НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ БІОРЕСУРСІВ І ПРИРОДОКОРИСТУВАННЯ УКРАЇНИ

Факультет інформаційних технологій

ПОГОДЖЕНО	**ДОПУСКАЄТЬСЯ ДО ЗАХИСТУ**
Декан факультету	Завідувач кафедри комп'ютерних наук
_____ Ігор БОЛБОТ	_____ Белла ГОЛУБ
(підпис)	(підпис)
«___» _____ 2026 р.	«___» _____ 2026 р.

БАКАЛАВРСЬКА КВАЛІФІКАЦІЙНА РОБОТА

На тему: «Програмне забезпечення веб-орієнтованого додатку для реалізації спортивного екіпірування»

Спеціальність «121 Інженерія програмного забезпечення»

Освітньо-професійна програма «Інженерія програмного забезпечення»

Гарант освітньої програми
к.т.н., доцент _____ **Ганна ВАЙГАНГ**
(підпис)

Керівник бакалаврської
кваліфікаційної роботи _____ **Віктор СЕМКО**
(підпис)

Виконав _____ **Максим ПАЩЕНКО**
(підпис)

Київ-2026

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ БІОРЕСУРСІВ І ПРИРОДОКОРИСТУВАННЯ УКРАЇНИ

Факультет інформаційних технологій

ЗАТВЕРДЖУЮ
Завідувач кафедри комп'ютерних наук

к.т.н., доцент _____ Белла ГОЛУБ

«___» _____ 2026 р.

ЗАВДАННЯ
ДО ВИКОНАННЯ БАКАЛАВРСЬКОЇ КВАЛІФІКАЦІЙНОЇ РОБОТИ
ЗДОБУВАЧУ

Пащенку Максиму Юрійовичу

Спеціальність «121 Інженерія програмного забезпечення»
Освітньо-професійна програма «Інженерія програмного забезпечення»
Тема бакалаврської кваліфікаційної роботи
«Програмне забезпечення веб-орієнтованого додатку для реалізації спортивного екіпірування»
затверджена наказом від «02» квітня 2026 р. № 940 «С»
Термін подання завершеної роботи на кафедру «22» травня 2026 р.
Вихідні дані до бакалаврської кваліфікаційної роботи (проєкту): Технічні та функціональні вимоги до веб-орієнтованого додатку для реалізації спортивного екіпірування, сучасні технології веброзробки, методи проєктування баз даних, засоби розробки користувацького інтерфейсу та серверної частини програмного забезпечення.
Перелік питань, що підлягають дослідженню: аналіз процесу реалізації спортивного екіпірування як об'єкта автоматизації, проєктування структури та інформаційного забезпечення веб-додатку, розробка клієнтської та серверної частин програмного забезпечення, проєктування бази даних, тестування функціональних можливостей системи, розробка рекомендацій щодо впровадження та експлуатації веб-орієнтованого додатку.
Дата видачі завдання «3» квітня 2026 р.

Керівник бакалаврської кваліфікаційної роботи _____ (підпис) **Віктор СЕМКО**

Завдання взяв до виконання _____ (підпис) **Максим ПАЩЕНКО**

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

API — Інтерфейс програмування застосунків (Application Programming Interface)

БД — База даних

ІС — Інформаційна система

ПЗ — Програмне забезпечення

СУБД — Система управління базами даних

UI / UX — Інтерфейс користувача / Досвід взаємодії з користувачем

CMS — Система керування контентом (Content Management System)

JSON — JavaScript Object Notation

SSR / CSR — Серверний рендеринг / Клієнтський рендеринг

ORM — Object-Relational Mapping

CI/CD — Continuous Integration / Continuous Deployment

SEO — Search Engine Optimization (Оптимізація для пошукових систем)

ВСТУП

Актуальність теми. У сучасному світі цифровізація бізнес-процесів та розвиток електронної комерції значною мірою визначають ефективність торгівлі. Швидкий темп життя, популяризація здорового способу життя та спорт-індустрії зумовлюють високий попит на якісне спортивне екіпірування. Традиційні методи торгівлі через фізичні магазини мають обмеження за геолокацією, часом роботи та асортиментом, що робить розробку сучасних веборієнтованих рішень вкрай актуальним завданням для бізнесу та споживачів.

З розвитком інформаційних технологій з’явилася можливість створювати інтерактивні веборієнтовані застосунки, які в режимі реального часу дозволяють користувачам фільтрувати каталоги товарів, керувати кошиком, здійснювати безпечні онлайн-платежі та миттєво отримувати сповіщення про статус замовлень. Такі системи покращують клієнтський досвід, оптимізують логістику та знижують операційні витрати підприємств.

Потреба у простому, надійному та швидкому інструменті для купівлі спортивних товарів стає дедалі актуальнішою. Сучасні вебтехнології дозволяють створювати такі системи у вигляді адаптивних застосунків із доступом з будь- якого пристрою. Інтернет-магазин повинен надавати інтерактивний каталог товарів із гнучкою фільтрацією за категоріями, брендами та розмірами, забезпечувати реактивне управління кошиком, підтримувати безпечну авторизацію та мати багаторівневий поділ ролей.

Мета цієї кваліфікаційної роботи - розробка веборієнтованого застосунку для продажу спортивного екіпірування, який зробить процес взаємодії покупця та продавця зручним, функціональним, надійним і швидким.

Об’єкт дослідження - процес розробки програмного забезпечення для веборієнтованої електронної комерції.

Предмет дослідження - методи, технології та засоби збору, обробки,

структурування, збереження та візуалізації даних інтернет-магазину у вебсередовищі.

Завдання дослідження

- Проаналізувати існуючі рішення та платформи у сфері e-commerce.
- Сформулювати функціональні та технічні вимоги до системи продажу товарів.
- Обґрунтувати вибір технологій, фреймворків та архітектурних підходів.
- Проєктувати архітектуру та логічну модель даних вебзастосунку.
- Розробити програмний комплекс відповідно до вимог.
- Провести тестування системи та оцінити її швидкодію та UI/UX.

1 СИСТЕМНИЙ АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Опис предметної області

Предметна область веборієнтованої електронної комерції охоплює сукупність процесів, пов'язаних із продажем, придбанням, маркетингом та обслуговуванням товарів через мережу Інтернет. У сфері реалізації спортивного екіпірування особливого значення набувають питання ефективного представлення товарів, організації зручного пошуку та фільтрації продукції, управління замовленнями та забезпечення якісної взаємодії між продавцем і покупцем.

Сучасний ринок спортивних товарів характеризується широким асортиментом продукції, до якого належать спортивний одяг, взуття, тренажери, захисне спорядження, аксесуари та спеціалізований інвентар для різних видів спорту. Покупці очікують отримання детальної інформації про товари, включаючи характеристики, розміри, матеріали виготовлення, фотографії високої якості та відгуки інших користувачів. Тому веборієнтований додаток повинен забезпечувати зручний доступ до такої інформації та підтримувати швидке оформлення замовлень.

Розвиток технологій веброзробки сприяє створенню сучасних електронних торговельних платформ, які забезпечують високу продуктивність, безпеку та масштабованість. Використання таких технологій, як React, Next.js, Node.js та хмарних сервісів зберігання даних, дозволяє реалізувати інтерактивний користувацький інтерфейс, ефективно обробляти запити клієнтів та забезпечувати стабільну роботу системи навіть при значному навантаженні.

Важливою складовою предметної області є організація бізнес-процесів електронної торгівлі. До них належать управління каталогом товарів, ведення обліку залишків на складі, обробка замовлень, інтеграція з платіжними системами та службами доставки. Автоматизація зазначених процесів дозволяє

підвищити ефективність роботи інтернет-магазину, скоротити час обробки замовлень та покращити якість обслуговування клієнтів.

Особливу роль у функціонуванні веборієнтованих систем електронної комерції відіграє архітектура програмного забезпечення. Використання сучасних підходів до розробки, зокрема клієнт-серверної архітектури, REST API та хмарної інфраструктури, забезпечує гнучкість, масштабованість і відмовостійкість системи. Це дозволяє ефективно підтримувати роботу веб-додатку, своєчасно оновлювати функціональні можливості та забезпечувати безпечне зберігання даних користувачів і замовлень.

1.2 Аналіз вимог до програмної системи

Вимоги до веб-орієнтованого додатку для реалізації спортивного екіпірування поділяються на функціональні та нефункціональні. Функціональні вимоги визначають набір можливостей, які повинна забезпечувати система для користувачів та адміністраторів. До них належать реєстрація та авторизація користувачів, перегляд каталогу товарів, пошук і фільтрація продукції за різними параметрами, додавання товарів до кошика, оформлення замовлень, управління особистим кабінетом, а також адміністрування асортименту товарів і обробка замовлень. Нефункціональні вимоги охоплюють питання продуктивності, надійності, безпеки, масштабованості та зручності використання системи.

Сучасні веб-технології дозволяють створювати ефективні платформи електронної комерції, які забезпечують швидку обробку запитів користувачів та комфортну взаємодію з інтерфейсом. Використання таких інструментів, як React, Next.js, Node.js та сучасних систем керування базами даних, сприяє створенню високопродуктивного програмного забезпечення, здатного підтримувати одночасну роботу великої кількості користувачів. Для сфери продажу спортивного екіпірування це особливо важливо, оскільки покупці очікують швидкого доступу до каталогу товарів, якісних зображень, детальних описів, характеристик та можливості оперативного оформлення замовлень.

Важливим аспектом розробки веб-орієнтованого додатку є забезпечення гнучкої та масштабованої архітектури. Використання клієнт-серверного підходу та REST API дозволяє розділити функціональність між фронтенд- та бекенд-компонентами системи, що спрощує подальший розвиток і супровід програмного продукту. Крім того, застосування хмарних сервісів і сучасних засобів зберігання даних забезпечує стабільну роботу системи, високу доступність сервісів та ефективне управління інформацією про товари, користувачів і замовлення.

Архітектурні рішення повинні також враховувати вимоги до інформаційної безпеки, зокрема захист персональних даних користувачів, безпечне зберігання облікових записів та контроль доступу до адміністративних функцій. Реалізація цих вимог сприяє підвищенню довіри користувачів до системи та забезпечує її надійне функціонування в умовах реальної експлуатації.

1.3 Моделювання предметної області

Для моделювання структури та функціональних можливостей веб-орієнтованого додатку використовується уніфікована мова моделювання UML (Unified Modeling Language). Вона забезпечує стандартизований підхід до візуального представлення компонентів системи, їх взаємодії та поведінки, що значно спрощує процес аналізу, проєктування та документування програмного забезпечення. У межах розробки веб-додатку для реалізації спортивного екіпірування були створені основні UML-діаграми, серед яких діаграма варіантів використання, діаграма класів, діаграма послідовності , діаграма станів (для сутності "Замовлення")

Діаграма прецедентів дозволяє визначити основних учасників системи та сценарії їх взаємодії з програмним забезпеченням. Основними акторами виступають покупець та адміністратор. Покупець має можливість переглядати каталог товарів, здійснювати пошук продукції, додавати товари до кошика, оформлювати замовлення та переглядати історію покупок. Адміністратор,

своєю чергою, керує асортиментом товарів, обробляє замовлення та здійснює контроль за функціонуванням системи.

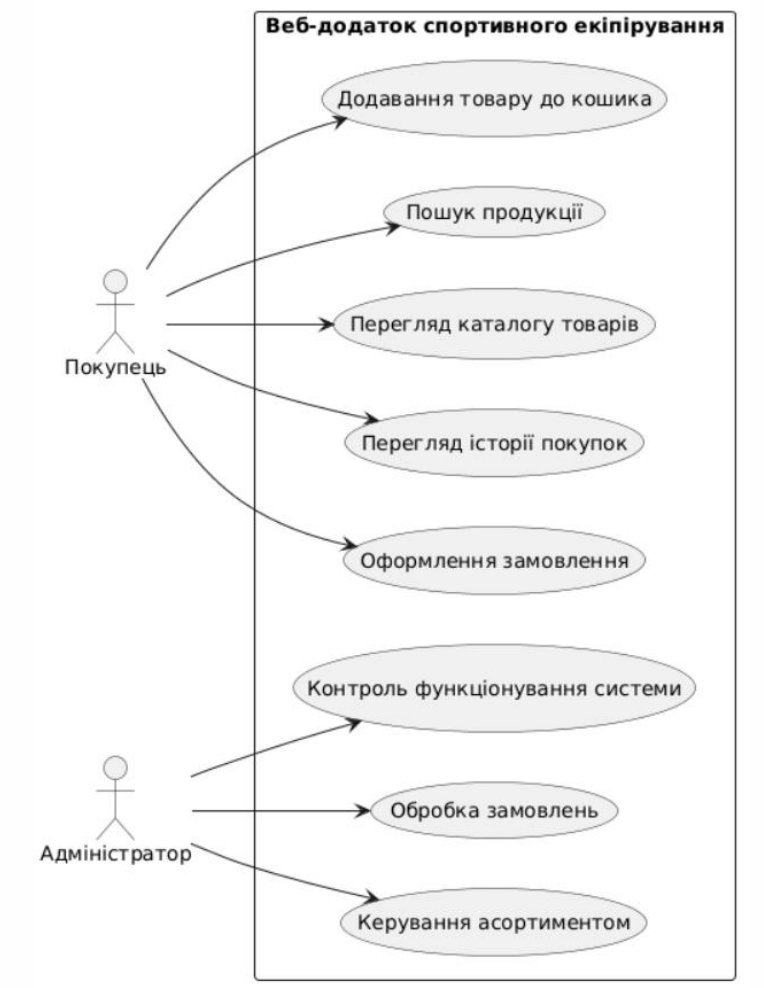

Рис 1.1 Діаграма прецедентів

Діаграма класів використовується для відображення структури програмного забезпечення та взаємозв'язків між його основними сутностями. До ключових класів системи належать користувач, товар, категорія, кошик, замовлення та адміністратор. Кожен клас містить набір атрибутів і методів, необхідних для реалізації відповідної бізнес-логіки веб-додатку.

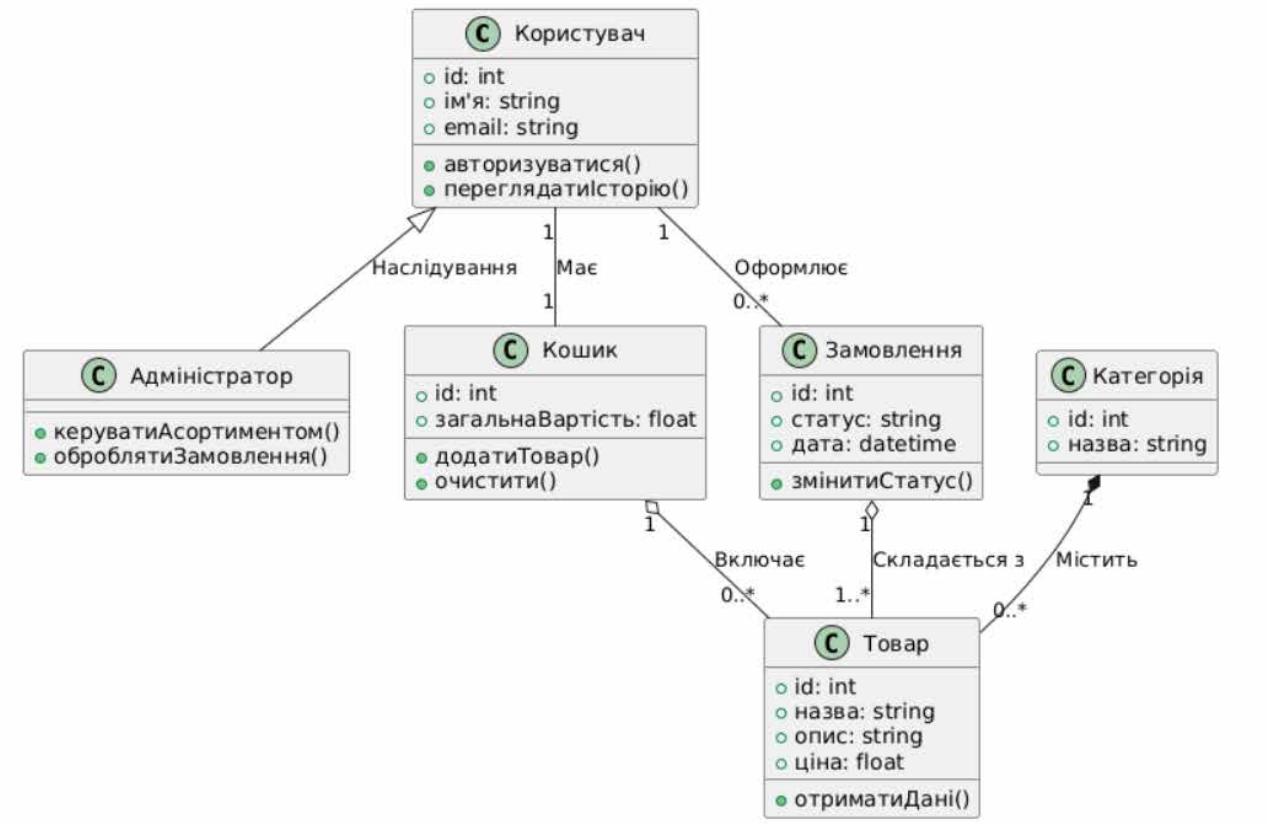

Рис 1.2 Діаграма класів

Для опису процесів обміну даними між компонентами системи застосовується діаграма послідовності. Вона демонструє порядок взаємодії між користувацьким інтерфейсом, серверною частиною та базою даних під час виконання таких операцій, як авторизація користувача, додавання товару до кошика або оформлення замовлення. Це дозволяє детально проаналізувати логіку роботи системи та виявити потенційні недоліки ще на етапі проєктування.

Діаграми діяльності та станів використовуються для моделювання бізнес-процесів і життєвого циклу окремих об'єктів системи. Наприклад, для сутності «Замовлення» можуть бути визначені стани «Створено», «Підтверджено», «Відправлено», «Доставлено» та «Скасовано». Таке представлення дозволяє формалізувати логіку зміни станів та забезпечити коректне виконання операцій у межах веб-додатку.

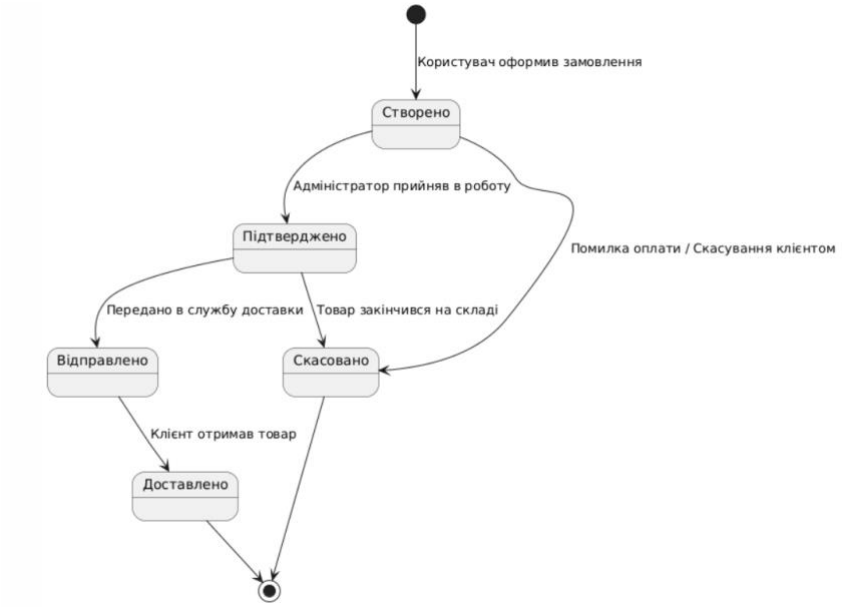

Рис 1.3 Діаграма станів (для сутності "Замовлення")

Застосування UML-моделювання під час розробки веб-орієнтованого додатку для реалізації спортивного екіпірування сприяє підвищенню якості проєктування програмного забезпечення, забезпечує кращу структурованість системи та спрощує її подальший супровід і розвиток.

1.4 Опис основних абстракцій веборієнтованого магазину

Веборієнтований магазин спортивного екіпірування являє собою інформаційну систему, що забезпечує автоматизацію процесів пошуку, вибору, придбання та доставки товарів користувачам через мережу Інтернет. Для моделювання предметної області доцільно використовувати набір основних абстракцій, кожна з яких відображає окремий об’єкт системи та містить відповідні атрибути й функціональні можливості.

Ключовими абстракціями веборієнтованого магазину є: Користувач, Товар, Категорія, Кошик, Замовлення, Платіж та Відгук. Взаємодія між цими сутностями забезпечує повний цикл електронної комерції – від перегляду каталогу до завершення покупки та оцінювання придбаного товару.

Абстракція Користувач описує зареєстрованого або незареєстрованого відвідувача системи. Вона містить інформацію про особисті дані, контактні

відомості, історію замовлень та налаштування облікового запису. Користувач може здійснювати пошук товарів, додавати їх до кошика, оформлювати замовлення та залишати відгуки.

Абстракція Товар є центральним елементом системи та містить дані про найменування продукції, опис, ціну, характеристики, наявність на складі, фотографії та інші параметри. Для спортивного екіпірування особливе значення мають характеристики розміру, матеріалу, бренду та призначення товару.

Абстракція Категорія використовується для структуризації асортименту магазину. Завдяки категоріям товари групуються за певними ознаками, що спрощує навігацію та пошук необхідної продукції. Наприклад, спортивне екіпірування може бути поділене на категорії одягу, взуття, аксесуарів та тренувального обладнання.

Абстракція Кошик призначена для тимчасового зберігання вибраних товарів перед оформленням покупки. Вона містить перелік товарів, їх кількість та розраховану загальну вартість замовлення. Кошик дозволяє користувачеві змінювати склад замовлення до моменту підтвердження покупки.

Абстракція Замовлення відображає факт придбання товарів користувачем. Вона включає інформацію про склад замовлення, дату створення, статус обробки, адресу доставки та підсумкову суму до сплати. Життєвий цикл замовлення може включати етапи створення, підтвердження, оплати, доставки та завершення.

Абстракція Платіж забезпечує обробку фінансових операцій у системі. Вона містить відомості про спосіб оплати, суму платежу, дату проведення транзакції та її поточний статус. Інтеграція з платіжними сервісами дозволяє автоматизувати процес підтвердження оплати та підвищує зручність використання системи.

Абстракція Відгук забезпечує механізм зворотного зв’язку між покупцями та магазином. Вона містить текстовий коментар, оцінку товару та інформацію про автора відгуку. Наявність системи відгуків сприяє підвищенню

довіри користувачів до магазину та допомагає потенційним покупцям приймати обґрунтовані рішення щодо придбання продукції.

Використання зазначених абстракцій дозволяє сформувати цілісну модель предметної області веборієнтованого магазину спортивного екіпірування. Такий підхід забезпечує чітку структуру програмної системи, спрощує її подальше проєктування та створює основу для масштабування функціональних можливостей у майбутньому.

Аналоги та існуючі рішення

Перед розробкою веборієнтованого магазину спортивного екіпірування доцільно провести аналіз існуючих рішень, представлених на ринку електронної комерції. Такий аналіз дозволяє визначити основні функціональні можливості сучасних інтернет-магазинів, виявити їх переваги та недоліки, а також сформувати вимоги до власного програмного продукту.

Одним із найбільш відомих міжнародних рішень є інтернет-магазин Nike, який пропонує широкий асортимент спортивного одягу, взуття та аксесуарів. Перевагами даного ресурсу є сучасний дизайн, зручна навігація, детальний опис товарів та персоналізовані рекомендації для користувачів. Водночас значна кількість функцій може ускладнювати роботу нових користувачів та збільшувати час завантаження сторінок.

Іншим популярним рішенням є інтернет-магазин Adidas, який забезпечує швидкий пошук товарів, систему фільтрації за різними характеристиками та підтримку адаптивного інтерфейсу для мобільних пристроїв. Серед переваг можна виділити високу якість представлення товарів та зручний процес оформлення замовлення. Недоліком є залежність частини функціоналу від зовнішніх сервісів та складність адміністрування великого каталогу продукції.

Серед українських аналогів можна виділити спеціалізовані інтернет-магазини спортивного спорядження, які забезпечують продаж спортивного одягу, інвентарю та аксесуарів. Такі ресурси, як правило, мають базовий набір функцій: каталог товарів, кошик, систему оформлення замовлень та особистий

кабінет користувача. Проте багато з них мають обмежені можливості персоналізації, недостатньо оптимізований інтерфейс або застарілий дизайн.

Проведений аналіз показав, що більшість сучасних інтернет-магазинів реалізують схожий набір функцій, зокрема реєстрацію та авторизацію користувачів, управління каталогом товарів, кошик покупок, оформлення замовлень, систему відгуків та інтеграцію з платіжними сервісами. Водночас значна увага приділяється швидкодії вебзастосунків, адаптивності інтерфейсу та забезпеченню безпеки персональних даних користувачів.

На основі аналізу існуючих рішень визначено, що розроблюваний веборієнтований магазин спортивного екіпірування повинен поєднувати зручний користувацький інтерфейс, ефективну систему пошуку та фільтрації товарів, можливість швидкого оформлення замовлень, а також засоби адміністрування каталогу продукції. Це дозволить створити конкурентоспроможний програмний продукт, який відповідатиме сучасним вимогам електронної комерції та забезпечуватиме комфортну взаємодію користувачів із системою.

Таблиця 1.1

Порівняння аналогів SportPoint

Критерії порівняння	Nike	Adidas	Sportmaster	SportPoint (розроблювана система)
Основне призначення	Продаж спортивного одягу та взуття бренду Nike	Продаж спортивних товарів бренду Adidas	Продаж спортивного одягу, взуття та інвентарю різних брендів	Продаж спортивного екіпірування різних категорій
Тип інтерфейсу	Адаптивний, сучасний	Адаптивний, сучасний	Адаптивний	Адаптивний SPA-інтерфейс
Категорії товарів	Одяг, взуття, аксесуари	Одяг, взуття, аксесуари	Одяг, взуття, тренажери, аксесуари	Одяг, взуття, аксесуари, спортивне спорядження
Пошук товарів	Розширений	Розширений	Базовий	Розширений
Фільтрація товарів	За ціною, розміром, кольором	За ціною, розміром, брендом	За основними параметрами	За категорією, ціною, брендом, розміром
Реєстрація користувачів	Наявна	Наявна	Наявна	Наявна

Критерії порівняння	Nike	Adidas	Sportmaster	SportPoint (розроблювана система)
Кошик покупок	Наявний	Наявний	Наявний	Наявний
Онлайн-оплата	Наявна	Наявна	Наявна	Наявна
Відгуки та рейтинги	Наявні	Наявні	Частково реалізовані	Наявні
Панель адміністратора	Обмежений доступ	Обмежений доступ	Наявна	Повноцінна система керування
Архітектура бази даних	Реляційна БД	Реляційна БД	Реляційна БД	Реляційна БД
Керування товарами	Так	Так	Так	Так
Керування замовленнями	Так	Так	Так	Так
Масштабованість	Висока	Висока	Середня	Висока
Технології розробки	Власні корпоративні рішення	Власні корпоративні рішення	Комерційна платформа	Next.js, React, PostgreSQL
Швидкодія інтерфейсу	Висока	Висока	Середня	Висока
Безпека даних	Висока	Висока	Висока	Висока

Аналіз аналогів показав, що більшість сучасних інтернет-магазинів спортивного екіпірування мають схожу функціональність, яка включає каталог товарів, систему пошуку та фільтрації, кошик покупок та механізм оформлення

замовлень. Водночас розроблювана система SportPoint відрізняється використанням сучасних вебтехнологій, адаптивним інтерфейсом, гнучкою архітектурою бази даних та повноцінною адміністративною панеллю, що забезпечує ефективне управління контентом і подальше масштабування програмного продукту.

1.5 Обґрунтування діаграми варіантів використання

Діаграма варіантів використання (Use Case Diagram) є одним із ключових інструментів моделювання вимог до програмної системи. Її основним призначенням є відображення взаємодії користувачів із веборієнтованим магазином спортивного екіпірування та визначення функціональних можливостей, які система надає різним категоріям користувачів.

Використання діаграми варіантів використання на етапі проєктування дозволяє сформувати цілісне уявлення про функціональні вимоги системи, встановити межі програмного продукту та визначити перелік основних сценаріїв роботи. Діаграма наочно демонструє, які дії можуть виконувати користувачі та адміністратори, а також показує взаємозв’язки між окремими функціями системи.

Для веборієнтованого магазину спортивного екіпірування основними акторами є Покупець та Адміністратор. Покупець взаємодіє із системою з метою перегляду каталогу товарів, пошуку необхідної продукції, перегляду детальної інформації про товар, додавання товарів до кошика, оформлення замовлення, здійснення оплати та залишення відгуків. Адміністратор відповідає за підтримку актуальності даних у системі та виконує функції керування товарами, категоріями, замовленнями та обліковими записами користувачів.

Обґрунтування вибору саме цих варіантів використання полягає в тому, що вони охоплюють усі основні бізнес-процеси електронної комерції. Сценарії покупця забезпечують реалізацію процесу придбання спортивного екіпірування від моменту пошуку товару до завершення покупки. Сценарії адміністратора забезпечують ефективне управління вмістом магазину та контроль за процесом виконання замовлень.

Побудова діаграми варіантів використання дозволяє виявити функціональні залежності між окремими процесами, уникнути дублювання функцій та сформувати основу для подальшого проєктування діаграм класів,

послідовностей і діяльності. Крім того, вона виступає засобом комунікації між замовником та розробником, забезпечуючи однозначне розуміння функціональних вимог до веборієнтованого магазину спортивного екіпірування.

Таким чином, діаграма варіантів використання є важливим етапом проєктування системи, оскільки дозволяє формалізувати взаємодію користувачів із програмним продуктом та забезпечує основу для подальшої реалізації його функціональних можливостей.

Відмінності запропонованого рішення

Однією з головних відмінностей розроблюваної системи є використання сучасного технологічного стеку, що включає Next.js, React та PostgreSQL. Це дозволяє забезпечити швидке завантаження сторінок, ефективну обробку запитів користувачів та надійне зберігання даних. На відміну від багатьох існуючих рішень, які використовують універсальні шаблонні платформи електронної комерції, система SportPoint розробляється з урахуванням особливостей предметної області та потреб кінцевих користувачів.

Важливою перевагою є реалізація адаптивного користувацького інтерфейсу, який забезпечує коректне відображення контенту на різних типах пристроїв. Це дозволяє користувачам комфортно працювати із системою незалежно від розміру екрана та типу пристрою.

На відміну від більшості аналогів, у розроблюваній системі передбачено розширену систему керування товарами та категоріями через адміністративну панель. Адміністратор має можливість централізовано керувати асортиментом продукції, змінювати характеристики товарів, контролювати замовлення та здійснювати моніторинг активності користувачів.

Особливістю SportPoint є оптимізована структура бази даних, яка забезпечує ефективне зберігання інформації про користувачів, товари, категорії, замовлення та платежі. Використання реляційної бази даних дозволяє

підтримувати цілісність даних та спрощує подальше розширення функціоналу системи.

Запропоноване рішення також передбачає підвищений рівень безпеки завдяки використанню механізмів автентифікації та авторизації користувачів, захисту персональних даних та контролю доступу до адміністративних функцій. Це сприяє забезпеченню надійності роботи системи та захисту конфіденційної інформації.

1.6 Постановка завдання

Для досягнення поставленої мети необхідно провести аналіз предметної області та існуючих аналогів, визначити функціональні та нефункціональні вимоги до програмного продукту, а також спроєктувати архітектуру системи та структуру бази даних. Особлива увага повинна бути приділена створенню адаптивного користувацького інтерфейсу, який забезпечуватиме комфортну взаємодію із системою незалежно від типу пристрою користувача.

У процесі розробки необхідно реалізувати каталог товарів із можливістю пошуку, сортування та фільтрації, систему реєстрації та авторизації користувачів, механізм формування кошика покупок і оформлення замовлень. Також система повинна забезпечувати роботу адміністративної панелі для керування товарами, категоріями, користувачами та замовленнями. Для зберігання інформації про товари, користувачів, платежі та замовлення необхідно реалізувати реляційну базу даних, яка гарантуватиме цілісність і надійність даних.

Важливим завданням є забезпечення безпечної роботи системи шляхом використання механізмів автентифікації, авторизації та захисту персональних даних користувачів. Крім того, необхідно реалізувати можливість залишення відгуків та оцінювання товарів, що сприятиме підвищенню рівня взаємодії між користувачами та магазином.

Результатом виконання роботи має стати функціональний веборієнтований магазин спортивного екіпірування SportPoint, який

забезпечуватиме повний цикл електронної комерції та відповідатиме сучасним вимогам до якості, продуктивності, безпеки та масштабованості програмного забезпечення.

1.7 Висновки до розділу 1

У першому розділі було проведено аналіз предметної області веборієнтованої електронної комерції у сфері реалізації спортивного екіпірування та визначено основні особливості функціонування сучасних інтернет-магазинів. Дослідження показало, що ефективність таких систем значною мірою залежить від зручності користувацького інтерфейсу, швидкодії, надійності обробки даних та можливості масштабування програмного забезпечення.

У ході аналізу вимог до програмної системи було визначено основні функціональні та нефункціональні вимоги, які повинні бути реалізовані у вебзастосунку. Зокрема, система має забезпечувати перегляд каталогу товарів, пошук і фільтрацію продукції, керування кошиком покупок, оформлення замовлень, адміністрування контенту та захист персональних даних користувачів. Також було встановлено необхідність використання сучасних вебтехнологій для досягнення високої продуктивності та стабільності роботи системи.

Під час моделювання предметної області було визначено основні сутності та взаємозв’язки між ними, а також обґрунтовано використання UML-діаграм для опису структури та поведінки програмного продукту. Виділено ключові абстракції системи, до яких належать користувач, товар, категорія, кошик, замовлення, платіж та відгук. Використання цих абстракцій дозволяє сформувати цілісну модель веборієнтованого магазину та забезпечити ефективну реалізацію його функціональних можливостей.

Проведений аналіз існуючих аналогів показав, що сучасні інтернет-магазини спортивного екіпірування мають схожий набір функцій, проте відрізняються рівнем адаптивності, продуктивності та можливостями

адміністрування. На основі отриманих результатів було сформовано вимоги до розроблюваної системи SportPoint та визначено її основні переваги, серед яких використання сучасного технологічного стеку, адаптивного інтерфейсу, оптимізованої структури бази даних та розширеної адміністративної панелі.

Таким чином, результати проведеного аналізу та проєктування створюють необхідне підґрунтя для подальшої розробки веборієнтованого магазину спортивного екіпірування SportPoint, який відповідатиме сучасним вимогам електронної комерції, забезпечуватиме високу якість обслуговування користувачів та підтримуватиме можливість подальшого розвитку і масштабування.

2. ПРОЄКТУВАННЯ ІНФОРМАЦІЙНОГО ТА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

2.1 Логічна модель у вигляді ER-Діаграми

Логічна модель даних є основою проєктування бази даних веборієнтованого магазину спортивного екіпірування SportPoint. Вона визначає структуру інформаційних об’єктів системи, їхні характеристики та взаємозв’язки, необхідні для забезпечення коректного функціонування всіх бізнес-процесів інтернет-магазину. Побудована ER-діаграма дозволяє формалізувати структуру даних, забезпечити їхню цілісність та створити основу для ефективної реалізації програмного продукту.

Однією з ключових сутностей моделі є таблиця *USER*, яка використовується для зберігання інформації про зареєстрованих користувачів системи. У таблиці містяться атрибути, що описують обліковий запис користувача, зокрема унікальний ідентифікатор, електронна адреса, пароль, персональні дані та контактна інформація. Кожен користувач може створювати декілька замовлень, залишати відгуки до товарів та взаємодіяти з функціоналом магазину відповідно до своєї ролі в системі.

Для представлення товарного асортименту використовується таблиця *PRODUCT*, яка містить інформацію про спортивне екіпірування, доступне для продажу. Для кожного товару зберігаються його назва, опис, вартість, кількість на складі, характеристики, фотографії та інші параметри, необхідні для відображення інформації покупцю. Товари належать до певних категорій, що спрощує навігацію та організацію каталогу.

Групування продукції реалізується за допомогою таблиці *CATEGORY*. Вона містить інформацію про категорії спортивних товарів, такі як одяг, взуття, аксесуари або спортивне спорядження. Між таблицями CATEGORY та PRODUCT існує зв’язок типу один до багатьох, оскільки одна категорія може

містити значну кількість товарів, тоді як кожен товар належить лише до однієї категорії.

Процес придбання товарів реалізується через таблицю *ORDER*, яка використовується для зберігання інформації про оформлені замовлення. Для кожного замовлення фіксуються дата створення, поточний статус виконання, загальна сума покупки та адреса доставки. Зв’язок між користувачем і замовленням реалізовано за принципом один до багатьох, що дозволяє зберігати історію покупок кожного клієнта.

Оскільки одне замовлення може містити декілька товарів, а один товар може входити до складу багатьох замовлень, у моделі використовується проміжна сутність *ORDER_ITEM*. Вона забезпечує реалізацію зв’язку багато до багатьох між товарами та замовленнями, а також містить інформацію про кількість кожного товару та його вартість на момент оформлення покупки.

Для обробки фінансових операцій передбачена таблиця *PAYMENT*, у якій зберігаються відомості про здійснені платежі. До основних атрибутів належать сума платежу, спосіб оплати, дата проведення операції та поточний статус транзакції. Кожен платіж безпосередньо пов’язаний із конкретним замовленням, що забезпечує контроль процесу оплати та підтвердження покупки.

Зворотний зв’язок від користувачів реалізується за допомогою таблиці *REVIEW*, яка містить оцінки та коментарі щодо придбаних товарів. Кожен відгук пов’язаний із конкретним користувачем та товаром, що дозволяє формувати рейтинг продукції та надавати потенційним покупцям додаткову інформацію для прийняття рішення про покупку.

Таким чином, логічна модель даних охоплює всі основні сутності, необхідні для функціонування веборієнтованого магазину спортивного екіпірування. Визначені зв’язки між таблицями забезпечують ефективне зберігання та обробку інформації, підтримують цілісність даних і створюють надійну основу для подальшої реалізації бази даних та серверної частини системи.

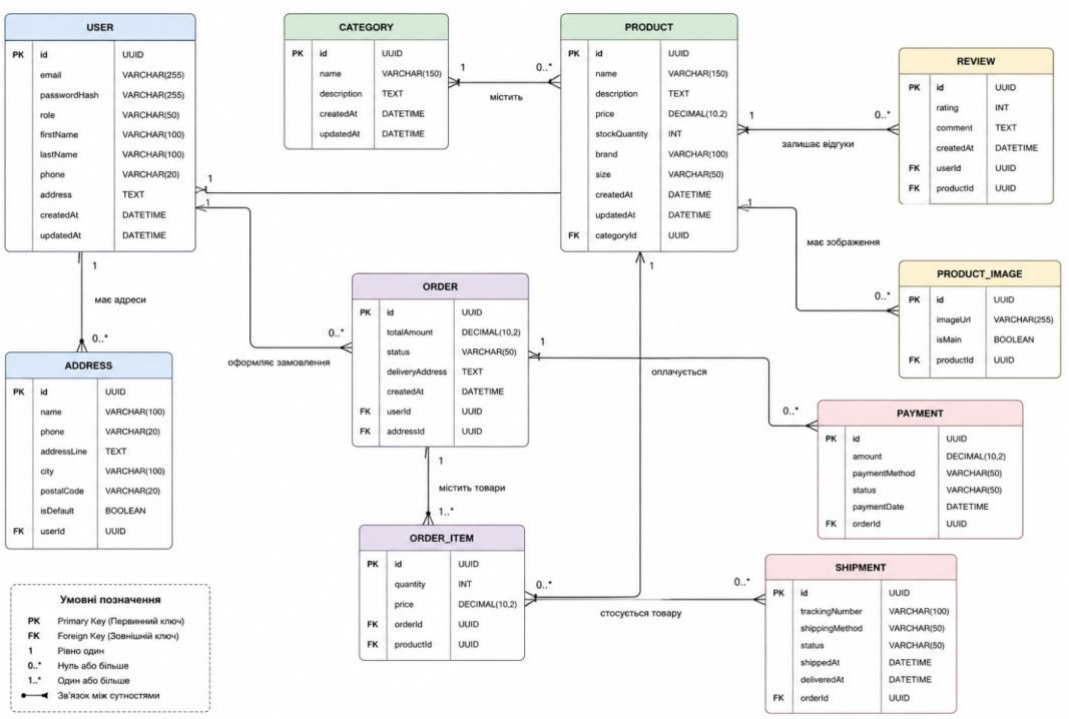

Рис. 2.1. Логічна модель даних у вигляді ER-діаграми

2.2 Діаграма класів та кооперацій

Для детального проєктування програмної системи веборієнтованого магазину спортивного екіпірування SportPoint використовуються діаграма класів та діаграма кооперацій, які дозволяють відобразити структуру програмного забезпечення та взаємодію його компонентів. Застосування цих діаграм сприяє формалізації архітектури системи, визначенню основних об’єктів предметної області та способів їхньої взаємодії під час виконання бізнес-процесів.

Діаграма класів використовується для представлення статичної структури системи. Вона відображає основні класи, їх атрибути, методи та взаємозв’язки між ними. У межах веборієнтованого магазину центральне місце займає клас User, який описує зареєстрованого користувача системи. Даний клас містить атрибути, що характеризують обліковий запис користувача, зокрема

ідентифікатор, електронну адресу, пароль, контактну інформацію та роль у системі. Методи класу забезпечують виконання операцій авторизації, оновлення персональних даних та перегляду історії замовлень.

Клас Product призначений для представлення товарів, доступних у каталозі магазину. Він містить інформацію про назву товару, опис, ціну, характеристики, наявність на складі та фотографії. Методи класу забезпечують отримання детальної інформації про товар, оновлення характеристик та перевірку доступності продукції.

Для структуризації асортименту використовується клас Category, який містить дані про категорії спортивного екіпірування та забезпечує групування товарів за певними ознаками. Один об’єкт категорії може бути пов’язаний із багатьма товарами, що реалізує відношення типу один до багатьох.

Процес оформлення покупки реалізується за допомогою класу Order, який містить інформацію про склад замовлення, дату створення, статус виконання та підсумкову вартість. Клас забезпечує функції створення, підтвердження та зміни статусу замовлення. З кожним замовленням пов’язаний клас OrderItem, що використовується для зберігання інформації про окремі товари, які входять до складу покупки.

Для реалізації механізму фінансових операцій використовується клас Payment, який містить дані про суму платежу, спосіб оплати, дату проведення транзакції та її поточний статус. Взаємодія між класами Order та Payment забезпечує контроль процесу оплати та підтвердження виконання замовлення.

Клас Review відповідає за реалізацію системи оцінювання товарів. Він містить текст відгуку, оцінку продукції, дату створення та інформацію про автора. Відгуки пов’язуються як із користувачами, так і з товарами, що дозволяє формувати рейтинг продукції та забезпечувати механізм зворотного зв’язку.

Взаємозв’язки між класами реалізуються через асоціації, агрегації та композиції. Користувач може створювати декілька замовлень та залишати багато відгуків. Категорія об’єднує групу товарів, а замовлення складається з

набору товарних позицій. Така структура забезпечує модульність системи та спрощує її подальше розширення.

Для відображення динамічної взаємодії між об’єктами використовується діаграма кооперацій. Вона демонструє обмін повідомленнями між компонентами системи під час виконання конкретних сценаріїв роботи користувача. Одним із основних сценаріїв є процес оформлення замовлення. Під час його виконання користувач взаємодіє з інтерфейсом каталогу товарів, додає обрані позиції до кошика та ініціює створення замовлення. Система звертається до бази даних для перевірки наявності товарів, формує запис про замовлення та передає інформацію до модуля обробки платежів.

Після підтвердження оплати система оновлює статус замовлення, зберігає відповідну інформацію в базі даних та надсилає користувачеві повідомлення про успішне оформлення покупки. Аналогічним чином реалізується взаємодія під час перегляду каталогу товарів, залишення відгуків або адміністрування асортименту через панель керування.

Таким чином, діаграма класів дозволяє описати структуру програмної системи та взаємозв’язки між її основними компонентами, тоді як діаграма кооперацій відображає механізми їхньої взаємодії під час виконання бізнес-процесів. Спільне використання цих діаграм забезпечує цілісне уявлення про архітектуру веборієнтованого магазину спортивного екіпірування SportPoint та створює основу для його подальшої реалізації.

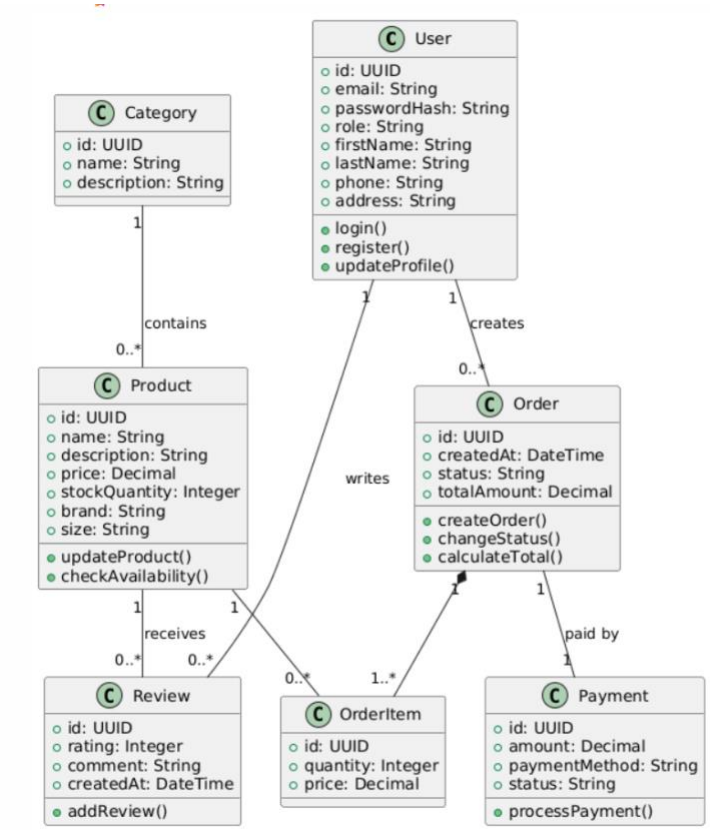

Рис. 2.2. Діаграма класів

2.3 Діаграма пакетів

Для відображення загальної архітектури програмної системи та логічного групування її компонентів використовується діаграма пакетів (Package Diagram). Даний тип UML-діаграми дозволяє представити структуру веборієнтованого магазину спортивного екіпірування SportPoint на високому рівні абстракції, розділивши систему на окремі функціональні модулі та показавши залежності між ними.

Використання діаграми пакетів сприяє підвищенню модульності програмного забезпечення, спрощує процес розробки та подальшого супроводу системи. Кожен пакет об'єднує пов'язані між собою класи, компоненти та сервіси, які реалізують певну частину функціональності вебзастосунку.

У розроблюваній системі SportPoint виділено декілька основних пакетів. Пакет Authentication відповідає за реєстрацію користувачів, автентифікацію,

авторизацію та керування сесіями. Він забезпечує контроль доступу до функцій системи та захист персональних даних користувачів.

Пакет User Management містить компоненти, пов'язані з керуванням профілями користувачів, обробкою особистих даних та історією замовлень. Він взаємодіє з модулем автентифікації та забезпечує роботу особистого кабінету покупця.

Пакет Catalog Management реалізує функціональність роботи з товарами та категоріями. До його складу входять класи, що відповідають за відображення каталогу, пошук продукції, фільтрацію товарів та керування асортиментом через адміністративну панель.

Пакет Order Management забезпечує обробку замовлень користувачів. Він реалізує створення замовлень, зміну їх статусів, зберігання історії покупок та взаємодію з платіжною системою. Саме цей пакет координує основні бізнес-процеси електронної комерції.

Для виконання фінансових операцій використовується пакет Payment Management, який відповідає за інтеграцію із сервісами онлайн-оплати, перевірку статусів платежів та фіксацію інформації про проведені транзакції.

Пакет Review Management реалізує механізм залишення відгуків та оцінок товарів. Він забезпечує взаємодію між користувачами та каталогом продукції, дозволяючи формувати рейтинг товарів і покращувати якість контенту магазину.

Окреме місце в архітектурі займає пакет Database Layer, який забезпечує взаємодію всіх модулів із базою даних. Через цей пакет здійснюються операції збереження, оновлення, пошуку та видалення інформації про користувачів, товари, замовлення, платежі та відгуки.

Між пакетами існують логічні залежності. Модулі керування товарами, замовленнями та відгуками використовують функціональність пакета Database Layer для доступу до даних. Пакет Order Management залежить від Catalog Management для отримання інформації про товари та від Payment Management

для проведення платежів. У свою чергу, більшість функціональних пакетів використовують можливості Authentication для перевірки прав доступу користувачів.

Таким чином, діаграма пакетів дозволяє відобразити високорівневу структуру веборієнтованого магазину SportPoint, визначити основні програмні модулі та взаємозв'язки між ними. Це забезпечує кращу організацію програмного коду, спрощує масштабування системи та створює основу для подальшої реалізації архітектури програмного забезпечення.

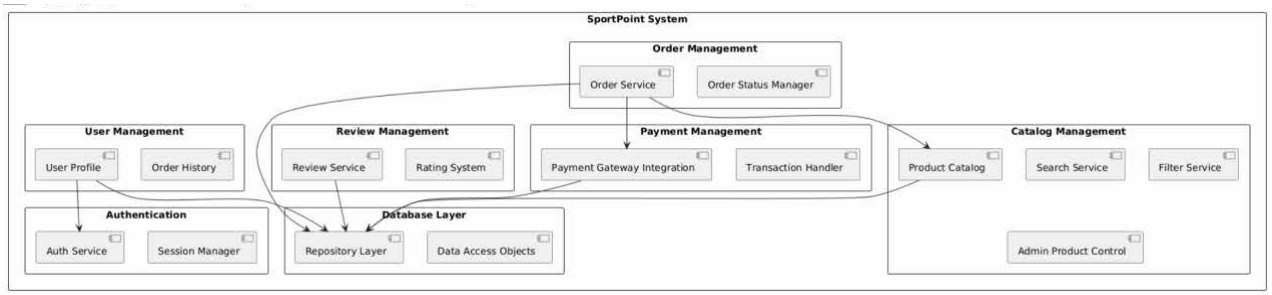

Рис. 2.3. Діаграма пакетів

2.4. Діаграма компонентів

Центральним елементом архітектури є компонент Веб-клієнт (Web Interface), який реалізує користувацький інтерфейс системи. Він містить підмодулі каталогу товарів, кошика, оформлення замовлення та особистого кабінету користувача. Через інтерфейси відобразитиКаталог(), оновитиКошик() та показатиЗамовлення() цей компонент забезпечує взаємодію користувача з системою та отримання актуальної інформації про продукцію.

Веб-клієнт взаємодіє з компонентом Backend-сервер, який відповідає за бізнес-логіку системи. Сервер обробляє запити користувачів, виконує перевірку даних, координує роботу модулів та забезпечує безпечний доступ до функцій магазину через інтерфейси обробитиЗапит(), керуватиДаними() та автентифікуватиКористувача().

Для управління доступом використовується компонент Authentication Service, який реалізує реєстрацію, авторизацію та керування сесіями

користувачів. Він взаємодіє з сервером через інтерфейси зареєструватиКористувача() та перевіритиТокен(), забезпечуючи захист персональних даних та контроль доступу до функціоналу системи.

Зберігання інформації здійснюється через компонент Database (База даних), який містить структури даних користувачів, товарів, замовлень, платежів і відгуків. Сервер взаємодіє з базою даних через інтерфейси зберегтиЗапис(), отриматиДані() та оновитиІнформацію(), що дозволяє підтримувати актуальний стан усіх бізнес-даних магазину.

Окремим важливим компонентом є Payment Gateway, який відповідає за проведення онлайн-оплат. Він інтегрується із сервером через інтерфейс ініціюватиПлатіж() та підтвердитиТранзакцію(), забезпечуючи безпечну обробку фінансових операцій та взаємодію із зовнішніми платіжними сервісами.

Компонент Order Service (Система замовлень) координує процес оформлення та обробки покупок. Він взаємодіє з каталогом товарів, платіжним шлюзом та базою даних через інтерфейси створитиЗамовлення(), оновитиСтатус() та отриматиІсторіюЗамовлень(), забезпечуючи повний життєвий цикл замовлення.

Для роботи з асортиментом використовується компонент Catalog Service, який відповідає за керування товарами, категоріями та пошуком продукції. Він надає інтерфейси отриматиТовари(), фільтруватиПродукцію() та пошукТовару(), що дозволяє формувати динамічний каталог спортивного екіпірування.

Додатково в системі присутній компонент Review Service, який реалізує функціональність відгуків та рейтингів товарів. Через інтерфейси додатиВідгук() та отриматиОцінки() він забезпечує взаємодію між користувачами та каталогом, формуючи систему оцінювання продукції.

Таким чином, діаграма компонентів дозволяє чітко відобразити фізичну структуру веб-застосунку SportPoint, визначити ключові програмні модулі та їх

взаємодію. Це сприяє підвищенню модульності системи, спрощує її масштабування та забезпечує зрозумілу архітектуру для подальшої розробки і підтримки.

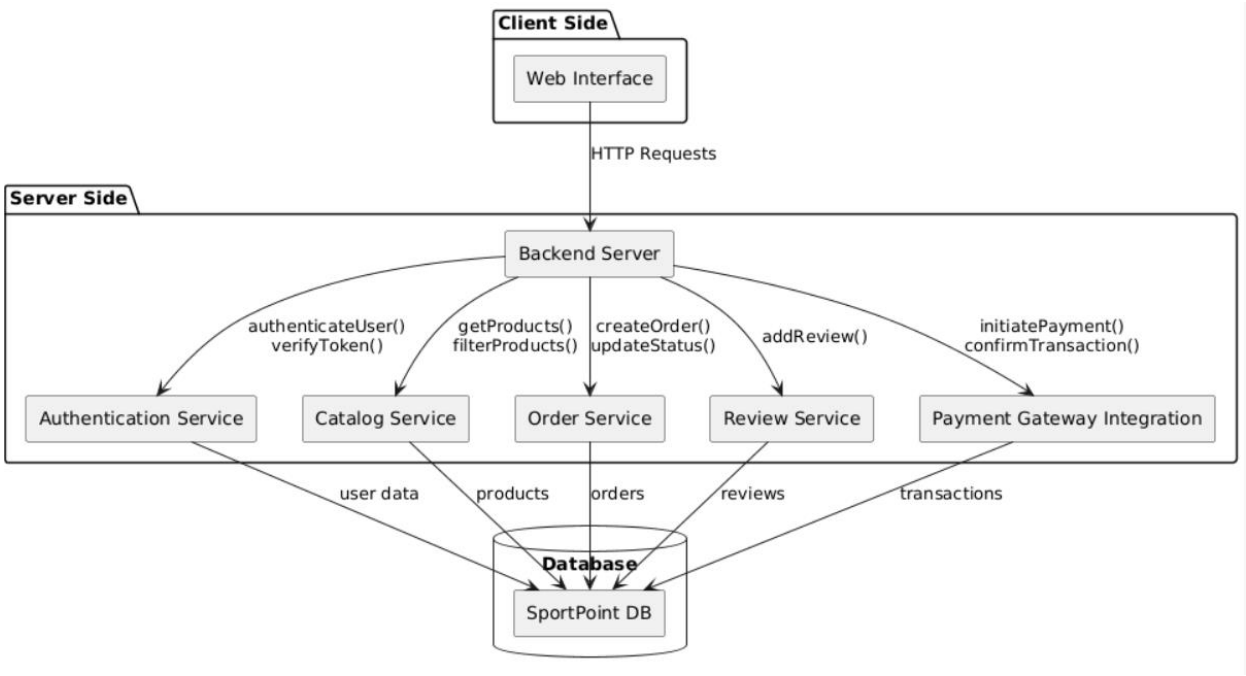

Рис. 2.4. Діаграма компонентів

2.5 Висновки до розділу 2

У другому розділі було виконано комплексне проєктування програмної системи веборієнтованого магазину спортивного екіпірування SportPoint, що охоплює основні етапи архітектурного та логічного моделювання системи.

На основі аналізу предметної області розроблено логічну модель даних, яка визначає ключові сутності системи, їхні атрибути та взаємозв'язки. Це дозволило сформувати цілісну структуру зберігання інформації, що забезпечує підтримку основних бізнес-процесів інтернет-магазину, таких як управління товарами, замовленнями, користувачами, платежами та відгуками.

У межах проєктування програмної архітектури створено діаграму класів і діаграму кооперацій, які відображають структуру програмних об'єктів та їхню взаємодію під час виконання основних сценаріїв роботи системи. Це дало змогу формалізувати логіку функціонування вебзастосунку та визначити основні сутності предметної області на рівні програмної реалізації.

Також розроблено діаграму пакетів, яка дозволила виконати логічне групування компонентів системи на функціональні модулі. Це сприяло підвищенню модульності архітектури, спрощенню супроводу системи та забезпеченню чіткого розподілу відповідальності між її складовими.

Окрім цього, побудовано діаграму компонентів, яка відображає фізичну структуру програмної системи та взаємодію між клієнтською частиною, серверною логікою, сервісами обробки даних і зовнішніми інтеграціями. Це дозволило чітко визначити архітектурні межі системи та механізми взаємодії її основних компонентів.

Таким чином, результати, отримані в другому розділі, формують повну та узгоджену архітектурну модель веборієнтованого магазину SportPoint. Розроблені діаграми забезпечують наочне представлення структури системи, сприяють підвищенню її масштабованості, підтримуваності та є основою для подальшої програмної реалізації.

3. РОЗРОБКА ІНФОРМАЦІЙНОГО ТА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

3.1 Система управління інформаційною базою даних

Вибір системи управління базою даних є одним із ключових етапів проєктування веборієнтованого магазину спортивного екіпірування SportPoint, оскільки саме СУБД забезпечує надійне зберігання, обробку та цілісність усіх даних системи.

Для реалізації бази даних обрано PostgreSQL, яка є однією з найпотужніших відкритих реляційних систем управління базами даних. Такий вибір зумовлений її високою стабільністю, продуктивністю та відповідністю принципам ACID, що гарантує надійне виконання транзакцій під час оформлення замовлень, проведення платежів та оновлення інформації про товари.

Важливою перевагою PostgreSQL є підтримка складних зв’язків між таблицями та можливість ефективної роботи з великими обсягами структурованих даних. Це є критично важливим для системи SportPoint, оскільки вона містить взаємопов’язані сутності, такі як користувачі, товари, категорії, замовлення, платежі та відгуки.

Окремо слід відзначити підтримку формату JSON/JSONB, що дозволяє зберігати гнучкі та розширювані характеристики товарів спортивного екіпірування. Це особливо корисно для реалізації каталогу, де різні категорії продукції можуть мати унікальні атрибути (наприклад, розмір, матеріал, тип навантаження, сезонність тощо).

СУБД PostgreSQL також забезпечує високий рівень масштабованості та оптимізації запитів, що дозволяє ефективно працювати з великою кількістю одночасних користувачів. Це є важливим для інтернет-магазину, де можливі пікові навантаження під час акцій або сезонних розпродажів.

Додатково система підтримує механізми індексації, що значно пришвидшує пошук товарів у каталозі, фільтрацію за параметрами та виконання аналітичних запитів. Це позитивно впливає на продуктивність вебзастосунку та покращує користувацький досвід.

Таким чином, використання PostgreSQL у проєкті SportPoint забезпечує надійну, масштабовану та ефективну основу для зберігання даних, що повністю відповідає вимогам сучасних веборієнтованих систем електронної комерції.

3.2 Розробка інформаційної бази

Інформаційна база побудована на основі логічної моделі даних, яка була розроблена в попередньому розділі. Вона включає основні сутності предметної області, такі як користувачі, товари, категорії, замовлення, елементи замовлень, платежі та відгуки. Кожна сутність реалізована у вигляді окремої таблиці в базі даних із чітко визначеними атрибутами та типами даних.

Таблиця USER використовується для зберігання інформації про зареєстрованих користувачів системи. Вона містить унікальний ідентифікатор, електронну пошту, пароль (у зашифрованому вигляді), ім’я, прізвище, контактні дані та роль користувача. Це дозволяє реалізувати механізми автентифікації та авторизації в системі.

Таблиця PRODUCT призначена для зберігання інформації про товари спортивного екіпірування. Вона включає назву товару, опис, ціну, кількість на складі, зображення та додаткові характеристики. Для підвищення гнучкості структури частина параметрів може зберігатися у форматі JSON, що дозволяє легко розширювати властивості товарів без зміни структури бази даних.

Таблиця CATEGORY забезпечує логічну класифікацію товарів за групами, такими як одяг, взуття, інвентар або аксесуари. Між таблицями CATEGORY та PRODUCT реалізовано зв’язок типу «один-до-багатьох», що дозволяє кожній категорії містити велику кількість товарів.

Процес оформлення покупок реалізується через таблицю ORDER, яка зберігає інформацію про замовлення користувачів. Вона містить дату

створення, статус замовлення, загальну суму та дані про доставку. Кожен користувач може мати декілька замовлень, що реалізує зв’язок «один-до-багатьох» між USER та ORDER.

Оскільки одне замовлення може включати декілька товарів, використовується проміжна таблиця ORDER_ITEM, яка реалізує зв’язок «багато-до-багатьох» між замовленнями та товарами. Вона додатково містить кількість товару та його ціну на момент покупки, що дозволяє зберігати історичну достовірність даних.

Для обробки фінансових операцій використовується таблиця PAYMENT, яка містить інформацію про платежі, включаючи суму, спосіб оплати, дату транзакції та її статус. Кожен платіж пов’язаний із конкретним замовленням, що забезпечує контроль процесу оплати та підтвердження покупки.

Система відгуків реалізована через таблицю REVIEW, яка дозволяє користувачам залишати оцінки та коментарі до товарів. Відгук містить текст, рейтинг, дату створення та зв’язки з користувачем і товаром, що забезпечує формування рейтингової системи продукції.

Таким чином, розроблена інформаційна база даних для системи SportPoint забезпечує повну підтримку бізнес-логіки вебзастосунку, гарантує цілісність даних та створює ефективну основу для роботи інтернет-магазину спортивного екіпірування.

3.3 Вибір інструментарію та технологічного стеку розробки

Для реалізації веборієнтованого магазину спортивного екіпірування SportPoint обрано сучасний технологічний стек, який забезпечує високу продуктивність, масштабованість та зручність розробки. Основою клієнтської та серверної частини вебзастосунку виступає фреймворк Next.js 15, що базується на бібліотеці React та дозволяє створювати повноцінні full-stack вебдодатки.

Використання Next.js 15 зумовлене його підтримкою сучасної архітектури App Router, яка забезпечує гнучке управління маршрутами,

серверними компонентами та оптимізованою структурою проєкту. Це дозволяє ефективно реалізувати сторінки каталогу товарів, кошика, оформлення замовлення та особистого кабінету користувача.

Однією з ключових переваг Next.js є підтримка Server-Side Rendering (SSR) та Static Site Generation (SSG), що значно покращує продуктивність системи та SEO-оптимізацію. Для інтернет-магазину це є критично важливим, оскільки забезпечує швидке завантаження сторінок та кращу видимість у пошукових системах.

Для реалізації серверної логіки використовуються API Routes та серверні компоненти Next.js, що дозволяє об’єднати frontend і backend в одному проєкті. Це спрощує архітектуру системи та зменшує складність взаємодії між клієнтською та серверною частинами.

Для роботи з базою даних PostgreSQL використовується ORM-рішення (наприклад, Prisma), яке забезпечує зручну взаємодію з даними, типізацію та автоматизацію SQL-запитів. Це підвищує надійність роботи з інформаційною базою та зменшує кількість помилок при розробці.

Для стилізації інтерфейсу застосовується сучасний підхід із використанням Tailwind CSS, який дозволяє швидко створювати адаптивний та привабливий UI без необхідності написання великої кількості кастомного CSS-коду. Це забезпечує єдиний стиль інтерфейсу та високу швидкість розробки.

У якості середовища розробки використовується Visual Studio Code, який надає широкі можливості для роботи з JavaScript/TypeScript проєктами, інтеграцію з Git та підтримку розширень для Next.js і Prisma.

Також для керування версіями коду застосовується система контролю версій Git, а репозиторій може бути розміщений на платформі GitHub, що забезпечує зручну командну роботу та збереження історії змін проєкту.

Таким чином, обраний технологічний стек на основі Next.js 15, React, PostgreSQL, Prisma та Tailwind CSS забезпечує сучасний підхід до розробки

вебзастосунку, високу продуктивність, гнучкість архітектури та можливість подальшого масштабування системи SportPoint.

3.4 Алгоритмізація та програмування модулів

Одним із основних процесів системи є алгоритм перегляду каталогу товарів. Користувач надсилає запит на отримання списку продукції, після чого система виконує звернення до бази даних, застосовує фільтрацію за категоріями, ціною або характеристиками та повертає сформований набір товарів для відображення у вебінтерфейсі. Додатково реалізується механізм пагінації для оптимізації завантаження великих обсягів даних.

Важливим алгоритмом є процес додавання товару до кошика. Після вибору користувачем конкретного товару система перевіряє його наявність на складі, визначає кількість доступного залишку та формує або оновлює запис у кошику. У разі повторного додавання товару виконується збільшення кількості позицій, що дозволяє уникнути дублювання записів.

Ключовим бізнес-процесом є алгоритм оформлення замовлення. Після ініціації користувачем оформлення покупки система перевіряє склад кошика, розраховує загальну суму замовлення, фіксує актуальні ціни товарів та створює запис у таблиці ORDER. Далі формується набір записів у таблиці ORDER_ITEM, після чого система блокує відповідну кількість товарів на складі для уникнення перевищення доступних залишків.

Наступним етапом є алгоритм обробки платежу. Система ініціює запит до платіжного шлюзу, передає суму замовлення та отримує статус транзакції. У разі успішної оплати оновлюється статус замовлення, а інформація про платіж зберігається в таблиці PAYMENT. У випадку помилки оплати замовлення переводиться у статус «очікує оплати» або «скасовано» залежно від бізнес-логіки.

Окремо реалізується алгоритм роботи з відгуками. Після завершення покупки користувач має можливість залишити відгук про товар. Система перевіряє факт придбання товару, після чого дозволяє створення запису в

таблиці REVIEW. Це забезпечує достовірність рейтингової системи та запобігає фальшивим оцінкам.

Також важливим є алгоритм автентифікації користувачів, який включає перевірку облікових даних, генерацію токена доступу та керування сесією. Усі захищені операції системи виконуються лише після успішної авторизації користувача, що забезпечує безпеку персональних даних.

Реалізація всіх алгоритмів здійснюється із використанням сучасних можливостей Next.js 15, серверних компонентів та API Routes. Для роботи з базою даних використовується ORM, що дозволяє ефективно виконувати CRUD-операції та забезпечує цілісність даних.

Таким чином, алгоритмізація та програмування модулів системи SportPoint забезпечують реалізацію основних бізнес-процесів інтернет-магазину, підвищують стабільність роботи системи та створюють основу для її подальшого розвитку та масштабування.

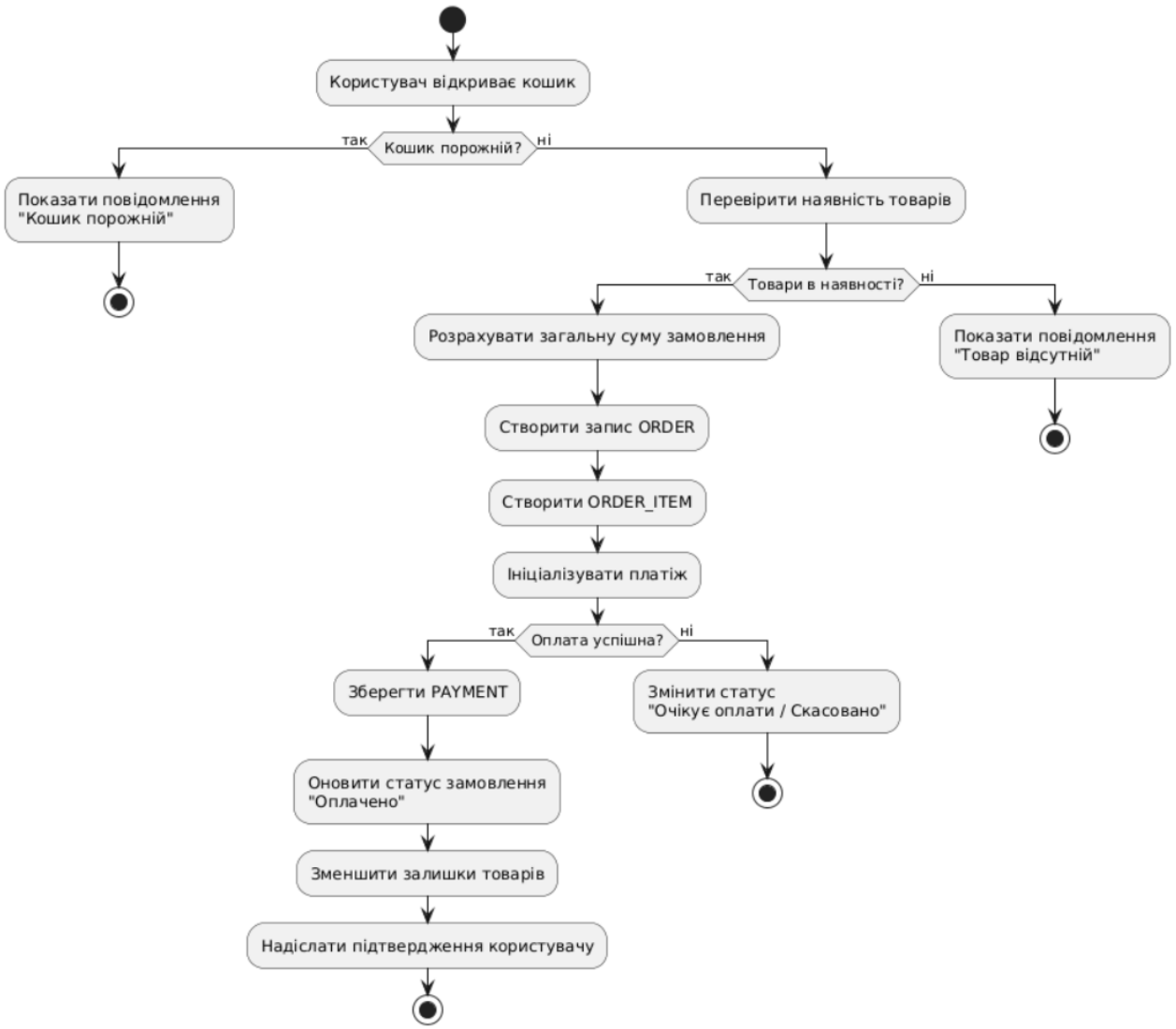

Рис. 3.1. Блок-схема алгоритму оформлення замовлення

3.5 Опис реалізованого програмного продукту

Реалізований програмний продукт SportPoint є веборієнтованим інтернет-магазином спортивного екіпірування, який забезпечує користувачам зручний доступ до каталогу товарів, фільтрацію продукції та взаємодію з інтерфейсом через сучасний адаптивний дизайн.

Головна сторінка системи містить логотип магазину, рядок пошуку та основні елементи навігації, що включають іконки профілю користувача, обраних товарів та кошика. Це забезпечує швидкий доступ до ключових функцій системи та покращує користувацький досвід.

Основним функціональним блоком є сторінка каталогу товарів, яка реалізує відображення продукції у вигляді карток. Кожна картка товару містить зображення, назву, артикул, ціну та кнопку для додавання до обраного. Такий підхід дозволяє користувачу швидко оцінити основні характеристики товару та прийняти рішення щодо покупки.

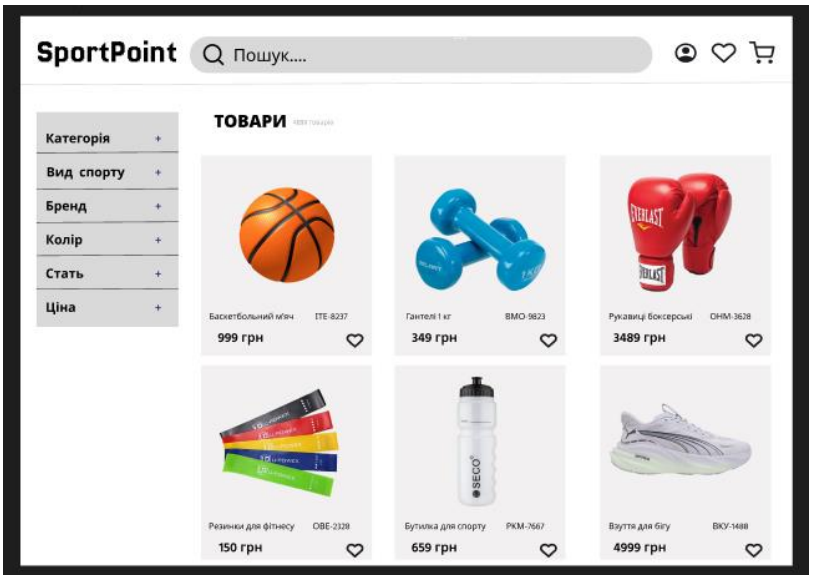

Рис 3.2. Загальний інтерфейс головної сторінки веб-додатку

Для зручності навігації реалізовано систему фільтрів, розташовану у лівій частині інтерфейсу. Вона дозволяє здійснювати сортування та відбір товарів за різними параметрами, такими як категорія, вид спорту, бренд, колір, стать та ціновий діапазон. Це забезпечує гнучкий пошук необхідної продукції та підвищує ефективність роботи з каталогом.

Представлений екран демонструє сторінку підкатегорії товарів, зокрема розділ балансувальних платформ. Тут реалізовано ієрархічну структуру каталогу, де користувач може переходити між категоріями та переглядати відповідні товари. Це сприяє логічній організації асортименту та спрощує навігацію по магазину.

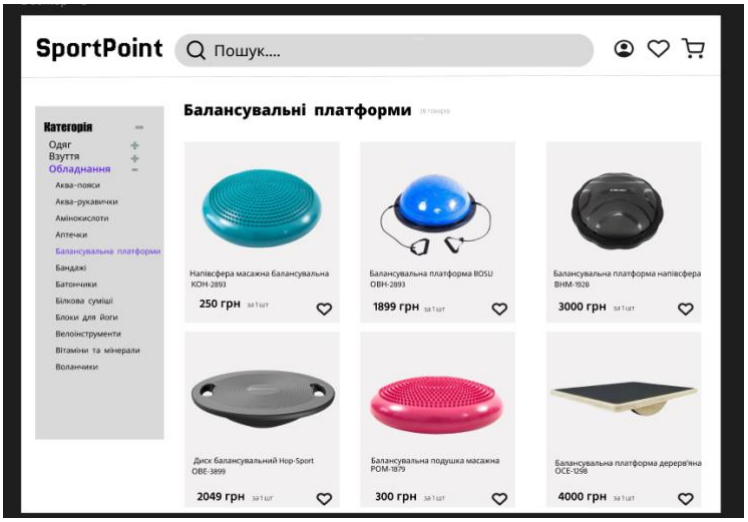

Рис 3.2. Інтерфейс сторінки певної категорії та фільтрація

Представлений екран демонструє сторінку реєстрації користувача вебзастосунку SportPoint. Інтерфейс виконаний у мінімалістичному стилі та містить центральну форму авторизації, що забезпечує швидкий доступ до основних функцій входу та створення нового облікового запису. У верхній частині сторінки розташована назва системи, що дозволяє ідентифікувати сервіс та підкреслює його брендову належність.

Для зручності користувачів реалізовано вкладки «Увійти» та «Зареєструватися», між якими можна перемикатися залежно від необхідної дії. Активна вкладка візуально виділена, що спрощує орієнтацію в інтерфейсі та робить процес взаємодії більш інтуїтивним.

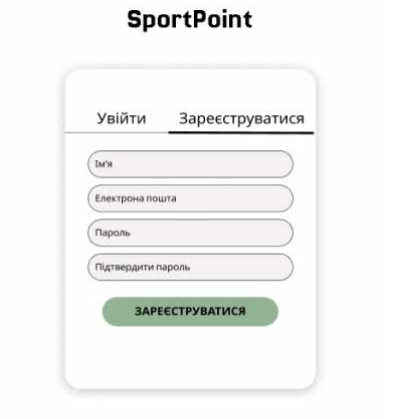

Рис 3.3. Інтерфейс сторінки Реєстрації

Для здійснення покупки користувач переходить на сторінку товару, де відображається детальна інформація про обрану продукцію. На сторінці представлено галерею зображень, назву товару, рейтинг, вартість, артикул та доступні розміри. Також користувач може ознайомитися з описом і технічними характеристиками товару, після чого додати його до кошика за допомогою кнопки «Купити». Така організація інтерфейсу забезпечує зручний перегляд продукції та спрощує процес прийняття рішення щодо її придбання.

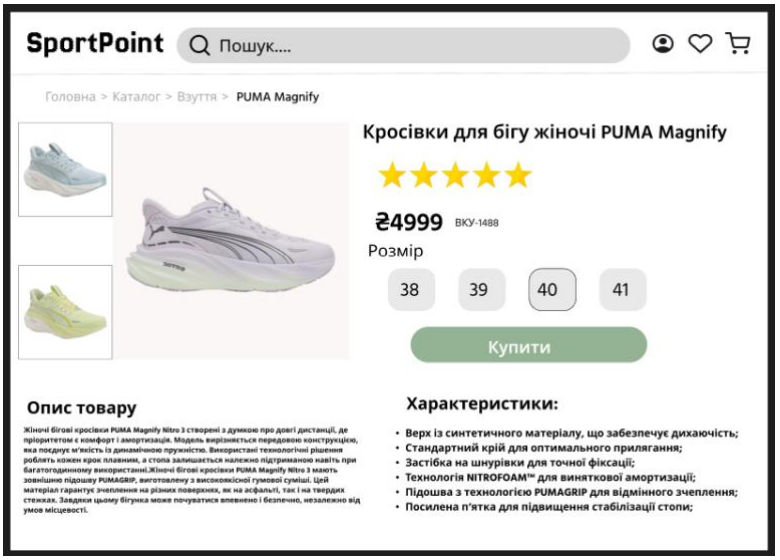

Рис 3.3. Інтерфейс сторінки товару

Після вибору товару та натискання кнопки «Купити» користувачу відображається модальне вікно підтвердження додавання товару до кошика. У вікні міститься коротка інформація про обраний товар, зокрема його назва, розмір та ціна. Користувач може продовжити покупки або перейти безпосередньо до кошика для подальшого оформлення замовлення. Такий підхід забезпечує зручний контроль дій користувача та покращує взаємодію з системою під час процесу покупки.

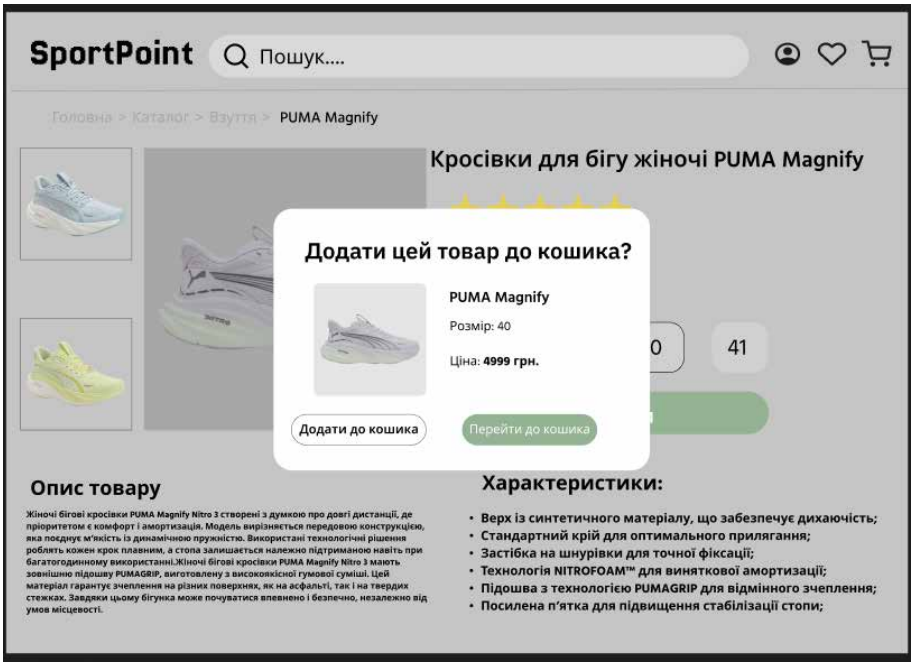

Рис 3.3. Модальне вікно підтвердження додавання товару

На сторінці оформлення замовлення користувач заповнює необхідні дані для доставки та оплати обраних товарів. Інтерфейс містить форму для введення контактної інформації, адреси доставки та вибору способу оплати. Також на сторінці відображається підсумок замовлення із переліком обраних товарів, їх кількістю, вартістю та загальною сумою до сплати. Після перевірки введених даних користувач може підтвердити замовлення за допомогою відповідної кнопки. Така організація інтерфейсу забезпечує зручний процес оформлення покупки та мінімізує ймовірність помилок під час введення даних.

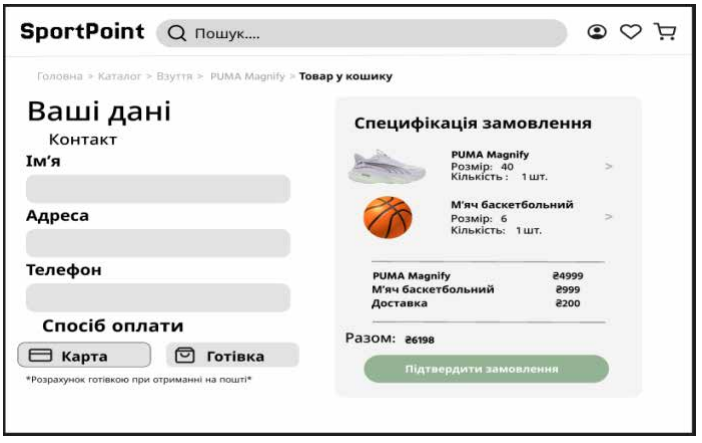

Рис 3.3. Інтерфейс сторінки оформлення замовлення

3.6 Висновки до розділу 3

У третьому розділі було розглянуто процес проєктування та реалізації програмного забезпечення веборієнтованого магазину спортивного екіпірування SportPoint. Обґрунтовано вибір системи управління базами даних PostgreSQL, яка забезпечує надійне зберігання інформації, підтримку транзакцій, високу продуктивність та можливість масштабування системи.

На основі розробленої логічної моделі створено інформаційну базу даних, що охоплює основні сутності предметної області: користувачів, товари, категорії, замовлення, платежі та відгуки. Визначено структуру таблиць, атрибути та зв’язки між ними, що забезпечує цілісність і узгодженість даних у системі.

Також було обрано сучасний технологічний стек розробки, до складу якого входять Next.js 15, React, PostgreSQL, Prisma та Tailwind CSS. Використання зазначених технологій дозволило реалізувати продуктивний, масштабований та зручний у супроводі вебзастосунок.

У процесі алгоритмізації було розроблено основні бізнес-процеси системи, зокрема перегляд каталогу товарів, роботу з кошиком, оформлення замовлень, обробку платежів, управління відгуками та автентифікацію користувачів. Реалізація цих алгоритмів забезпечує коректне функціонування всіх ключових модулів інтернет-магазину.

У результаті виконаних робіт створено функціональний програмний продукт SportPoint, який надає користувачам можливість переглядати каталог спортивного екіпірування, здійснювати пошук і фільтрацію товарів, реєструватися в системі, формувати замовлення та виконувати покупки через сучасний вебінтерфейс. Розроблене програмне забезпечення відповідає поставленим вимогам, забезпечує зручність використання та може слугувати основою для подальшого розвитку й розширення функціональних можливостей системи.

4. РЕКОМЕНДАЦІЇ ЩОДО ВПРОВАДЖЕННЯ ТА ЕКСПЛУТАЦІЇ СИСТЕМИ

4.1. Тестування системи

Тестування програмного забезпечення є важливим етапом розробки веборієнтованого магазину спортивного екіпірування SportPoint, оскільки дозволяє перевірити коректність роботи реалізованого функціоналу та забезпечити відповідність системи поставленим вимогам.

У процесі тестування було виконано перевірку основних функціональних модулів системи. Зокрема, протестовано механізми реєстрації та авторизації користувачів, перегляд каталогу товарів, роботу пошукової системи та фільтрів, додавання товарів до кошика, оформлення замовлень і взаємодію з базою даних.

Для перевірки коректності роботи інтерфейсу виконано функціональне тестування вебзастосунку в різних сценаріях використання. Було перевірено правильність відображення сторінок, коректність переходів між розділами та обробку введених користувачем даних. Особливу увагу приділено перевірці адаптивності інтерфейсу та стабільності роботи елементів керування.

Також проведено тестування бази даних, під час якого перевірялися операції створення, читання, оновлення та видалення записів. Результати тестування підтвердили коректність реалізації зв’язків між таблицями та збереження цілісності даних під час виконання транзакцій.

Для оцінювання продуктивності системи було виконано навантажувальне тестування, яке показало стабільну роботу вебзастосунку при одночасній роботі декількох користувачів. Перевірка підтвердила відсутність критичних помилок, що можуть впливати на працездатність основних бізнес-процесів.

За результатами проведеного тестування встановлено, що вебзастосунок SportPoint коректно виконує всі передбачені функції, забезпечує надійне зберігання та обробку даних, а також відповідає вимогам, визначеним на етапі

проєктування. Отримані результати свідчать про готовність програмного продукту до впровадження та подальшої експлуатації.

4.2 Вимоги до апаратного та програмного забезпечення

Для забезпечення стабільної роботи веборієнтованого магазину спортивного екіпірування SportPoint необхідно дотримуватися певних вимог до апаратного та програмного забезпечення як серверної, так і клієнтської частини системи.

Серверна частина вебзастосунку повинна функціонувати на комп'ютері або в хмарному середовищі, що підтримує роботу платформи Node.js та системи управління базами даних PostgreSQL. Мінімальні апаратні вимоги для сервера включають двоядерний процесор із тактовою частотою не менше 2,0 ГГц, 4 ГБ оперативної пам'яті та не менше 20 ГБ вільного дискового простору. Для забезпечення стабільної роботи в умовах підвищеного навантаження рекомендується використовувати сервер із чотириядерним процесором, 8 ГБ оперативної пам'яті та SSD-накопичувачем.

У якості операційної системи можуть використовуватися сучасні версії Linux (Ubuntu Server, Debian), Windows Server або інші системи, сумісні з Node.js та PostgreSQL. Для запуску вебзастосунку необхідно встановити середовище виконання Node.js версії 20 або новішої, систему управління базами даних PostgreSQL, а також залежності проєкту, визначені у файлі package.json.

Клієнтська частина системи не потребує спеціалізованого обладнання та може використовуватися на персональних комп'ютерах, ноутбуках, планшетах і смартфонах. Мінімальні вимоги передбачають наявність сучасного веббраузера з підтримкою стандартів HTML5, CSS3 та JavaScript. Для коректної роботи рекомендується використовувати останні версії браузерів Google Chrome, Mozilla Firefox, Microsoft Edge або Safari.

Для розробки та супроводу програмного продукту рекомендується використовувати середовище розробки Visual Studio Code, систему контролю

версій Git та платформу GitHub для зберігання вихідного коду. Додатково можуть використовуватися інструменти адміністрування бази даних PostgreSQL та засоби моніторингу серверного навантаження.

Таким чином, зазначені апаратні та програмні вимоги забезпечують коректне функціонування вебзастосунку SportPoint, стабільну роботу його компонентів та можливість подальшого розвитку системи.

4.3 Склад інсталяційного пакет

Інсталяційний пакет веборієнтованого магазину спортивного екіпірування SportPoint містить набір програмних компонентів, необхідних для розгортання, налаштування та подальшої експлуатації системи. Його структура забезпечує швидке встановлення програмного продукту та спрощує процес адміністрування.

До складу інсталяційного пакету входять вихідні файли вебзастосунку, розробленого з використанням фреймворку Next.js, включаючи клієнтську та серверну частини системи. Також пакет містить конфігураційні файли проєкту, необхідні для налаштування параметрів середовища виконання, підключення до бази даних та роботи зовнішніх сервісів.

Окремим компонентом є структура бази даних PostgreSQL, яка включає SQL-скрипти створення таблиць, зв’язків, індексів та початкових службових даних. Це дозволяє автоматизувати процес підготовки інформаційної бази під час встановлення системи.

До інсталяційного пакету також входять файли залежностей проєкту, описані у файлі package.json, які забезпечують встановлення необхідних програмних бібліотек та модулів. Для роботи системи використовуються бібліотеки React, Next.js, Prisma, Tailwind CSS та інші компоненти, що забезпечують реалізацію функціональних можливостей вебзастосунку.

Крім основних програмних модулів, пакет містить документацію користувача та адміністратора, інструкцію з встановлення, налаштування та запуску системи. Це дозволяє спростити процес впровадження програмного продукту та зменшити час на його підготовку до експлуатації.

4.4 Висновки до розділу 4

У четвертому розділі було розглянуто питання впровадження, експлуатації та перевірки працездатності веборієнтованого магазину спортивного екіпірування SportPoint. Сформовано рекомендації щодо розгортання програмного забезпечення, налаштування серверного середовища, забезпечення безпеки даних та організації процесу супроводу системи.

У ході роботи визначено вимоги до апаратного та програмного забезпечення, необхідного для стабільного функціонування вебзастосунку. Встановлено мінімальні та рекомендовані характеристики серверної інфраструктури, а також перелік програмних компонентів, необхідних для розгортання та експлуатації системи.

Особливу увагу приділено тестуванню програмного продукту. Було перевірено коректність роботи основних функціональних модулів, зокрема механізмів реєстрації та авторизації користувачів, роботи каталогу товарів, кошика, оформлення замовлень та взаємодії з базою даних. Результати тестування підтвердили працездатність системи та відповідність реалізованого функціоналу встановленим вимогам.

Також визначено склад інсталяційного пакету, який включає вихідний код вебзастосунку, конфігураційні файли, схему бази даних, програмні залежності та супровідну документацію. Це забезпечує можливість швидкого розгортання та подальшого супроводу системи.

Таким чином, результати, отримані в межах четвертого розділу, підтверджують готовність вебзастосунку SportPoint до впровадження та практичного використання. Розроблена система забезпечує стабільну роботу,

зручність експлуатації, можливість масштабування та відповідає сучасним вимогам до програмних засобів електронної комерції.

ВИСНОВКИ

У результаті виконання кваліфікаційної роботи було розроблено веборієнтований застосунок для продажу спортивного екіпірування SportPoint, який забезпечує автоматизацію основних процесів електронної комерції та відповідає сучасним вимогам до функціональності, продуктивності, безпеки та масштабованості програмного забезпечення.

У ході дослідження проведено аналіз предметної області веборієнтованої електронної комерції у сфері реалізації спортивного екіпірування, визначено особливості функціонування сучасних інтернет-магазинів та досліджено існуючі аналоги. Виконаний аналіз дозволив сформувати функціональні та нефункціональні вимоги до програмної системи, а також обґрунтувати вибір архітектурних підходів і сучасних вебтехнологій.

У процесі проєктування було виконано моделювання предметної області із використанням UML-діаграм, визначено основні сутності системи, їхні взаємозв’язки та логіку функціонування. Розроблено логічну модель даних у вигляді ER-діаграми, діаграми класів, пакетів, компонентів і кооперацій, що дозволило сформувати цілісну архітектурну модель вебзастосунку та забезпечити структурованість програмного рішення.

Для реалізації програмного продукту обрано сучасний технологічний стек, що включає Next.js 15, React, PostgreSQL, Prisma та Tailwind CSS. Використання зазначених технологій забезпечило створення продуктивного, адаптивного та масштабованого вебзастосунку з можливістю ефективної взаємодії між клієнтською та серверною частинами системи.

У межах розробки створено інформаційну базу даних, яка забезпечує зберігання та обробку інформації про користувачів, товари, категорії, замовлення, платежі та відгуки. Реалізовано основні бізнес-процеси системи, зокрема механізми реєстрації та авторизації користувачів, перегляду каталогу продукції, пошуку та фільтрації товарів, формування кошика, оформлення замовлень, обробки платежів і взаємодії із системою відгуків.

У процесі тестування підтверджено коректність роботи основних функціональних модулів вебзастосунку, стабільність взаємодії з базою даних, працездатність алгоритмів обробки замовлень та адаптивність користувацького інтерфейсу. Отримані результати свідчать про відповідність реалізованого програмного продукту поставленим вимогам і готовність системи до впровадження та практичного використання.

Практичне значення роботи полягає у створенні функціонального веборієнтованого магазину спортивного екіпірування, який забезпечує зручний процес взаємодії між покупцем і продавцем, підвищує ефективність управління асортиментом продукції та може бути використаний як основа для подальшого розвитку електронної комерції у сфері спортивних товарів.

Таким чином, поставлена мета роботи досягнута, а всі визначені завдання виконані. Розроблений вебзастосунок SportPoint відповідає сучасним вимогам до систем електронної комерції, забезпечує зручність використання, високу продуктивність, надійність та можливість подальшого функціонального розширення.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Sommerville I. Software Engineering. – 10th ed. – Boston : Pearson, 2016. – 816 p.
2. Pressman R. S., Maxim B. R. Software Engineering: A Practitioner’s Approach. – 9th ed. – New York : McGraw-Hill Education, 2019. – 992 p.
3. Fowler M. UML Distilled: A Brief Guide to the Standard Object Modeling Language. – 3rd ed. – Boston : Addison-Wesley, 2004. – 208 p.
4. Ambler S. W. The Elements of UML 2.0 Style. – Cambridge : Cambridge University Press, 2005. – 328 p.
5. Elmasri R., Navathe S. Fundamentals of Database Systems. – 7th ed. – Boston : Pearson, 2016. – 1272 p.
6. Silberschatz A., Korth H., Sudarshan S. Database System Concepts. – 7th ed. – New York : McGraw-Hill Education, 2019. – 1376 p.
7. Date C. J. An Introduction to Database Systems. – 8th ed. – Boston : Pearson, 2003. – 1024 p.
8. PostgreSQL Global Development Group. PostgreSQL Documentation [Electronic resource]. – Access mode: https://www.postgresql.org/docs/
9. React Documentation [Electronic resource]. – Access mode: https://react.dev/
10. Next.js Documentation [Electronic resource]. – Access mode: https://nextjs.org/docs
11. Prisma ORM Documentation [Electronic resource]. – Access mode: https://www.prisma.io/docs
12. Tailwind CSS Documentation [Electronic resource]. – Access mode: https://tailwindcss.com/docs
13. Mozilla Developer Network (MDN) Web Docs [Electronic resource]. –

Access mode: https://developer.mozilla.org/

14. Gamma E., Helm R., Johnson R., Vlissides J. Design Patterns: Elements of Reusable Object-Oriented Software. – Boston : Addison-Wesley, 1994. – 395 p.
15. Evans E. Domain-Driven Design: Tackling Complexity in the Heart of Software. – Boston : Addison-Wesley, 2003. – 560 p.
16. Martin R. C. Clean Architecture: A Craftsman's Guide to Software Structure and Design. – Boston : Pearson, 2017. – 432 p.
17. Martin R. C. Clean Code: A Handbook of Agile Software Craftsmanship. – Boston : Prentice Hall, 2008. – 464 p.
18. Laudon K. C., Traver C. G. E-Commerce: Business, Technology, Society. – 18th ed. – Pearson, 2022. – 912 p.
19. Welling L., Thomson L. PHP and MySQL Web Development. – 5th ed. – Addison-Wesley, 2016. – 768 p.
20. Google Developers. Web Performance Optimization Guide [Electronic resource]. – Access mode: https://developers.google.com/
21. Nielsen J. Usability Engineering. – Boston : Morgan Kaufmann, 1994. – 362 p.
22. ISO/IEC 25010:2011 Systems and Software Engineering — Systems and Software Quality Requirements and Evaluation (SQuaRE). – Geneva : ISO, 2011.
23. Fielding R. Architectural Styles and the Design of Network-based Software Architectures : Doctoral dissertation. – University of California, Irvine, 2000.
24. Oracle Corporation. SQL Language Reference [Electronic resource]. – Access mode: https://docs.oracle.com/

25. Git Documentation [Electronic resource]. – Access mode: https://git-scm.com/doc
26. GitHub Documentation [Electronic resource]. – Access mode: https://docs.github.com/
27. Visual Studio Code Documentation [Electronic resource]. – Access mode: https://code.visualstudio.com/docs
28. Node.js Documentation [Electronic resource]. – Access mode: https://nodejs.org/docs/latest/api/

ДОДАТКИ

ДОДАТОК А

```
import React, { useState } from 'react';
import './ProductPage.css';

const ProductPage = ({ product }) => {
  const [selectedSize, setSelectedSize] = useState(null);
  const [isModalOpen, setIsModalOpen] = useState(false);

  const handleAddToCart = () => {
    if (!selectedSize) {
      alert("Будь ласка, оберіть розмір перед додаванням у кошик.");
      return;
    }
    setIsModalOpen(true);
  };

  return (
    <div className="product-page-container">
      <nav className="breadcrumbs">
        <span>Головна</span> > <span>Каталог</span> > <span>Взуття</span> >
<span>{product.name}</span>
      </nav>

      <main className="product-details">
        <div className="product-gallery">
          <div className="thumbnails">
            {product.images.map((img, index) => (
              <img key={index} src={img.thumbnail} alt={`Thumbnail ${index}`} />
            ))}
          </div>
          <div className="main-image">
            <img src={product.images[0].fullSize} alt={product.name} />
          </div>
        </div>

        <div className="product-info">
```

```
    <h1>{product.name}</h1>
    <div className="rating">★★★★★ (148)</div>
    <div className="price">{product.price} грн</div>

    <div className="size-selector">
      <p>Розмір</p>
      <div className="sizes">
        {[38, 39, 40, 41].map(size => (
          <button
            key={size}
            className={`size-btn ${selectedSize === size ? 'active' : ''}`}
            onClick={() => setSelectedSize(size)}
          >
            {size}
          </button>
        ))}
      </div>
    </div>

    <button className="buy-button" onClick={handleAddToCart}>
      Купити
    </button>

    <div className="details-section">
      <div className="description">
        <h3>Опис товару</h3>
        <p>{product.description}</p>
      </div>
      <div className="characteristics">
        <h3>Характеристики:</h3>
        <ul>
          {product.characteristics.map((item, idx) => (
            <li key={idx}>{item}</li>
          ))}
        </ul>
      </div>
    </div>
  </div>
</main>
```

```
      {isModalOpen && (
        <div className="modal-overlay">
          <div className="modal-content">
            <h3>Додати цей товар до кошика?</h3>
            <div className="modal-product-summary">
              <img src={product.images[0].thumbnail} alt={product.name} />
              <div>
                <p>{product.name}</p>
                <p>Розмір: {selectedSize}</p>
                <p>Ціна: {product.price} грн</p>
              </div>
            </div>
            <div className="modal-actions">
              <button onClick={() => setIsModalOpen(false)}>Продовжити
покупки</button>
              <button className="go-to-cart">Перейти до кошика</button>
            </div>
          </div>
        </div>
      )}
    </div>
  );
};

export default ProductPage;
```

ДОДАТОК Б

```
const express = require('express');
const router = express.Router();
const Product = require('../models/Product');

router.get('/', async (req, res) => {
  try {
    const {
      search,
      category,
      sportType,
      minPrice,
      maxPrice,
      page = 1,
      limit = 12
    } = req.query;

    let query = {};

    if (search) {
      query.name = { $regex: search, $options: 'i' };
    }

    if (category) {
      query.category = category;
    }

    if (sportType) {
```

```
    query.sportType = sportType;
  }

  if (minPrice || maxPrice) {
    query.price = {};
    if (minPrice) query.price.$gte = Number(minPrice);
    if (maxPrice) query.price.$lte = Number(maxPrice);
  }

  const skip = (page - 1) * limit;

  const products = await Product.find(query)
    .skip(skip)
    .limit(Number(limit))
    .sort({ createdAt: -1 });

  const totalItems = await Product.countDocuments(query);

  res.status(200).json({
    success: true,
    data: products,
    pagination: {
      totalItems,
      totalPages: Math.ceil(totalItems / limit),
      currentPage: Number(page)
    }
  });

} catch (error) {
```

```
    console.error('Помилка отримання каталогу товарів:', error);
    res.status(500).json({
      success: false,
      message: 'Внутрішня помилка сервера при завантаженні каталогу'
    });
  }
});

module.exports = router;
```

ДОДАТОК В

```
import React, { useState } from 'react';
import './AuthModal.css';

const AuthModal = () => {
  const [activeTab, setActiveTab] = useState('register');

  const [formData, setFormData] = useState({
    name: '',
    email: '',
    password: '',
    confirmPassword: ''
  });

  const handleInputChange = (e) => {
    const { name, value } = e.target;
    setFormData(prevState => ({
      ...prevState,
      [name]: value
    }));
  };

  const handleSubmit = async (e) => {
    e.preventDefault();
    if (activeTab === 'register') {
      if (formData.password !== formData.confirmPassword) {
        alert('Паролі не співпадають!');
        return;
      }
```

```
    console.log('Реєстрація користувача:', formData);
  } else {
    console.log('Вхід користувача:', formData.email, formData.password);
  }
};

return (
  <div className="auth-container">
    <h2 className="logo">SportPoint</h2>

    <div className="auth-card">
      <div className="auth-tabs">
        <button
          className={`tab ${activeTab === 'login' ? 'active' : ''}`}
          onClick={() => setActiveTab('login')}
        >
          Увійти
        </button>
        <button
          className={`tab ${activeTab === 'register' ? 'active' : ''}`}
          onClick={() => setActiveTab('register')}
        >
          Зареєструватися
        </button>
      </div>

      <form onSubmit={handleSubmit} className="auth-form">
        {activeTab === 'register' && (
          <input
```

```
    type="text"
    name="name"
    placeholder="Ім'я"
    value={formData.name}
    onChange={handleInputChange}
    required
  />
)}

<input
  type="email"
  name="email"
  placeholder="Електронна пошта"
  value={formData.email}
  onChange={handleInputChange}
  required
/>

<input
  type="password"
  name="password"
  placeholder="Пароль"
  value={formData.password}
  onChange={handleInputChange}
  required
/>

{activeTab === 'register' && (
  <input
```

```
            type="password"
            name="confirmPassword"
            placeholder="Підтвердити пароль"
            value={formData.confirmPassword}
            onChange={handleInputChange}
            required
          />
        )}

        <button type="submit" className="submit-btn">
          {activeTab === 'login' ? 'УВІЙТИ' : 'ЗАРЕЄСТРУВАТИСЯ'}
        </button>
      </form>
    </div>
  </div>
 );
};

export default AuthModal;
```

ДОДАТОК Г

```
const mongoose = require('mongoose');

const productSchema = new mongoose.Schema({
  name: {
    type: String,
    required: [true, "Назва товару є обов'язковою"],
    trim: true,
    maxlength: [100, "Назва не може перевищувати 100 символів"]
  },
  price: {
    type: Number,
    required: [true, "Вкажіть ціну товару"],
    min: [0, "Ціна не може бути від'ємною"]
  },
  description: {
    type: String,
    required: [true, "Додайте опис товару"]
  },
  category: {
    type: String,
    enum: ['Одяг', 'Взуття', 'Обладнання', 'Аксесуари'],
    required: true
  },
  sportType: {
    type: String,
    required: true
  },
  characteristics: [{
```

```
    type: String
  }],
  sizes: [{
    type: Number
  }],
  images: [{
    thumbnail: String,
    fullSize: String
  }],
  rating: {
    average: {
      type: Number,
      default: 0,
      min: 0,
      max: 5
    },
    count: {
      type: Number,
      default: 0
    }
  },
  inStock: {
    type: Boolean,
    default: true
  }
}, {
  timestamps: true
});
```

```
productSchema.index({ name: 'text', description: 'text' });

module.exports = mongoose.model('Product', productSchema);
```

ДОДАТОК Д

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ БІОРЕСУРСІВ І ПРИРОДОКОРИСТУВАННЯ УКРАЇНИ

Факультет інформаційних технологій

Декларація про академічну доброчесність та використання ШІ

Я, Пащенко Максим Юрійович, здобувач(ка) НУБіП України, усвідомлюючи відповідальність за порушення академічної доброчесності, цією заявою підтверджую, що:

1. Моя бакалаврська кваліфікаційна робота на тему «Програмне забезпечення веб-орієнтованого додатку для реалізації спортивного екіпірування» є результатом моєї особистої праці.
2. Усі запозичення з текстів інших авторів мають відповідні посилання згідно з чинними стандартами.
3. Щодо використання інструментів ШІ зазначаю, що:

інструменти ШІ Gemini були використані для:

- перекладу іншомовних джерел;
- допомоги у написанні/відлагодженні програмного коду;

Я розумію, що надання недостовірної інформації може призвести до скасування результатів захисту та відрахування з числа здобувачів НУБіП України.

«___»_____ 2026 р. _____ **Максим ПАЩЕНКО**

(підпис)