МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
МИКОЛАЇВСЬКИЙ НАЦІОНАЛЬНИЙ АГРАРНИЙ УНІВЕРСИТЕТ

Факультет менеджменту
Кафедра економічної кібернетики, комп'ютерних наук
та інформаційних технологій

Кваліфікаційна наукова
праця на правах рукопису

АРАПЕНКО Вікторія Ігорівна

УДК 004.451.1:004.738.5:339.138(043.2)

КВАЛІФІКАЦІЙНА РОБОТА

РОЗРОБКА ТА ІМПЛЕМЕНТАЦІЯ ВЕБ-ЗАСТОСУНКУ ЕЛЕКТРОННОЇ КОМЕРЦІЇ ДЛЯ РЕАЛІЗАЦІЇ ОДЯГУ ТА ВЗУТТЯ

Спеціальність 122 «Комп'ютерні науки»
Галузь знань – 12 «Інформаційні технології»

Подається на здобуття освітнього ступеня бакалавра

Кваліфікаційна робота містить результати власних досліджень. Використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело

Арапенко В. І.

Науковий керівник:
Пархоменко Олександр Юрійович,
кандидат фізико-математичних наук, доцент

Завідувач кафедри:
Тищенко Світлана Іванівна,
кандидат педагогічних наук, доцент

Миколаїв–2025

АНОТАЦІЯ

Арапенко В. І. Розробка та імплементація веб-застосунку електронної комерції для реалізації одягу та взуття. – Кваліфікаційна наукова праця на правах рукопису.

Робота на здобуття освітнього ступеня бакалавра за спеціальністю 122 «Комп'ютерні науки». Миколаївський національний аграрний університет, Миколаїв, 2025.

Об'єктом дослідження є процеси розробки та використання веб-застосунків у сфері електронної комерції.

Предмет дослідження – інструменти, методи та технології розробки веб-застосунків для продажу товарів через інтернет.

Метою роботи є створення веб-застосунку для електронної комерції, що забезпечує ефективну платформу для продажу одягу та взуття із функціональністю управління каталогом товарів, кошиком та оформленням замовлень, інтегрованого з базою даних та орієнтованого на оптимізацію взаємодії з користувачем.

Робота складається з пояснювальної записки, яка містить вступ, три розділи, висновки, список використаних джерел та додатки.

У першому розділі розглянуто теоретичні основи веб-технологій та електронної комерції, здійснено аналіз ключових етапів проєктування архітектури застосунків і визначено вимоги до побудови ефективних онлайн-платформ.

Другий розділ присвячено обґрунтуванню вибору архітектурних рішень і технологій розробки веб-застосунку. Проведено порівняльний аналіз існуючих електронних платформ і сформульовано технічні вимоги до функціональності майбутнього застосунку.

У третьому розділі висвітлено процеси практичної реалізації веб-застосунку: розробку основних модулів, інтеграцію з базою даних, тестування працездатності та оптимізацію продуктивності. Окрему увагу приділено питанням забезпечення доступності веб-застосунку для всіх категорій

користувачів. Запропоновано рекомендації для подальшого розвитку функціоналу і підвищення стійкості системи.

Результати дослідження обговорено на всеукраїнських науково-практичних конференціях та представлені у вигляді тез доповідей.

Кваліфікаційна робота викладена на 71 сторінках, містить 4 таблиці, 8 рисунків, список використаних джерел включає 30 найменування.

Ключові слова: веб-застосунок, Django, електронна комерція, база даних, інтерфейс користувача, оптимізація, тестування, онлайн-торгівля.

ANNOTATION

Arapenko V.I. Development and implementation of a web-based e-commerce application for the sale of clothing and footwear. - Qualifying scientific work on the rights of a manuscript.

Work for a bachelor's degree in speciality 122 «Computer Science». Mykolayiv National Agrarian University, Mykolaiv, 2025.

The object of research is the processes of developing and using web applications in the field of e-commerce.

The subject of the study is tools, methods and technologies for developing web applications for selling goods over the Internet.

The purpose of the study is to create a web-based e-commerce application that provides an effective platform for selling clothes and shoes with the functionality of managing a product catalogue, shopping cart and ordering, integrated with a database and focused on optimising user experience.

The paper consists of an explanatory note, which includes an introduction, three chapters, conclusions, a list of references and appendices.

The first section discusses the theoretical foundations of web technologies and e-commerce, analyses the key stages of application architecture design, and identifies the requirements for building effective online platforms.

The second section is devoted to the rationale for choosing architectural solutions and technologies for developing a web application. A comparative analysis of existing electronic platforms is carried out and technical requirements for the functionality of the future application are formulated.

The third section highlights the processes of practical implementation of the web application: development of the main modules, integration with the database, performance testing and performance optimisation. Special attention is paid to ensuring the accessibility of the web application for all categories of users. Recommendations for further development of the functionality and improvement of the system's sustainability are offered.

The research results were discussed at all-Ukrainian scientific and practical conferences and presented in the form of abstracts.

The qualification work is set out on 71 pages, contains 4 tables, 8 figures, the list of references includes 30 titles.

Keywords: web application, Django, e-commerce, database, user interface, optimisation, testing, online trading.

ЗМІСТ

ВСТУП

Актуальність теми дослідження обумовлена стрімким розвитком електронної комерції у світі, що потребує створення ефективних веб-застосунків для забезпечення онлайн-продажів товарів та послуг. На сучасному етапі розвиток цифрової економіки відкриває нові можливості для підприємств малого та середнього бізнесу у галузі електронної торгівлі, водночас висуваючи високі вимоги до якості веб-ресурсів, їх функціональності, надійності та зручності використання. Створення веб-застосунків із застосуванням сучасних фреймворків, зокрема Django, стає важливим інструментом підвищення ефективності бізнес-процесів.

Необхідність дослідження обґрунтовується тим, що для конкурентного функціонування на ринку електронної комерції необхідно забезпечити не лише присутність у мережі, а й створити технічно досконалі, доступні та оптимізовані веб-застосунки, здатні забезпечити високу якість користувацького досвіду. Водночас важливою є також підтримка й подальший розвиток таких ресурсів.

Мета і завдання дослідження. Метою кваліфікаційної роботи є створення веб-застосунку для електронної комерції, що забезпечує ефективну платформу для продажу одягу та взуття із функціональністю управління каталогом товарів, кошиком та оформленням замовлень, інтегрованого з базою даних та орієнтованого на оптимізацію взаємодії з користувачем.

Відповідно до мети поставлено і вирішено теоретичні, методичні і практичні завдання:

- проаналізувати теоретичні основи створення веб-застосунків у сфері електронної комерції;
- здійснити розробку основних модулів веб-застосунку: каталогу товарів, кошика, оформлення замовлення;
- реалізувати інтеграцію веб-застосунку з базою даних для зберігання інформації про товари та замовлення;

– провести тестування функціональності та доступності веб-застосунку;

– здійснити оптимізацію веб-застосунку для підвищення його ефективності;

– надати рекомендації щодо розширення функціональності та подальшої підтримки розробленого веб-застосунку.

Методи дослідження включають:

– Аналіз сучасних платформ електронної комерції.

– Методи веб-програмування (HTML, CSS, Django, Python).

– Інтеграція з базами даних (PostgreSQL).

– Тестування функціональності та продуктивності веб-застосунків.

– Аналіз безпеки веб-застосунків.

Інформаційна база дослідження складається з офіційної документації фреймворку Django, наукових публікацій у сфері веб-розробки, електронної комерції, а також матеріалів профільних сайтів та статистичних даних щодо розвитку інтернет-торгівлі.

Практичне значення результатів дослідження полягає у розробці веб-застосунку для електронної комерції, який може бути впроваджений у діяльність підприємств для автоматизації процесів продажу товарів через інтернет. Отримані результати сприяють підвищенню ефективності роботи підприємств та розширенню можливостей електронної торгівлі.

Основні положення роботи апробовані на Всеукраїнських науково-практичних конференціях:

– XI Всеукраїнська конференція «Екологічні та соціальні аспекти розвитку економіки в умовах євроінтеграції», м. Миколаїв, 23-25 жовтня 2024 р. Тема доповіді: «Екологічні аспекти впровадження веб-застосунків електронної комерції в Fashion-індустрії»;

– III Всеукраїнська конференція «Управління механізмами гарантування фінансово-економічної безпеки соціально-економічних систем різних рівнів функціонування», м. Миколаїв, 20-22 листопада 2024 р. Тема доповіді: «Вплив

цифрових інструментів на фінансово-економічну безпеку підприємств аграрної галузі»;

– 37-ма студентська науково-теоретична конференція «Участь молоді у розбудові агропромислового комплексу країни», м. Миколаїв, 18-19 березня 2025 р. Тема доповіді: «Електронна комерція як інструмент залучення молоді до розвитку агропромислового комплексу»;

РОЗДІЛ 1
ТЕОРЕТИЧНІ ОСНОВИ ВЕБ-ТЕХНОЛОГІЙ ТА ЕЛЕКТРОННОЇ КОМЕРЦІЇ

1.1. Огляд сучасних технологій веб-розробки

Сучасні технології веб-розробки представляють собою комплексну екосистему інструментів, фреймворків та мов програмування, які постійно еволюціонують та адаптуються до нових вимог цифрового світу. HTML5, CSS3 та JavaScript формують фундаментальну тріаду технологій front-end розробки, забезпечуючи структуру, стилізацію та інтерактивність веб-сторінок відповідно.

Для реалізації серверної частини веб-застосунків однією з найпопулярніших технологій є Django – фреймворк на мові програмування Python, що забезпечує обробку запитів, взаємодію з базами даних та логіку функціонування веб-застосунку.

Значний прогрес у розвитку веб-стандартів дозволив створювати більш динамічні та адаптивні інтерфейси, які однаково добре функціонують на різних пристроях та платформах. Сучасні браузери підтримують широкий спектр API та нативних функцій, що раніше вимагали використання сторонніх плагінів або складних обхідних рішень. React, Angular та Vue.js стали домінуючими фреймворками для розробки користувацьких інтерфейсів, пропонуючи потужні інструменти для створення складних веб-додатків. Компонентний підхід до розробки, який вони впровадили, дозволяє створювати масштабовані та легко підтримувані додатки [**Error! Reference source not found.**, **Error! Reference source not found.**]. Віртуальний DOM та реактивне оновлення даних значно покращили продуктивність веб-додатків. Системи збірки та транспіляції коду, такі як Webpack та Babel, дозволяють використовувати найновіші можливості JavaScript, одночасно забезпечуючи сумісність із старішими браузерами. Менеджери пакетів npm та yarn спростили процес управління залежностями та розгортання проектів.

Node.js революціонізував серверну розробку, дозволивши використовувати JavaScript на стороні сервера. Express.js, Koa та NestJS стали популярними фреймворками для створення масштабованих серверних додатків. Використання єдиної мови програмування для front-end та back-end розробки значно спростило процес створення повноцінних веб-додатків [**Error! Reference source not found.**]. RESTful API та GraphQL надають гнучкі способи обміну даними між клієнтом та сервером, дозволяючи створювати ефективні та швидкі додатки. Мікросервісна архітектура дозволяє розробляти складні системи як набір незалежних сервісів. Технології контейнеризації, такі як Docker та Kubernetes, змінили підхід до розгортання та масштабування веб-додатків. Хмарні платформи AWS, Google Cloud та Azure надають широкий спектр сервісів для хостингу та управління веб-додатками. Serverless архітектура дозволяє розробникам зосередитися на написанні бізнес-логіки, не турбуючись про інфраструктуру. Continuous Integration та Continuous Deployment (CI/CD) автоматизують процеси тестування та розгортання, забезпечуючи швидку доставку нових функцій користувачам. Моніторинг та аналітика допомагають відстежувати продуктивність та виявляти проблеми в режимі реального часу.

Прогресивні веб-додатки (PWA) стирають межу між веб-сайтами та нативними мобільними додатками. Service Workers забезпечують офлайн-функціональність та push-повідомлення. WebAssembly відкриває нові можливості для високопродуктивних веб-додатків, дозволяючи виконувати код, написаний на C++, Rust та інших мовах програмування, безпосередньо в браузері. WebGL та WebGPU надають доступ до апаратного прискорення для створення складної графіки та анімацій. Технології віртуальної та доповненої реальності стають доступними через веб-інтерфейси. CSS-фреймворки та методології, такі як Tailwind CSS, Bootstrap та CSS-in-JS, пропонують різні підходи до стилізації веб-додатків. Системи дизайну та компонентні бібліотеки спрощують створення узгоджених користувацьких інтерфейсів. Препроцесори SASS та LESS розширюють можливості CSS, додаючи змінні, міксини та інші

корисні функції. Адаптивний дизайн та CSS Grid дозволяють створювати гнучкі макети, що адаптуються до різних розмірів екрану. Анімації та переходи покращують користувацький досвід та роблять інтерфейси більш інтуїтивними.

Технології тестування та забезпечення якості коду стали невід'ємною частиною процесу розробки. Jest, Mocha та Cypress дозволяють створювати автоматизовані тести для різних аспектів веб-додатків. ESLint та Prettier забезпечують дотримання стандартів кодування та форматування. TypeScript додає статичну типізацію до JavaScript, покращуючи надійність та підтримуваність коду. Code reviewing та pair programming стали стандартними практиками в командній розробці. Оптимізація продуктивності веб-додатків включає використання кешування, lazy loading та code splitting. Системи збірки автоматично оптимізують та мініфікують код для продакшену. Content Delivery Networks (CDN) забезпечують швидку доставку статичного контенту користувачам по всьому світу. Технології стиснення та оптимізації зображень допомагають зменшити розмір завантажуваних ресурсів. Метрики Web Vitals допомагають оцінювати та покращувати користувацький досвід.

Безпека веб-додатків забезпечується використанням HTTPS, CORS та інших механізмів захисту. OAuth2 та JWT надають безпечні способи автентифікації та авторизації користувачів. CSP та інші заголовки безпеки захищають від поширених веб-вразливостей. Сканери безпеки та penetration testing допомагають виявляти та усувати потенційні загрози. Шифрування даних та безпечне зберігання паролів стали стандартними вимогами для сучасних веб-додатків. Розробка веб-додатків стає все більш доступною завдяки low-code та no-code платформам. Візуальні конструктори та drag-and-drop інтерфейси дозволяють створювати прості веб-сайти без глибоких технічних знань. Headless CMS системи надають гнучкі способи управління контентом для сучасних веб-додатків API-first підхід до розробки дозволяє створювати масштабовані та інтегровані системи [**Error! Reference source not found.**]. Штучний інтелект та машинне навчання починають інтегруватися в процеси веб-розробки.

Технології реального часу, такі як WebSocket та Server-Sent Events, дозволяють створювати інтерактивні додатки з миттєвим оновленням даних. Firebase та подібні платформи надають готові рішення для real-time функціональності. WebRTC забезпечує можливість прямої комунікації між браузерами для відео- та аудіо-зв'язку. Push-повідомлення та веб-хуки дозволяють тримати користувачів в курсі важливих подій. Екосистема веб-розробки включає також інструменти для аналітики та відстеження поведінки користувачів. Google Analytics та подібні сервіси надають цінні дані про використання веб-додатків. A/B тестування допомагає оптимізувати користувацький досвід та конверсію. Системи моніторингу помилок, такі як Sentry, допомагають швидко виявляти та виправляти проблеми в продакшені.

SEO та доступність стали критично важливими аспектами веб-розробки. Семантична розмітка та структуровані дані покращують видимість у пошукових системах. WAI-ARIA та інші важливі міжнародні стандарти веб-доступності відіграють ключову роль у забезпеченні повноцінного використання веб-додатків людьми з різними формами обмежених можливостей та особливих потреб. Комплексна оптимізація веб-ресурсів для мобільних пристроїв різних типів, включаючи швидкість завантаження контенту та адаптивність інтерфейсу, суттєво впливає на позиції сайту в результатах пошукових систем та загальну видимість для користувачів. Глибока інтеграція сучасних веб-додатків із різноманітними зовнішніми сервісами та прикладними програмними інтерфейсами (API) значно розширює їх базові функціональні можливості та потенціал. Сучасні платіжні системи, популярні соціальні мережі, передові картографічні сервіси та численні інші API-інтерфейси надають розробникам потужні інструменти для створення багатофункціональних та інтерактивних веб-додатків. Використання технології webhook-ів та спеціалізованих інтеграційних платформ істотно спрощує процес налаштування та підтримки взаємодії між різноманітними програмними системами. Впровадження гнучкої мікросервісної архітектури разом із

використанням API Gateway створює ефективну основу для управління складними системами інтеграцій та забезпечує їх масштабованість.

1.2. Архітектура та компоненти веб-застосунків

Сучасна архітектура веб-застосунків базується на багаторівневій структурі, що забезпечує чіткий розподіл відповідальності між компонентами. Презентаційний рівень відповідає за взаємодію з користувачем та відображення інтерфейсу. Рівень бізнес-логіки містить основні алгоритми та правила обробки даних. Рівень доступу до даних забезпечує взаємодію з базами даних та зовнішніми джерелами інформації. Така структура дозволяє оптимізувати процеси розробки та підтримки веб-застосунків.

Таблиця 1.1 Основні компоненти веб-застосунків та їх функції

Рівень архітектури	**Компоненти**	**Основні функції**
Презентаційний	Frontend-компоненти, UI елементи, JavaScript-обробники	Відображення інтерфейсу, взаємодія з користувачем
Бізнес-логіка	Сервіси, контролери, валідатори	Обробка даних, реалізація бізнес-правил
Доступ до даних	Репозиторії, ORM, драйвери БД	Взаємодія з базами даних, кешування
Інтеграційний	API Gateway, адаптери, конектори	Взаємодія з зовнішніми системами
Інфраструктурний	Сервери, балансувальники, моніторинг	Забезпечення роботи середовища виконання, масштабування, та стабільності

Джерело: сформовано автором

Модульна архітектура сучасних веб-систем забезпечує гнучке масштабування та ефективне оновлення окремих компонентів без впливу на

роботу всієї системи в цілому. Клієнтська частина сучасного веб-застосунку складається з множини взаємопов’язаних компонентів, які виконують свої функції безпосередньо у браузері кінцевого користувача. Динамічне DOM-дерево формує комплексну структуру веб-сторінки та забезпечує розширені можливості для інтерактивної взаємодії з користувачем. Вбудований JavaScript-движок ефективно обробляє різноманітні користувацькі сценарії та забезпечує безперебійне керування динамічним оновленням контенту на сторінці. Сучасні системи клієнтської маршрутизації надають зручні інструменти для швидкої та плавної навігації між різними функціональними розділами веб-застосунку. Спеціалізовані менеджери станів здійснюють централізований контроль над даними та відстежують їх зміни в межах всього клієнтського застосунку. Інтерактивні компоненти користувацького інтерфейсу формують привабливу візуальну частину застосунку та забезпечують ефективну обробку всіх типів взаємодії з користувачем.

Серверна архітектура веб-застосунків реалізує обробку запитів від клієнтів та керує бізнес-логікою. Веб-сервер обробляє HTTP-запити та розподіляє навантаження між компонентами застосунку. Контролери обробляють вхідні запити та координують роботу різних частин системи. Багатофункціональні сервіси реалізують складну бізнес-логіку та забезпечують виконання чітко визначених правил обробки різноманітних даних у системі [**Error! Reference source not found.**]. Спеціалізовані репозиторії надають зручний рівень абстракції для ефективної та безпечної роботи з різними типами баз даних та сховищ інформації. Інтелектуальні системи кешування значно оптимізують загальну продуктивність застосунку шляхом тимчасового зберігання та швидкого доступу до найбільш часто використовуваних даних. Надійна взаємодія між клієнтською та серверною частинами сучасного веб-застосунку здійснюється через різноманітні протоколи та стандартизовані формати передачі даних. Добре спроектований REST API забезпечує уніфікований та зручний інтерфейс для ефективного обміну даними між різними компонентами системи. Технологія WebSocket надає можливість

встановлення швидкого двостороннього зв’язку між клієнтом та сервером у реальному часі. Сучасний GraphQL надає клієнтським застосункам гнучкі можливості для точного запиту саме тих даних, які необхідні в конкретний момент. Комплексні системи автентифікації та авторизації забезпечують надійний контроль доступу до всіх захищених ресурсів веб-застосунку. Багаторівневі механізми валідації ретельно перевіряють коректність та цілісність даних на всіх рівнях взаємодії в системі.

Бази даних є ключовим компонентом архітектури веб-застосунків, забезпечуючи надійне зберігання та управління даними. Реляційні бази даних забезпечують структуроване зберігання інформації та підтримку транзакцій. Сучасні NoSQL рішення забезпечують виняткову гнучкість та ефективність при зберіганні та обробці різноманітних типів неструктурованих даних у масштабних системах [**Error! Reference source not found.**]. Високопродуктивні системи кешування суттєво оптимізують швидкість доступу до найбільш затребуваної інформації шляхом її тимчасового зберігання у швидкій пам’яті. Передові механізми реплікації даних гарантують надійну роботу та повну відмовостійкість розподілених баз даних у випадку часткових збоїв системи. Комплексні системи резервного копіювання забезпечують надійне збереження та можливість швидкого відновлення критично важливих даних у разі виникнення серйозних технічних проблем чи збоїв обладнання. Архітектура мікросервісів розділяє веб-застосунок на незалежні компоненти, кожен з яких відповідає за конкретну функціональність.

Різноманітні мікросервіси здійснюють надійну комунікацію між собою через чітко визначені стандартизовані протоколи та уніфіковані програмні інтерфейси. Потужна система оркестрації забезпечує ефективне керування процесами розгортання та динамічного масштабування розподілених мікросервісів. Спеціалізовані сервіс-реєстри надають гнучкі можливості для автоматичного виявлення та налагодження безперебійної комунікації між множиною сервісів. Надійний механізм Circuit breaker ефективно запобігає виникненню небезпечних каскадних відмов у складній розподіленій системі.

Комплексні системи моніторингу постійно відстежують поточний стан та ключові показники продуктивності кожного окремого сервісу в реальному часі. Автоматизовані системи збірки та розгортання значно спрощують та прискорюють процеси підготовки та публікації складних веб-застосунків у виробниче середовище. Сучасні системи контролю версій надійно зберігають повну історію змін та забезпечують ефективну координацію роботи розподіленої команди розробників. Гнучкі CI/CD пайплайни повністю автоматизують критично важливі процеси тестування та розгортання програмного забезпечення. Технологія контейнеризації гарантує надійну ізоляцію та повну портативність всіх компонентів системи між різними середовищами. Розвинені системи оркестрації контейнерів ефективно керують їх повним життєвим циклом та забезпечують гнучке горизонтальне масштабування. Багаторівневі системи моніторингу безперервно відстежують загальний стан інфраструктури та ключові метрики продуктивності веб-застосунку. Комплексні компоненти безпеки глибоко інтегровані у всі функціональні рівні сучасної архітектури веб-застосунку. Надійні системи аутентифікації ретельно перевіряють цифрову особистість користувачів та забезпечують безпечний контрольований доступ до захищених ресурсів. Багатофункціональні механізми авторизації здійснюють гнучкий контроль прав доступу до різних компонентів функціональності системи. Сучасні алгоритми шифрування надійно захищають конфіденційні користувацькі дані під час їх передачі мережею та довготривалого зберігання. Спеціалізовані системи аудиту детально відстежують та протоколюють всі критично важливі операції в системі. Потужний Web Application Firewall забезпечує комплексний захист від різноманітних веб-атак та гарантує належний рівень безпеки на рівні мережевого протоколу.

Системи кешування та оптимізації продуктивності забезпечують швидку роботу веб-застосунків. Кеш браузера зберігає статичні ресурси для швидкого завантаження сторінок. CDN розподіляє контент ближче до кінцевих користувачів. Серверне кешування оптимізує доступ до даних та зменшує

навантаження на бази даних. Вдосконалені механізми стиснення суттєво зменшують загальний обсяг даних, що передаються між компонентами системи, оптимізуючи використання мережевих ресурсів та прискорюючи час відгуку застосунку. Інтелектуальні системи балансування навантаження забезпечують оптимальний розподіл вхідних користувацьких запитів між множиною доступних серверів для досягнення максимальної продуктивності та стабільності роботи системи. Компоненти для аналітики та моніторингу забезпечують збір та аналіз даних про роботу веб-застосунку. Комплексні системи логування безперервно зберігають та аналізують детальну інформацію про всі важливі події та виникаючі помилки в роботі застосунку. Сучасні APM-рішення забезпечують глибокий моніторинг продуктивності та автоматично виявляють потенційні вузькі місця у функціонуванні системи. Спеціалізовані системи збору метрик постійно накопичують важливу статистичну інформацію про використання різноманітних системних ресурсів. Інтерактивні інструменти візуалізації даних надають зручні можливості для ефективного аналізу прихованих трендів та виявлення важливих патернів у роботі системи. Надійні системи сповіщень оперативно інформують відповідальних спеціалістів про виникнення критичних подій та серйозних проблем у роботі застосунку. Сучасна архітектура веб-застосунків включає потужні компоненти для ефективної роботи з різноманітним медіа-контентом. Високопродуктивні системи обробки зображень виконують комплексну оптимізацію графічного контенту для коректного відображення на різних типах пристроїв. Спеціалізовані стримінгові сервіси забезпечують безперебійну передачу якісного аудіо та відео контенту кінцевим користувачам. Гнучкі системи конвертації форматів ефективно адаптують різноманітні медіа-файли для використання на різних цільових платформах. Оптимізовані механізми кешування значно прискорюють процеси доставки медіа-контенту до кінцевих користувачів. Глобальні мережі доставки контенту (CDN) гарантують максимально швидкий доступ до медіа-файлів для користувачів по всьому світу. Комплексні інтеграційні компоненти забезпечують надійну взаємодію

веб-застосунку з численними зовнішніми системами та сервісами. Централізований API Gateway ефективно керує доступом та викликами до зовнішніх програмних інтерфейсів. Масштабовані системи черг повідомлень реалізують надійну асинхронну комунікацію між різними компонентами системи. Універсальні адаптери інтеграції забезпечують стандартизовану взаємодію з різноманітними протоколами та форматами даних. Інтелектуальні системи трансформації виконують коректну конвертацію інформації між різними форматами даних. Вдосконалені механізми повторних спроб (retry) гарантують максимальну надійність всіх інтеграційних взаємодій у системі.

Компоненти для роботи з геоданими забезпечують функціональність, пов’язану з локацією та картографією. Сервіси геокодування перетворюють адреси в координати та навпаки. Високоточні системи маршрутизації виконують складні обчислення для визначення оптимальних маршрутів між різними географічними точками з урахуванням багатьох факторів. Інтерактивні компоненти візуалізації забезпечують зручне відображення динамічних карт та різноманітних геопросторових даних для користувачів. Спеціалізовані сервіси геопошуку надають розширені можливості для ефективного пошуку різноманітних об’єктів за складними географічними критеріями. Оптимізовані системи кешування суттєво прискорюють доступ до часто використовуваних картографічних даних та зменшують навантаження на сервери. Комплексні компоненти для роботи з користувацьким контентом забезпечують зручне керування процесами створення та ефективної модерації різноманітної інформації від користувачів системи. Безпечні системи завантаження файлів гарантують надійне збереження та швидкий доступ до різноманітних користувацьких матеріалів. Інтелектуальні механізми модерації автоматично фільтрують небажаний та потенційно шкідливий контент для підтримки якості системи. Надійні системи версіонування детально зберігають повну історію змін користувацького контенту для можливості відстеження та відновлення даних. Гнучкі компоненти категоризації забезпечують зручну організацію великих обсягів контенту за різноманітними темами та інформативними тегами.

Високопродуктивні пошукові системи надають користувачам можливість максимально швидкого доступу до необхідного користувацького контенту за різними критеріями пошуку.

Архітектурні компоненти для масштабування забезпечують здатність системи обробляти зростаюче навантаження. Інтелектуальні системи балансування ефективно розподіляють вхідні користувацькі запити між доступними серверами для оптимального використання обчислювальних ресурсів. Гнучкі механізми автоматичного масштабування динамічно додають необхідні обчислювальні ресурси при виявленні зростання навантаження на систему. Високопродуктивні системи шардингу забезпечують оптимальний розподіл великих обсягів даних між декількома незалежними базами даних для підвищення продуктивності. Надійні компоненти реплікації гарантують безперебійну роботу та повну відмовостійкість системи шляхом створення синхронізованих копій даних. Інтелектуальні механізми деградації функціональності автоматично адаптують роботу системи для збереження її базової працездатності при виникненні критичних пікових навантажень.

1.3. Особливості проектування систем електронної комерції

Проектування систем електронної комерції базується на глибокому розумінні бізнес-процесів та потреб користувачів. Архітектура таких систем повинна забезпечувати надійну обробку транзакцій та безпеку даних користувачів. Каталог товарів та система управління замовленнями формують основу функціональності e-commerce платформи. Інтеграція з платіжними системами та логістичними сервісами розширює можливості платформи. Системи аналітики та відстеження поведінки користувачів допомагають оптимізувати продажі. Масштабованість архітектури дозволяє системі справлятися зі зростаючим навантаженням та сезонними піками активності [**Error! Reference source not found.**].

Таблиця 1.2 Ключові елементи системи електронної комерції

Компонент	Функціональність	Критичні вимоги
Каталог товарів	Управління асортиментом, категоризація, пошук	Швидкий пошук, актуальність даних
Система замовлень	Обробка замовлень, статуси, історія	Надійність, відмовостійкість
Платіжна система	Обробка оплат, повернення коштів	Безпека, відповідність стандартам
Логістика	Доставка, трекінг, управління запасами	Точність, швидкість обробки
Аналітика	Збір метрик, звіти, прогнозування	Актуальність даних, масштабовані

Джерело: сформовано автором

Користувацький інтерфейс відіграє ключову роль у конверсії відвідувачів у покупців. Зручна навігація та пошук допомагають користувачам швидко знаходити потрібні товари. Детальні картки товарів з фотографіями та описами надають всю необхідну інформацію для прийняття рішення про покупку. Система фільтрації та сортування спрощує вибір товарів за різними параметрами. Оптимізований процес оформлення замовлень забезпечує максимально простий та інтуїтивно зрозумілий шлях від вибору товару до успішного завершення покупки на торговельній платформі. Професійно розроблений адаптивний дизайн гарантує комфортне використання всіх функцій платформи на різноманітних пристроях, від настільних комп’ютерів до мобільних телефонів, автоматично підлаштовуючись під розмір екрану та особливості взаємодії [**Error! Reference source not found.**].

Управління товарним асортиментом вимагає розробки складної системи категорій та атрибутів. Кожен товар може мати множину характеристик та варіантів виконання. Система повинна підтримувати різні типи цін та знижок для різних груп користувачів. Контроль залишків та автоматичне оновлення наявності товарів запобігає продажу відсутніх позицій. Механізми імпорту та експорту даних спрощують роботу з великими каталогами товарів. Підтримка різних валют та мов розширює географію продажів. Система обробки

замовлень є критичним компонентом e-commerce платформи. Кожне замовлення проходить через декілька статусів від створення до доставки. Інтеграція з складською системою дозволяє автоматично резервувати товари під замовлення. Комплексна система повідомлень оперативно інформує користувачів про всі зміни у статусі їхніх замовлень через різні канали комунікації [**Error! Reference source not found.**]. Ефективні механізми повернення та обміну товарів забезпечують швидку обробку всіх типів післяпродажних операцій відповідно до встановлених правил магазину. Повна історія замовлень та інтерактивна система відгуків від покупців формують надійну репутацію магазину та допомагають майбутнім клієнтам приймати обґрунтовані рішення щодо покупок.

Безпека платформи електронної комерції охоплює захист персональних даних та платіжних операцій. Шифрування даних забезпечує конфіденційність інформації при передачі та зберіганні. Системи виявлення шахрайства аналізують транзакції на предмет підозрілої активності. Надійна система багатофакторної аутентифікації забезпечує комплексний захист облікових записів користувачів від несанкціонованого доступу. Регулярні професійні аудити безпеки допомагають оперативно виявляти та ефективно усувати потенційні вразливості в системі захисту. Отримана PCI DSS сертифікація офіційно підтверджує повну відповідність платформи міжнародним стандартам безпеки обробки платіжних карток. Інтегровані маркетингові інструменти забезпечують ефективне залучення нових та утримання існуючих клієнтів через різні канали комунікації. Розумна система персоналізованих рекомендацій автоматично пропонує релевантні товари на основі детального аналізу історії переглядів та попередніх покупок користувача. Гнучкі програми лояльності активно заохочують клієнтів до повторних покупок через привабливу систему накопичувальних бонусів та персональних знижок. Таргетований email-маркетинг своєчасно інформує цільову аудиторію про появу нових товарів та спеціальні акційні пропозиції. Систематичне A/B тестування допомагає постійно оптимізувати показники конверсії та покращувати загальний

користувацький досвід на платформі. Глибока інтеграція з популярними соціальними мережами суттєво розширює доступні канали просування товарів та послуг.

Аналітичні системи забезпечують збір та аналіз даних про поведінку користувачів та ефективність продажів. Відстеження воронки продажів допомагає виявляти проблемні місця в процесі покупки. Глибокий аналіз покинутих кошиків надає цінні дані для розробки ефективних стратегій повернення потенційних клієнтів до завершення покупки. Розумна сегментація користувачів забезпечує можливість створення високоефективних персоналізованих маркетингових кампаній для різних груп клієнтів. Передове прогнозування попиту дозволяє суттєво оптимізувати процеси управління товарними запасами та мінімізувати витрати. Комплексні системи бізнес-аналітики надають потужні інструменти для прийняття обґрунтованих стратегічних рішень на основі реальних даних. Оптимізовані логістичні компоненти забезпечують максимально ефективну доставку товарів до кінцевих покупців. Глибока інтеграція з провідними службами доставки дозволяє миттєво розраховувати точну вартість та очікувані терміни доставки замовлень. Інтерактивна система відстеження посилок надає користувачам актуальну інформацію про поточний статус доставки їхніх замовлень. Інтелектуальна оптимізація маршрутів доставки допомагає істотно зменшити загальні витрати на логістичні операції. Автоматизоване управління складськими запасами гарантує своєчасне поповнення популярних товарів та оптимальний рівень складських залишків. Комплексна автоматизація процесів комплектації та відправки значно прискорює обробку клієнтських замовлень на всіх етапах.

Система підтримки клієнтів включає різні канали комунікації з покупцями. Онлайн-чат забезпечує миттєву допомогу користувачам під час покупки. Продуманна система тікетів забезпечує ефективну організацію та швидку обробку всіх клієнтських звернень службою підтримки. Розширена база знань містить детальні відповіді на найбільш поширені питання користувачів

платформи. Інтегрована система збору та аналізу відгуків допомагає постійно покращувати якість обслуговування клієнтів [**Error! Reference source not found.**]. Інтелектуальні автоматизовані відповіді на типові користувацькі запити суттєво зменшують навантаження на операторів підтримки. Потужні інтеграційні компоненти забезпечують надійне об'єднання e-commerce платформи з іншими важливими бізнес-системами компанії. Глибока інтеграція з корпоративною ERP системою гарантує безперебійну синхронізацію актуальних даних про товари та клієнтські замовлення. Комплексна CRM система ефективно накопичує та аналізує детальну інформацію про клієнтів та їх взаємодію з онлайн-магазином. Спеціалізовані бухгалтерські системи повністю автоматизують процеси фінансового обліку та формування обов'язкової звітності. Розширена інтеграція з популярними маркетплейсами значно збільшує доступні канали продажів та розширює клієнтську базу. Відкритий API платформи надає розробникам зручні можливості для створення мобільних додатків та інших спеціалізованих клієнтських застосунків.

Для забезпечення конкурентоспроможності системи електронної комерції необхідно інтегрувати платформи веб-аналітики, такі як Google Analytics або Hotjar. Це дозволяє відстежувати поведінку користувачів, аналізувати ефективність рекламних кампаній і виявляти вузькі місця в процесах продажів. Збір даних про покинуті кошики або найпопулярніші категорії товарів сприяє розробці стратегій підвищення конверсії.

Застосування алгоритмів машинного навчання дозволяє персоналізувати контент та рекомендації. Наприклад, системи рекомендацій аналізують історію покупок та переглядів для пропонування релевантних товарів. Персоналізовані email-розсилки також сприяють утриманню клієнтів та збільшенню повторних продажів.

Таблиця 1.3 Компоненти користувацького досвіду в e-commerce системах

Елемент UX	**Реалізація**	**Вплив на конверсію**
Навігація	Меню, фільтри, пошук	Швидкий доступ до товарів
Картка товару	Фото, опис, характеристики	Повнота інформації для

Елемент UX	Реалізація	Вплив на конверсію
		прийняття рішення
Кошик	Управління замовленням, розрахунок вартості	Зручність оформлення покупки
Оплата	Різні платіжні методи, безпека	Довіра користувачів
Підтримка	Чат, FAQ, зворотній зв’язок	Допомога у вирішенні проблем
Персоналізація	Рекомендації, історія покупок	Підвищення лояльності

Джерело: сформовано автором

Висока швидкість завантаження сторінок є критично важливим фактором для зручності користувачів. Використання кешування, CDN (Content Delivery Network) та оптимізації зображень допомагає мінімізувати час відгуку системи. Окрім цього, правильне налаштування серверного середовища знижує навантаження на бази даних під час пікової активності.

Сучасні e-commerce системи повинні підтримувати взаємодію з користувачами через різні канали: веб-сайт, мобільний додаток, соціальні мережі, email та фізичні магазини. Синхронізація даних про запаси, замовлення та клієнтів між усіма платформами дозволяє створити цілісний користувацький досвід.

1.4. Технології онлайн-платежів та управління базами даних

Сучасні технології онлайн-платежів базуються на складній інфраструктурі, що забезпечує безпечну обробку фінансових транзакцій. Платіжні шлюзи виступають посередниками між магазином та банківськими системами, обробляючи авторизацію карток та проведення платежів. Токенізація забезпечує безпечне зберігання платіжних даних, замінюючи реальні номери карток унікальними ідентифікаторами. Системи 3D Secure додають додатковий рівень захисту при онлайн-платежах. Моніторинг транзакцій в реальному часі допомагає виявляти підозрілу активність.

Шифрування даних на всіх етапах забезпечує конфіденційність платіжної інформації. Управління базами даних у системах електронної комерції вимагає особливого підходу через високі навантаження та вимоги до надійності. Реляційні бази даних забезпечують цілісність та узгодженість даних про замовлення та транзакції. Розподілені бази даних дозволяють масштабувати систему горизонтально для обробки зростаючого навантаження. Системи реплікації забезпечують високу доступність даних та захист від відмов [**Error! Reference source not found.**]. Механізми резервного копіювання гарантують збереження критично важливої інформації. Оптимізація запитів та індексування покращують продуктивність системи.

Альтернативні методи оплати розширюють можливості проведення платежів у системах електронної комерції. Електронні гаманці спрощують процес оплати, зберігаючи платіжні дані користувачів. Системи мобільних платежів дозволяють проводити транзакції за допомогою смартфонів. Криптовалютні платежі надають додаткові опції для міжнародних транзакцій. Сучасні платіжні системи на основі QR-кодів стрімко набувають широкої популярності в різних регіонах світу завдяки своїй простоті, швидкості та зручності здійснення безконтактних платежів через мобільні пристрої. Інтеграція з локальними платіжними методами розширює географію продажів.

Обробка платіжних транзакцій вимагає врахування різних сценаріїв та станів. Надійні системи управління транзакціями гарантують повну атомарність та узгодженість всіх фінансових операцій у платформі електронної комерції. Ефективні механізми відкату транзакцій забезпечують можливість швидкого скасування помилкових операцій та відновлення попереднього стану даних. Розподілені системи транзакцій координують складні операції між різними компонентами платформи для збереження цілісності даних. Детальне журналювання транзакцій надає можливість проведення повного аудиту та швидкого відновлення даних у разі збоїв. Оптимізована асинхронна обробка платежів значно покращує загальну масштабованість платіжної системи. Комплексна безпека баз даних є критично важливим аспектом у сучасних

системах електронної комерції. Багаторівневі системи контролю доступу чітко обмежують можливості користувачів та застосунків при роботі з даними. Надійне шифрування даних забезпечує надійний захист конфіденційної інформації від будь-якого несанкціонованого доступу. Постійний аудит операцій дозволяє детально відстежувати всі зміни в базі даних для забезпечення прозорості. Інтелектуальні системи виявлення вторгнень забезпечують активний захист від різноманітних зовнішніх атак. Регулярні професійні оцінки безпеки допомагають своєчасно виявляти та оперативно усувати потенційні вразливості в системі.

Інтеграція платіжних систем вимагає дотримання міжнародних стандартів та протоколів. Міжнародний стандарт PCI DSS встановлює суворі вимоги до безпечної обробки платіжних карток в електронній комерції. Надійні протоколи OAuth та OpenID Connect забезпечують високозахищену автентифікацію користувачів у системі. Стандартизовані API платіжних систем гарантують уніфіковану взаємодію з різними платіжними провайдерами. Системи ефективного управління версіями API забезпечують повну сумісність з різними клієнтськими додатками. Постійний моніторинг API дозволяє оперативно відстежувати якість надання сервісів. Комплексна оптимізація продуктивності баз даних включає впровадження різноманітних стратегій та передових технологій. Інтелектуальне кешування часто використовуваних даних суттєво зменшує загальне навантаження на базу даних. Ефективне партиціонування великих таблиць значно покращує швидкість виконання запитів до масштабних наборів даних. Професійне налаштування конфігурації серверів баз даних оптимізує раціональне використання системних ресурсів. Глибокий аналіз планів виконання запитів допомагає швидко виявляти потенційні проблеми з продуктивністю. Безперервний моніторинг ключових метрик продуктивності забезпечує своєчасне виявлення та усунення можливих проблем. Ефективне управління грошовими потоками вимагає максимально точного обліку та регулярної звітності. Автоматизовані системи розрахунків оптимізують складні процеси виплат продавцям та повернень коштів покупцям. Глибока інтеграція з

сучасними бухгалтерськими системами гарантує абсолютно точний фінансовий облік всіх операцій. Автоматична генерація детальних фінансових звітів надає цінну інформацію для комплексного бізнес-аналізу. Інтелектуальні системи узгодження платежів оперативно виявляють будь-які розбіжності в транзакціях. Повна автоматизація податкових розрахунків значно спрощує процес дотримання вимог законодавства. Сучасні NoSQL бази даних ефективно доповнюють традиційні реляційні системи у специфічних сценаріях використання. Оптимізовані документоорієнтовані бази даних забезпечують ефективне зберігання складних структур даних. Спеціалізовані графові бази даних максимально оптимізують роботу з сильно взаємопов’язаними даними. Високопродуктивні бази даних типу "ключ-значення" гарантують надшвидкий доступ до даних кешування. Колоночні бази даних забезпечують оптимальну продуктивність при виконанні складних аналітичних запитів. Гнучкі мультимодельні бази даних ефективно комбінують різні підходи до організації зберігання даних. Комплексні системи обробки повернень та відшкодувань вимагають підвищеної уваги до безпеки та регулярного аудиту операцій. Надійні механізми верифікації повернень ефективно запобігають виконанню шахрайських операцій. Повна автоматизація процесів повернення коштів значно прискорює обробку клієнтських запитів. Глибока інтеграція з платіжними системами забезпечує максимально швидке проведення всіх відшкодувань. Оперативні системи сповіщень своєчасно інформують клієнтів про поточний статус їх повернень. Регулярний аудит операцій повернення гарантує повну прозорість всіх процесів для всіх учасників. Оптимальний географічний розподіл баз даних суттєво покращує загальну доступність та продуктивність системи. Надійні системи геореплікації забезпечують безперебійну синхронізацію даних між різними географічними регіонами. Інтелектуальні механізми маршрутизації запитів автоматично направляють користувачів до найближчих доступних серверів. Комплексні стратегії відновлення після збоїв гарантують повну безперервність роботи системи. Постійний моніторинг мережевих затримок допомагає оптимізувати загальну

продуктивність системи. Гнучкі системи балансування навантаження ефективно розподіляють вхідні запити між доступними серверами. Глибока аналітика платіжних операцій надає надзвичайно цінну інформацію для розвитку бізнесу. Інтелектуальні системи виявлення шахрайства постійно аналізують патерни транзакцій для виявлення підозрілої активності. Точне прогнозування грошових потоків допомагає ефективно планувати фінансову діяльність компанії. Детальний аналіз конверсії платежів оперативно виявляє проблемні місця в процесі оплати. Розумна сегментація клієнтів за платіжною поведінкою значно покращує ефективність маркетингового таргетингу. Інтерактивні системи візуалізації даних суттєво спрощують процес аналізу великих обсягів транзакцій. Складна міграція даних та оновлення систем вимагають надзвичайно ретельного планування та виконання. Продумані стратегії міграції гарантують повну безперервність роботи сервісів під час масштабних оновлень. Спеціалізовані інструменти версіонування схем баз даних забезпечують надійний контроль над змінами структури даних. Ефективні процедури відкату дозволяють швидко повернутися до попереднього стабільного стану у разі виникнення проблем. Комплексне тестування міграцій на повних копіях даних значно зменшує потенційні ризики. Повна автоматизація процесів міграції мінімізує можливість виникнення людських помилок. Глибока інтеграція з зовнішніми фінансовими системами істотно розширює функціональні можливості платформи. Надійне підключення до захищених банківських API повністю автоматизує процеси проведення платежів. Комплексна інтеграція з системами еквайрингу забезпечує зручний прийом платежів у фізичних точках продажу. Глобальні системи міжнародних переказів значно розширюють географічні можливості платіжної системи. Вбудована підтримка різних валют та автоматична конвертація суттєво спрощують ведення міжнародної торгівлі. Ефективна інтеграція з системами факторингу помітно покращує процеси управління грошовими потоками. Повна відповідність суворим регуляторним вимогам є критично важливим аспектом сучасних платіжних систем [**Error! Reference source not found.**].

Комплексні системи KYC та AML забезпечують ретельну перевірку клієнтів та постійний моніторинг транзакцій. Надійне зберігання документації повністю відповідає всім вимогам регуляторів щодо термінів зберігання даних. Автоматизована генерація регуляторної звітності значно спрощує процеси звітування перед контролюючими органами. Потужні системи контролю змін забезпечують детальний аудит всіх модифікацій в системі. Регулярні комплексні перевірки відповідності допомагають підтримувати системи в актуальному стані згідно з вимогами регуляторів.

Висновки до розділу 1

У першому розділі було розглянуто теоретичні основи веб-технологій та електронної комерції, що є фундаментом для подальшої розробки веб-застосунку. Проаналізовано сучасні підходи до побудови архітектури веб-додатків, особливості їхньої реалізації та ключові компоненти, що забезпечують стабільну, безпечну і масштабовану роботу системи. Значну увагу приділено інструментам front-end і back-end розробки, а також технологіям, які використовуються для інтеграції, управління даними і реалізації бізнес-логіки.

Також було визначено специфіку проєктування систем електронної комерції, враховуючи вимоги до зручності інтерфейсу, безпеки платіжних операцій, ефективного управління товарним асортиментом і замовленнями. Розглянуто основні підходи до реалізації користувацького досвіду, що безпосередньо впливають на конверсію, та технології, які забезпечують високу продуктивність і надійність електронних платформ.

Отже, сучасні веб-технології надають широкі можливості для створення ефективних та конкурентоспроможних e-commerce рішень. Отримані теоретичні знання стануть основою для практичної реалізації веб-застосунку в наступних розділах роботи.

РОЗДІЛ 2
АНАЛІЗ РИНКУ ВЕБ-ЗАСТОСУНКІВ ДЛЯ ЕЛЕКТРОННОЇ КОМЕРЦІЇ ТА ОБҐРУНТУВАННЯ ТЕХНОЛОГІЙ РОЗРОБКИ ВЕБ-ЗАСТОСУНКУ

2.1. Аналіз вимог до веб-застосунку для продажу одягу та взуття

Проектування веб-застосунку для продажу одягу та взуття потребує детального аналізу вимог, щоб забезпечити його ефективність, зручність для користувачів і відповідність сучасним стандартам. Розглянемо ключові аспекти, що мають бути враховані під час розробки такого застосунку, використовуючи стек технологій Django, Python, HTML, CSS і базу даних PostgreSQL.

Каталог товарів є основним елементом функціоналу, що потребує чіткої структури. Користувачі повинні мати можливість переглядати товари за категоріями, такими як "чоловічий одяг", "жіночий одяг" та "взуття". Додатково, необхідно впровадити функції фільтрування за цінами та акціями, що значно полегшує процес вибору. Функція пошуку є також важливою для швидкого доступу до потрібної продукції.

Кошик для покупок повинен дозволяти додавання товарів із можливістю їх редагування або видалення. Веб-застосунок повинен автоматично підраховувати загальну вартість товарів, враховуючи акції або знижки. Інтерфейс має бути простим і зрозумілим навіть для новачків. Процес оформлення замовлення повинен включати кілька етапів: введення контактної інформації, вибір способу доставки та оплати. Для зручності користувачів доцільно передбачити збереження цих даних у профілі користувача, що скоротить час оформлення повторних замовлень.

Реєстрація та авторизація користувачів є важливими для персоналізації сервісу. Користувачі повинні мати змогу створювати особисті кабінети, де зберігатимуться історія покупок, обрані товари та налаштування. Реалізація цієї функції за допомогою вбудованих механізмів фреймворку Django у поєднанні з базою даних PostgreSQL забезпечує створення надійного, динамічного та зручного інтерфейсу.

Для сайту повинна бути реалізована система онлайн-оплати. Інтеграція з популярними платіжними системами, такими як PayPal, Stripe або локальні сервіси, забезпечить зручність і безпеку платежів. Дані про транзакції повинні зберігатися у базі даних PostgreSQL для подальшої обробки та звітності. Інтерфейс веб-застосунку повинен бути адаптивним, щоб забезпечити коректне відображення на різних пристроях, включаючи смартфони, планшети та настільні комп'ютери. Використання CSS медіа-запитів і фреймворків, таких як Bootstrap, дозволить досягти цієї мети [**Error! Reference source not found.**].

У цьому контексті важливо забезпечити правильне відображення всіх даних про товар. Кожна товарна позиція повинна містити вичерпну інформацію, включаючи фотографії високої якості, детальні описи, характеристики, а також інформацію про наявність на складі і можливі варіанти доставки. Багатофункціональність і наочність таких елементів підвищують рівень довіри користувачів і дають їм можливість більш точно оцінити продукт.

Також є необхідною інтеграція з соціальними мережами для залучення трафіку та створення маркетингових кампаній. Користувачі можуть мати можливість реєструватися або входити в особистий кабінет через акаунти в соціальних мережах, що спрощує процес входу і заохочує більше людей стати користувачами веб-застосунку. Інтеграція з платформами на зразок Facebook, Instagram або Google дозволяє не лише полегшити процес реєстрації, але й стимулювати соціальні взаємодії, що підвищує рівень залученості користувачів. У той же час, маркетингові інструменти мають відігравати важливу роль у просуванні веб-застосунку. Впровадження системи акцій, спеціальних пропозицій і знижок може значно підвищити лояльність користувачів. Наприклад, система бонусних балів, яку можна накопичувати за покупки, а потім обмінювати на знижки чи безкоштовну доставку, є ефективним інструментом для збільшення частоти покупок. Також, важливо забезпечити можливість персоналізувати пропозиції на основі історії покупок або вподобань користувача.

Адаптивний дизайн є невід'ємною частиною сучасних веб-застосунків, оскільки користувачі все більше взаємодіють з ними через мобільні пристрої. Використання медіа-запитів у CSS дозволяє створювати такі макети, які будуть коректно відображатися на різних розмірах екранів, від маленьких смартфонів до великих моніторів. Крім того, необхідно враховувати різні орієнтації екранів, оскільки користувачі можуть переходити між портретним і ландшафтним режимами на мобільних пристроях.

Процес оформлення замовлення має бути не тільки простим, але й безпечним. Веб-застосунок повинен використовувати останні стандарти шифрування для захисту даних про користувачів, особливо при передачі платіжної інформації. Це забезпечить не тільки конфіденційність, а й запобігання різного роду атакам, таким як крадіжка даних або шахрайство з картковими реквізитами.

Веб-застосунок для продажу одягу та взуття також має бути інтегрований з системами аналітики. Наприклад, інструменти, такі як Google Analytics, можуть допомогти відслідковувати конверсії, середній час перебування на сайті і кількість відмов, що дозволяє своєчасно реагувати на зміни в попиті та покращувати користувацький досвід.

Масштабованість системи є важливою для підтримки зростання кількості користувачів і товарів. Архітектура бази даних PostgreSQL повинна передбачати розширення обсягів даних без втрати продуктивності. Для цього можна використовувати індексацію та оптимізацію таблиць.

Таблиця 2.1 Порівняльний аналіз технологій для розробки веб-застосунку

Категорія	Технологія	Сфера застосування	Переваги	Недоліки
Front-end	HTML5	Структура веб-сторінок	Семантичність, кросбраузерність	Обмежена динамічність
	CSS3	Оформлення, адаптивність	Медіа-запити, адаптивні макети	Вимагає продуманої структури

Категорія	Технологія	Сфера застосування	Переваги	Недоліки
	React	SPA, інтерфейси	Компонентний підхід, швидкість	Високий поріг входу
Back-end	Python	Серверна логіка	Простота, читабельність коду	Менш ефективний для heavy-load
	Django	Швидка веб-розробка	Вбудовані засоби, безпека	Менше гнучкості, ніж у мікрофреймворках
	Node.js	JavaScript-сервер	Асинхронність, одна мова для front/back	Вимоги до оптимізації
Бази даних	PostgreSQL	Реляційні дані, аналітика	Транзакції, JSONB, надійність	Складніша конфігурація
	MongoDB	NoSQL дані	Гнучка структура, масштабованість	Відсутність строгих зв’язків

Джерело: складено автором

Розробка веб-застосунку для продажу одягу та взуття вимагає ретельно підібраного набору технологій, що забезпечить оптимальне поєднання функціональності, зручності використання, продуктивності та безпеки. Обраний стек Django, Python, HTML, CSS і PostgreSQL дозволяє створити сучасне, гнучке та масштабоване рішення, яке відповідає вимогам сучасного ринку електронної комерції.

– Django є потужним фреймворком для веб-розробки, створеним на мові Python. Його головна мета – прискорення процесу створення веб-застосунків, зосереджуючись на зручності, безпеці та масштабованості. Django включає в себе безліч вбудованих функцій, таких як панель адміністратора, маршрутизація, шаблонізатор, ORM (система керування базами даних), що дозволяє розробникам зосередитися на логіці застосунку, а не на

низькорівневих деталях. Завдяки принципу «DRY» (Don’t Repeat Yourself) та модульній структурі, Django забезпечує ефективність розробки та легкість у підтримці коду. Фреймворк ідеально підходить для реалізації як простих сайтів, так і складних веб-платформ [**Error! Reference source not found.**, **Error! Reference source not found.**].

– Python є сучасною мовою програмування загального призначення, яка вирізняється простою та зрозумілою синтаксичною структурою. Вона активно використовується у веб-розробці, автоматизації, аналізі даних, машинному навчанні та інших галузях. Python є ідеальним вибором для створення веб-застосунків завдяки своїй гнучкості та великій кількості бібліотек. Його читаємий код сприяє швидкому розумінню логіки програми та зменшенню помилок, що робить Python популярним серед розробників різного рівня. У поєднанні з фреймворком Django ця мова дозволяє створювати масштабовані й безпечні веб-рішення [**Error! Reference source not found.**].

– HTML є основою веб-розробки, його використовують для створення структури веб-сторінок. Це мова розмітки, яка дозволяє позначати і описувати елементи веб-сторінки, їхнє розташування та взаємодію. Веб-сторінки без HTML були б просто набором необроблених текстових файлів або зображень без чіткої структури. HTML є стандартом, який підтримується всіма веб-браузерами, і дозволяє розробникам створювати функціональні, зручні для користувачів інтерфейси. Усі веб-застосунки, незалежно від складності, базуються на HTML як основному засобі розмітки контенту.

– CSS використовується для оформлення веб-сторінок, що дозволяє задати стилі для елементів HTML [**Error! Reference source not found.**]. За допомогою CSS можна визначати кольори, шрифти, відступи, позиціювання елементів, а також створювати анімації та інші ефекти. CSS забезпечує гнучкість у налаштуванні зовнішнього вигляду веб-застосунків, що дає можливість створювати красиві та адаптивні інтерфейси. Веб-дизайн сьогодні є важливим аспектом успіху застосунків, тому використання CSS є обов’язковим для досягнення гарного вигляду інтерфейсу.

– PostgreSQL є потужною системою управління реляційними базами даних з відкритим вихідним кодом. Вона підтримує широкий спектр функцій, зокрема транзакції, перевірки цілісності даних, складні запити та розширення. PostgreSQL забезпечує надійне зберігання інформації та високу продуктивність, що робить її ідеальним вибором для веб-застосунків з великими обсягами даних [**Error! Reference source not found.**]. Її використання у зв'язці з Django дозволяє ефективно взаємодіяти з базою даних завдяки вбудованому ORM, що спрощує маніпуляції з даними на рівні коду. PostgreSQL вважається однією з найкращих СУБД для побудови сучасних, масштабованих і стабільних веб-сервісів.

Адаптивний дизайн є обов'язковою умовою для забезпечення зручності використання. Він реалізується за допомогою CSS медіа-запитів і сучасних технологій, що гарантує коректне відображення веб-застосунку на пристроях із різними розмірами екранів.

Поєднання цих технологій дозволяє створити веб-застосунок, який відповідає сучасним стандартам у сфері електронної комерції. Використання Django, Python, HTML, CSS і PostgreSQL забезпечує гнучкість розробки, високу продуктивність та конкурентоспроможність продукту, водночас дозволяючи легко підтримувати й розширювати систему в майбутньому.

2.2. Огляд існуючих рішень на ринку електронної комерції

Вивчення ринку електронної комерції показує, що успіх платформи залежить від кількох ключових факторів, таких як інтуїтивність інтерфейсу, швидкість завантаження сторінок, безпека транзакцій та якість обслуговування клієнтів. На сучасному ринку розвиваються нові технології, які надають користувачам ще більшу зручність при покупках онлайн. Ці зміни не лише покращують досвід покупців, але й створюють нові можливості для компаній з надання своїх товарів і послуг [**Error! Reference source not found.**].

Одним з основних аспектів електронної комерції є забезпечення високої швидкості завантаження сайтів. Швидкість є критичним фактором у збереженні

користувачів і в успішному завершенні покупок. Якщо веб-сайт завантажується занадто довго, це може призвести до втрати потенційних клієнтів, які не будуть готові чекати. Тому важливо оптимізувати контент і використовувати сучасні методи кешування, такі як CDN, для швидшого доступу до сторінок, а також зменшення розміру файлів для мобільних версій сайтів. У відповідь на зростаючі вимоги покупців до персоналізації, багато платформ почали використовувати технології штучного інтелекту для прогнозування і рекомендацій. Штучний інтелект дозволяє аналізувати поведінку користувачів і на основі цього формувати персоналізовані рекомендації. Такі системи не лише покращують досвід покупок, але й підвищують лояльність клієнтів, оскільки вони відчувають, що їх інтереси і переваги враховуються.

Використання мобільних додатків стало невід'ємним аспектом життя людини. Мобільні додатки надають компаніям можливість створити більш персоналізоване і зручне середовище для покупок. Завдяки push-сповіщенням компанії можуть оперативно інформувати користувачів про нові надходження, знижки та акції, що допомагає підвищити рівень продажів. Одним з важливих аспектів є безпека транзакцій. Покупці повинні бути впевнені, що їхні особисті дані не будуть викрадені або використані не за призначенням. З цією метою сайти електронної комерції використовують сучасні методи шифрування і системи автентифікації, такі як двофакторна аутентифікація. Це дозволяє мінімізувати ризики для клієнтів та забезпечити їм спокій при здійсненні покупок.

Успішні платформи також надають різні методи оплати, щоб задовольнити потреби різних категорій користувачів. Користувачі повинні мати змогу вибирати між традиційними банківськими картами, електронними гаманцями, а також новітніми способами оплати, такими як криптовалюти. Всі ці опції дозволяють підвищити комфорт і забезпечити безпечні транзакції для користувачів. Ще однією важливою тенденцією є інтеграція платіжних і логістичних сервісів безпосередньо в процес покупки. Це дозволяє користувачам отримувати точну інформацію про доступні способи доставки, а

також відстежувати свої замовлення в режимі реального часу. Це значно підвищує задоволеність клієнтів і допомагає утримувати їх на платформі.

Зростання популярності голосових помічників є ще одним цікавим трендом. Голосові помічники дозволяють користувачам здійснювати покупки за допомогою голосових команд, що забезпечує новий рівень зручності. Така інтеграція може значно покращити досвід користувачів, які віддають перевагу зручності та швидкості при виборі товарів.

Важливою частиною стратегії онлайн-продажів є інвестиції в SEO (пошукову оптимізацію) і маркетинг у соціальних мережах. Хороша видимість в пошукових системах забезпечує високий рівень трафіку на сайт, а маркетингові кампанії в соціальних мережах допомагають привертати увагу до бренду. Системи таргетингу, які використовують соціальні мережі, дозволяють точно визначити цільову аудиторію і запропонувати їй релевантні пропозиції. Особливо важливою є оптимізація мобільної версії сайтів для зручного шопінгу на смартфонах і планшетах. Більшість покупок сьогодні здійснюються через мобільні пристрої, і тому мобільна версія сайту має бути не менш зручною та швидкою, ніж десктопна версія. Використання адаптивного дизайну є стандартом для успішних онлайн-платформ.

Згідно з останніми тенденціями, інтернет-магазини все більше орієнтуються на екологічно чисті та стійкі практики. Компанії впроваджують еко-дружні способи упаковки і доставки товарів, а також надають покупцям можливість вибору еко-варіантів товарів, що сприяє збереженню довкілля. Це відповідає на зростаючий інтерес споживачів до сталого розвитку та екологічної відповідальності брендів.

Інтеграція чат-ботів на сайтах для надання цілодобової підтримки клієнтів стала важливою частиною сучасного електронного бізнесу. Чат-боти дозволяють автоматизувати відповіді на часті запитання, сприяючи підвищенню рівня задоволення клієнтів. Вони також можуть допомогти в процесі вибору товару, рекомендаціях або навіть обробці замовлень.

Аналіз даних є ще одним критичним елементом в електронній комерції. Інтернет-магазини активно використовують аналітику для збору даних про поведінку користувачів і для подальшого вдосконалення своїх платформ. Це дозволяє з’ясувати, які товари користуються найбільшим попитом, а також виявити можливі проблеми в процесі покупки, що може допомогти оптимізувати досвід користувачів. Зручність обміну та повернення товарів також є важливим фактором, який визначає успіх платформи. Покупці повинні мати чітке уявлення про процес повернення товару, а також про можливі варіанти обміну. Прозорі правила повернення допомагають підвищити довіру до інтернет-магазину і зменшити кількість незадоволених клієнтів. Системи лояльності, які нагороджують користувачів за частоту покупок або за рекомендації друзям, стають популярними в електронній комерції. Такі програми допомагають не лише утримувати існуючих клієнтів, але й залучати нових. Вони створюють додаткову мотивацію для покупців повертатись на платформу та здійснювати нові покупки.

Технології віртуальної реальності (VR) та доповненої реальності (AR) поступово входять у світ електронної комерції. З їх допомогою покупці можуть більш детально ознайомитись із товаром перед покупкою, що особливо актуально для інтернет-магазинів, які продають одяг або меблі. Ці інновації підвищують рівень залучення і допомагають зробити процес покупки більш інтерактивним.

Ще одним важливим аспектом є розвиток omnichannel-стратегії, що передбачає інтеграцію різних каналів продажу, таких як онлайн-платформи, фізичні магазини та мобільні додатки. Це дозволяє клієнтам здійснювати покупки через будь-який зручний для них канал і отримувати безперебійний досвід у всіх точках контакту з брендом. Зростаюча популярність сервісів підписки є ще одним фактором, що визначає тенденції на ринку електронної комерції. Інтернет-магазини все більше пропонують моделі підписки, які дозволяють користувачам отримувати товари регулярно, що зручно для тих,

хто робить покупки повторно або хоче скористатися спеціальними пропозиціями.

Сучасний ринок електронної комерції багатий на різноманітні платформи, які пропонують покупцям зручність, широкий вибір товарів та інтуїтивний інтерфейс. Для аналізу розглянемо три приклади: офіційний сайт бренду Adidas, інтернет-магазин Answear, а також український ритейлер YesOriginal.

Adidas

Офіційний сайт Adidas є прикладом високоякісного рішення для електронної комерції, що вдало поєднує стиль, функціональність і зручність [**Error! Reference source not found.**].

– Інтерфейс і дизайн

Сайт вирізняється мінімалістичним дизайном, який відповідає фірмовому стилю бренду. Використання великої кількості білого простору, акцентних кольорів і чітких шрифтів забезпечує зручність сприйняття інформації. Адаптивний дизайн дозволяє комфортно використовувати сайт на будь-яких пристроях. Також, основна увага приділяється зручності навігації та швидкому доступу до основних категорій товарів.

– Каталог і навігація

Каталог чітко структурований за категоріями: одяг, взуття, аксесуари, розпродаж. Функціонал фільтрів дозволяє швидко відсіяти товари за ціною, розміром, кольором і навіть матеріалом. Особливістю є рекомендаційна система, що пропонує товари на основі історії переглядів, завдяки чому покращується персоналізація покупок. Розумне сортування товарів забезпечує ефективний перегляд.

– Технічні рішення

Сайт використовує сучасні технології, такі як React для динамічних компонентів і оптимізації продуктивності. Інтеграція з платіжними системами дозволяє здійснювати покупки швидко і безпечно. Використання CDN (Content

Delivery Network) сприяє зменшенню часу завантаження сторінок, що покращує користувацький досвід.

Answear

Answear є популярною платформою, що пропонує великий асортимент товарів відомих брендів [**Error! Reference source not found.**].

– Інтерфейс і структура

Дизайн сайту орієнтований на масового споживача: акцент на яскравих банерах із розпродажами і промоціями, зручна навігація та адаптивний дизайн для мобільних пристроїв. Структура сайту дозволяє користувачам швидко знайти товар за категоріями і сортами, а також порівнювати кілька товарів одночасно, що підвищує ефективність покупок.

– Особливості.

Платформа пропонує багатомовний інтерфейс, що дозволяє залучати покупців із різних країн. Зручний кошик із можливістю порівняння товарів сприяє покращенню користувацького досвіду, дозволяючи приймати більш обґрунтовані рішення при покупках. Ці можливості покращують взаємодію користувача з платформою.

– Технології

Використання AJAX забезпечує динамічне оновлення вмісту сторінки без перезавантаження, що дозволяє значно пришвидшити взаємодію користувача з сайтом. Інтеграція із соціальними мережами та e-mail маркетинг підвищують ефективність продажів, дозволяючи бренду підтримувати контакт з покупцями через різні канали. Це створює інтегровану стратегію взаємодії з аудиторією.

YesOriginal

YesOriginal – це український інтернет-магазин, який спеціалізується на продажу оригінального взуття та аксесуарів [**Error! Reference source not found.**].

– Дизайн і зручність

Дизайн сайту спрямований на локальну аудиторію. Простий інтерфейс із мінімальною кількістю зайвих елементів дозволяє зосередитися на товарах. Сайт має інтуїтивно зрозуміле меню і відповідно адаптовану версію для мобільних пристроїв, що забезпечує безперебійний доступ до магазину на будь-якому пристрої.

– Каталог і акції

Розділи каталогу чітко визначені: чоловічий, жіночий одяг, аксесуари. Акції та знижки відображаються на головній сторінці, що привертає увагу покупців. Сайт постійно оновлюється, щоб користувачі могли отримати найсвіжішу інформацію про нові надходження і сезонні знижки.

– Локалізація і підтримка

Платформа адаптована для українського ринку, включаючи інтеграцію з локальними платіжними системами і службами доставки. Реалізована багатоканальна підтримка клієнтів через телефон, e-mail і месенджери. Це дозволяє покупцям отримати швидку допомогу в разі виникнення запитань щодо замовлення чи товару.

Проаналізувавши три платформи можна зробити невелике порівняння. Adidas зосереджується на глобальному масштабі, пропонуючи інноваційні технічні рішення і максимальну продуктивність. Відзначається висока якість технологій і зручність використання сайту, що забезпечує відмінний досвід для користувачів по всьому світу. Answear є універсальним ритейлером із широким асортиментом і сучасними інтеграціями, підкреслюючи увагу до динамічного контенту та персоналізації. YesOriginal орієнтований на локального споживача, акцентуючи увагу на простоті та доступності, а також на інтеграції з українськими платіжними і кур’єрськими системами, що забезпечує високий рівень зручності для місцевих покупців.

2.3. Обґрунтування вибору інструментів та технологій для розробки веб- застосунку

Розробка веб-застосунку вимагає ретельного вибору інструментів та технологій, оскільки від цього залежить ефективність, надійність і масштабованість майбутнього продукту. Для створення веб-застосунку обґрунтованим є вибір таких технологій, як Django, Python, HTML, CSS та система управління базами даних PostgreSQL.

HTML виступає фундаментом структури веб-застосунку. Це дає змогу організувати зрозумілу, логічну та семантичну розмітку сторінок, яка полегшує роботу з адаптивним дизайном і забезпечує коректне відображення контенту на різних пристроях.

CSS відповідає за візуальну частину та забезпечує створення сучасного дизайну. Завдяки використанню таких технік, як Flexbox і Grid, можливо реалізовувати адаптивні макети з гнучкою структурою. У практиці веб-розробки часто використовуються CSS-фреймворки, зокрема Bootstrap, який надає велику кількість готових стилізованих компонентів. Це дозволяє прискорити розробку інтерфейсу та зосередитись на основній функціональності, зменшуючи обсяг ручного кодування стилів.

Django як фреймворк для розробки веб-застосунків на мові Python забезпечує реалізацію більшості функцій на серверній стороні. Його вбудовані інструменти дозволяють ефективно реалізовувати обробку запитів, маршрутизацію, роботу з базою даних та формування динамічного HTML-контенту без потреби у складних клієнтських скриптах. Це дозволяє зосередитись на надійності та структурованості бізнес-логіки, спрощуючи розробку та супровід веб-застосунку [**Error! Reference source not found.**, **Error! Reference source not found.**].

Для реалізації веб-застосунку доцільним є використання інтегрованого середовища розробки PyCharm. Це середовище значно спрощує процес створення Django-застосунків, забезпечуючи автодоповнення коду, перевірку синтаксису в режимі реального часу, підтримку шаблонів та інструменти для

налагодження. Крім того, PyCharm має інструменти для запуску серверів розробки та управління міграціями бази даних, що значно полегшує процес супроводу та тестування веб-застосунку. Завдяки високому рівню інтеграції з Django, PyCharm дозволяє організувати ефективне середовище для командної та індивідуальної розробки, сприяючи підвищенню якості програмного коду та оптимізації розробницького процесу [**Error! Reference source not found.**].

Для розробки веб-застосунків, що включають збереження та обробку даних, необхідно використовувати систему керування базами даних. PostgreSQL є однією з найпопулярніших реляційних систем керування базами даних (СКБД). Вона є відкритою і широко використовуваною завдяки своїй надійності, високій швидкодії та простоті в управлінні. PostgreSQL ідеально підходить для веб-застосунків середньої та великої складності, оскільки вона підтримує великі обсяги даних і має потужні інструменти для їх обробки та зберігання. Крім того, PostgreSQL має великий набір інтерфейсів для роботи з різними мовами програмування, що дозволяє без зусиль інтегрувати її в проект.

Одним з головних переваг використання PostgreSQL є її здатність працювати з великими обсягами даних. Вона дозволяє створювати та виконувати складні запити до бази даних, що є необхідним для динамічних веб-застосунків, де дані можуть змінюватися в режимі реального часу. База даних PostgreSQL також підтримує механізми забезпечення цілісності даних, що дозволяє уникнути помилок при роботі з даними та забезпечує їх безпеку. Завдяки широкому поширенню PostgreSQL, для цієї технології є велика кількість інструментів для адміністрування, що дозволяє зменшити час на розробку та підтримку системи.

Одним з важливих аспектів, що впливають на вибір технологій для розробки веб-застосунку, є підтримка стандартів. HTML і CSS забезпечують структуру та візуальне оформлення інтерфейсу, гарантуючи коректне відображення у всіх сучасних веб-браузерах і на різних пристроях. Django, як фреймворк на Python, повністю підтримує шаблонізацію HTML та інтеграцію CSS, що дозволяє швидко реалізовувати зручні й адаптивні інтерфейси. Крім

того, ці технології мають велику кількість інструментів для тестування та оптимізації, що дозволяє забезпечити високу якість продукту. У разі використання PostgreSQL для зберігання даних можна бути впевненим у надійності бази та її підтримці в найпоширеніших середовищах хостингу.

Крім того, вибір Django, Python, HTML, CSS і PostgreSQL забезпечує можливість розширення застосунку в майбутньому. Якщо проект вимагає нових функцій, таких як інтеграція з іншими сервісами або додавання складних функцій для обробки даних, ці технології дозволяють легко інтегрувати нові рішення. Наприклад, можна створювати нові моделі в Django, які автоматично формують таблиці у базі даних PostgreSQL, або підключати зовнішні API для обміну даними, що значно розширює функціональність застосунку. Вибір цих технологій також зумовлений великою кількістю документації, спільнот і ресурсів для навчання. Оскільки Django, Python, HTML, CSS та PostgreSQL є відкритими стандартами, для них доступні тисячі статей, відеоуроків, форумів та інших матеріалів, які допомагають розробникам швидко освоїти необхідні інструменти. Завдяки цьому підтримка проекту, вирішення проблем і пошук рішень для виникаючих питань відбуваються значно швидше. Важливою перевагою використання HTML і CSS у складі цього стеку є забезпечення кросплатформності: веб-застосунки працюють коректно на комп’ютерах, планшетах і смартфонах. Це дає змогу досягти максимальної аудиторії користувачів і забезпечити доступ до застосунку з різних типів пристроїв.

Веб-застосунки, розроблені з використанням Django, Python, HTML, CSS та PostgreSQL, відрізняються високою продуктивністю і масштабованістю. Django забезпечує чітке розділення логіки та інтерфейсу, що спрощує розробку складних систем, які можуть обслуговувати велику кількість користувачів одночасно. Python, як гнучка мова програмування, дозволяє реалізовувати складні алгоритми обробки даних, а PostgreSQL – ефективно зберігати та обробляти великі обсяги інформації.

Обраний стек технологій також надає можливості для інтеграції з різними сторонніми сервісами та платформами. Django має потужні інструменти для

підключення API, що дозволяє реалізовувати платіжні системи, авторизацію через соцмережі, аналітику тощо. Це критично важливо для створення сучасних веб-застосунків, які потребують постійної взаємодії з іншими сервісами. У процесі розробки веб-застосунку важливо не лише правильно обрати технології для реалізації, а й розуміти, як ці інструменти будуть взаємодіяти між собою для досягнення бажаних результатів. Стек Django, Python, HTML, CSS та база даних PostgreSQL не лише забезпечують належний рівень функціональності, а й дають змогу створювати зручні та ефективні веб-рішення. У цьому контексті важливо звернути увагу на низку аспектів, що обґрунтовують вибір таких технологій.

Почнемо з того, що HTML є основним інструментом для створення структури веб-застосунку. Це мова розмітки, яка визначає, як виглядатимуть різні елементи сторінки: текст, зображення, посилання, таблиці, форми тощо. Веб-розробники використовують HTML для визначення основної архітектури веб-сторінки, що дозволяє забезпечити логічну і зручну навігацію для користувача. Завдяки цьому розробники можуть працювати над додаванням функцій і адаптацією дизайну без необхідності переписувати код для кожного окремого елементу. CSS, у свою чергу, відіграє важливу роль у відокремленні зовнішнього вигляду веб-сторінки від її структури. Завдяки цьому розробники можуть змінювати кольори, шрифти, розміри, відступи та інші параметри без необхідності змінювати HTML-код. Це дозволяє зручно працювати над оформленням сторінки і робити її максимально адаптивною до різних розмірів екрану та пристроїв. CSS також дозволяє створювати складні макети за допомогою таких інструментів, як Flexbox і CSS Grid, що робить процес верстки набагато ефективнішим і гнучким [**Error! Reference source not found.**].

Django як серверний фреймворк забезпечує повноцінну інтеграцію з PostgreSQL, автоматично створює структуру бази даних на основі описаних у коді моделей, а також дозволяє реалізовувати операції з даними без необхідності вручну писати SQL-запити. Це значно прискорює процес розробки та знижує ризик помилок.

Важливою складовою частиною веб-застосунку є також база даних, і PostgreSQL є одним з найпопулярніших рішень для роботи з реляційними базами даних. PostgreSQL забезпечує надійне зберігання, обробку та отримання даних, які можуть бути пов’язані між собою зовнішніми ключами. Це забезпечує зручний спосіб управління великими обсягами структурованої інформації. Завдяки PostgreSQL можна легко реалізувати функції, пов’язані з користувачами, авторизацією, транзакціями, товарами, замовленнями та іншими аспектами функціонування веб-застосунку. Вибір саме PostgreSQL для роботи з базами даних обумовлений також її надійністю та підтримкою транзакцій. Для веб-застосунків, де важливо зберігати цілісність даних, PostgreSQL є оптимальним варіантом. Завдяки механізмам ACID (атомарність, узгодженість, ізоляція, довговічність) можна бути впевненим у тому, що навіть у випадку збою системи всі зміни в базі даних будуть гарантовано виконані або скасовані.

Окрім того, PostgreSQL є дуже ефективною при роботі з великими обсягами даних. Вона підтримує індексацію, що дозволяє пришвидшити пошук та вибірку даних із великих таблиць. Це є важливим аспектом для веб-застосунків, де потрібна швидка обробка запитів до бази даних для забезпечення високої продуктивності. PostgreSQL також забезпечує можливість реплікації, що дозволяє організувати резервне копіювання і масштабування бази даних.

Однією з ключових особливостей Django є вбудована система ORM (Object-Relational Mapping). ORM дозволяє працювати з базою даних за допомогою об’єктно-орієнтованого підходу, замінюючи необхідність написання складних SQL-запитів. Це спрощує роботу з базою даних і знижує ймовірність помилок у коді. PostgreSQL, як реляційна система управління базами даних, забезпечує надійне зберігання та обробку великих обсягів структурованої інформації, підтримуючи транзакції та забезпечуючи цілісність даних.

З точки зору продуктивності та безпеки, PostgreSQL також має ряд важливих переваг. Вона підтримує різні методи шифрування даних і має вбудовані механізми для забезпечення захисту від SQL-ін’єкцій. Це особливо важливо для веб-застосунків, де обробляються чутливі дані користувачів, наприклад, персональна інформація або платіжні реквізити. Безпека бази даних є критично важливою для забезпечення довіри користувачів і захисту від зловмисних атак. PostgreSQL також підтримує механізм транзакцій, що дозволяє обробляти кілька запитів до бази даних в межах однієї операції. Це дозволяє забезпечити консистентність даних і гарантує, що всі зміни будуть застосовані коректно. Наприклад, якщо веб-застосунок здійснює кілька взаємопов’язаних операцій (наприклад, додавання товару до кошика і оновлення кількості на складі), транзакція дозволяє виконати ці операції в межах єдиної угоди.

Що стосується інтеграції з іншими сервісами, то вибір цього стека також є оптимальним для створення масштабованих веб-застосунків. Якщо веб-застосунок потребує інтеграції з платіжними системами, сторонніми API для отримання даних або іншими зовнішніми сервісами, цей стек дозволяє легко реалізувати такі функції. Наприклад, можна інтегрувати платіжні системи, які потребують обробки даних про транзакції, або підключити сторонні API для отримання актуальної інформації про погоду чи фінансові курси.

Завдяки великій спільноті розробників і підтримці таких великих фреймворків, як Django, цей стек технологій дозволяє створювати сучасні веб-застосунки, які швидко завантажуються, є масштабованими і забезпечують високий рівень користувацького досвіду. Django забезпечує ефективне управління маршрутизацією, обробкою форм, безпекою, доступом до бази даних та логікою додатку, що значно прискорює процес розробки. Завдяки вбудованій системі шаблонів, можна реалізовувати інтерактивні інтерфейси, які динамічно реагують на дії користувача і швидко оновлюють вміст сторінки. Наявність численних бібліотек Python, а також CSS-фреймворків (наприклад, Bootstrap або Tailwind CSS, які легко інтегруються з Django), дозволяє значно

зменшити час розробки та підвищити ефективність роботи. Вони надають розробникам готові рішення, які можна налаштувати під специфічні потреби проекту, що дозволяє економити час і ресурси. Ще однією перевагою цього стеку є можливість його подальшого розширення. Якщо в майбутньому з'являться нові функції або потреба в нових технологіях, можна легко інтегрувати інші інструменти і сервіси. Наприклад, можна додавати інтерфейси для роботи з іншими базами даних, підключати нові платформи для обробки даних або впроваджувати інші фреймворки для поліпшення функціональності. Ця гнучкість дозволяє масштабувати застосунок відповідно до вимог часу.

Також важливо відзначити, що вибір цього стеку має великий потенціал для створення SEO-дружніх веб-застосунків. HTML і CSS забезпечують правильну структуру сторінки, що дозволяє зручно індексувати контент пошуковими системами. Django дозволяє формувати URL-адреси, мета-теги, карти сайту та інші елементи, важливі для SEO. У випадку потреби в динамічному формуванні контенту також можлива інтеграція з JavaScript, проте основний вміст веб-застосунку залишається доступним для пошукових систем. PostgreSQL, у свою чергу, забезпечує ефективне зберігання даних, які можуть бути використані для побудови звітів, аналітики та генерації контенту для SEO-оптимізованих сторінок. [**Error! Reference source not found.**].

Обраний стек технологій для веб-розробки є універсальним і має великий потенціал для створення потужних, швидких і безпечних веб-застосунків. Використання Django, Python, HTML, CSS та PostgreSQL дозволяє розробникам створювати високоякісні продукти, які відповідають сучасним вимогам користувачів і забезпечують гнучкість та масштабованість проекту.

Висновки до розділу 2

Згідно проведеного аналізу, вибір архітектурних рішень і технологій для розробки веб-застосунку у сфері електронної комерції зумовлений актуальними вимогами ринку, високими очікуваннями користувачів щодо якості цифрових сервісів, а також необхідністю забезпечення надійності, гнучкості та

масштабованості системи. У рамках 2 розділу було охарактеризовано ключові вимоги до функціоналу веб-застосунків для продажу одягу та взуття, зокрема: підтримка каталогу товарів із фільтрацією, реалізація кошика, багаторівневий процес оформлення замовлення, авторизація, інтеграція з платіжними системами, адаптивність дизайну, а також підтримка аналітичних інструментів і механізмів персоналізації. Проведений огляд існуючих платформ на ринку, таких як Adidas, Answear та YesOriginal, дозволив виявити як глобальні, так і локальні тенденції розвитку електронної комерції, в тому числі використання сучасних фронтенд-технологій, систем безпеки, штучного інтелекту для персоналізації та мобільних додатків.

З урахуванням функціональних вимог до системи та тенденцій розвитку ринку, доцільним є використання технологічного стеку, до якого входять Python, Django, HTML, CSS і PostgreSQL. Таке поєднання забезпечує не лише високий рівень продуктивності та безпеки, а й дозволяє створювати масштабовані, гнучкі у підтримці та зручні для кінцевого користувача веб-рішення. Водночас для ефективної реалізації веб-застосунку обґрунтовано обрання середовище розробки PyCharm, яке надає зручні інструменти для автодоповнення коду, налагодження, управління базами даних і запуску серверів. Це значно підвищує продуктивність і якість розробки, дозволяючи повноцінно реалізувати переваги обраного стеку технологій.

РОЗДІЛ 3
ПРОЕКТУВАННЯ ТА ІМПЛЕМЕНТАЦІЯ ВЕБ-ЗАСТОСУНКУ ЕЛЕКТРОННОЇ КОМЕРЦІЇ

3.1. Проектування архітектури веб-застосунку

У сучасних умовах цифрової трансформації бізнесу та стрімкого зростання популярності електронної комерції, ефективна організація архітектури веб-застосунків відіграє ключову роль у забезпеченні їх стабільності, масштабованості та зручності використання. Користувачі очікують не лише на якісний контент, а й на високу швидкість завантаження, логічну структуру, інтуїтивно зрозумілий інтерфейс і можливість безперешкодного здійснення взаємодії із сервісом. У зв’язку з цим, на етапі розробки інформаційних систем – зокрема тих, що належать до сфери електронної комерції – надзвичайно важливою є побудова архітектури, яка дозволяє ефективно реалізувати функціональні вимоги та забезпечити гнучкість щодо подальших змін [**Error! Reference source not found.**].

Розробка веб-застосунку для продажу товарів через Інтернет включає низку архітектурних рішень, що стосуються як фронтенд-частини (інтерфейсу користувача), так і бекенду (серверної логіки, обробки даних і взаємодії з базою даних). При цьому вибір відповідної програмної платформи, структурування коду, взаємозв’язки між модулями, способи передачі та обробки даних мають бути чітко спроєктовані й реалізовані на початкових етапах проєктування. Належна архітектура гарантує мінімізацію помилок у майбутньому, покращення продуктивності та зручність у супроводі й розширенні функціоналу.

Веб-застосунок побудований за класичною клієнт-серверною архітектурною моделлю. У цій моделі клієнтська частина (веб-браузер користувача) відповідає за відображення інтерфейсу та взаємодію з користувачем, тоді як серверна частина (на основі фреймворку Django) відповідає за обробку запитів, виконання бізнес-логіки та взаємодію з базою

даних. Уся інформація про товари, замовлення, користувачів тощо зберігається в реляційній базі даних PostgreSQL, яка інтегрована з Django через ORM (Object-Relational Mapping).

Клієнтська частина реалізована за допомогою HTML-шаблонів Django, які поєднуються з CSS та JavaScript [**Error! Reference source not found.**]. Шаблони зберігаються в директоріях кожного додатку, зокрема в templates/carts/, templates/goods/, templates/main/. Відображення динамічних даних реалізується за допомогою вбудованого механізму шаблонів Django (Template Language) і, за потреби, додаткових кастомних тегів, наприклад у файлі *carts_tags.py*.

Серверна частина реалізована за допомогою фреймворку Django, який обробляє HTTP-запити та забезпечує відповідь клієнту у вигляді HTML. Основні компоненти включають *views.py* у кожному додатку, де реалізовано логіку обробки запитів, models.py для визначення структури бази даних, та urls.py для маршрутизації.

Система використовує PostgreSQL як систему управління базами даних. Всі взаємодії з БД реалізовано через Django ORM, що забезпечує абстракцію над SQL-запитами. У файлі *settings.py* описано підключення до бази даних.

```
DATABASES = {
    'default': {
        'ENGINE': 'django.db.backends.postgresql',
        'NAME': 'home',
        'USER': 'home',
        'PASSWORD': 'home',
        'HOST': 'localhost',
        'PORT': '5432',
    }
}
```

Рисунок 3.1 – Підключення до бази даних PostgreSQL

Джерело: створено автором

Структура застосунку побудована навколо ідеї модульності: кожен окремий функціональний блок реалізовано у вигляді додатку (app), який містить усі необхідні компоненти для реалізації конкретної частини логіки. Такий підхід дозволяє розділити відповідальність між різними частинами

системи, полегшує тестування, повторне використання коду та підтримку в майбутньому.

Усі додатки інтегруються у єдину систему через основний конфігураційний модуль, який включає маршрути, налаштування, точки входу та загальні параметри проєкту. У середовищі Django застосунок є сукупністю таких модулів, де кожен з них має свою власну структуру каталогів, що включає *models.py* (моделі даних), *views.py* (представлення, які реалізують бізнес-логіку), *urls.py* (маршрути), *templates/* (шаблони для формування HTML-інтерфейсу) та *admin.py* (інтеграція моделей у адміністративну панель).

У контексті даного проєкту ключовими додатками є *goods*, *carts*, *orders*, *users* та *main*. Кожен із них реалізує свою частину функціональності: наприклад, *goods* керує товарним каталогом, *carts* реалізує механізм кошика, *orders* відповідає за обробку замовлень, *users* – за автентифікацію та профілі користувачів, а *main* – за загальні сторінки інтерфейсу.

Таке архітектурне рішення дозволяє легко масштабувати систему, додаючи нові додатки або модифікуючи існуючі без потреби вносити зміни в інші частини проєкту. Проєкт організовано у вигляді набору додатків (apps), кожен з яких реалізує окрему частину функціональності. Основні додатки:

- goods: відповідає за управління товарами, їх категоріями, характеристиками;
- carts: реалізує логіку додавання, перегляду та видалення товарів із кошика користувача. Має власні шаблони, теги та модулі;
- orders: відповідає за обробку замовлень, включаючи створення, зміну статусу, історію;
- users: реалізує автентифікацію, реєстрацію, управління профілем користувача;
- main: відповідає за головну сторінку сайту та загальні елементи UI (інтерфейс користувача).

Кожен додаток містить файли:

- models.py – опис моделей бази даних;

- views.py – логіка обробки запитів;
- urls.py – маршрути для доступу до функціоналу;
- templates/ – шаблони HTML.

Django працює за шаблоном MVT (Model-View-Template), який структурно схожий на MVC, але адаптований для веб-середовища:

- Model – визначає структуру та зв’язки даних, зберігає дані у базі через ORM;
- View – містить бізнес-логіку, викликається при відповідному HTTP-запиті;
- Template – відповідає за представлення даних у HTML.

Ця архітектура дозволяє розділити відповідальність, спростити тестування та підвищити масштабованість.

Типова взаємодія між компонентами виглядає так:

1. Користувач надсилає HTTP-запит (наприклад, відкриває сторінку товару).
2. Запит надходить у *urls.py*, де маршрутизатор визначає відповідну функцію у *views.py*.
3. Представлення (view) звертається до моделі (model) за даними.
4. Отримані дані передаються до шаблону (template).
5. Згенерована HTML-сторінка повертається користувачеві.

Глобальний файл маршрутизації – це *urls.py* у головному конфігураційному модулі (в даному випадку – у папці app). У ньому визначаються основні шляхи, які включають маршрути кожного окремого додатку (рис. 3.2):

```
from django.contrib import admin
from django.urls import include, path
from django.conf.urls.static import static

from app import settings


urlpatterns = [
    path('admin/', admin.site.urls),
    path('', include('main.urls', namespace='main')),
    path('catalog/', include('goods.urls', namespace='catalog')),
    path('user/', include('users.urls', namespace='user')),
    path('cart/', include('carts.urls', namespace='cart')),
    path('orders/', include('orders.urls', namespace='orders')),
]
```

Рисунок 3.2 – Функція маршрутизації запитів між додатками веб-застосунку

Джерело: створено автором

Цей фрагмент коду демонструє делегування URL-адрес з основного рівня до відповідних додатків. Наприклад:

- Запити на /catalog/ будуть оброблятись у goods/urls.py;
- /user/ – у users/urls.py;
- /cart/ – у carts/urls.py тощо.

Django реалізує вбудовану систему автентифікації. У додатку users/ реалізовано кастомну модель користувача (через AUTH_USER_MODEL), а також логіку входу/виходу та реєстрації. Використовуються:

- захист від CSRF (вставка {% csrf_token %} у форми);
- хешування паролів;
- обмеження доступу через @login_required або PermissionRequiredMixin.

Запроєктована архітектура веб-застосунку відповідає принципам модульності, масштабованості та безпеки. Використання фреймворку Django, шаблону MVT та бази даних PostgreSQL дозволяє ефективно реалізувати бізнес-логіку, забезпечити надійне зберігання даних і створити зручний інтерфейс для користувача.

3.2. Реалізація основних модулів (каталог товарів, кошик, оформлення замовлення)

У цьому підрозділі розглянемо реалізацію основних функціональних модулів веб-застосунку для електронної комерції, розробленого з використанням фреймворку Django. Зокрема, зосередимо увагу на таких компонентах, як каталог товарів, кошик, оформлення замовлення.

Каталог товарів є ключовим елементом веб-застосунку, що дозволяє

```
class Categories(models.Model):
    name = models.CharField(max_length=150, unique=True, verbose_name='Назва')
    slug = models.SlugField(max_length=200, unique=True, blank=True, null=True, verbose_name='URL')

class Products(models.Model):
    name = models.CharField(max_length=150, unique=True, verbose_name='Назва')
    slug = models.SlugField(max_length=200, unique=True, blank=True, null=True, verbose_name='URL')
    description = models.TextField(blank=True, null=True, verbose_name='Опис')
    image = models.ImageField(upload_to='goods_images', blank=True, null=True, verbose_name='Зображення')
    price = models.DecimalField(default=0.00, max_digits=7, decimal_places=2, verbose_name='Ціна')
    discount = models.DecimalField(default=0.00, max_digits=4, decimal_places=2, verbose_name='Знижка в
%')
    quantity = models.PositiveIntegerField(default=0, verbose_name='Кількість')
    category = models.ForeignKey(to=Categories, on_delete=models.CASCADE, verbose_name='Категорія')
```

користувачам переглядати доступні продукти (див. додаток А). У Django для цього створено моделі Categories та Products, які описують відповідно категорії та товари (рис. 3.3).

Рисунок 3.3 – Моделі Categories та Products для каталогу товарів

Джерело: створено автором

Модель Categories визначає назву та унікальний ідентифікатор (slug) для кожної категорії.

Модель Products містить інформацію про назву товару, опис, зображення, ціну, знижку, кількість на складі та зв’язок із відповідною категорією.

Ці моделі дозволяють ефективно організувати та зберігати інформацію про товари та їх категорії, забезпечуючи зручний доступ до даних для подальшого відображення на веб-сторінках.

У веб-застосунку реалізація механізму додавання товарів до кошика базується на використанні класу CartAddView, що наслідує View та міксин CartMixin. Такий підхід забезпечує чітке розділення відповідальностей і дозволяє повторно використовувати логіку кошика. Функція працює асинхронно за допомогою Ajax-запиту, що дозволяє оновлювати вміст кошика без перезавантаження сторінки, покращуючи зручність користування. Візуалізацію реалізованого функціоналу кошика представлено у додатку Б.

Нижче наведено фрагмент коду, який реалізує цю функціональність (рис.3.4):

```
class CartAddView(CartMixin, View):
    def post(self, request):
        product_id = request.POST.get("product_id")
        product = Products.objects.get(id=product_id)

        cart = self.get_cart(request, product=product)

        if cart:
            cart.quantity += 1
            cart.save()
        else:
            Cart.objects.create(user=request.user if request.user.is_authenticated else None,
                                session_key=request.session.session_key if not
request.user.is_authenticated else None,
                                product=product, quantity=1)

        response_data = {
            "message": "Товар доданий до кошика",
            'cart_items_html': self.render_cart(request)
        }

        return JsonResponse(response_data)
```

Рисунок 3.4 – Клас додавання товару до кошика

Джерело: створено автором

У цьому коді за допомогою request.POST.get("product_id") отримується ідентифікатор товару, що додається. Якщо товар уже є в кошику, збільшується його кількість. Інакше створюється новий об’єкт Cart. Стан кошика оновлюється як для авторизованих користувачів (через user), так і для гостей (через session_key). Відповідь формується у форматі JSON, що дозволяє інтерактивно оновлювати HTML-вміст кошика.

Після формування кошика користувач може перейти до оформлення замовлення (див. додаток В). Для цього створюються дві моделі: Order та OrderItem.

Модель Order зберігає загальну інформацію про замовлення, зокрема:

- користувача, який зробив замовлення (user);
- дату створення замовлення (created_timestamp);
- контактний номер (phone_number);
- ознаку необхідності доставки та адресу доставки (requires_delivery, delivery_address);
- спосіб оплати (наприклад, післяплата – payment_on_get);
- статус оплати (is_paid);
- поточний статус обробки (status).

Це дозволяє системі гнучко управляти замовленнями в залежності від їхнього стану та вимог користувача.

Ось частина моделі Order, яка відповідає за зберігання цих даних (рис. 3.5):

```
class Order(models.Model):
    user = models.ForeignKey(to=User, on_delete=models.SET_DEFAULT, blank=True, null=True,
verbose_name="Користувач", default=None)
    created_timestamp = models.DateTimeField(auto_now_add=True, verbose_name="Дата створення замовлення")
    phone_number = models.CharField(max_length=20, verbose_name="Номер телефону")
    requires_delivery = models.BooleanField(default=False, verbose_name="Потрібна доставка")
    delivery_address = models.TextField(null=True, blank=True, verbose_name="Адреса доставки")
    payment_on_get = models.BooleanField(default=False, verbose_name="Оплата при отриманні")
    is_paid = models.BooleanField(default=False, verbose_name="Оплачено")
    status = models.CharField(max_length=50, default='В обробці', verbose_name="Статус замовлення")
```

Рисунок 3.5 – Модель Order для зберігання інформації про замовлення

Джерело: створено автором

Модель OrderItem описує кожен товар у складі замовлення. Вона містить посилання на замовлення (order), товар (product), його назву, ціну на момент замовлення, кількість, а також дату продажу (рис. 3.6):

```
class OrderItem(models.Model):
    order = models.ForeignKey(to=Order, on_delete=models.CASCADE, verbose_name="Замовлення")
    product = models.ForeignKey(to=Products, on_delete=models.SET_DEFAULT, null=True,
verbose_name="Продукт", default=None)
    name = models.CharField(max_length=150, verbose_name="Назва")
    price = models.DecimalField(max_digits=7, decimal_places=2, verbose_name="Ціна")
    quantity = models.PositiveIntegerField(default=0, verbose_name="Кількість")
    created_timestamp = models.DateTimeField(auto_now_add=True, verbose_name="Дата продажу")
```

Рисунок 3.6 – Модель OrderItem, що представляє одиницю товару в замовленні

Джерело: створено автором

Ця модель дозволяє відстежувати детальну інформацію про кожен елемент замовлення, зберігаючи історичні дані про ціни й кількість, що особливо корисно для аналітики або звітності.

У підсумку можна зазначити, що реалізація основних модулів веб-застосунку на базі Django забезпечує надійну та гнучку архітектуру системи електронної комерції. Каталог товарів, представлений через моделі категорій та продуктів, дозволяє зручно структурувати асортимент і відображати актуальну інформацію для користувачів. Механізм кошика, реалізований із використанням класових представлень і міксинів, забезпечує гнучке управління

вмістом кошика без необхідності перезавантаження сторінки, що значно покращує досвід користування. Система оформлення замовлень, представлена моделями Order та OrderItem, дозволяє зберігати всю необхідну інформацію для обробки та подальшого аналізу замовлень. Така структурована та модульна реалізація функціональних компонентів створює міцну основу для подальшого розвитку веб-застосунку, включаючи додавання нових функцій, таких як онлайн-оплати, або розширене управління доставкою.

3.3. Інтеграція з базою даних для зберігання інформації про товари та замовлення

Для забезпечення надійного зберігання інформації про товари, замовлення та користувачів веб-застосунок інтегровано із реляційною системою управління базами даних PostgreSQL. Як було зазначено у попередньому підрозділі, підключення до бази даних реалізується через налаштування у файлі settings.py, де визначено основні параметри доступу до СКБД. Завдяки використанню механізму об’єктно-реляційного відображення (ORM), що є вбудованою частиною Django, взаємодія із базою даних відбувається на високому рівні абстракції без необхідності безпосереднього написання SQL-запитів.

Основні моделі даних, такі як Product, Order та OrderItem, вже було розглянуто раніше. Вони визначають структуру таблиць у базі даних і автоматично перетворюються на відповідні таблиці під час виконання міграцій. Сама процедура створення базової структури відбувається шляхом використання команди python manage.py migrate, яка застосовує всі зміни моделей до бази даних.

На етапі розробки і тестування система міграцій Django дозволяє швидко оновлювати структуру бази, що сприяє гнучкій адаптації до змін вимог. Для підтвердження правильності створення структури бази даних доцільно навести приклад загальної схеми бази даних у pgAdmin, яка демонструє взаємозв’язки

між основними таблицями, включаючи Product, Order, OrderItem (див. додаток Г).

Інтеграція з базою даних супроводжується використанням адміністративної панелі Django (/admin/), яка надає зручний інтерфейс для роботи з даними. Завдяки налаштуванню моделей у файлах *admin.py*, розробник отримує можливість здійснювати перегляд, редагування та управління об’єктами бази даних без необхідності безпосереднього доступу до СКБД. Це істотно полегшує процес налагодження і тестування системи на етапах розробки. Для наочності слід продемонструвати інтерфейс адміністративної панелі Django із прикладами відображення замовлень та товарів (рис. 3.7).

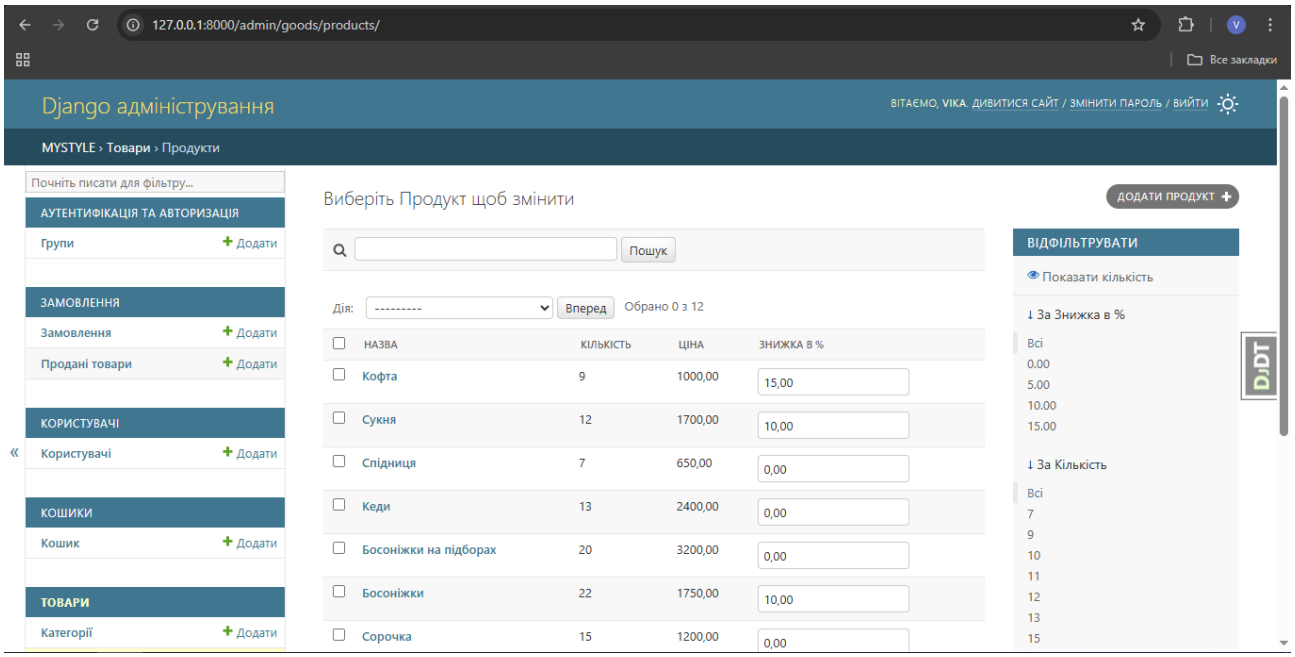


Рисунок 3.7 – Відображення товарів у адміністративній панелі Django

Джерело: створено автором

Окрім товарів, адміністративна панель Django також відображає інформацію про замовлення користувачів. На наступному рисунку продемонстровано інтерфейс управління замовленнями, де відображаються такі атрибути, як користувач, статус замовлення, необхідність доставки та стан оплати (рис. 3.8).

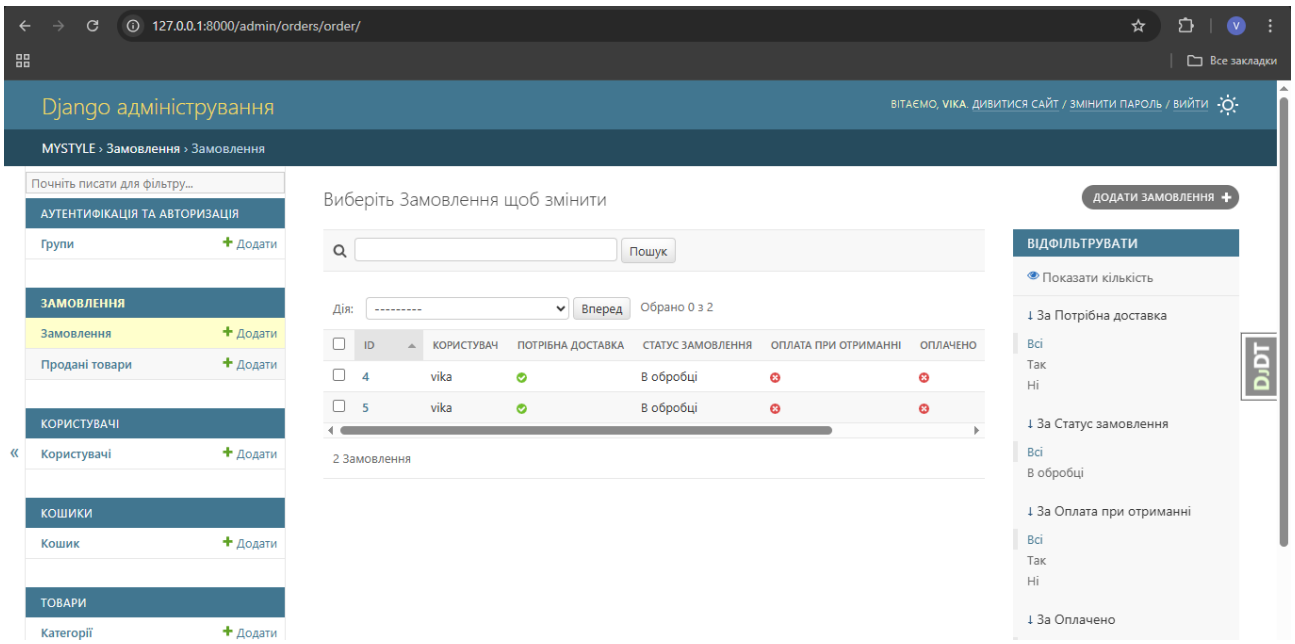


Рисунок 3.8 – Відображення замовлень у адміністративній панелі Django

Джерело: сформовано автором

У межах реалізації інтеграції забезпечено зв’язки між моделями за допомогою зовнішніх ключів (ForeignKey), що гарантує підтримку цілісності даних на рівні бази. Наприклад, кожен об’єкт замовлення (Order) пов’язується із користувачем, а кожна товарна одиниця у замовленні (OrderItem) асоціюється із відповідним товаром (Product). Це дозволяє зберігати історичну інформацію про покупки, включаючи актуальну ціну товару на момент здійснення замовлення.

Інтерактивна взаємодія між веб-застосунком та базою даних здійснюється у реальному часі. Наприклад, під час додавання товару до кошика або створення замовлення зміни негайно фіксуються у таблицях бази даних, що забезпечує актуальність даних і мінімізує ризики втрати інформації.

Таким чином, інтеграція веб-застосунку із базою даних PostgreSQL забезпечує надійне зберігання критично важливої інформації про товари та замовлення, підтримує високий рівень цілісності даних та дозволяє ефективно керувати інформацією через вбудовані інструменти Django.

3.4. Тестування та оптимізація веб-застосунку

Етап тестування та оптимізації є завершальним у процесі створення веб-застосунку, оскільки саме він забезпечує якість, стабільність та зручність подальшого використання програмного продукту. На даному етапі основна увага приділяється перевірці коректності реалізації функціоналу, відповідності логіки застосунку до поставлених цілей, а також пошуку та усуненню помилок, які могли виникнути під час розробки.

Перш за все, здійснено ручне тестування ключових компонентів інтерфейсу користувача. Перевірено працездатність форм, відображення сторінок, навігацію між розділами, функції реєстрації, входу та виходу користувача, а також обробку можливих помилок у запитах. Під час тестування було виявлено кілька незначних логічних неточностей, зокрема неправильну обробку деяких виняткових ситуацій та недостатню інформативність повідомлень про помилки. Після внесення відповідних коригувань, було забезпечено правильну реакцію системи на всі типові дії користувача.

Особливу увагу приділено тестуванню адаптивності веб-застосунку для різних типів пристроїв. Вебінтерфейс було перевірено на екранах різних розмірів, включно з мобільними телефонами, планшетами та десктопами. В результаті тестування було вдосконалено стилі CSS для коректного масштабування контенту та взаємодії з елементами керування, що підвищило загальний рівень зручності користування.

У межах тестування веб-застосунку було проведено оцінювання доступності розробленого ресурсу за допомогою інструменту WAVE Evaluation Tool, який є одним із поширених сервісів для аналізу відповідності веб-сторінок вимогам доступності.

Під час аналізу були виявлені окремі зауваження щодо доступності інтерфейсу (див. додаток Д). Зокрема, зафіксовано одну критичну помилку, пов’язану з некоректною ARIA-атрибуцією (Broken ARIA reference), що може впливати на сприйняття контенту користувачами, які застосовують допоміжні

технології. Також система виявила низку попереджень, зокрема щодо відсутності альтернативних текстів у частини елементів керування формами, а також пропуски рівнів заголовків.

Відзначено, що основні елементи сторінки, такі як навігаційне меню, пошук, заголовки та контентні блоки, мають відповідну семантичну структуру, що сприяє базовій доступності сайту. Водночас наявність деяких недоліків вказує на необхідність подальшої роботи з удосконалення доступності веб-застосунку, аби забезпечити відповідність сучасним стандартам WCAG (Web Content Accessibility Guidelines) та підвищити зручність користування ресурсом для всіх категорій користувачів [**Error! Reference source not found.**].

У процесі оптимізації було проаналізовано навантаження на сервер, кількість запитів до бази даних та ефективність їх обробки. Виконано оптимізацію запитів за рахунок використання методів попереднього завантаження пов’язаних об’єктів, що дозволило суттєво знизити кількість повторних звернень до бази. Також впроваджено базове кешування найбільш часто використовуваних даних, що дало змогу скоротити час відгуку сторінок.

Ще одним напрямом оптимізації стала робота з фронтенд-частиною застосунку. Було зменшено обсяг завантажуваних ресурсів, оптимізовано зображення, оновлено структуру HTML-шаблонів та очищено код від зайвих елементів. У результаті таких дій покращилася загальна швидкодія та продуктивність застосунку.

Проведене тестування та оптимізація веб-застосунку дозволили виявити й усунути недоліки функціоналу, покращити адаптивність і швидкодію, а також підвищити рівень доступності інтерфейсу. Це забезпечило належний рівень готовності продукту до практичного використання.

3.5. Рекомендації щодо розширення функціональності та подальшої підтримки застосунку

У процесі розробки та впровадження веб-застосунку для електронної комерції, спрямованого на продаж одягу та взуття, було реалізовано базову

функціональність, що включає каталог товарів, систему кошика, оформлення замовлення та інтеграцію з базою даних. Проте для забезпечення довготривалої актуальності, конкурентоспроможності та задоволення змінних потреб користувачів доцільно передбачити можливості розширення функціоналу та забезпечення стабільної підтримки веб-застосунку.

Серед пріоритетних напрямів розвитку можна визначити впровадження модуля онлайн-платежів, що дозволить користувачам здійснювати оплату безпосередньо через сайт, підвищуючи зручність користування та конверсію продажів.

Для покращення маркетингової ефективності можна впровадити систему розсилок та інструменти персоналізованих рекомендацій товарів на основі попередніх переглядів та замовлень. Це сприятиме зростанню продажів та підвищенню лояльності клієнтів.

Щодо технічної підтримки веб-застосунку, важливим етапом розвитку є запровадження системи автоматизованого тестування основних модулів веб-застосунку. Наявність тестового покриття дозволить своєчасно виявляти помилки на ранніх стадіях змін та підвищити загальну надійність системи. Також доцільно передбачити регулярні процедури оновлення залежностей фреймворку Django та інших бібліотек для гарантування безпеки застосунку.

З точки зору масштабованості, можлива подальша оптимізація бази даних, зокрема шляхом індексування часто використовуваних полів та оптимізації складних запитів. При зростанні обсягів даних рекомендовано застосувати горизонтальне масштабування бази або перехід на розподілену архітектуру.

Таким чином, запровадження запропонованих удосконалень дозволить забезпечити стійкий розвиток веб-застосунку, покращити його функціональні можливості, зручність користування та відповідність сучасним вимогам ринку електронної комерції.

Висновки до розділу 3

У третьому розділі кваліфікаційної роботи було висвітлено етапи розробки, інтеграції, тестування та оптимізації веб-застосунку для електронної комерції, орієнтованого на продаж одягу та взуття. В результаті проведеної роботи створено повноцінну систему, що реалізує базову функціональність: каталог товарів, кошик для покупок, систему оформлення замовлень і зберігання даних за допомогою інтеграції з базою даних.

Здійснено ручне тестування ключових функцій застосунку, проаналізовано його адаптивність для різних пристроїв, а також проведено оцінювання доступності за допомогою спеціалізованого інструменту WAVE. Виявлені недоліки були усунені, що дозволило підвищити якість взаємодії користувачів із застосунком. Виконано низку заходів з оптимізації як серверної, так і клієнтської частини системи: оптимізовано обробку запитів до бази даних, впроваджено кешування і мінімізацію обсягів завантажуваних ресурсів, що покращило загальну продуктивність застосунку.

Окрему увагу приділено формуванню подальших напрямів розвитку розробленого веб-застосунку. Планується впровадження модуля онлайн-платежів, створення системи персоналізованих рекомендацій товарів та підвищення ефективності маркетингової взаємодії з користувачами. Також намічено заходи щодо технічної підтримки: впровадження автоматизованого тестування основних модулів, регулярне оновлення технологічної бази, оптимізація структури бази даних та забезпечення масштабованості веб-застосунку в умовах зростання навантаження.

Отже, результати реалізації третього розділу дозволили досягти поставлених завдань щодо створення функціонально завершеного веб-застосунку, що відповідає сучасним стандартам якості, та окреслили стратегію його подальшого вдосконалення і підтримки для ефективної роботи в реальних умовах ринку електронної комерції.

ВИСНОВКИ

У процесі виконання кваліфікаційної роботи було комплексно досліджено теоретичні основи та практичні аспекти створення веб-застосунку для електронної комерції, орієнтованого на продаж одягу та взуття. Робота охоплювала аналіз сучасних тенденцій розвитку електронної комерції, принципів побудови веб-застосунків на основі фреймворку Django, а також етапи розробки, тестування та оптимізації готового програмного продукту.

У теоретичній частині кваліфікаційної роботи розглянуто основні поняття та особливості електронної комерції, виділено переваги використання фреймворку Django для створення веб-застосунків, проаналізовано архітектуру MVC (Model-View-Controller) та механізми взаємодії з базами даних. Окрему увагу було приділено питанням забезпечення якості веб-ресурсів, адаптивності інтерфейсу та доступності для користувачів з різними можливостями.

Практична частина роботи була присвячена розробці веб-застосунку з базовим функціоналом, що включає каталог товарів, систему кошика, оформлення замовлення та інтеграцію з базою даних для надійного зберігання інформації про товари та замовлення. Було реалізовано основні моделі бази даних із дотриманням принципів цілісності даних та організовано їх взаємодію через зовнішні ключі. Структура веб-застосунку забезпечує зручність користування, логічну навігацію та належний рівень обробки даних.

На етапі тестування здійснено перевірку працездатності функціональних модулів, тестування адаптивності інтерфейсу на різних пристроях, а також аналіз доступності веб-застосунку за допомогою інструменту WAVE Evaluation Tool. Виявлені недоліки були усунені, що дозволило покращити якість веб-ресурсу відповідно до сучасних стандартів.

Важливою складовою роботи стало визначення основних напрямів подальшого розвитку веб-застосунку. Сформульовано рекомендації щодо розширення функціональності, зокрема шляхом впровадження системи онлайн-платежів, персоналізованих рекомендацій товарів, автоматизації тестування

основних модулів і оптимізації бази даних для забезпечення масштабованості й безпеки веб-застосунку.

Узагальнюючи результати виконаної роботи, можна констатувати, що всі поставлені завдання були успішно реалізовані. Створений веб-застосунок відповідає вимогам функціональності, надійності та зручності використання, має потенціал для подальшого розвитку й масштабування відповідно до потреб ринку електронної комерції.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Ворошилов Д. О. Інформаційна технологія проектування інформаційної системи електронної комерції : кваліфікац. роб. магістра / Сумський державний університет. Суми, 2024. URL: https://essuir.sumdu.edu.ua/bitstream-download/123456789/95857/1/Voroshylov_mag_rob.pdf

2. Дмитренко А. О. Дослідження особливостей використання методів визначення рекомендацій в системах електронної комерції з обмеженим асортиментом товарів : кваліфікац. роб. магістра. Харків, 2024. URL: https://openarchive.nure.ua/server/api/core/bitstreams/eedc6a6c-7e94-4fda-9834-75b039f173bc/content

3. Довганюк Л. О. Алгоритмічно-програмний комбінований метод генерації коду шаблонів компонентів веб-застосунків : магістер. дис. Київ, 2020. URL: https://ela.kpi.ua/server/api/core/bitstreams/e8624a21-edcb-4701-9ecd-9f7d4e677bd3/content

4. Кузнєцова Г. О. Розробка веб-додатку для надання абітурієнтам НТУ «Дніпровська політехніка» психологічної підтримки з використанням мови програмування TypeScript : бакалавр. роб. Дніпро, 2024. URL: https://ir.nmu.org.ua/bitstream/handle/123456789/167376/12220%D0%B71_%D0%9A%D1%83%D0%B7%D0%BD%D1%94%D1%86%D0%BE%D0%B2%D0%B02.pdf

5. Левчук В. В. Обґрунтування ефективних інструментів розробки та тестування веб-доступності сайтів : кваліфікац. роб. магістра. Львів, 2024. URL: http://repository.lnau.edu.ua:8080/jspui/bitstream/123456789/982/1/Levchuk%20Volodymyr.pdf

6. Матусевич М. М. Інформаційна технологія розповсюдження технологічних товарів з модулем оплати замовлення : кваліфікац. роб. магістра / Сумський державний університет. Суми, 2022. URL: http://essuir.sumdu.edu.ua:8080/bitstream-download/123456789/88598/1/Matusevich_mag_rob.pdf

7. Мухін М. О. Розробка інформаційного веб-сайту криптовалют з використанням технологій ASP.NET Core, Angular та Entity Framework : бакалавр. роб. Одеса, 2024.

URL: http://eprints.library.odeku.edu.ua/id/eprint/13162/1/Мухін.pdf

8. Різниця між фронтенд і бекенд розробкою [Електронний ресурс]. URL: https://foxminded.ua/ru/front-end-back-end-raznica/ (дата звернення: 10.03.2025).

9. Розробка інтернет-магазину – основні характеристики та функціональність [Електронний ресурс]. URL: https://web24.pro/rozrobka-sajtiv-blog/rozrobka-internet-magazyn-osnovni-harakterystyky-ta-funkczionalnist/ (дата звернення: 10.03.2025).

10. Степаненко В. В. Обґрунтування ефективних інструментів розробки користувацьких інтерфейсів інтернет-магазину : кваліфікац. роб. магістра. Львів, 2024. URL: http://repository.lnau.edu.ua:8080/jspui/bitstream/123456789/981/1/Stepanenko.pdf

11. Твердовський Д. Л. Розробка веб-застосунку для архітектурно-будівельного конструкторського бюро Project UA : бакалавр. роб. Одеса, 2024. URL: http://eprints.library.odeku.edu.ua/id/eprint/13172/1/Твердовський.pdf

12. Тимошенко Н. Ю. Електронна комерція як стратегічний напрям розвитку бізнесу. 2024. URL: https://eco-science.net/wp-content/uploads/2024/11/11.24._topic_Natalya-Tymohenko-62-73.pdf (дата звернення: 22.02.2025).

13. Хапченко О. В., Лисенко О. М. Модифікована класифікація тифлотехнічних навігаційних систем та обґрунтування вибору архітектури розроблюваного навігаційного рішення // Вчені записки. 2022. № 5202233.

14. Харченко В. Є. Система аналізу цифрових платежів : кваліфікац. роб. магістра. Київ, 2023.

URL: http://studtheses.nubip.edu.ua:8080/handle/123456789/3581

15. Що таке Django? [Електронний ресурс].

URL: https://tutorial.djangogirls.org/uk/django/ (дата звернення: 04.03.2025).

16. Adidas – Офіційний інтернет-магазин в Україні [Електронний ресурс]. URL: https://surl.li/nolxva (дата звернення: 10.02.2025).

17. Angular vs React vs Vue in 2024 [Електронний ресурс]. URL: https://www.f22labs.com/blogs/angular-vs-react-vs-vue-in-2023/ (дата звернення: 14.03.2025).

18. Answear – офіційний сайт. URL: https://answear.ua (дата звернення: 12.03.2025).

19. Bootstrap [Електронний ресурс]. URL: https://getbootstrap.com/ (дата звернення: 05.02.2025).

20. Trudeau C. Django in Action. 2024. URL: https://pragprog.com/titles/ctdjango/django-in-action/ (дата звернення: 07.03.2025).

21. Django documentation [Електронний ресурс]. URL: https://docs.djangoproject.com/en/3.2/ (дата звернення: 06.02.2025).

22. Hoang T. Business Management App with Django. 2023. URL: https://github.com/HoangBusinessApp (дата звернення: 08.03.2025).

23. HTML5 & CSS3 Developer’s Guide [Електронний ресурс]. URL: https://developer.mozilla.org/en-US/docs/Web/HTML/Reference (дата звернення: 15.02.2025).

24. JavaScript. Підручник. Основи веб-програмування [Електронний ресурс]. URL: https://w3schoolsua.github.io/js/index.html#gsc.tab=0 (дата звернення: 13.02.2025).

25. Modern Web Architecture: Adopting Headless, API-First, and Cloud-Native Approaches. 2024. URL: https://medium.com/@eitbiz/a-modern-approach-to-web-development-headless-api-first-cloud-native-0a77f100e0b1 (дата звернення: 10.03.2025).

26. NoSQL – переваги та недоліки нереляційних баз даних [Електронний ресурс]. URL: https://www.hostinger.com/tutorials/nosql-vs-sql (дата звернення: 18.03.2025).

27. PostgreSQL [Електронний ресурс]. URL: https://www.postgresql.org/ (дата звернення: 02.03.2025).

28. PyCharm documentation [Електронний ресурс]. URL: https://www.jetbrains.com/pycharm/documentation/ (дата звернення: 11.03.2025).

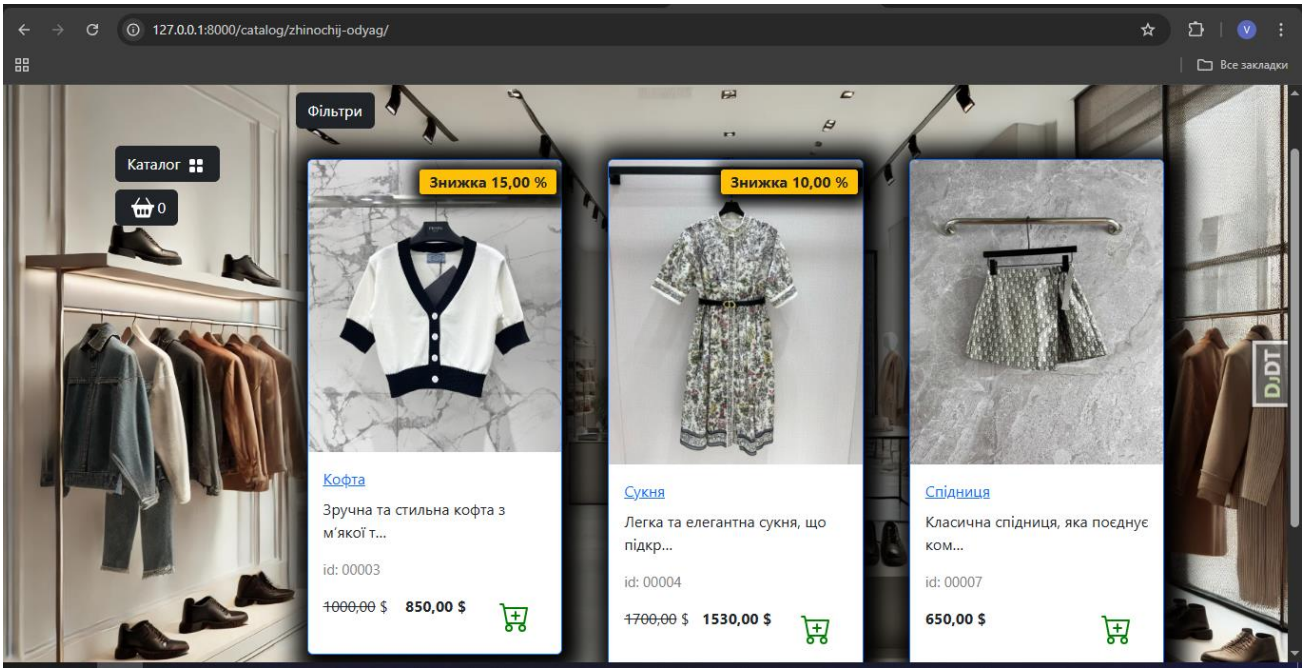
29. Python documentation [Електронний ресурс]. URL: https://www.python.org/doc/ (дата звернення: 03.02.2025).

30. YesOriginal – офіційний інтернет-магазин. URL: https://surl.li/jwzhyp (дата звернення: 06.02.2025).

ДОДАТКИ

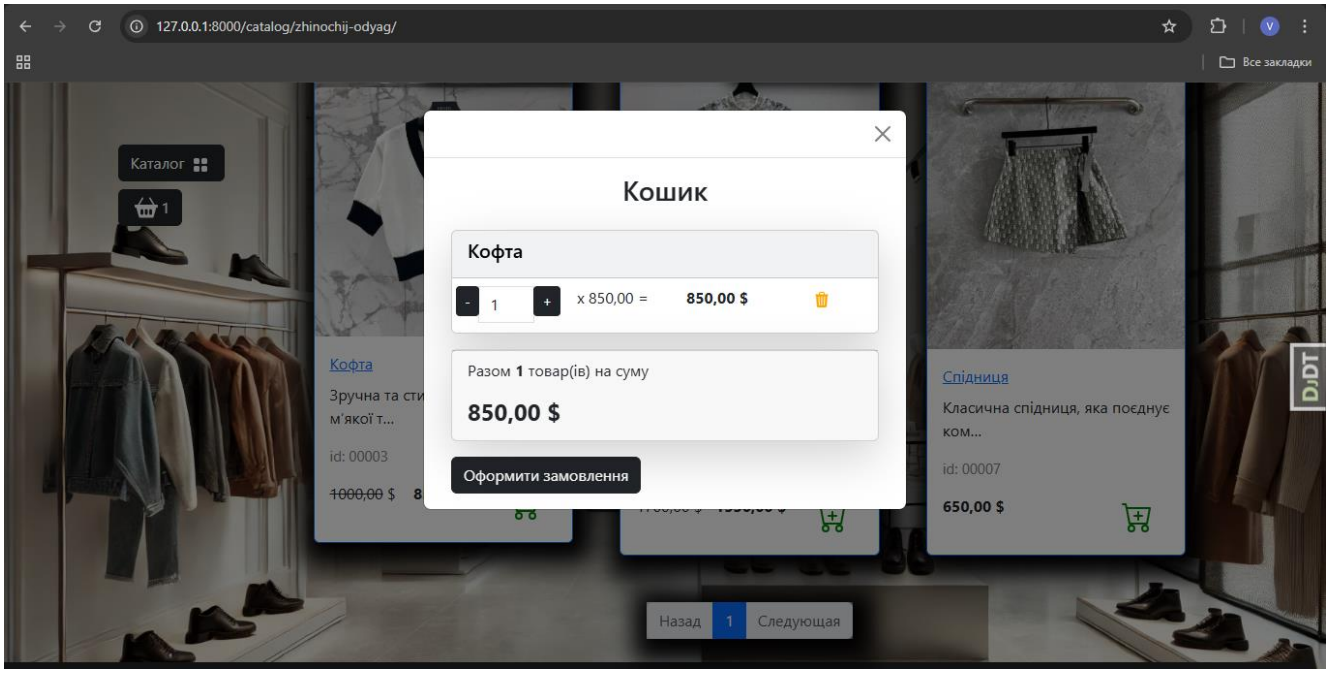
ДОДАТОК А

Каталог товарів



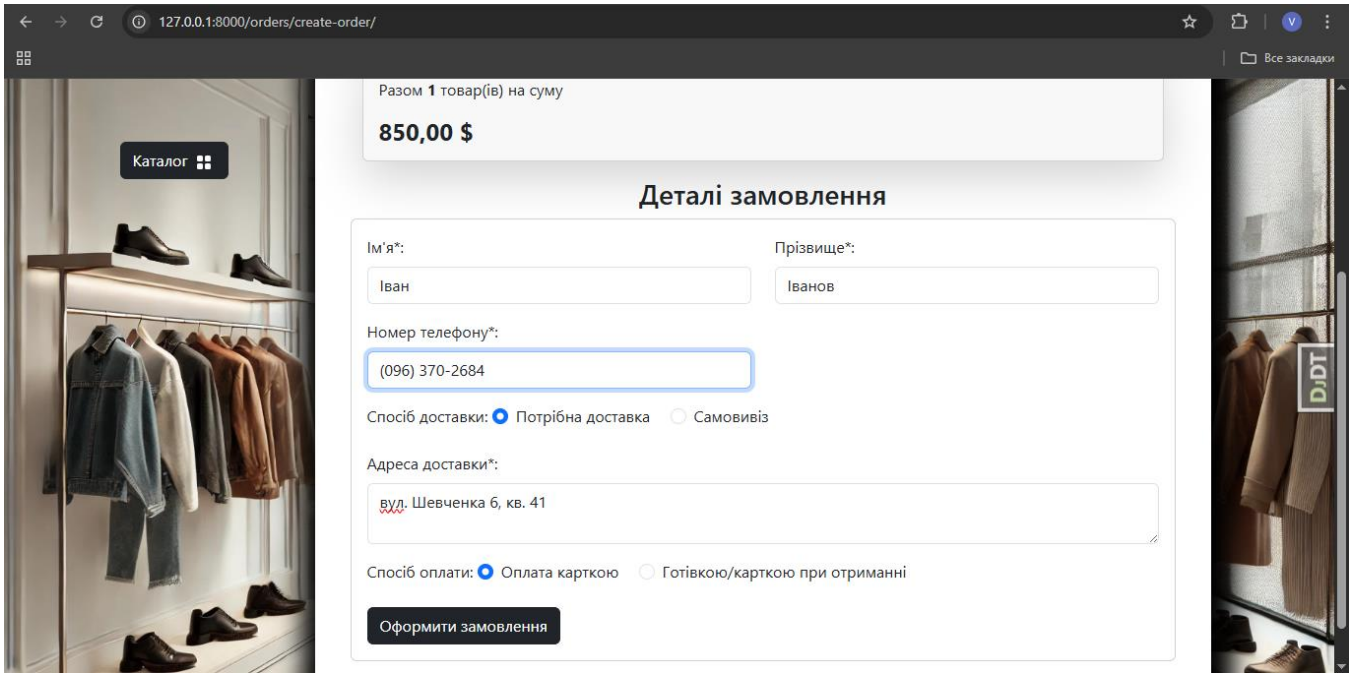
ДОДАТОК Б

Додавання товару до кошика



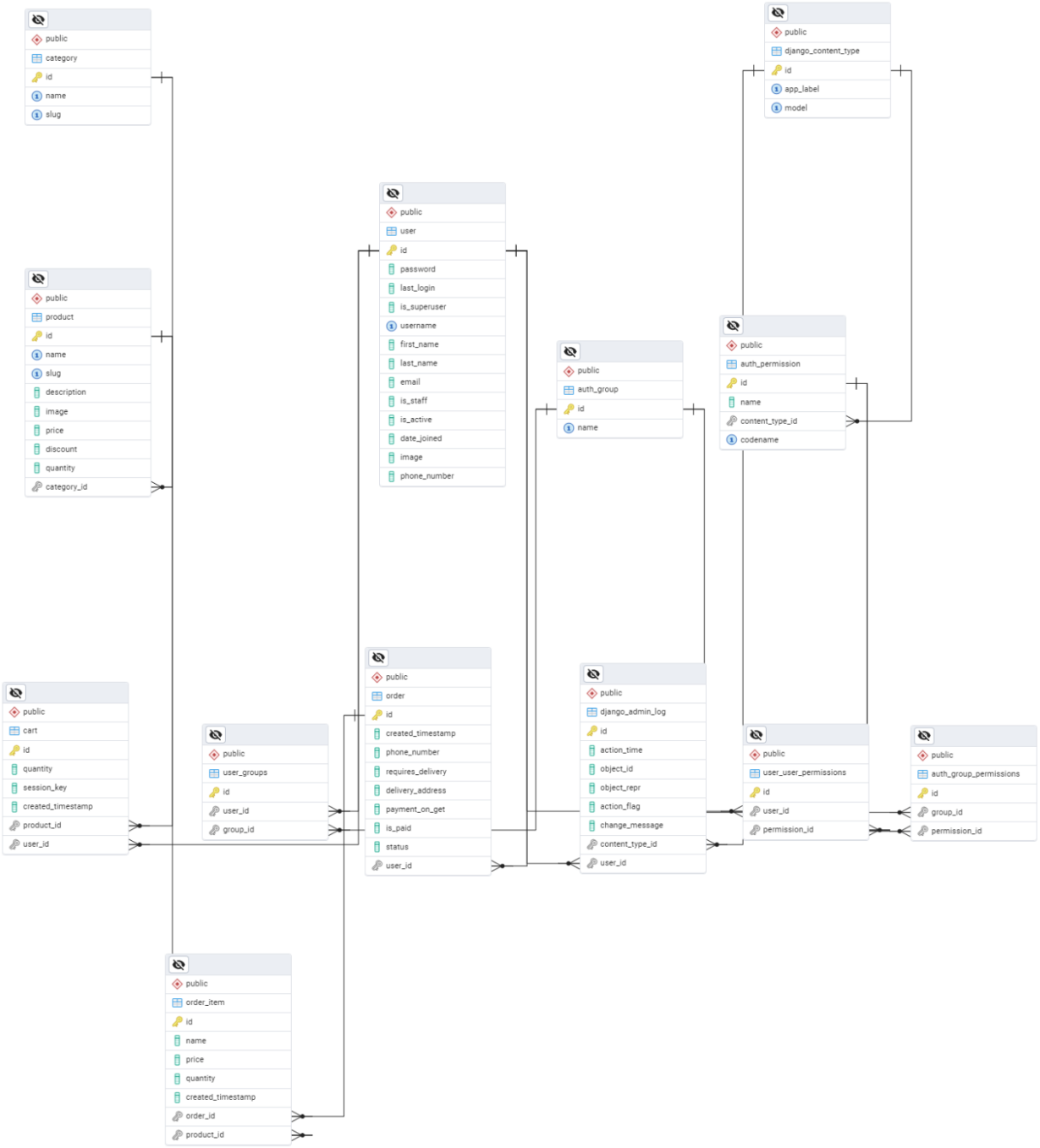
ДОДАТОК В

Оформлення замовлення



ДОДАТОК Г

Структура бази даних у pgAdmin



ДОДАТОК Д

Перевірка веб-застосунку на доступність