування веб-додатку, а також впровадження та підтримка сайту.

Метою цієї роботи є розробки веб-сайту засобами Django та React, а також набуття практичних навичок у розробці веб-додатків. Результатом роботи буде функціональний веб-сайт, який може бути використаний як інтернет-магазин.

1 ТЕОРЕТИЧНІ ВІДОМОСТІ

1.1 Поняття інформаційної системи

Інформаційна система (ІС) – це сукупність взаємопов’язаних компонентів, які співпрацюють для збору, зберігання, обробки та передачі інформації з метою забезпечення підтримки прийняття рішень, керування процесами та виконання інших завдань у певній організації або діловій сфері.

Основні складові ІС включають:

– апаратну частину: це фізичні компоненти, такі як комп’ютери, сервери, мережеве обладнання, сховища даних та інші пристрої, необхідні для обробки та зберігання інформації;
– програмне забезпечення: це програми та системи, які використовуються для обробки даних та виконання функцій ІС (це можуть бути програми для управління базами даних, веб-додатки, системи управління вмістом, системи аналізу даних та інші);
– інформаційні ресурси: це дані та інформація, які збираються, зберігаються та обробляються в ІС (це можуть бути дані про клієнтів, продукти, фінансові показники, звіти, документи та інші види інформації);
– людські ресурси: це люди, які працюють з ІС, включаючи розробників, адміністраторів, користувачів та інших фахівців, які забезпечують роботу ІС;
– процедури та правила: це визначені процеси, стандарти та правила, які регулюють використання ІС і визначають порядок роботи з даними та інформацією.

Інформаційна система може бути розроблена для різних цілей, таких як автоматизація бізнес-процесів, обробка та аналіз даних, комунікація та співпраця, прийняття рішень, надання послуг тощо. Вона допомагає

організаціям ефективніше управляти інформацією, покращувати продуктивність та приймати обґрунтовані рішення [1].

1.2 Основні вимоги до інформаційної системи

Основні вимоги до інформаційної системи варіюються в залежності від конкретних потреб та цілей організації. Однак, деякі загальні вимоги можуть включати:

- функціональність: ІС повинна виконувати потрібні функції та завдання для підтримки бізнес-процесів та виконання робочих завдань (це можуть бути функції збору та обробки даних, автоматизація процесів, забезпечення безпеки даних тощо);
- надійність: ІС повинна бути надійною, тобто забезпечувати стабільну та безперебійну роботу, уникати втрати даних та забезпечувати можливість відновлення після виникнення помилок або відмов;
- безпека: ІС повинна забезпечувати захист інформації від несанкціонованого доступу, втрати, пошкодження чи зловживання (це може включати застосування аутентифікації, авторизації, шифрування даних та інших методів безпеки);
- ефективність та продуктивність: ІС повинна працювати швидко та ефективно, забезпечуючи швидкий доступ до інформації, оптимальне використання ресурсів та високу продуктивність при обробці даних;
- масштабованість: ІС повинна бути здатною до розширення та масштабування, щоб враховувати зростаючі потреби організації і обсяги даних;
- легкість використання та інтуїтивний інтерфейс: ІС повинна мати зрозумілий та зручний інтерфейс, який дозволяє користувачам

легко виконувати свої завдання та забезпечує зручну навігацію та взаємодію;

– сумісність та інтеграція: ІС повинна бути сумісною з іншими системами, джерелами даних або додатками, що дозволяє обмінюватися даними та інтегруватися з іншими системами для забезпечення повної функціональності та обмеження дублювання даних;
– підтримка та обслуговування: ІС повинна мати забезпечену підтримку, документацію та можливість обслуговування, щоб забезпечити безперебійну роботу системи, вирішувати проблеми та оновлювати її відповідно до змін потреб організації.

1.3 Основні аспекти створення інформаційної системи

Вибір технологій: визначення найкращих технологій і мов програмування для розробки вашого сайту. Це можуть бути HTML, CSS, JavaScript, PHP, Python тощо.

Структура сайту: визначення логічної структури сайту, включаючи головні сторінки, підрозділи, меню навігації та посилання між ними.

Розробка фронтенду: написання HTML, CSS та JavaScript коду для створення зовнішнього вигляду та взаємодії користувача з сайтом. Це включає розміщення елементів, стилізацію, анімацію та валідацію форм.

Розробка бекенду: написання серверної логіки та бази даних для обробки запитів користувачів, збереження даних, автентифікації, взаємодії з зовнішніми сервісами тощо. Це може включати використання мов програмування, таких як PHP, Python, Ruby, або фреймворків, таких як Laravel, Django, Ruby on Rails.

Безпека: забезпечення захисту від потенційних загроз безпеки, таких як злами, вразливості вводу даних, атаки на базу даних. Це включає

застосування належних заходів безпеки, які включають валідацію даних, захист від SQL-ін’єкцій, використання шифрування тощо.

Оптимізація продуктивності: виконання оптимізацій для покращення швидкості завантаження сайту, ефективного використання ресурсів сервера, кешування даних, стиснення файлів тощо.

Тестування та налагодження: проведення тестування функціональності та виправлення помилок, які можуть виникнути під час роботи сайту. Це включає перевірку різних сценаріїв взаємодії користувача, валідацію форм, перевірку сумісності з різними браузерами тощо.

Розгортання: публікація сайту на веб-сервері та його налаштування для доступу з Інтернету. Це включає налаштування доменного імені, хостингу, бази даних та інші аспекти, щоб ваш сайт був доступним для відвідувачів [2].

1.4 Вимоги до інтернет-магазину

Основні вимоги, які часто ставляться до інтернет-магазину, включають наступні аспекти:

а) функціонал:
- реєстрація та авторизація користувачів;
- пошук та фільтрація товарів;
- додавання товарів у кошик та оформлення замовлення;
- оплата замовлення онлайн;
- керування замовленнями та їх статусами;
- можливість залишати відгуки та рейтинги для товарів;
- керування категоріями товарів та їх асортиментом;
- оптимізоване відображення товарів та швидкість завантаження сторінок;

б) дизайн та користувацький інтерфейс:

- привабливий та зручний дизайн;
- інтуїтивно зрозумілий користувацький інтерфейс;
- адаптивний дизайн, що добре працює на різних пристроях;

в) безпека:

- захищена передача даних за допомогою SSL-шифрування;
- захист від несанкціонованого доступу та атак на магазин;
- захист персональних даних користувачів та операцій з платежами;

г) адміністративна панель:

- можливість керування товарами, категоріями, замовленнями та користувачами;
- зручний інтерфейс для адміністрування та аналізу даних;

д) інтеграція з іншими сервісами:

- інтеграція з платіжними системами для онлайн-оплати;
- інтеграція з системами доставки та відстеження замовлень;

е) SEO-оптимізація:

- правильні метатеги та URL-структура;
- можливість додавання метатегів та описів для товарів;
- швидка завантаженість сторінок та оптимізований контент [3].

1.5 Back-end програмування

Бекенд програмування – це процес створення серверної логіки та функціональності, яка підтримує роботу веб-додатків і забезпечує взаємодію з базою даних, зовнішніми сервісами та передачу даних на фронтенд.

Розглянемо, що включають основні аспекти бекенд програмування.

Вибір технологій і мов програмування. Визначення найкращих технологій і мов програмування для створення серверної логіки. Це можуть

бути мови програмування, такі як PHP, Python, Ruby, Java, або фреймворки, такі як Laravel, Django, Ruby on Rails, Spring тощо.

Розробка API. Створення API (Application Programming Interface) для забезпечення комунікації між фронтендом та бекендом. Це включає визначення маршрутів, обробку запитів, передачу та отримання даних у форматі JSON або XML.

Робота з базами даних. Розробка схеми бази даних та взаємодія з нею. Це включає створення таблиць, виконання запитів (наприклад, вибірка, вставка, оновлення, видалення даних), реалізацію зв’язків між таблицями та оптимізацію запитів.

Безпека. Забезпечення захисту даних та сервера. Це включає валідацію та фільтрацію вхідних даних, захист від атак (наприклад, XSS, CSRF), шифрування даних, автентифікацію та авторизацію користувачів.

Управління сесіями та станом. Реалізація механізмів управління сесіями, збереження стану користувача, робота з кукісами (cookies).

Інтеграція з зовнішніми сервісами. Робота з API сторонніх сервісів, таких як платіжні системи, соціальні мережі, поштові сервіси тощо.

Тестування. Розробка тестових сценаріїв і виконання автоматизованих тестів для перевірки працездатності і надійності серверної логіки. Це можуть бути функціональні тести, юніт-тести, інтеграційні тести тощо.

Масштабування. Забезпечення можливості масштабування системи для оптимальної продуктивності та швидкодії. Це може включати в себе використання кешування, балансування навантаження, розподілення даних тощо.

Документація. Створення документації для розробників і користувачів, яка описує структуру, функціональність і використання бекенду.

1.5.1 Рейтинг мов програмування для Back-end розробки

Python є однією з найпопулярніших мов програмування для back-end розробки. Він має чистий синтаксис, велику кількість розширень та бібліотек, як Django та Flask, які полегшують розробку веб-додатків.

JavaScript здобуває все більшу популярність для back-end розробки за допомогою платформи Node.js. Вона дозволяє використовувати JavaScript і на стороні сервера, що спрощує розробку повного стеку JavaScript.

Java є широко використовуваною мовою для back-end розробки. Вона має потужні фреймворки, такі як Spring та Java EE, і широко застосовується у великих підприємствах.

Ruby: ruby та фреймворк ruby on rails набули популярності завдяки своїй простоті та зручності використання. Вони дозволяють швидко розробляти веб-додатки та мають багато готових рішень.

Php є однією з найбільш поширених мов програмування для back-end розробки. Вона має широку підтримку та велику спільноту розробників.

C# використовується головним чином на платформі .NET для розробки веб-додатків. Вона має потужні фреймворки, такі як ASP.NET та ASP.NET Core [4].

1.6 Front-end програмування

Front-end програмування – це процес розробки користувацького інтерфейсу веб-додатків. Воно включає створення веб-сторінок, їх оформлення, взаємодію з користувачем та інші аспекти, які стосуються клієнтської частини додатків.

Розглянемо основні аспекти front-end програмування.

Веб-мови. Використання HTML (HyperText Markup Language), CSS

(Cascading Style Sheets) та JavaScript як основних мов для створення веб-сторінок. HTML використовується для структуризації контенту, CSS – для оформлення сторінок, а JavaScript – для взаємодії з користувачем та реалізації динамічної функціональності.

Розмітка сторінок. Створення HTML-структури веб-сторінок, включаючи відповідне розташування елементів, вкладеність та навігаційні елементи.

Оформлення сторінок. Використання CSS для стилізації веб-сторінок, включаючи кольори, шрифти, розміри, межі, фонові зображення та інші візуальні ефекти. Це допомагає забезпечити привабливий та зручний для користувача дизайн.

Взаємодія з користувачем. Використання JavaScript для реалізації взаємодії з користувачем, такої як обробка подій, анімації, валідація форм, взаємодія з API та інші динамічні функції. Це дозволяє створювати інтерактивні та реактивні веб-додатки.

Адаптивний дизайн. Розробка веб-сторінок, що добре пристосовуються до різних розмірів екранів і пристроїв, зокрема мобільних телефонів, планшетів та настільних комп’ютерів. Це забезпечує зручний перегляд і взаємодію з додатком на різних пристроях.

Тестування та оптимізація. Перевірка функціональності, сумісності та продуктивності веб-сторінок на різних браузерах та пристроях. Виявлення та усунення помилок, а також оптимізація швидкодії і завантаження сторінок.

Використання фреймворків та бібліотек. Використання готових фреймворків та бібліотек, таких як React, Angular або Vue.js, для спрощення розробки і покращення продуктивності.

Документація. Створення документації, яка описує структуру, функціональність та використання фронт-енд частини веб-додатка.

1.6.1 Рейтинг мов програмування для Front-end розробки

Javascript є основною мовою програмування для фронтенд-розробки. Вона підтримується всіма сучасними браузерами і має широку екосистему і багато корисних бібліотек і фреймворків, таких як React, Angular і Vue.js.

TypeScript є суперсетом JavaScript, який додає статичну типізацію і додаткові функції до мови. Він набуває популярності серед розробників, оскільки допомагає виявляти помилки на етапі розробки і поліпшує масштабованість проєктів.

React є бібліотекою JavaScript для побудови користувацьких інтерфейсів. Він дозволяє розробникам створювати компоненти з якими легко працювати, повторно використовувати та керувати станом додатків.

Angular є повноцінним фреймворком для фронтенд-розробки, що надає багато функціональних можливостей, таких як двунаправлене зв'язування даних і систему модулів. Він підтримується компанією Google і широко використовується в великих проєктах.

Vue.js є прогресивним фреймворком JavaScript, який набуває популярності своєю простотою використання і легкістю вивчення. Він має чудову документацію, активну спільноту та здатний до розширення [4].

1.7 Аналіз проблем та викликів, пов'язаних з інтернет магазинами

У сучасному світі Інтернет-магазини здобули широку популярність та стали невід'ємною частиною електронної комерції. Однак, разом зі зростанням популярності та використанням Інтернет-магазинів, з'явилися певні проблеми та виклики, які необхідно враховувати при розробці та використанні інформаційних систем для цих магазинів.

Безпека та захист інформації. Інтернет-магазини стикаються з загрозами безпеки, такими як хакерські атаки, крадіжка конфіденційної

інформації та зловживання персональними даними клієнтів. Важливо розробляти системи, що забезпечують надійний захист даних, використовуючи шифрування, механізми авторизації та контролю доступу.

Відповідність законодавству. Інтернет-магазини повинні дотримуватись законодавства, зокрема у сфері захисту персональних даних, електронної комерції, оподаткування та інших відповідних правил. Недотримання законодавства може мати серйозні правові наслідки, тому важливо розробляти системи, які враховують ці вимоги та забезпечують відповідність.

Масштабованість та продуктивність. Інтернет-магазини можуть зіткнутися з проблемами масштабованості та продуктивності, особливо при збільшенні обсягів продажів та кількості користувачів. Важливо розробляти системи, які можуть ефективно масштабуватись та забезпечувати швидкість обробки запитів, швидкий доступ до бази даних та оптимізацію завантаження серверів.

Управління запасами та логістика. Інтернет-магазини повинні ефективно вести управління запасами, ураховуючи попит та прогнозування замовлень. Крім того, вони повинні впроваджувати ефективну систему логістики для доставки товарів клієнтам вчасно та надійно.

Конкуренція та ринкова позиція. Зараз існує велика конкуренція серед Інтернет-магазинів у багатьох галузях. Для успішної роботи необхідно проводити аналіз ринку, визначати свою унікальну пропозицію та виробляти стратегії маркетингу для залучення та утримання клієнтів.

Взаємодія з клієнтами. Інтернет-магазини повинні забезпечувати зручну та ефективну взаємодію з клієнтами. Це може включати онлайн-консультації, систему зворотного зв’язку, відгуки та оцінки товарів, програми лояльності та інші інструменти для поліпшення задоволення клієнтів.

Технологічні зміни та інновації. Технологічний прогрес швидко змінюється, і Інтернет-магазини повинні бути готові до впровадження нових

інновацій та технологічних рішень. Наприклад, це можуть бути використання штучного інтелекту, аналітики даних, розширеної реальності та інші сучасні технології для поліпшення користувацького досвіду та ефективності бізнесу.

Враховуючи ці проблеми та виклики, розроблювання інформаційної системи Інтернет-магазину повинно бути орієнтоване на розв’язання цих проблем. Основні підходи до цього можуть включати:

- ретельний аналіз безпеки та розробка механізмів захисту даних, включаючи шифрування передачі та зберігання інформації, аутентифікацію та авторизацію користувачів, а також систему контролю доступу;
- використання ефективних алгоритмів та технологій для забезпечення продуктивності системи, таких як кешування даних, оптимізація запитів до бази даних, розподілене зберігання та обробка даних;
- впровадження системи управління запасами та логістики, що дозволить ефективно керувати складами, прогнозувати попит та автоматизувати процеси доставки товарів;
- ретельне вивчення ринку та розробка стратегій маркетингу, що допоможуть залучити та утримати клієнтів (важливо проводити аналіз конкурентів, визначати сильні сторони та унікальні пропозиції Інтернет-магазину);
- впровадження інноваційних технологій та рішень, які поліпшать користувацький досвід та ефективність бізнесу (це можуть бути інтелектуальні алгоритми рекомендацій, використання аналітики даних для прогнозування попиту, інтерактивні елементи веб-сайту та інші інноваційні рішення).

В результаті ретельного аналізу проблем та викликів, а також врахування сучасних тенденцій, розробка інформаційної системи Інтернет-магазину може стати ефективним інструментом для розвитку бізнесу та забезпечення конкурентоспроможності на ринку електронної комерції. Для

успішної реалізації проєкту Інтернет-магазину, слід враховувати наступні кроки:

– визначення функціональних вимог;
– проєктування бази даних;
– розробка користувацького інтерфейсу;
– розробка системи безпеки;
– розробка системи управління запасами;
– інтеграція платіжних систем;
– розробка системи зворотного зв’язку та відгуків;
– впровадження аналітичних засобів;
– тестування та впровадження;
– підтримка та післяпродажне обслуговування.

Аналіз проблем та викликів, пов’язаних з Інтернет-магазинами, допоможе розробити інформаційну систему, яка буде відповідатинеобхідним вимогам та вирішувати проблеми, що можуть виникати в процесі функціонування Інтернет-магазину. Аналізуючи проблеми, пов’язані з безпекою, продуктивністю, управлінням запасами, конкуренцією та взаємодією з клієнтами, можна визначити ключові аспекти, які слід враховувати при розробці інформаційної системи Інтернет-магазину.

Під час розробки системи слід звернути увагу на такі аспекти:

– безпека;
– продуктивність;
– управління запасами;
– конкуренція та маркетинг;
– взаємодія з клієнтами;
– мобільна адаптація;
– система аналітики та звітності;
– забезпечення масштабованості;
– постійне вдосконалення;
– правові аспекти.

Узагальнюючи, розробка інформаційної системи для Інтернет-магазину повинна включати аналіз проблем та викликів, пов’язаних з таким видом бізнесу. Це допоможе врахувати ключові аспекти, такі як безпека, продуктивність, управління запасами, конкуренція та взаємодія з клієнтами. Планування, розробка та впровадження системи мають забезпечити ефективне функціонування Інтернет-магазину, покращення задоволення клієнтів та досягнення успіху в лектронній комерції [5].

2 ПРОЄКТУВАННЯ

2.1 Інформаційна система інтернет-магазин

Інформаційна система інтернет-магазину є комплексним програмним забезпеченням, яке дозволяє здійснювати електронну торгівлю через Інтернет. Вона включає в себе різні компоненти та функціонал, які допомагають забезпечити ефективне функціонування магазину та задоволення потреб клієнтів. Розглянемо основні аспекти інформаційної системи інтернет-магазину.

Веб-інтерфейс: розробка зручного та привабливого веб-інтерфейсу, який дозволяє клієнтам переглядати товари, здійснювати пошук, розміщувати замовлення та здійснювати оплату. Це включає створення дизайну, розташування елементів, структуру сторінок та інші аспекти користувацького інтерфейсу.

Каталог товарів: розробка системи каталогу товарів, включаючи ієрархічну структуру категорій, описи, характеристики, зображення та ціни. Це дозволяє клієнтам швидко знаходити потрібні товари та отримувати необхідну інформацію про них.

Кошик покупок: реалізація функціоналу кошика покупок, який дозволяє клієнтам додавати товари до кошика, змінювати їх кількість, видаляти або редагувати. Крім того, це забезпечує розрахунок загальної суми замовлення та підрахунок вартості доставки.

Управління замовленнями: функціонал для обробки замовлень, включаючи створення замовлення, відстеження статусу, підтвердження оплати, формування рахунку та надсилання повідомлень покупцям про станїх замовлення.

Система оплати: інтеграція з різними системами оплати, такими як кредитні картки, електронні платежі, платіжні системи, що дозволяють

клієнтам зручно та безпечно здійснювати оплату за замовлення.

Управління запасами: модуль для відстеження та управління запасами товарів, включаючи кількість наявних одиниць, перезамовлення та оновлення інформації про запаси.

Аналітика та звіти: забезпечення можливості аналізу даних, статистики продажів, поведінки клієнтів та іншої важливої інформації, що допомагає приймати обґрунтовані рішення щодо розвитку та покращення магазину.

2.2 Опис інформаційної системи інтернет-магазину меблів

Інформаційна система для інтернет-магазину меблів є комплексним рішенням, яке об’єднує різні функції та процеси, пов’язані з управлінням та функціонуванням магазину в онлайн-середовищі. Основною метою такої системи є забезпечення зручного та ефективного процесу замовлення, оплати та доставки меблів для клієнтів. Опис нижче є загальним описом інформаційної системи для інтернет-магазину меблів.

Веб-платформа: інформаційна система базується на веб-платформі, що дозволяє клієнтам переглядати асортимент меблів, робити замовлення та здійснювати оплату онлайн. Веб-платформа повинна бути зручною, інтуїтивно зрозумілою та мобільно-дружньою.

Каталог меблів: система містить повний каталог доступних меблів, включаючи зображення, опис, ціни та технічні характеристики. Клієнти можуть шукати меблі за категоріями, фільтрувати за параметрами та переглядати детальну інформацію про кожен товар.

Замовлення та оплата: клієнти можуть робити замовлення меблів через систему, додавати товари до кошика, обирати варіанти доставки та способи оплати. Система повинна надати зручний та безпечний процес оплати, включаючи онлайн-платежі, кредитні карти або інші електронні

способи оплати.

Управління замовленнями: адміністратори магазину повинні мати доступ до системи управління замовленнями, де вони можуть відстежувати статус замовлень, підтверджувати їх, встановлювати дати доставки та спілкуватися з клієнтами щодо їх замовлень.

Управління запасами: система повинна відстежувати наявність товарів на складі, автоматично оновлювати інформацію про доступність товарів та сповіщати про необхідність поповнення запасів.

Система оцінок та відгуків: клієнти можуть залишати відгуки та оцінки на продукти, що дозволяє іншим користувачам оцінити якість меблів та вибрати найкращі варіанти.

Система доставки: інформаційна система повинна мати інтегровану систему доставки, яка дозволяє обробляти замовлення, відстежувати доставку та забезпечувати зв’язок з кур’єрами або постачальниками послуг доставки.

Аналітика та звіти: система може забезпечувати аналітичні звіти про продажі, замовлення, популярність товарів та іншу статистику, що допомагає власнику магазину приймати стратегічні рішення.

2.3 Види діаграм

Існує кілька видів діаграм, які використовуються для моделювання інформаційних систем. Короткий огляд деяких основних типів діаграм наведено нижче.

Діаграми класів (*Class diagrams*). Використовуються для відображення класів, їх атрибутів, методів та взаємозв’язків між класами. Ці діаграми допомагають моделювати структуру системи та його об’єкти.

Діаграми послідовності (*Sequence diagrams*). Показують послідовність взаємодії між об’єктами системи. Вони використовуються для відображення

обміну повідомленнями між об’єктами впродовж певного часу.

Діаграми активності (*Activity diagrams*). Використовуються для моделювання послідовності дій або процесів в системі. Ці діаграми дозволяють візуалізувати алгоритми, бізнес-процеси та логіку системи.

Діаграми компонентів (*Component diagrams*). Використовуються для моделювання фізичних компонентів системи та їх залежностей. Ці діаграми допомагають відображати архітектурну структуру системи та організацію компонентів.

Діаграми розгортання (*Deployment diagrams*). Використовуються для моделювання фізичного розміщення компонентів системи на апаратному забезпеченні. Ці діаграми допомагають відображати інфраструктуру системи та взаємодію зовнішніх ресурсів.

Діаграми взаємодії (*Interaction diagrams*). Це загальна категорія діаграм, яка включає діаграми послідовності та діаграми співпраці (Collaboration diagrams). Діаграми взаємодії використовуються для моделювання взаємодії між об’єктами системи.

2.3.1 Створення діаграми класів

Діаграма класу є важливим інструментом в об’єктно-орієнтованому моделюванні, який використовується для візуалізації класів, їх взаємозв’язків та структури системи. Основною метою діаграми класу є показати структуру класів, їх атрибути та методи, а також зв’язки між класами.

Кілька кроків для створення діаграми класу:

– визначення класів;
– визначення атрибутів;
– визначення методів;
– розмістіть класи на діаграмі;
– додайте атрибути та методи;

– додайте додаткові деталі;

– перевірте та налаштуйте діаграму.

Рисунок 2.1 відображає діаграму класів.

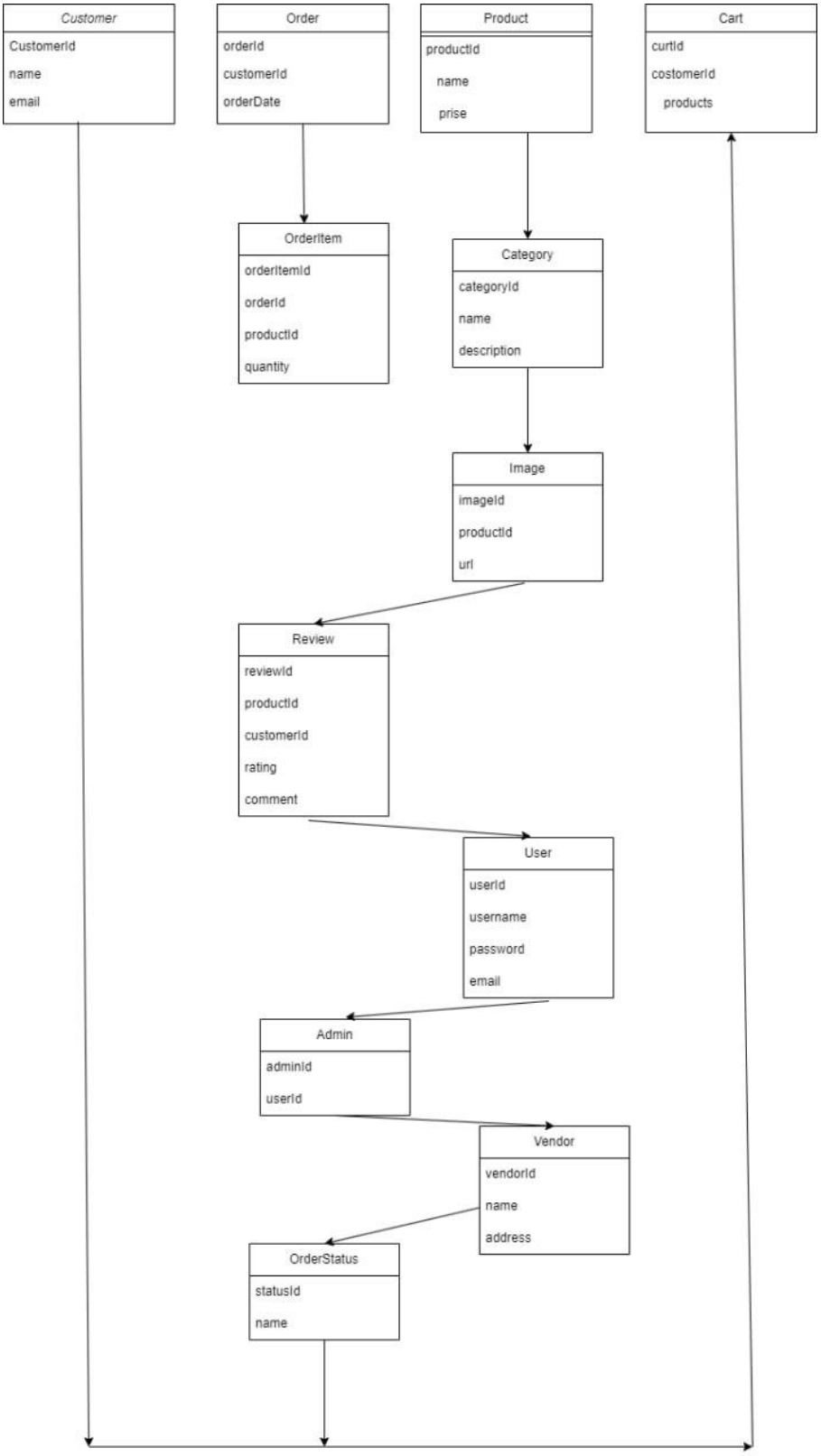

Рисунок 2.1 – Діаграма класів

2.3.2 Діаграма використання

Якщо розглянути більш детально діаграму використання (див. рис. 2.2), то бачимо 3 групи користувачів: незареєстрований користувач (або гість), зареєстрований користувач та адміністратор.

Що стосується незареєстрованого користувача, то він має можливість зареєструватися та переглянути інформацію про товари, а саме: продивитись товари або, нажавши на посилання «детальніше», ознайомитись з детальною інформацією про окремий товар. А також незареєстрований користувач може користуватися пошуком у гулі. Для того щоб, зареєструватися необхідно ввести електронну адресу, логін та пароль.

У можливості зареєстрованого користувача включається можливість незареєстрованого користувача прегляду інформації про товари, а також додаються нові функції: авторизація, редагування профілю та управління корзиною.

2.3.3 ER-діаграма

На рисунку 2.3 можемо побачити ER-діаграму.

Зв’язки:

- користувач має багато замовлень (User to Order: one-to-many);
- кожне замовлення належить одному користувачу (Order to User: many-to-one);
- кожне замовлення містить багато товарів (Order to Product: many-to-many);
- кожен товар може бути включений в багато замовлень (Product to Order: many-to-many);
- користувач має один кошик, але в кошику може бути багато товарів (User to Cart: one-to-many).

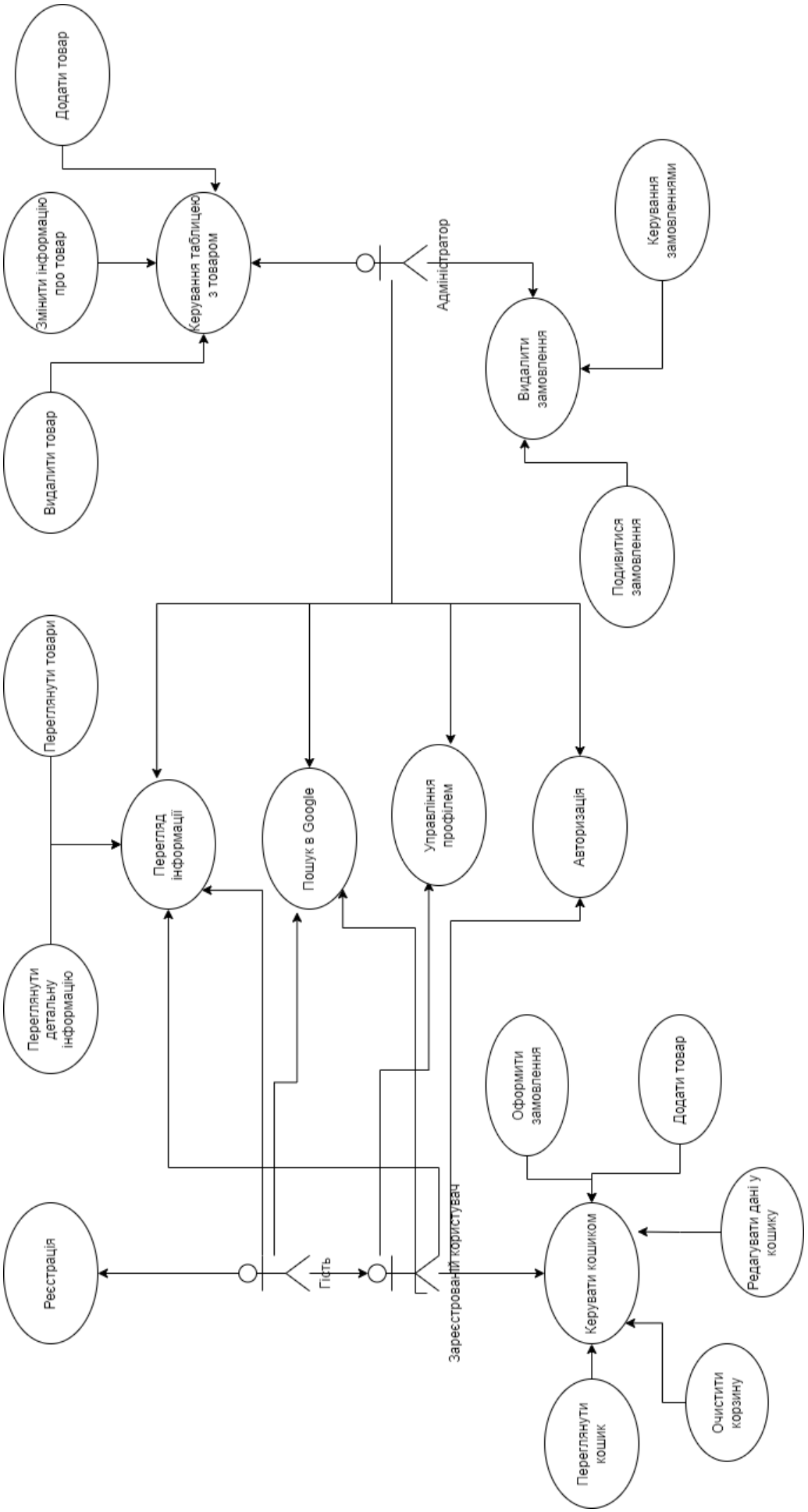

Рисунок 2.2 – Діаграма використання

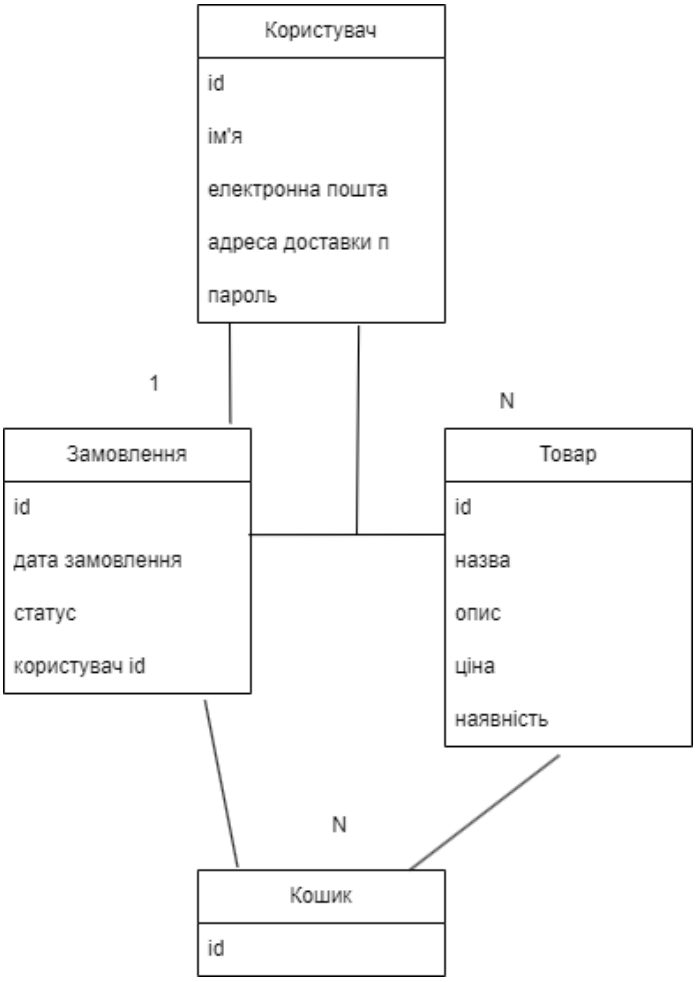

Рисунок 2.3 – ER-діаграма

2.4 Бази даних для Back-end розробки

При розробці backend-додатків інформаційних систем використовуються різні типи баз даних, залежно від потреб проєкту. Розглянемо основні типи баз даних, які використовуються для backend-розробки.

Реляційні бази даних (RDBMS). Це тип баз даних, які зберігають дані у вигляді таблиць зі зв’язками між ними. Популярні реляційні бази даних включають MySQL, PostgreSQL та Oracle. Вони часто використовуються в багатьох веб-додатках та мають потужні можливості для роботи зі структурованими даними.

NoSQL бази даних. Це тип баз даних, які використовують іншу модель

зберігання даних, не засновану на реляційній моделі. NoSQL бази даних включають MongoDB, Cassandra, Redis та інші. Вони часто використовуються для зберігання нереляційних даних, таких як документи, ключ-значення, графи тощо, і надають велику масштабованість та швидкодію.

Колоночно-орієнтовані бази даних. Це тип баз даних, які зберігають дані у вигляді колонок, а не рядків, що дозволяє ефективно зберігати та обробляти великі обсяги даних. Популярні колоночно-орієнтовані бази даних включають Apache Cassandra та Amazon Redshift.

Графові бази даних. Це тип баз даних, які засновані на математичній теорії графів і використовують графову модель для зберігання та обробки даних. Графові бази даних, такі як Neo4j, дозволяють ефективно працювати зі зв’язками між даними та виконувати складні операції пошуку і аналізу графових структур.

СУБД MySQL – це система управління базами даних (СУБД), яка надає можливість зберігання, керування та отримання доступу до структурованих даних. Вона є однією з найпопулярніших відкритих СУБД і використовується для розробки різноманітних веб-додатків та систем [**Ошибка! Источник ссылки не найден.**].

Розглядаючи більш детально діаграму бази даних (рис. 2.4), бачимо 4 таблиці: «Ціна», «Товар», «Замовлення» та «Клієнт».

Рисунок 2.4 – Діаграма бази даних

Таблиця «Ціна» зберігає інформацію о вартості товару.

В таблицю «Товар» входять такі атрибути як: назва товару, інформація про виробника, категорія товару, зображення товару, а також його опис.

Таблиця «Замовлення» включає: дату замовлення, кількість товару, адресу замовника та його номер телефону.

Таблиця «Клієнт» зберігає інформацію про клієнта: фамілію та ім’я, логін та пароль для входу, статус.

3 РОЗРОБКА

3.1 Створення Back-end частини. Початок проєкту

Створення Back-end частини інтернет-магазину на django включає ряд кроків і процесів. django – це популярний веб-фреймворк, який дозволяє швидко і ефективно розробляти веб-додатки на мові програмування python. Розглянемо основні кроки розробки back-end частини на django.

Встановлення Django. Почніть з встановлення Django на вашому сервері або робочій станції. Виконайте встановлення Python і пакету Django за допомогою менеджера пакетів, такого як pip.

Створення проєкту. Використовуйте команду Django для створення нового проєкту. Ця команда створить необхідну структуру каталогів і основні файлів для вашого проєкту.

Визначення моделей даних. Визначте моделі даних для вашого інтернет-магазину меблів. Моделі в Django відображають таблиці бази даних і використовуються для збереження даних.

Створення міграцій. Використовуйте команди Django для створення міграцій, які дозволять створити таблиці бази даних, відповідні до вашого визначення моделей.

Реалізація логіки бізнес-логіки. Додайте код для обробки логіки вашого інтернет-магазину меблів. Це може включати обробку замовлень, оплату, каталог товарів, пошук і багато іншого.

Налагодження і тестування. Проведіть тестування вашої Back-end частини, переконайтеся, що функціональність працює коректно і відповідає вимогам. Використовуйте інструменти Django для налагодження і виявлення помилок.

Розгортання. Після завершення розробки розгорніть вашу Back-end частину на веб-сервері. Django надає різні способи розгортання, включаючи використання серверів WSGI, як Apache або Nginx.

Розпочнемо роботу над проєктом Django на бекенді [6].

Інсталюємо Django (рис. 3.1).

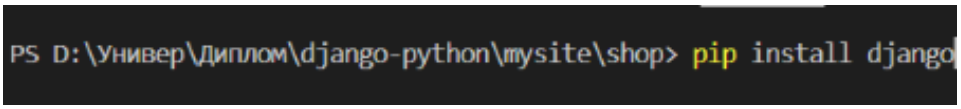

Рисунок 3.1 – Інсталяція Django

Створимо новий проєкт Django (рис. 3.2).

Рисунок 3.2 – Створення проєкту Django

Перейдемо до каталогу проєкту (рис. 3.3).

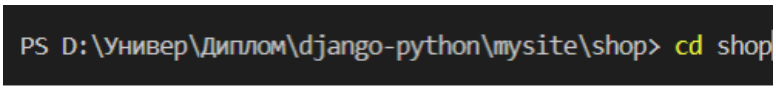

Рисунок 3.3 – Перехід до каталогу проєкту

Django розділяє функціональність на додатки. Створимо новий додаток (рис. 3.4).

Рисунок 3.4 – Створення нового додатку

Створимо міграції (рис. 3.5).

Рисунок 3.5 – Міграції

3.2 Створення Front-end частини

Створення Front-end частини веб-додатку за допомогою JavaScript та React включає в себе наступні кроки [7]:

– створення проєкту;
– розробка компонентів;
– визначення стилів для компонентів;
– реалізація взаємодії між компонентами та зв’язок з серверною частиною додатку;
– тестування та налагодження коду;
– підготовка для публікації.

3.3 Реєстрація користувачів в інтернет-магазині

3.3.1 Програмна реалізація

Реєстрація користувачів в інтернет-магазині є процесом, за допомогою якого нові користувачі створюють облікові записи і отримують доступ до функціоналу магазину.

На сторінці реєстрації користувач заповнює необхідну інформацію, таку як електронна пошта, пароль, ім’я, прізвище та інші деталі, які потрібні для створення облікового запису.

Серверна частина перевіряє введені дані на валідність, таку як правильний формат електронної пошти або унікальність облікового запису.

Якщо введені дані успішно пройшли валідацію, інформація про користувача зберігається в базі даних. Часто пароль зберігається у хешованому вигляді для забезпечення безпеки.

Деякі системи вимагають підтвердження електронної пошти, надсилаючи посилання для активації облікового запису на вказану адресу

електронної пошти. Користувач повинен перейти за посиланням, щоб підтвердити свою адресу.

Після успішної реєстрації користувач отримує повідомлення про успіх та може бути автоматично перенаправлений на сторінку входу.

В деяких випадках після реєстрації користувач автоматично автентифікується і отримує доступ до особистого кабінету без необхідності повторного входу.

На рисунку 3.6 зображено код, який використовувався для створення реєстраційної сторінки.

```
import React, { useState } from 'react';
import axios from 'axios';

const RegistrationForm = () => {
  const [email, setEmail] = useState('');
  const [password, setPassword] = useState('');

  const handleRegistration = async (e) => {
    e.preventDefault();

    try {
      const response = await axios.post('/api/register', {
        email,
        password,
      });
      console.log(response.data);
    } catch (error) {
      console.error(error);
    }
  };

  return (
    <form onSubmit={handleRegistration}>
      <input
        type="email"
        placeholder="Email"
        value={email}
        onChange={(e) => setEmail(e.target.value)}
      />
      <input
        type="password"
        placeholder="Password"
        value={password}
        onChange={(e) => setPassword(e.target.value)}
      />
      <button type="submit">Register</button>
    </form>
  );
};

export default RegistrationForm;
```

Рисунок 3.6 – Код реєстраційної форми

3.3.2 Створення сторінки авторизації та реєстрації

Для створення сторінки авторизації (рис. 3.7) та реєстрації (рис. 3.8) можна використати React та його бібліотеки.

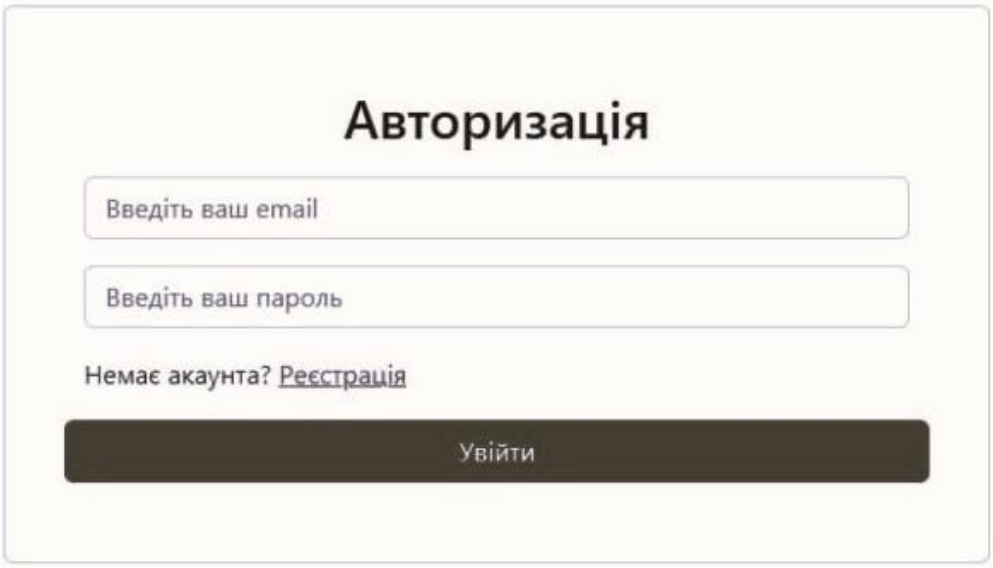

Рисунок 3.7 – Сторінка авторизації

Сторінка реєстрації складається з декількох полів, таких як: поле електронної пошти, пароля та кнопки зареєструватися.

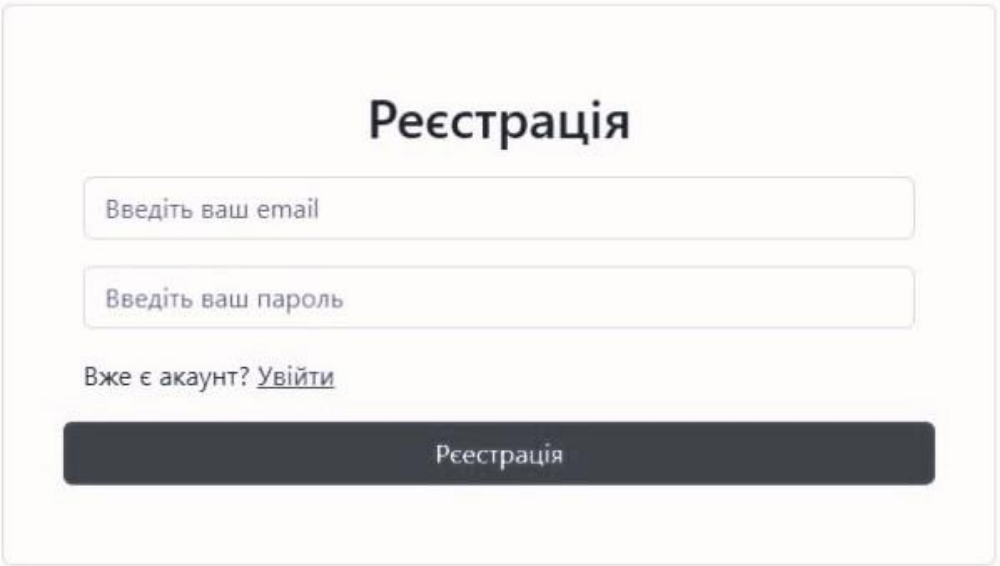

Рисунок 3.8 – Сторінка реєстрації

Після реєстрації, дані користувача зберігаються в базі даних для подальшого використання сайту користувачем [8].

3.4 Тестування

3.4.1 Створення головної сторінки інтернет-магазину

Створення головної сторінки інтернет-магазину (рис. 3.9) є важливою складовою частиною розробки інформаційної системи. Головна сторінка є першим контактом користувачів з магазином, тому вона повинна бути привабливою, зручною для навігації та ефективно передавати ключову інформацію.

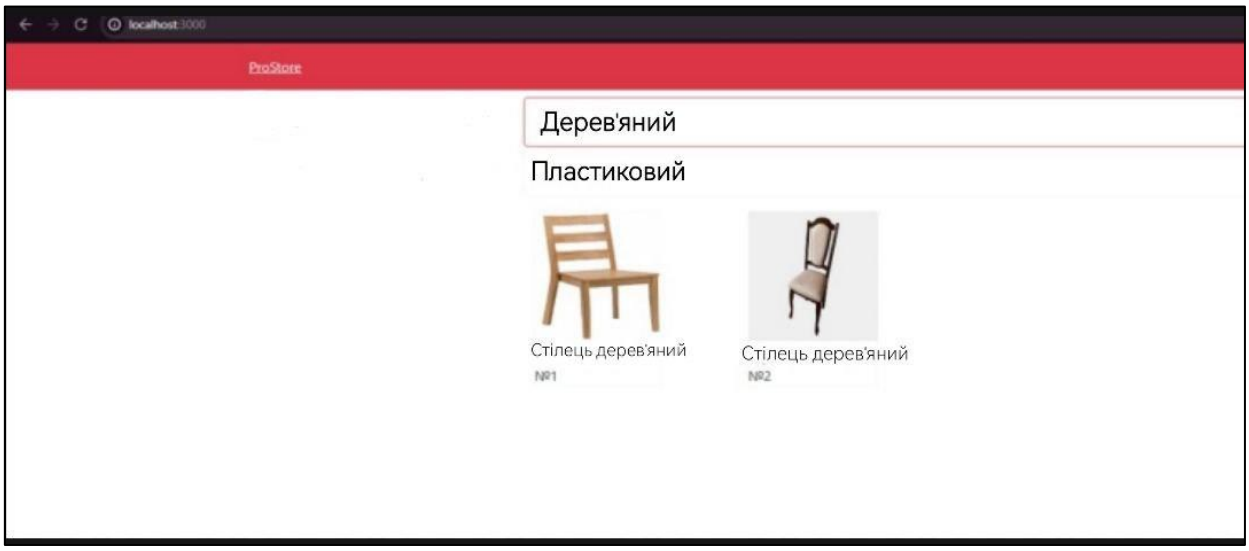

Рисунок 3.9 – Головна сторінка сайту

При створенні головної сторінки інтернет-магазину можна врахувати деякі критерії. Розглянемо їх далі.

Привабливий дизайн. Використовуйте привабливі кольори, візуальні ефекти та якісні зображення для привернення уваги користувачів. Забезпечте чистий та організований макет, щоб зробити головну сторінку привітною та

простою для сприйняття.

Зручна навігація. Розташуйте основні елементи навігації, такі як меню, пошукову панель та категорії товарів, відповідно до стандартних місць або за допомогою інтуїтивно зрозумілих символів. Забезпечте легкий доступ до важливих розділів магазину, таких як нові продукти, розпродажі або акції.

Виділення акційних товарів. На головній сторінці можна виділити акційні товари або найпопулярніші продукти, щоб залучити увагу користувачів та стимулювати їх до покупок. Використовуйте великі зображення, цікаві заголовки та вказівки до деталей товару.

Корисна інформація. Представте корисну інформацію на головній сторінці, таку як контактні дані, умови доставки, повернення товару або спеціальні пропозиції для нових користувачів. Забезпечте швидкий доступ до інформації про контакти та підтримку, щоб користувачі мали змогу отримати відповіді на свої запитання.

Відгуки і рейтинги. Якщо у вашому магазині є відгуки та рейтинги товарів, можна вивести деякі з них на головну сторінку. Це допоможе встановити довіру до магазину та його продукції, а також стимулюватиме користувачів до здійснення покупок.

Адаптивний дизайн. Забезпечте, щоб головна сторінка була адаптивною до різних пристроїв, таких як комп’ютери, планшети та смартфони. Впевніться, що макет адаптується до розміру екрану та зберігає свою функціональність та зручність використання на різних пристроях.

3.4.2 Створення сторінки товару

При створенні сторінки товару (див. рис. 3.10) можна додати деякі компоненти. Розглянемо їх далі.

Зображення товару має бути велике та якісне, яке дозволяє користувачам детально роздивитися товар та зрозуміти, як він виглядає.

Опис товару докладний, включаючи його характеристики, особливості, матеріали, розміри, вагу, кольори тощо. Треба забезпечте чітку структуру опису, щоб користувачам було легко знайти необхідну інформацію.

Ціна товару та інформація про його наявність. Кнопка «Додати до кошика» або «Купити», щоб користувачі могли зробити покупку безпосередньо зі сторінки товару.

Якщо є відгуки та рейтинги товару, виведіть їх на сторінці товару. Відобразіть пов’язані товари або додаткові пропозиції на сторінці товару. Це можуть бути схожі товари, акційні пропозиції, додаткові аксесуари або рекомендації на підставі покупок інших користувачів.

Додайте кнопки поширення на соціальних мережах, щоб користувачі могли легко поділитися товаром зі своїми друзями та знайомими.

Дозвольте користувачам додавати товар до свого списку бажань для подальшого зручного перегляду або замовлення.

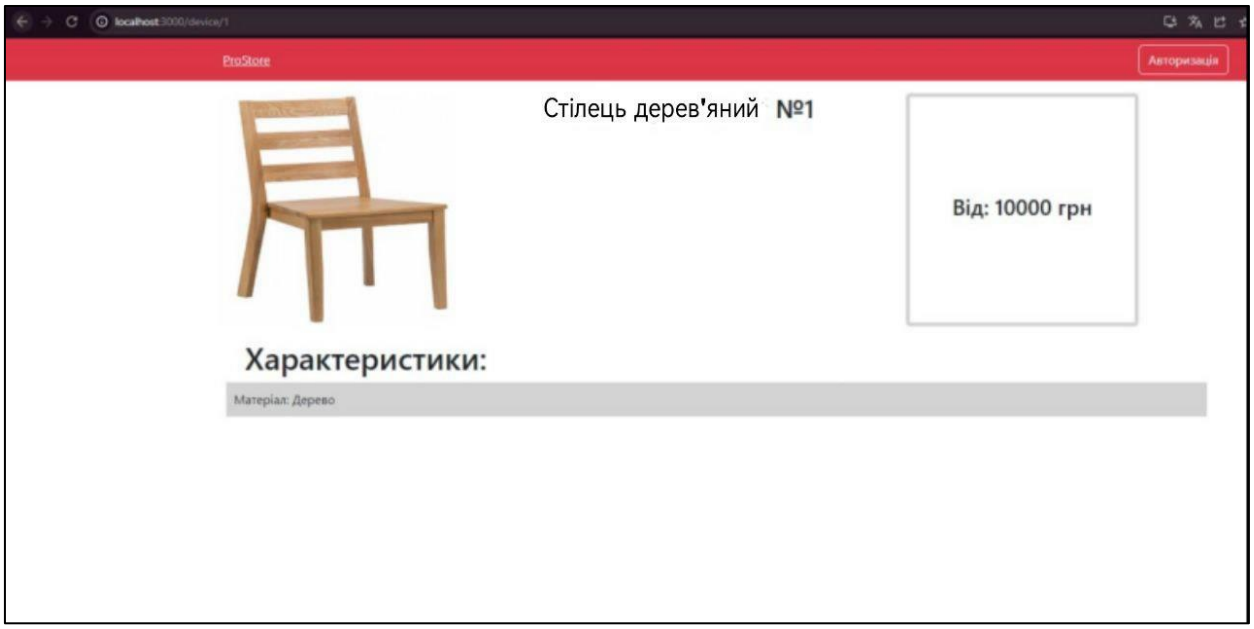

Рисунок 3.10 – Сторінка товару

ВИСНОВКИ

У цій дипломній роботі було проведено розробку інформаційної системи інтернет-магазину з використанням Django та React. Результатом роботи є функціональна веб-платформа, яка дозволяє користувачам зручно переглядати продукти, робити покупки, реєструватися та авторизуватися в системі.

Під час розробки було використано фреймворк Django для забезпечення швидкого та ефективного створення серверної частини системи. Django забезпечує високу продуктивність, розширюваність та безпеку, що робить його ідеальним вибором для розробки інтернет-магазину.

Також було використано фреймворк React для створення користувацького інтерфейсу інтернет-магазину. React дозволяє створювати компоненти, які можна повторно використовувати, що спрощує розробку та підтримку фронтенду системи. Крім того, React забезпечує швидку відповідь інтерфейсу та покращує загальний досвід користувача.

Під час розробки були враховані проблеми та виклики, пов’язані з інтернет-магазинами, такі як конкуренція, безпека, логістика, мобільність, клієнтське обслуговування, маркетинг та аналітика. Шляхом врахування цих факторів та впровадження відповідних функціональних модулів було досягнуто створення ефективної та конкурентоспроможної інформаційної системи.

Отже, результати цієї дипломної роботи свідчать про успішну розробку інформаційної системи інтернет-магазину з використанням Django та React. Розроблена система має потенціал для успішного використання в реальних умовах та забезпечення зручного та безпечного онлайн-шопінгу для користувачів.

ПЕРЕЛІК ПОСИЛАНЬ

1. Теорія інформаційної системи. URL: https://ela.kpi.ua/bitstream/123456789/33651/1/PIS_KL.pdf (дата звернення: 10.03.2023).
2. Вимоги до інтернет-магазину. URL: https://sysale.ua/uk/trebovaniya-k-vizualnoj-sostavlyayushhej-i-dizajnu-internet-magazinov/ (дата звернення: 16.03.2023).
3. Рейтинг мов програмування 2023. URL: https://itc.ua/ua/novini/rejtyng-populyarnyh-v-ukrayini-mov-programuvannya-2022-roku-v-liderah-javascript-java-j-python/ (дата звернення: 21.03.2023).
4. Теоретичні відомості Django. URL: https://developer.mozilla.org/en-US/docs/Learn/Server-side/Django (дата звернення: 30.03.2023).
5. База даних MySQL. URL: https://www.w3schools.com/MySQL/default.asp (дата звернення: 05.04.2023).
6. Програмвання на Python. URL: https://www.w3schools.com/python/ (дата звернення: 12.04.2023).
7. Мова програмування JavaScript. URL: https://www.w3schools.com/js/ (дата звернення: 17.04.2023).
8. Фреймворк React. URL: https://www.tutorialspoint.com/reactjs/index.htm (дата звернення: 23.04.2023).

ДОДАТОК А

Код програмного забезпечення

А.1 Файл package.json

```
{
"name": "prostore",
"version": "0.1.0",
"private": true,
"dependencies": {
"@testing-library/jest-dom": "^5.16.5",
"@testing-library/react": "^13.4.0",
"@testing-library/user-event": "^13.5.0",
"react": "^18.2.0",
"react-dom": "^18.2.0",
"react-scripts": "5.0.1",
"web-vitals": "^2.1.4"
},
"scripts": {
"start": "react-scripts start",
"build": "react-scripts build",
"test": "react-scripts test",
"eject": "react-scripts eject"
},
"eslintConfig": {
"extends": [
"react-app",
"react-app/jest"
]
```

```
},
"browserslist": {
"production": [
">0.2%",
"not dead",
"not op_mini all"
],
"development": [
"last 1 chrome version",
"last 1 firefox version",
"last 1 safari version"
]
},
"devDependencies": {
"webpack-cli": "^5.1.1"
}
}
```

A.2 Файл Login.js

```
import React from 'react';

const Login = () => {
const [email, setEmail] = useState('');
const [password, setPassword] = useState('');

const handleEmailChange = (event) => {
setEmail(event.target.value);
};
```

```
const handlePasswordChange = (event) => {
setPassword(event.target.value);
};

const handleLogin = (event) => {
event.preventDefault();
// Виконати логіку авторизації
// Наприклад, відправити дані на сервер для перевірки
};

return (
<div>
<h2>Авторизація</h2>
<form onSubmit={handleLogin}>
<div>
<label>Email:</label>
<input type="email" value={email} onChange={handleEmailChange} />
</div>
<div>
<label>Пароль:</label>
<input type="password" value={password} onChange={handlePasswordChange}
/>
</div>
<button type="submit">Увійти</button>
</form>
</div>
);
};
export default Login;
```

А.3 Файл Models.py

```
from django.db import models

class Product(models.Model):
name = models.CharField(max_length=100)
price = models.DecimalField(max_digits=10, decimal_places=2)
description = models.TextField()
# Додаткові поля

def __str__(self):
return self.name
```