Міністерство освіти і науки України
Тернопільський національний технічний університет імені Івана Пулюя

Факультет комп'ютерно-інформаційних систем і програмної інженерії
(повна назва факультету)
Кафедра комп'ютерних наук
(повна назва кафедри)

КВАЛІФІКАЦІЙНА РОБОТА

на здобуття освітнього ступеня

бакалавр
(назва освітнього ступеня)

на тему: Розробка інтернет-магазину з функціональними можливостями аналітики, авторизації та оптимізації інтерфейсу

Виконав: студент IV курсу, групи СНс-41
спеціальності 122 Комп’ютерні науки
(шифр і назва спеціальності)

(підпис) Самарик В.М. (прізвище та ініціали)

Керівник (підпис) Липак Г.І. (прізвище та ініціали)

Нормоконтроль (підпис) Шимчук Г.В. (прізвище та ініціали)

Завідувач кафедри (підпис) Боднарчук І.О. (прізвище та ініціали)

Рецензент (підпис) Тиш Є.В. (прізвище та ініціали)

Тернопіль
2026

Міністерство освіти і науки України
Тернопільський національний технічний університет імені Івана Пулюя

Факультет комп'ютерно-інформаційних систем і програмної інженерії
(повна назва факультету)
Кафедра комп'ютерних наук
(повна назва кафедри)

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____ Боднарчук І.О.
(підпис) (прізвище та ініціали)
«___» _червня_ 2026 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

на здобуття освітнього ступеня Бакалавр
(назва освітнього ступеня)
за спеціальністю 122 Комп'ютерні науки
(шифр і назва спеціальності)
Студентці Самарик Вікторії Мирославівні
(прізвище, ім'я, по батькові)
1. Тема роботи Розробка інтернет-магазину з функціональними можливостями аналітики, авторизації та оптимізації інтерфейсу

Керівник роботи Липак Галина Ігорівна, к. соц. комун., доцент кафедри КН
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)
Затверджені наказом ректора від «*14*» *травня* 2026 року № *4/9-237*
2. Термін подання студентом завершеної роботи 22 червня 2026 р.
3. Вихідні дані до роботи методичні вказівки до виконання кваліфікаційної роботи, наукові праці з питань електронної комерції та споживчої поведінки, документація використаних технології React, Node.js, MongoDB.
4. Зміст роботи (перелік питань, які потрібно розробити)
Вступ. 1. Аналіз предметної області. 2. Проєктування архітектури інтернет-магазину. 3. Програмна реалізація та аналіз результатів. 4. Безпека життєдіяльності та основи охорони праці Висновки. Перелік джерел. Додатки.

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень, слайдів)
1. Титульний. 2. Аналіз предметної області. 3. Огляд існуючих рішень. 4. Формування вимог та постановка завдання для інтернет-магазину. 5. Середовище розробки та основні технології. 6. Проєктування структури бази даних. 7. Проєктування основних модулів інтернет-магазину і механізмів обробки. 8. Проєктування інтерфейсу. 9. Реалізація клієнтської і серверної частин та взаємодії з базою даних. 10. Реалізація оформлення замовлення. 11. Реалізація модуля для адміністратора. 12. Тестування та аналіз користувацького інтерфейсу. 13. Висновки. 14. Завершальний слайд.

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Безпека життєдіяльності, основи охорони праці	Мариненко С. Ю., к.т.н, доцент кафедри МТ		

7. Дата видачі завдання 26 січня 2026 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів роботи	Термін виконання етапів роботи	Примітка
1.	Ознайомлення з завданням до кваліфікаційної роботи	26.01.2026	
2.	Підбір та опрацювання літературних джерел по темі кваліфікаційної роботи	27.01.2026-16.02.2026	
3.	Виконання дослідження щодо сучасних інтернет-	17.02.2026-10.05.2026	
	магазинів та їх функціональних можливостей.		
	Розроблення вимог до інтернет-магазину		
4.	Оформлення розділу «Аналіз предметної області»	11.05.2026-17.05.2026	
5.	Оформлення розділу «Проєктування архітектури	18.05.2026-24.05.2026	
	інтернет-магазину»		
6.	Оформлення розділу «Програмна реалізація та	25.05.2026-31.05.2026	
	аналіз результатів роботи»		
7.	Виконання завдання до підрозділу «Безпека життєдіяльності»	01.06.2026-08.06.2026	
8.	Виконання завдання до підрозділу «Основи охорони	01.06.2026-08.06.2026	
	праці»		
9.	Оформлення кваліфікаційної роботи	09.06.2026-11.06.2026	
10.	Нормоконтроль	12.06.2026-15.06.2026	
11.	Перевірка на плагіат	16.06.2026	
12.	Попередній захист кваліфікаційної роботи	18.06.2026	
13.	Захист кваліфікаційної роботи	28.06.2026	

Студент _____ Самарик В.М.
(підпис) (прізвище та ініціали)

Керівник роботи _____ Липак Г.І.
(підпис) (прізвище та ініціали)

АНОТАЦІЯ

Розробка інтернет-магазину з функціональними можливостями аналітики, авторизації та оптимізації інтерфейсу // Кваліфікаційна робота освітнього ступеня «Бакалавр» // Самарик Вікторія Мирославівна // Тернопільський національний технічний університет імені Івана Пулюя, факультет комп'ютерно-інформаційних систем і програмної інженерії, кафедра комп'ютерних наук // Тернопіль, 2026 // С. 81, рис. – 22, табл. – 7, додат. – 5, бібліогр. – 41.

Ключові слова: інтернет-магазин одягу, React, Node.js, MongoDB, REST API, авторизація користувачів, пошук і фільтрація товарів, кошик, оформлення замовлення, адміністративна панель, аналітика, тегізація товарів.

Кваліфікаційна робота присвячена розробці інтернет-магазину з функціональними можливостями аналітики, авторизації і автентифікації користувачів та удосконаленню користувацького інтерфейсу.

У першому розділі кваліфікаційної роботи проаналізовано сучасні інтернет-магазини одягу, розглянуто особливості інтерфейсної взаємодії, авторизації, аналітики, здійснено огляд наявних аналогів та сформовано вимоги до розроблюваної системи.

У другому розділі кваліфікаційної роботи обґрунтовано вибір технологій, спроєктовано клієнт-серверну архітектуру інтернет-магазину, структуру бази даних, функціональну організацію модулів, механізми пошуку, фільтрації, двомовності та контролю доступу.

У третьому розділі кваліфікаційної роботи подано програмну реалізацію клієнтської частини, серверної логіки, оформлення замовлення, адміністративного модуля та засобів аналітики, а також виконано перевірку працездатності основних користувацьких і службових сценаріїв.

Об’єкт дослідження: процес функціонування інтернет-магазину одягу як інформаційної системи електронної торгівлі.

Предмет дослідження: програмні засоби, моделі даних та інтерфейсні рішення для створення інтернет-магазину.

ANNOTATION

Development of an Online Store with Analytics, Authorization, and Interface Optimization Features // Qualification work of the educational level «Bachelor» // Samaryk Viktoriia Myroslavivna // Ternopil Ivan Pulyu National Technical University, Faculty of Computer Information Systems and Software Engineering, Department of Computer Sciences // Ternopil, 2026 // P. 80, fig. – 22, tabl. – 7, annexes – 5, references – 41.

Keywords: online clothing store, React, Node.js, Express, MongoDB, REST API, user authentication, product search and filtering, shopping cart, order placement, admin panel, analytics, product tagging.

The qualification thesis is devoted to the development of an online store with analytics, authentication and authorization, and enhancement of the user interface functionality.

The first chapter analyzes modern online clothing stores, examines the features of interface interaction, authentication, and analytics, reviews existing analogues, and defines the requirements for the system being developed.

The second chapter substantiates the choice of technologies, designs the client-server architecture of the online store, the database structure, the functional organization of modules, and the mechanisms of search, filtering, bilingual support, and access control.

The third chapter presents the software implementation of the client side, server logic, order placement, administrative module, and analytics tools. It also includes the verification of the main user and service scenarios.

Object of research: the functioning process of an online clothing store as an e-commerce information system.

Subject of research: software tools, data models, and interface solutions for creating an online store.

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ І ТЕРМІНІВ

API (англ. Application Programming Interface) – інтерфейс програмування застосунків.

ER (англ. Entity-Relationship) – модель «сутність-зв'язок» для опису структури даних.

HTTP (англ. HyperText Transfer Protocol) – протокол передавання гіпертексту.

JSON (англ. JavaScript Object Notation) – текстовий формат подання та обміну даними.

JWT (англ. JSON Web Token) – маркер автентифікації та передавання службових даних.

REST (англ. Representational State Transfer) – архітектурний стиль передавання стану представлення.

REST API (англ. Application Programming Interface), побудований за принципами REST – програмний інтерфейс взаємодії за архітектурним стилем REST.

SPA (англ. Single Page Application) – односторінковий клієнтський інтерфейс.

UI (англ. User Interface) – користувацький інтерфейс.

UML (англ. Unified Modeling Language) – уніфікована мова моделювання.

URL (англ. Uniform Resource Locator) – уніфікований локатор ресурсу.

UX (англ. User Experience) – користувацький досвід.

БД – база даних.

ВДТ – відеодисплейний термінал.

Бекенд – серверна частина програмної системи, що забезпечує обробку даних, бізнес-логіку та взаємодію з базою даних.

Фронтенд – клієнтська частина програмної системи, що відповідає за відображення інтерфейсу та взаємодію з користувачем.

ЗМІСТ

ВСТУП

Актуальність теми. Інтернет-магазини посідають важливе місце у сучасній цифровій торгівлі, забезпечуючи представлення асортименту, взаємодію з користувачем, оформлення замовлень і супровід продажів у межах єдиного інформаційного середовища. В умовах зростання вимог до зручності користування цифровими сервісами, поширення локальних брендів, підвищеної уваги до інклюзивності, національної ідентичності та соціальної відповідальності розробка інтернет-магазину з функціональними можливостями аналітики, авторизації та оптимізації інтерфейсу є актуальним напрямком дослідження у галузі комп'ютерних наук.

Мета і задачі дослідження. Метою даної кваліфікаційної роботи освітнього рівня «Бакалавр» є розробка інтернет-магазину з функціональними можливостями аналітики, авторизації та оптимізації інтерфейсу. Для досягнення поставленої мети потрібно виконати ряд завдань, зокрема:

– проаналізувати предметну область та сучасні аналоги інтернет-магазинів одягу;

– сформувати функціональні, адміністративні, безпекові та інтерфейсні вимоги до системи;

– обґрунтувати вибір технологій для реалізації клієнтської і серверної частин;

– спроєктувати архітектуру інтернет-магазину та структуру бази даних;

– реалізувати механізми пошуку, фільтрації, двомовності, авторизації та оформлення замовлення;

– розробити адміністративний модуль для керування товарами, користувачами і замовленнями;

– провести перевірку працездатності основних користувацьких і службових сценаріїв.

Об'єкт дослідження – процес функціонування інтернет-магазину одягу як інформаційної системи електронної торгівлі.

Предмет дослідження – методи і засоби проєктування та програмної реалізації клієнтської і серверної частин інтернет-магазину, структури бази даних, пошуку, фільтрації, двомовності та адміністративного керування.

Практичне значення одержаних результатів. Результати роботи можуть бути використані як основа для впровадження цифрового торговельного рішення для локального бренду одягу. Реалізовані модулі каталогу, пошуку, фільтрації, двомовності, авторизації, оформлення замовлення та адміністративного керування придатні до подальшої адаптації, розширення й інтеграції з додатковими аналітичними, платіжними та логістичними сервісами.

РОЗДІЛ 1. АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Аналіз сучасних інтернет-магазинів одягу та їх функціональних можливостей

У сучасних умовах розвитку електронної комерції інтернет-магазин є не лише каналом продажу, а повноцінною цифровою системою взаємодії з користувачем [1]. Для сфери продажу одягу це особливо важливо, оскільки покупець має швидко знайти потрібну позицію, оцінити її характеристики, порівняти декілька варіантів і виконати замовлення без зайвих ускладнень. У сучасних дослідженнях підкреслюється, що зручність електронної комерції, якість цифрового досвіду та емоційне сприйняття платформи безпосередньо впливають на намір здійснення покупки [2], [3].

Базовими функціональними можливостями інтернет-магазину одягу є каталог товарів, сторінка окремого товару, кошик, оформлення замовлення, реєстрація та вхід користувача. Проте конкурентоспроможність таких систем дедалі більше визначається не самим фактом наявності цих функцій, а якістю їх реалізації. Для користувача важливими є логічна структура каталогу, швидкодія інтерфейсу, адаптивність під мобільні пристрої, зручні сценарії навігації та мінімальна кількість кроків до завершення покупки [3], [4].

Для інтернет-магазинів одягу характерна робота з великою кількістю атрибутів товару. На відміну від багатьох інших типів електронної комерції, користувач часто орієнтується не лише на назву виробу, а й на категорію, бренд, колір, розмір, стиль, сезонність, матеріал і тематичні теги. У зв'язку з цим важливого значення набувають засоби багатокритеріальної фільтрації та сортування, які скорочують час пошуку й підвищують ефективність взаємодії з каталогом. Для сучасних комерційних платформ також доцільно передбачати більш гнучку предметну тегізацію, що дає змогу будувати тематичні добірки товарів і підтримувати змістовну навігацію за смисловими ознаками [5], [6].

Ще однією суттєвою вимогою є якісне візуальне та інформаційне подання товару. Користувач очікує структурованого опису, інформації про доступність, розміри, матеріали, способи доставки та повернення. Дослідження якості цифрових платформ підтверджують, що інформаційна повнота, системна якість і сервісна складова формують довіру до платформи та позитивно впливають на намір купівлі [3], [7].

Для проєктів електронної комерції UX/UI є не просто візуальним оформленням, а інструментом керування користувацькою поведінкою. У сфері продажу одягу від інтерфейсу залежить, наскільки легко користувач перейде від етапу ознайомлення з асортиментом до додавання товару в кошик та завершення замовлення. Онлайн-атмосфера, послідовність інтерфейсних рішень і зрозуміла взаємодія з елементами сторінки впливають на залученість користувача, рівень його задоволеності та підсумкову конверсію [2], [8].

В українському контексті до функціональних і візуальних вимог інтернет-магазину одягу доцільно додати ще й ціннісно-смисловий вимір. Сучасний користувач дедалі частіше звертає увагу не лише на характеристики товару, а й на позицію бренду, походження продукції, натуральність тканин, екологічність виробництва, наявність елементів національної ідентичності, а також підтримку інклюзивності та соціальної відповідальності. Для ринку одягу це може виявлятися через окремі категорії або теги на кшталт «натуральні тканини», «еко», «вишиванка», а також через візуальні образи людей з різним досвідом і тілесністю, зокрема людей на кріслах колісних чи з протезами [7], [9].

У межах цієї роботи така орієнтація розглядається як проєктна гіпотеза: сайт українського бренду може не лише виконувати комерційну функцію, а й формувати у користувача відчуття причетності до ширшого суспільного контексту. Якщо платформа демонструє підтримку України, воїнів, громадянського суспільства та соціально відповідальних цінностей, це може посилювати емоційну залученість користувача. У такому випадку взаємодія з інтернет-магазином сприймається не лише як купівля товару, а як дія, що має додатковий особистий і суспільний сенс. Саме це відчуття користі, причетності

та ціннісної відповідності може бути пов’язане з евдемонічним щастям споживчого досвіду, який у подальшому здатний впливати на довіру, лояльність і повторну взаємодію з брендом [10].

Отже, сучасний інтернет-магазин одягу повинен поєднувати функції електронного каталогу, засоби персоналізованої навігації, зручний інтерфейсний сценарій купівлі та ціннісно орієнтовану комунікацію з користувачем. Саме ця сукупність вимог визначає напрями подальшого аналізу авторизації, аналітики, аналогів та постановки завдання на розробку власного рішення.

1.2 Аналіз засобів авторизації, аналітики та інтерфейсної взаємодії в сучасних інтернет-магазинах

Авторизація є однією з ключових складових сучасних інформаційних систем електронної комерції. Її призначення полягає не лише в ідентифікації користувача, а й у забезпеченні доступу до персоналізованих функцій: історії замовлень, списку обраного, зміни особистих даних і перегляду поточного статусу покупок. У наукових працях, присвячених поведінці споживача у сфері електронної комерції, питання довіри, безпеки та сприйняття ризику розглядаються як критичні чинники формування наміру до купівлі [10].

Не менш важливою є роль аналітики у вебзастосунках електронної комерції. Збір і обробка даних про популярність товарів, джерела переходів, етапи відмови від покупки та характеристики замовлень дозволяють оцінити ефективність платформи й приймати обґрунтовані рішення щодо її вдосконалення. З позицій сучасних досліджень цифрового ритейлу саме аналітична складова допомагає перетворити інтернет-магазин із простого каталогу в керовану систему, орієнтовану на зростання конверсії та утримання клієнта [9], [11].

Для інтернет-магазину одягу, який орієнтується не лише на продаж, а й на формування ціннісної взаємодії з користувачем, аналітика може охоплювати також поведінку в межах тематичних категорій і смислових маркерів. Зокрема,

доцільно відстежувати перегляди товарів із тегами «натуральні тканини», «еко», «українське виробництво», «благодійне» або інших категорій, що відображають соціальну відповідальність бренду. Це дозволяє оцінити, які ціннісні характеристики товару є важливими для користувачів, і надалі враховувати ці дані під час удосконалення каталогу, фільтрації та комунікації платформи [3].

Інтерфейсна взаємодія в сучасних інтернет-магазинах повинна будуватись за принципом передбачуваності та логічної послідовності. Користувач має швидко розуміти, як перейти до потрібної категорії, як застосувати фільтри, як переглянути товар і як завершити замовлення. Наукові огляди з онлайн-ритейлу підкреслюють, що залученість користувача зростає тоді, коли інтерфейс підтримує чітку навігацію, інформативність і відчуття контролю над процесом взаємодії [3], [8].

Окремо слід враховувати розмежування ролей доступу в системі. Для звичайного користувача пріоритетними є функції перегляду каталогу, оформлення замовлення та керування власним профілем, тоді як адміністративний персонал повинен мати доступ до редагування товарів, контролю замовлень, аналітичних показників і службових операцій. Таке відокремлення доступу знижує ризик помилкових або несанкціонованих дій і відповідає підходам до побудови стійких цифрових платформ, де безпекові обмеження та бізнес-функції повинні бути логічно розподілені між різними типами користувачів [1], [4].

Таким чином, у розроблюваній системі доцільно поєднати надійну авторизацію, базові аналітичні засоби, розмежування ролей доступу та послідовну інтерфейсну логіку. Окрему увагу варто приділити можливості аналізу взаємодії користувачів із тематичними категоріями й ціннісними маркерами товарів. Це створює основу для практичної реалізації інтернет-магазину, що буде не лише функціональним, а й керованим, безпечним, зручним і змістовно релевантним для цільової аудиторії.

1.3 Огляд існуючих аналогів і визначення їх переваг та недоліків

Для формування обґрунтованого підходу до розробки інтернет-магазину доцільно проаналізувати існуючі рішення, представлені на ринку. Порівняння аналогів дозволяє виявити найбільш вдалі підходи до структури каталогу, організації пошуку, фільтрації, подання товарів та оформлення замовлення, а також визначити типові обмеження, які знижують зручність користування.

У межах аналізу доцільно розглянути як мультибрендові платформи, так і монобрендові магазини. Перші зазвичай мають більш потужні засоби навігації та фільтрації, оскільки працюють із широким асортиментом. Другі частіше демонструють вищу цілісність візуальної комунікації та простіший шлях користувача до покупки. Дослідження моделей роздрібної торгівлі безпосередньо споживачам також показують, що сучасна брендована електронна комерція орієнтується не лише на продаж, а й на цілісний цифровий досвід взаємодії з покупцем [6].

Порівняльний аналіз аналогів, представлений у таблиці 1.1, показує, що сильними сторонами успішних рішень є структурований каталог, фільтрація за релевантними параметрами, адаптивність, якісна картка товару та зрозумілий сценарій оформлення замовлення. Водночас поширеними недоліками залишаються перевантаження сторінок великою кількістю елементів, недостатньо гнучка локалізація окремих атрибутів й обмежена предметна тегізація. Міжнародні бренди зазвичай демонструють вищий рівень стандартизації інтерфейсу, масштабованості каталогу та технічної послідовності, проте їхня комунікація часто є більш універсальною й менш прив’язаною до локального соціокультурного контексту. Українські бренди, навпаки, частіше використовують національні мотиви, локальне виробництво, благодійні колекції та соціально відповідальні повідомлення, однак не завжди мають настільки розвинені інструменти фільтрації й персоналізації.

Таблиця 1.1 – Порівняльна характеристика аналогів інтернет-магазинів одягу

Аналог	Ключові переваги	Слабкі сторони
Kasta [12]	Широкий асортимент, багатокритеріальна фільтрація, зручна структура каталогу	Перевантаження сторінок великою кількістю візуальних елементів
ANSWEAR [13]	Поєднання великого асортименту з цілісним візуальним стилем і зрозумілою навігацією	Менший акцент на предметній тегізації та локалізованих атрибутах товару
MustHave [14]	Сильна айдентика бренду, чиста подача контенту, простий шлях користувача до покупки	Менш розвинена багатокритеріальна фільтрація через компактніший каталог
Zara [15]	Виразний UX/UI, сильна візуальна ієрархія, цілісна брендова комунікація	Недостатньо гнучка фільтрація для деталізованого предметного добору
Overthesea [16]	Сучасна естетика, проста навігація, цілісний монобрендовий досвід	Базова система фільтрації порівняно з більшими мультибрендовими платформами
Dodo Socks [17]	Виразна українська ідентичність, локальне виробництво, смислові та благодійні колекції	Велика кількість візуальних акцентів може ускладнювати швидкий добір товару
SVARGA [18]	Сильний національний маркер, акцент на культурній спадщині та натуральних матеріалах	Вузька етнокультурна спеціалізація зменшує універсальність каталогу

Отже, для розроблюваного інтернет-магазину доцільно поєднати сильні сторони мультибрендових платформ і монобрендових рішень: залишити зручну багатокритеріальну фільтрацію, забезпечити структуровану картку товару, передбачити двомовність, чітке розмежування ролей доступу та додати адміністративні інструменти для керування каталогом і аналітикою. Окремо доцільно закласти механізми предметної тегізації, що дадуть змогу виокремлювати товари за натуральністю матеріалів, екологічними характеристиками, національними мотивами та іншими смисловими маркерами,

важливими для українського користувача. Саме таке рішення буде покладене в основу подальшого проєктування архітектури, даних і функціональних модулів системи.

1.4 Формування вимог до інтернет-магазину

На основі аналізу предметної області, наукових джерел і розглянутих аналогів можна сформувати сукупність вимог до розроблюваного інтернет-магазину. Доцільно розподілити їх на користувацькі, адміністративні, безпекові та інтерфейсні, оскільки саме така структура дає змогу пов'язати бізнес-потреби, технічну реалізацію та якість користувацького досвіду [3], [11].

З огляду на сформульовану проєктну гіпотезу, для головної сторінки доцільно передбачити окремий комунікаційний блок, що підкреслюватиме соціальну відповідальність бізнесу та його чітку позицію щодо суспільно важливих питань. Такий блок не повинен замінювати каталог чи комерційні елементи, але може виконувати роль емоційного та ціннісного маркера, який підсилює довіру до бренду й формує відчуття змістовної взаємодії з платформою [1], [9]. Систематизація вимог у вигляді таблиці 1.2 дозволяє створити чітку основу для подальшого проєктування структури системи, визначення її модулів і реалізації пріоритетних функцій.

Таблиця 1.2 – Узагальнені вимоги до розроблюваного інтернет-магазину

Група вимог	**Основний зміст вимог**
1	**1**
Користувацькі	Перегляд каталогу товарів, сторінки товару, пошук за текстовим запитом, багатокритеріальна фільтрація, сортування, кошик, оформлення замовлення, реєстрація, авторизація та автентифікація, профіль користувача, історія замовлень, двомовність, тематичні теги на кшталт «натуральні тканини», «еко», «код нації», «вишиванка», вибір способу оплати, тестова онлайн-оплата.

Продовження таблиці 1.2

1	2
Адміністративні	Керування товарами, користувачами та замовленнями, редагування даних товарів, контроль статусів замовлень, доступ до аналітичної інформації, розмежування прав доступу, керування тегами та інформаційними блоками головної сторінки, завантаження та оновлення зображень товарів.
Безпекові	Серверна перевірка складу замовлення, контроль кількості товару, перерахунок сум на сервері, обмеження доступу до службових функцій, захист персональних даних користувача та його замовлень, хешування паролів користувачів.
Інтерфейсні	Адаптивність, зручна навігація, логічна структура каталогу, інформативна картка товару, швидкий доступ до фільтрів, послідовний сценарій оформлення замовлення, читабельність елементів інтерфейсу, інклюзивна візуальна комунікація, національні та соціально відповідальні акценти на головній сторінці.

1.5 Постановка завдання на розробку інтернет-магазину

Метою кваліфікаційної роботи є розробка інтернет-магазину одягу, який поєднуватиме зручний користувацький інтерфейс, двомовний каталог товарів, пошук і багатокритеріальну фільтрацію, авторську систему атрибутів товару, авторизацію, кошик, оформлення замовлення, тестову онлайн-оплату, адміністративну панель і базові засоби аналітики. Результатом має стати інтернет-магазин, орієнтований на практичне використання, безпечну роботу з користувацькими даними та подальше масштабування.

Під час формулювання мети, завдань і загальної структури роботи доцільно враховувати вимоги до побудови та оформлення кваліфікаційної роботи, визначені методичними вказівками кафедри [19]. Для досягнення поставленої мети необхідно виконати такі завдання: дослідити предметну область і наявні аналоги; сформувати функціональні та нефункціональні вимоги;

спроєктувати архітектуру клієнтської та серверної частин; визначити структуру бази даних і модель взаємодії між сутностями; реалізувати основні модулі каталогу, авторизації, кошика, замовлень та адміністрування; забезпечити перевірку коректності роботи системи.

1.6 Висновок до першого розділу

У першому розділі проаналізовано предметну область інтернет-магазинів одягу, їхні базові та розширені функціональні можливості, роль авторизації, аналітики й інтерфейсної взаємодії. Визначено, що сучасний інтернет-магазин має поєднувати каталог товарів, пошук, фільтрацію, кошик, оформлення замовлення, персоналізовані функції користувача, адміністративне керування та зручний UX/UI. Окремо встановлено, що для магазину одягу важливими є не лише стандартні характеристики товару, а й матеріал, колір, стиль, тематичні теги, локальний контекст і ціннісна комунікація бренду.

Огляд аналогів показав, що сильними сторонами успішних рішень є структурований каталог, якісна картка товару, зручна навігація, адаптивність і зрозумілий сценарій покупки. На основі цього сформовано користувацькі, адміністративні, безпекові та інтерфейсні вимоги до розроблюваної системи.

Сформульоване завдання безпосередньо визначає зміст наступних розділів роботи. У розділі 2 буде обґрунтовано проєктні рішення, архітектуру системи, структуру даних і логіку взаємодії модулів, зокрема засоби тегізації товарів, організації головної сторінки та способи відображення ціннісних маркерів бренду. У розділі 3 буде описано практичну реалізацію інтернет-магазину, використані технології, ключові програмні компоненти та результати перевірки працездатності створеного інтернет-магазину, включаючи реалізацію елементів інтерфейсу, що підкреслюють інклюзивність, соціальну відповідальність і український контекст платформи.

РОЗДІЛ 2. ПРОЄКТУВАННЯ АРХІТЕКТУРИ ІНТЕРНЕТ-МАГАЗИНУ

У другому розділі розглянуто проєктні рішення, необхідні для створення інтернет-магазину одягу. Основну увагу приділено вибору технологій, побудові клієнт-серверної архітектури, структурі бази даних, функціональним модулям системи, механізмам обробки даних і проєктуванню користувацького інтерфейсу.

2.1 Обґрунтування вибору технологій для розробки інтернет-магазину

Під час проєктування інтернет-магазину одягу технологічний стек доцільно визначати з урахуванням реалізації базового функціоналу, подальшого розширення системи, структури каталогу, складності бізнес-логіки та розвитку адміністративної частини. Для досліджуваної системи обрано React, Node.js, Express, MongoDB та набір допоміжних бібліотек, пов'язаних із маршрутизацією, HTTP-взаємодією, інтерфейсом, оплатою, локалізацією й роботою із зображеннями.

Для клієнтської частини обрано React. Ця бібліотека дає змогу будувати інтерфейс як сукупність незалежних і повторно використовуваних компонентів, що відповідає підходу, описаному в офіційній документації React [20], [21]. Для інтернет-магазину одягу це має практичне значення, оскільки в його структурі багаторазово повторюються картки товарів, блоки фільтрації, форми авторизації, панелі оформлення замовлення та адміністративні списки. Компонентна архітектура полегшує супровід таких інтерфейсів і дає змогу вносити локальні зміни без повного перепроєктування сторінки.

Важливою перевагою React є також зручна побудова SPA-інтерфейсу з маршрутизацією на стороні клієнта. Для цього в системі передбачено використання react-router-dom, який забезпечує перехід між сторінками без повного перезавантаження документа [22]. У системі наявні як звичайні сторінки перегляду товарів, так і захищені сторінки для авторизованого користувача та

адміністратора. Перехід між такими сценаріями відбувається без повного перезавантаження сторінки, що позитивно впливає на швидкодію та сприйняття інтерфейсу. Навіть незначні затримки при зміні сторінок каталогу, пошуку чи кошика впливають на користувацький досвід [3].

Серверну частину передбачено реалізувати на Node.js із використанням Express. Це дозволяє використовувати JavaScript і на клієнтському, і на серверному рівні, що спрощує узгодження логіки між частинами системи. Express виконує роль вебфреймворку: приймає HTTP-запити, визначає маршрути API, підключає проміжні обробники та передає запити до відповідних серверних функцій. Такий підхід відповідає можливостям маршрутизації та використання проміжних обробників, описаним в офіційній документації Express [23].

Для серверної частини також можна було б використати Django [24], Laravel [25] або Spring Boot [26]. Однак у межах цієї системи доцільнішим є підхід на базі Node.js та Express, оскільки він узгоджується з клієнтською частиною й не потребує введення додаткової мови програмування. Обраний стек забезпечує достатню функціональність для реалізації каталогу, замовлень, перекладів і адміністрування без надлишкового ускладнення архітектури.

Для зберігання даних обрано MongoDB у поєднанні з Mongoose. Вибір БД, яка використовує документно-орієнтовану модель, пов'язаний зі змінністю атрибутів товарів [27]. Окрім традиційних назви, бренду та ціни, система використовує тканину, колір, тематичні теги, двомовні назви й описи, параметри доступності, вкладені структури замовлень і словникові відповідники для перекладу. Документно-орієнтована модель дозволяє еволюційно розширювати такі сутності без перетворення структури даних на надмірно жорстку схему з великою кількістю допоміжних таблиць. Mongoose, своєю чергою, дає змогу описувати схеми документів, моделі та правила роботи з даними на рівні застосунку [28].

MongoDB не розглядається як єдино можливий варіант для такого класу систем. Реляційні СУБД також придатні для надійного зберігання замовлень,

користувачів і товарного каталогу. Водночас у межах досліджуваної системи документна модель є зручною для роботи зі змінними характеристиками товарів, вкладеними адресами доставки та позиціями замовлення, а також для поступового розширення структури колекцій без суттєвих міграційних ускладнень.

Обмін даними між клієнтською та серверною частинами передбачено через REST API. Такий підхід забезпечує прозорий розподіл відповідальності між рівнем подання і рівнем бізнес-логіки, дозволяє будувати зрозумілі кінцеві точки для роботи з товарами, користувачами, замовленнями та перекладами, а також полегшує локальне тестування підсистем. Для наявної структури даних і сценаріїв взаємодії цього підходу достатньо; складніші моделі на кшталт GraphQL не створювали б істотної функціональної переваги.

Окремо слід зазначити допоміжні інструменти, використані в реалізації. На клієнтській частині це react-router-dom для маршрутизації, axios для HTTP-запитів [29], react-bootstrap для базових адаптивних інтерфейсних компонентів, react-helmet-async для керування метаданими сторінок, react-toastify для системи повідомлень, react-google-charts для відображення аналітики та @paypal/react-paypal-js для інтеграції оплати. На серверній частині додатково застосовано jsonwebtoken для токенів доступу, bcryptjs для роботи з паролями, multer і cloudinary для завантаження зображень. Використання такого набору бібліотек відповідає структурі повнофункціонального інтернет-магазину.

Для роботи із зображеннями товарів мають значення multer і cloudinary [30]. Multer відповідає за приймання файлів на серверному рівні, а Cloudinary використовується як хмарне середовище для зберігання, оптимізації та подальшого використання зображень у каталозі. Оскільки на користувацький досвід безпосередньо впливає швидкість завантаження і якість відображуваного візуального контенту, рішення про винесення зображень у хмарне сховище зменшує навантаження на локальну структуру проєкту і полегшує подальше масштабування каталогу.

Розділення клієнтської та серверної частин створює передумови для подальшого розвитку системи у напрямку мультибрендової платформи, розширення ролей користувачів, запровадження складнішої аналітики, рекомендаційних механізмів або зовнішніх інтеграцій. Дослідження, присвячені еластичним архітектурам цифрових сервісів і сервісно орієнтованим системам, також підкреслюють доцільність модульного поділу та ізоляції відповідальностей [4], [6]. Узагальнений вибір технологій і їхнє призначення в системі наведено в таблиці 2.1.

Таблиця 2.1 – Вибір технологій для реалізації інтернет-магазину

Технологія	**Призначення**	**Обґрунтування вибору**
React	Реалізація клієнтської частини та побудова SPA-інтерфейсу	Компонентний підхід, маршрутизація, зручне повторне використання елементів каталогу, кошика, форм і адміністративних сторінок.
Node.js + Express	Реалізація серверної логіки та REST API	Єдина мова програмування з фронтендом, гнучка маршрутизація, проміжні обробники авторизації, зручне структурування бізнес-логіки.
MongoDB + Mongoose	Зберігання даних каталогу, користувачів, замовлень і перекладів	Гнучка документна модель для товарів зі змінними атрибутами, зручне розширення структури даних і схемна валідація.
REST API	Обмін даними між клієнтом і сервером	Прозора взаємодія між підсистемами, стандартизовані кінцеві точки, зручність тестування та подальшого масштабування.
Допоміжні бібліотеки	Маршрутизація, UI, аналітика, повідомлення, оплата, завантаження файлів	Дозволяють не дублювати базову інфраструктурну логіку й зосередитися на предметній частині магазину.

Як видно з таблиці 2.1, обраний стек охоплює всі ключові рівні системи: клієнтський інтерфейс, серверну логіку, зберігання даних, API-взаємодію та

допоміжні функції. Це дає змогу проєктувати інтернет-магазин як цілісну клієнт-серверну систему, яку можна надалі адаптувати під потреби невеликого локального магазину: розширювати каталог, додавати нові атрибути товарів, змінювати адміністративні функції та поступово підключати додаткові сервіси.

2.2 Проєктування архітектури інтернет-магазину

Архітектуру інтернет-магазину побудовано за клієнт-серверним принципом. У такій моделі клієнтська частина відповідає за відображення інтерфейсу, маршрутизацію, збір дій користувача і формування HTTP-запитів, тоді як бекенд приймає ці запити, виконує перевірку прав доступу, реалізує бізнес-логіку та звертається до бази даних. Подібний розподіл ролей є типовим для сучасних цифрових сервісів і добре узгоджується з підходами до побудови інформаційних систем із чітким поділом на рівень подання, прикладний рівень та рівень даних [9].

На клієнтському рівні систему спроєктовано як SPA на базі React. Центральним вузлом є компонент App, який містить маршрутизацію, глобальну навігацію, бічне меню категорій та перемикання між основними екранами. У структурі інтерфейсу можна виокремити три великі групи сторінок: сторінки відкритого доступу, персоналізовані сторінки користувача та адміністративні сторінки. Така сегментація задає логіку подальшого розмежування доступу й окреслює основні потоки взаємодії з системою.

До сторінок відкритого доступу належать головна сторінка, сторінка пошуку, сторінка окремого товару, кошик, а також сторінки реєстрації й авторизації. Персоналізована частина охоплює оформлення замовлення, вибір способу оплати, перегляд профілю та історії замовлень. Адміністративна частина містить панель керування, списки користувачів, товарів і замовлень, а також форми редагування відповідних сутностей. Така архітектурна сегментація забезпечує зрозумілу відповідність між предметною логікою системи та її інтерфейсною реалізацією.

Серверний рівень побудовано навколо server.js, який виконує роль точки входу до системи. У ньому налаштовується підключення до MongoDB, реєструються засоби обробки JSON і даних форм, визначаються службові кінцеві точки та підключаються тематичні маршрутизатори: productRoutes, userRoutes, orderRoutes, translationRoutes, uploadRoutes, exchangeRoutes, seedRoutes [21]. Така композиція робить систему модульною, оскільки бізнес-логіка групується не за технічними деталями, а за предметними підсистемами.

У роботі інтернет-магазину простежується послідовний ланцюг обробки запиту. Користувач виконує дію в інтерфейсі; React-компонент або екран формує запит до REST API; серверний маршрут приймає його й передає через проміжні обробники авторизації та додаткових перевірок; після цього виконується прикладна логіка і звернення до колекцій MongoDB через моделі Mongoose; далі сервер формує JSON-відповідь, а клієнтська частина оновлює стан сторінки. Послідовність обробки запиту в інтернет-магазині наведено на рисунку 2.1.

Рисунок 2.1 – Схема обробки запиту в інтернет-магазині

На рисунку 2.1 показано, як окремі архітектурні рівні взаємодіють у межах одного користувацького сценарію. Запит проходить шлях від дії користувача в інтерфейсі до серверного маршруту, проміжної перевірки, звернення до бази даних і повернення відповіді у форматі JSON. Саме така послідовність пояснює, чому клієнтська частина, серверна логіка та база даних розглядаються як окремі, але пов’язані рівні системи.

Архітектура системи поєднує стандартний каталог із двомовністю, тематичними тегами, валютними допоміжними даними та адміністративною аналітикою. У структурному відношенні система виходить за межі базової схеми «товари-користувачі-замовлення» і включає додатковий інформаційний шар. Це

відповідає вимогам систем електронної торгівлі, у яких важливими є не тільки операції купівлі, а й керування досвідом користувача, смисловою навігацією та аналітичними даними [11].

Оскільки клієнтська частина на React будується переважно через компоненти, контексти та маршрути, класична діаграма класів не повністю відображає її структуру. Тому для опису фронтенду доцільно використати умовну UML-подібну схему, яка показує основні компоненти, екрани та зв’язки між ними.

На рисунку 2.2 узагальнено основні фронтенд-сутності. UML-подібна схема відображає структурні залежності між центральним компонентом клієнтської частини, ключовими екранами, спільними компонентами та сервісами локалізації і стану.

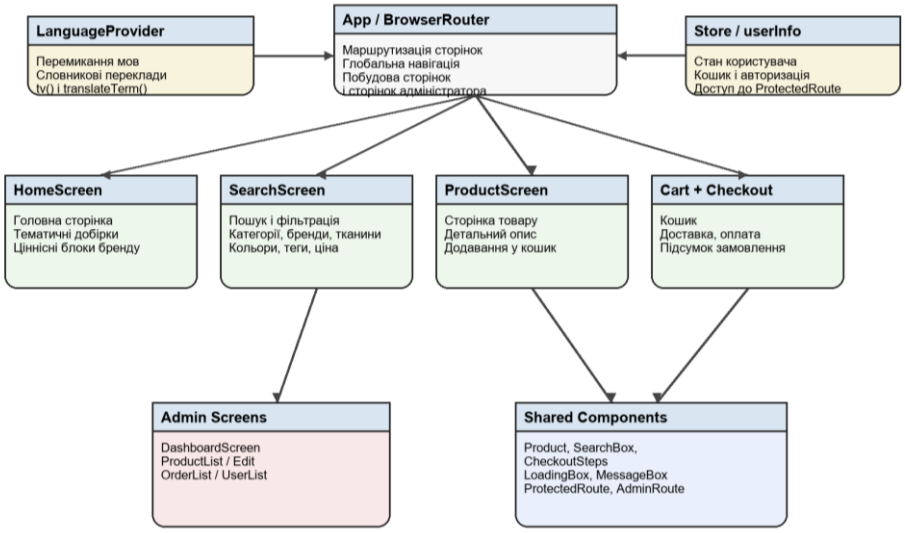

Рисунок 2.2 – Умовна діаграма класів і компонентів фронтенду

Таке UML-подання дає змогу наочно показати взаємозв’язки між основними компонентами клієнтської частини.

Окремо доцільно подати й серверну частину, де структура ближча до класичної моделі: маршрутизатори працюють із моделями даних, проміжними обробниками перевірки доступу та службовими модулями. Рисунок 2.3 відображає умовну діаграму класів і компонентів серверної частини.

Маршрутизатори групують прикладну логіку навколо моделей даних і проміжних обробників авторизації, а server.js виконує координаційну функцію.

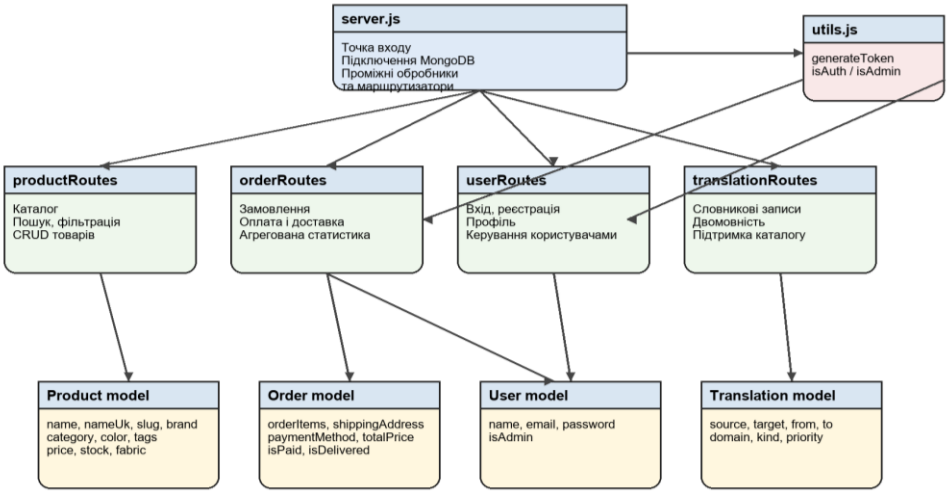

Рисунок 2.3 – Умовна діаграма класів і компонентів бекенду

На рисунку 2.3 показано структуру серверної частини. Маршрутизатори групують прикладну логіку навколо моделей даних і проміжних обробників перевірки доступу, а server.js виконує координаційну функцію. У поєднанні з рисунком 2.1 ця схема показує не лише склад серверного рівня, а й його місце в загальному ланцюгу обробки запиту. Спроєктоване REST API реалізує принципи REST і забезпечує однаковий підхід до роботи з товарами, користувачами та замовленнями.

2.3 Проєктування структури бази даних

Проєктування бази даних виконано з урахуванням основних сутностей предметної області: користувачів, товарів, замовлень і словникових перекладів. Для кожної сутності створено окрему колекцію MongoDB, що описується через схему Mongoose. Така модель поєднує гнучкість документного підходу з елементами структурного контролю: визначаються обов'язкові поля, типи даних, ознаки унікальності окремих атрибутів та автоматичні часові мітки [27].

Колекція User призначена для облікових записів користувачів і містить ім'я, електронну адресу, пароль та ознаку адміністратора. З погляду предметної логіки ця сутність є базовою для персоналізованих сценаріїв: вона визначає, хто може оформлювати замовлення, переглядати власну історію та хто має доступ до адміністративного функціоналу. Поле isAdmin у поєднанні з серверними перевірками доступу формує просту рольову модель.

Колекція Product є центральною для каталогу інтернет-магазину. У ній зберігаються як базові характеристики товару, так і атрибути, що безпосередньо підтримують пошук і фільтрацію: назви двома мовами, бренд, категорія, колір, тканина, тематичні теги, описи, ціна й кількість на складі. Така модель підтримує не лише сторінку окремого товару, а й розширену каталогізацію за культурними, екологічними та стилістичними ознаками асортименту.

Колекція Order реалізована як окрема сутність із вкладеним масивом orderItems. Для кожної товарної позиції в замовленні фіксуються назва, альтернативна українська назва, кількість, зображення, ціна та посилання на початковий товар. Система зберігає не тільки зв'язок із товаром, а й фіксує ключові характеристики позиції на момент купівлі. Це дозволяє коректно відтворювати історію замовлень навіть у разі подальших змін у каталозі.

Колекція Translation виконує словникову функцію. У ній зберігаються пари слів і термінів для перекладу між мовами, пріоритет використання, галузь застосування та тип запису. Наявність окремої колекції перекладів дає змогу централізовано керувати предметною двомовністю і не дублювати всі можливі текстові варіанти в кожній позиції каталогу.

Окремі колекції не існують ізольовано, а формують спільну модель даних, у якій користувачі пов’язані із замовленнями, замовлення — з товарними позиціями, а товари — з атрибутами каталогу та словниковими значеннями. Така структура дає змогу підтримувати основні сценарії роботи системи: перегляд товарів, пошук і фільтрацію, оформлення замовлення, перегляд історії покупок, адміністрування каталогу та двомовне відображення предметних даних.

Структура БД охоплює не лише базові сутності інтернет-магазину, а й додатковий шар локалізації. Основні колекції бази даних, їхні поля та призначення наведено в таблиці 2.2.

Таблиця 2.2 – Основні колекції бази даних

Колекція	**Основні поля**	**Призначення**
User	name, email, password, isAdmin, timestamps	Зберігання облікових записів користувачів і розмежування ролей доступу.
Product	name, nameUk, slug (адресний ідентифікатор), image, brand, category, color, tags, description, descriptionUk, price, countInStock, fabric, timestamps	Зберігання каталогу товарів і підтримка пошуку, фільтрації, сторінки товару та тематичних добірок.
Order	orderItems, shippingAddress, paymentMethod, paymentResult, itemsPrice, shippingPrice, discountPrice, totalPrice, user, isPaid, isDelivered, timestamps	Фіксація оформлених замовлень, статусів оплати й доставки та історії покупок.
Translation	source, target, from, to, priority, domain, kind, timestamps	Підтримка словникових перекладів для двомовного відображення предметних термінів і значень каталогу.

Логічні зв'язки між колекціями мають виразну структуру: один користувач може створювати багато замовлень; кожне замовлення пов'язане з одним користувачем; кожне замовлення містить багато товарних позицій; товарні позиції асоційовані з конкретними товарами каталогу. Додатково колекція Translation підтримує інформаційний шар локалізації, який використовується багатьма інтерфейсними сценаріями. З погляду теорії інформаційних систем така модель відповідає багатосутнісній структурі сервісної платформи [11].

ER-діаграма, представлена на рисунку 2.4, відображає основні колекції та логіку зв'язків між ними, сформовану на етапі проєктування даних.

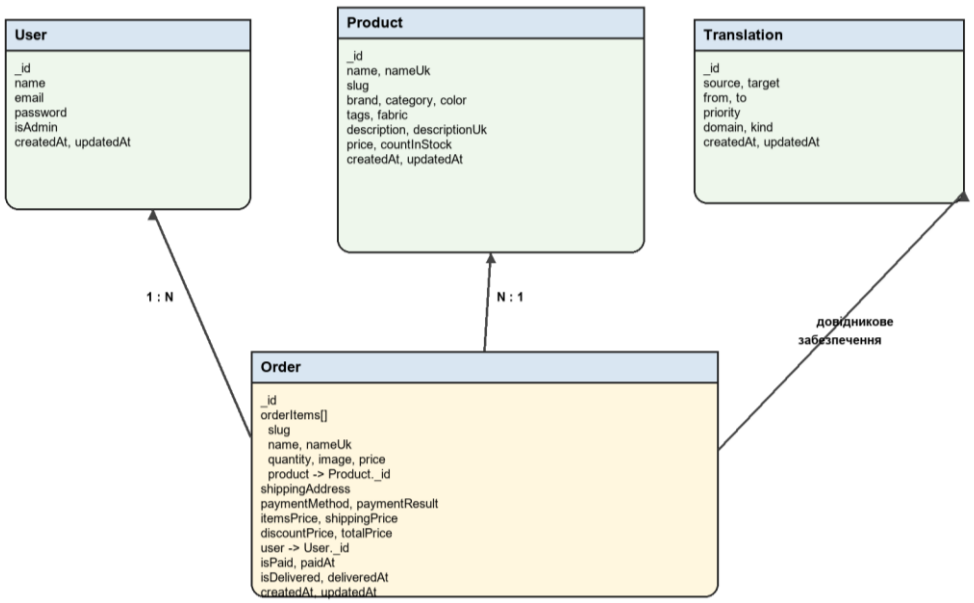

Рисунок 2.4 – ER-діаграма бази даних інтернет-магазину

Розширена база даних подана у додатку А, тоді як подана ER-модель дає змогу коротко формалізувати зв’язки між колекціями ще до реалізації запитів і перевірити цілісність структури даних. Вона складає чотири основні колекції інтернет-магазину та зв’язки між ними (див. рис. 2.4). Центральними сутностями є User, Product і Order, оскільки саме вони забезпечують сценарії реєстрації, перегляду товарів, оформлення замовлення та перегляду історії покупок. Колекція Translation доповнює основну модель і відповідає за підтримку двомовного відображення предметних значень.

Спроєктована структура БД відповідає основним сценаріям роботи інтернет-магазину: зберіганню каталогу, облікових записів, замовлень і словникових перекладів. Окремою перевагою є авторська система атрибутів товару в колекції Product. Вона підтримує не лише базовий опис, ціну й залишок на складі, а й тканину, колір, бренд, категорію, тематичні теги, двомовні назви та описи. Документна модель залишає можливість для подальшого розширення: додавання нових характеристик товарів, розширення ролей користувачів, деталізації замовлень і розвитку локалізованого каталогу [27].

2.4 Проєктування функціональних модулів системи

Функціональну структуру інтернет-магазину організовано за модульним принципом. Кожен модуль охоплює завершену частину предметної логіки та взаємодіє з іншими модулями через визначені дані й сценарії. Така побудова є зручною з погляду супроводу, оскільки зміни локалізуються в межах відповідної підсистеми й не призводять до надмірного змішування відповідальностей у коді [20].

Базовим модулем є каталог товарів. Він забезпечує отримання списку позицій, посторінковий перегляд, відображення карток товарів, перехід до сторінки окремого товару та використання тематичних добірок і фільтрів. Саме цей модуль формує основний простір взаємодії користувача з асортиментом.

Окремий модуль становлять пошук і фільтрація. Він підтримує текстові запити, багатовибіркові критерії, сортування, відбір за наявністю та ціновим діапазоном. У структурі системи цей модуль тісно пов'язаний із каталогом, проте має власну логіку формування умов запиту та ранжування результатів.

Модуль авторизації та профілю охоплює реєстрацію, вхід, збереження токена, перегляд і редагування персональних даних, а також доступ до історії замовлень. Із цим модулем безпосередньо пов'язані механізми захищених маршрутів, які розмежовують доступ для неавторизованого користувача, авторизованого користувача та адміністратора.

Модуль кошика, оформлення замовлення та оплати забезпечує накопичення обраних позицій, введення адреси доставки, вибір способу оплати, підрахунок вартості й передавання даних на сервер. У межах предметної логіки цей модуль поєднує результати роботи каталогу, профілю користувача та серверних перевірок замовлення.

Адміністративна частина охоплює модулі керування товарами, користувачами, замовленнями та аналітикою. Через неї виконуються створення нових позицій каталогу, редагування й видалення товарів, перегляд списків користувачів, контроль замовлень та аналіз агрегованих показників продажів.

Завдяки цьому інтернет-магазин виконує не лише функцію продажу, а й функцію оперативного керування даними.

Окреме місце у функціональній структурі посідає підсистема двомовності. Вона об'єднує статичні словники інтерфейсних фраз, динамічні словникові переклади предметних термінів, перемикання мови та логіку відображення двомовних полів. Унаслідок цього локалізація інтегрується не в окремі фрагменти, а в наскрізну модель роботи каталогу.

Схема взаємодії модулів і таблиця відповідності модулів файлам реалізації дають змогу зіставити функціональну структуру системи з фактичною організацією коду. Схему взаємодії функціональних модулів наведено на рисунку 2.5.

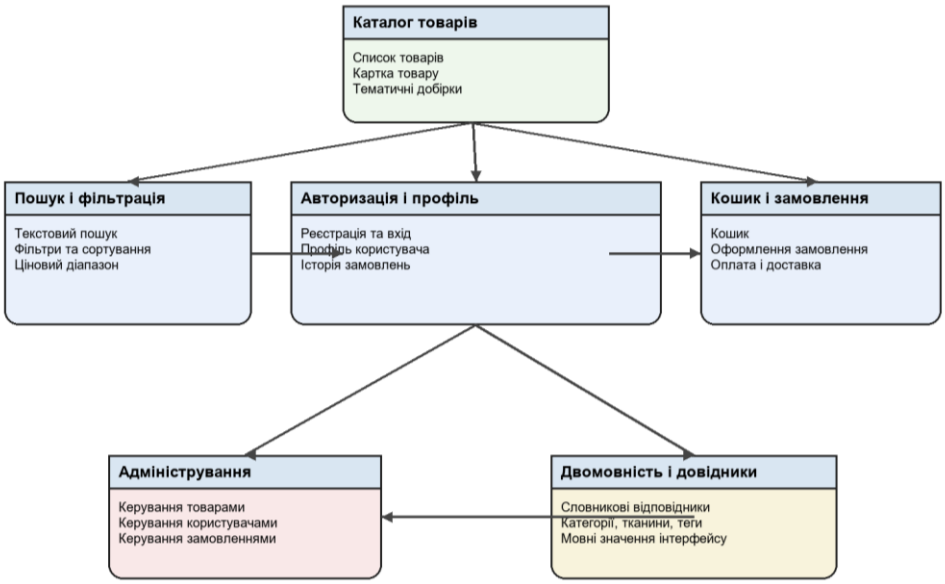

Рисунок 2.5 – Схема взаємодії функціональних модулів

Схема (див. рис. 2.5) відображає взаємодію між каталогом товарів, пошуком і фільтрацією, авторизацією, кошиком, замовленнями, адмініструванням і двомовністю. Таке подання точніше відображає внутрішні зв'язки між модулями, ніж перелік безпосередньо після текстового опису. Відповідність функціональних модулів основним файлам наведено в таблиці 2.3.

Таблиця 2.3 – Відповідність функціональних модулів основним файлам реалізації

Модуль	**Фронтенд**	**Бекенд**
Каталог товарів	HomeScreen.js, ProductScreen.js, Product.js	productRoutes.js, productModel.js
Пошук і фільтрація	SearchScreen.js, SearchBox.js	productRoutes.js, exchangeRoutes.js
Авторизація і профіль	SigninScreen.js, SignupScreen.js, ProfileScreen.js	userRoutes.js, userModel.js, utils.js
Оформлення замовлення	CartScreen.js, ShippingAddressScreen.js, PaymentMethodScreen.js, PlaceOrderScreen.js, OrderScreen.js	orderRoutes.js, orderModel.js, utils.js
Адміністрування	DashboardScreen.js, ProductListScreen.js, OrderListScreen.js, UserListScreen.js	orderRoutes.js, productRoutes.js, userRoutes.js
Двомовність	i18n.js	translationRoutes.js, translationModel.js

Функціональні модулі мають відображення і на клієнтському, і на серверному рівнях. Відтак це означає, що структура системи не є набором окремих сторінок, а побудована як взаємопов’язана модульна модель. Кожен модуль має власну зону відповідальності, але водночас бере участь у спільних сценаріях: перегляді каталогу, пошуку товарів, оформленні замовлення, роботі з профілем, адмініструванні та двомовному відображенні даних.

Спроєктована функціональна структура дає змогу розглядати інтернет-магазин як систему взаємопов’язаних модулів. Окремі колекції не існують ізольовано, а формують спільну модель даних, у якій користувачі пов’язані із замовленнями, замовлення — з товарними позиціями, а товари — з атрибутами каталогу та словниковими значеннями [20], [27].

Такий підхід спрощує подальше розширення: до системи можна додавати нові фільтри, адміністративні функції, способи оплати, аналітичні показники або додаткові мовні елементи без повного перегляду всієї архітектури.

2.5 Проєктування основних механізмів обробки даних

Ключові механізми обробки даних пов'язані з пошуком, фільтрацією, перевіркою цілісності замовлення, двомовністю та контролем доступу. Саме вони визначають, наскільки система є функціонально повною та придатною до практичного використання.

Пошук у каталозі реалізовано не як просту перевірку збігів, а як вагову модель релевантності. Кожен токен пошукового запиту перевіряється за назвами двома мовами, категорією, брендом, тканиною, кольором, тегами й описом. Далі формується інтегральний показник релевантності, що дозволяє впорядковувати результати відповідно до близькості до наміру користувача. Така логіка є доцільною для магазину одягу, оскільки пошуковий запит нерідко формулюється через ознаки або асоціації, а не через точну назву товару.

Механізм фільтрації підтримує одночасний вибір кількох категорій, брендів, тканин, кольорів, тегів, статусу наявності та меж ціни. З технічної точки зору це реалізовано через універсальну побудову умов запиту до MongoDB на основі параметрів URL [31]. Завдяки цьому система може гнучко комбінувати критерії без множення кількості кінцевих точок і без дублювання логіки на клієнтській та серверній частинах.

Важливе значення для надійної роботи системи має серверний перерахунок замовлення. Під час створення покупки сервер повторно звертається до бази даних, звіряє товари, перевіряє кількість і наявність на складі, а вже потім самостійно обчислює суму товарів, доставку, знижку та підсумкову вартість. У системах електронної торгівлі такий підхід є необхідним, оскільки дані, передані з браузера, не можуть вважатися достатньою підставою для фінансових розрахунків [4].

Контроль доступу реалізовано на двох рівнях. Перший рівень охоплює загальну автентифікацію й поділ ролей за допомогою JWT-токена, `isAuth` та `isAdmin`. Другий рівень становить контекстна перевірка приналежності

конкретного ресурсу, наприклад замовлення. Це забезпечує як розмежування ролей, так і захист від несанкціонованого доступу до чужих даних.

Механізм двомовності вбудовано не лише в статичні інтерфейсні тексти, а й у предметну навігацію по каталогу. Переклад охоплює категорії, тканини, кольори, теги й частину описів, тому локалізація має прикладне значення в роботі з асортиментом. Фрагмент реалізації гнучкого текстового пошуку та обчислення релевантності наведено у лістингу 2.1.

Лістинг 2.1 – Фрагмент реалізації гнучкого текстового пошуку та обчислення релевантності

```
const buildTokenFilter = (token) => {
  const escapedToken = escapeRegExp(token);
  return {
    $or: [
      { name: { $regex: escapedToken, $options: 'i' } },
      { nameUk: { $regex: escapedToken, $options: 'i' } },
      { category: { $regex: escapedToken, $options: 'i' } },
      { brand: { $regex: escapedToken, $options: 'i' } },
      { fabric: { $regex: escapedToken, $options: 'i' } },
      { color: { $regex: escapedToken, $options: 'i' } },
      { tags: { $elemMatch: { $regex: escapedToken, $options:
'i' } } },
      { description: { $regex: escapedToken, $options: 'i' } },
      { descriptionUk: { $regex: escapedToken, $options: 'i' }
},
    ],
  };
};
const buildSearchScoreExpression = (searchQuery, searchTokens)
=> {
  const scoreExpressions = [];

  if (searchQuery && searchQuery !== 'all') {
    scoreExpressions.push(
      buildWeightedMatchExpression('$name', searchQuery, 30),
      buildWeightedMatchExpression('$nameUk', searchQuery, 30),
      buildWeightedMatchExpression('$category', searchQuery,
22),
      buildWeightedMatchExpression('$brand', searchQuery, 20),
      buildWeightedMatchExpression('$slug', searchQuery, 18)
    );
  }
  for (const token of searchTokens) {
    scoreExpressions.push(
      buildWeightedMatchExpression('$name', token, 12),
```

```
        buildWeightedMatchExpression('$nameUk', token, 12),
        buildWeightedMatchExpression('$category', token, 9),
        buildWeightedMatchExpression('$brand', token, 8)
      );
    }
    return { $add: [...scoreExpressions, 0] };
};
```

Наведений у лістингу 2.1 фрагмент показує формування вагової моделі релевантності. Більша вага надається назвам товару, категорії, бренду та адресному ідентифікатору, оскільки ці поля найчастіше визначають основний зміст пошукового запиту. Інші поля, зокрема тканина, колір, теги й опис, розширюють пошук і дають змогу враховувати додаткові ознаки товару. Це дає змогу відокремити релевантні результати від випадкових часткових збігів і зробити пошук більш придатним для каталогу одягу, де користувач часто шукає не точну назву, а поєднання характеристик.

Лістинг 2.2 демонструє побудову багатокритеріальної фільтрації через універсальний набір умов, який використовується в загальній кінцевій точці пошуку. Такий підхід дозволяє поєднувати текстовий запит, категорії, бренди, тканини, кольори, теги, наявність і ціновий діапазон в одному механізмі. Завдяки цьому користувач може поступово звужувати перелік товарів, а система зберігає єдину логіку обробки параметрів без дублювання окремих сценаріїв для кожного фільтра.

Лістинг 2.2 – Фрагмент реалізації багатокритеріальної фільтрації товарів

```
const buildSearchFilters = (query) => {
  const category = query.category || '';
  const brand = query.brand || '';
  const fabric = query.fabric || '';
  const color = query.color || '';
  const tag = query.tag || '';
  const stock = query.stock || '';
  const searchQuery = query.query ? query.query.trim() : '';
  const minPrice = getNumberOrNull(query.minPrice);
  const maxPrice = getNumberOrNull(query.maxPrice);
  const searchTokens = searchQuery
    .split(/\s+/)
    .map((token) => token.trim())
    .filter(Boolean);
```

```
  const queryFilter =
    searchTokens.length > 0 && searchQuery !== 'all'
      ? { $and: searchTokens.map((token) =>
buildTokenFilter(token)) }
      : {};
  return {
    filters: {
      ...queryFilter,
      ...buildMultiValueFilter('category', category),
      ...buildMultiValueFilter('brand', brand),
      ...buildMultiValueFilter('fabric', fabric),
      ...buildMultiValueFilter('color', color),
      ...(tag ? { tags: { $in: parseMultiValue(tag) } } : {}),
      ...(stock === 'in' ? { countInStock: { $gt: 0 } } : {}),
      ...(stock === 'out' ? { countInStock: { $lte: 0 } } : {}),
      ...(minPrice !== null || maxPrice !== null
        ? { price: { ...(minPrice !== null ? { $gte: minPrice }
: {}), ...(maxPrice !== null ? { $lte: maxPrice } : {}) } }
        : {}),
    },
    searchQuery,
    searchTokens,
  };
};
```

Окремо варто зазначити, що параметри фільтрації можуть мати не лише навігаційне, а й аналітичне значення. Якщо система фіксуватиме, які саме фільтри користувачі застосовують найчастіше, ці дані можна буде використовувати для подальшого аналізу попиту й поведінкових сценаріїв. Наприклад, частота вибору тканини, кольору, категорії, цінового діапазону, наявності або тематичних тегів може показувати, які характеристики товару є найважливішими під час прийняття рішення.

На етапі проєктування можна сформулювати гіпотезу, що для користувачів інтернет-магазину одягу найбільш значущими є не лише стандартні фільтри за категорією й ціною, а й атрибути, пов’язані з матеріалом, стилем, культурним маркером, екологічністю або наявністю товару. У межах подальшого розвитку системи цю гіпотезу можна перевірити шляхом збору статистики використання фільтрів, аналізу послідовності дій користувача, порівняння частоти застосування різних параметрів і зіставлення цих даних із фактом додавання товару в кошик або оформлення замовлення.

Лістинг 2.3 ілюструє механізм серверної перевірки замовлення: система повторно отримує товари з бази даних, перевіряє їх наявність, формує склад замовлення на основі актуальних даних каталогу та самостійно обчислює підсумкові суми. Це зменшує залежність від даних, переданих із клієнтської частини, і підвищує надійність оформлення замовлення.

Лістинг 2.3 – Фрагмент серверної перевірки замовлення та перерахунку підсумкової суми

```
const canAccessOrder = (order, user) =>
  user.isAdmin || order.user.toString() === user._id;
orderRouter.post(
  '/',
  isAuth,
  expressAsyncHandler(async (req, res) => {
    const productIds = req.body.orderItems.map((item) =>
item._id);
    const products = await Product.find({ _id: { $in: productIds
} });
    const productsById = new Map(
      products.map((product) => [product._id.toString(),
product])
    );
    const orderItems = [];
    for (const item of req.body.orderItems) {
      const product = productsById.get(item._id);
      if (!product || product.countInStock < item.quantity) {
        return res.status(400).send({ message: 'Product out of
stock' });
      }
      orderItems.push({
        slug: product.slug,
        name: product.name,
        nameUk: product.nameUk || '',
        quantity: item.quantity,
        image: product.image,
        price: product.price,
        product: product._id,
      });
    }
    const itemsPrice = round2(
      orderItems.reduce((sum, item) => sum + item.price *
item.quantity, 0)
    );
    const shippingPrice = itemsPrice > 100 ? round2(0) :
round2(10);
    const discountPrice = round2(0.05 * itemsPrice);
    const totalPrice = round2(itemsPrice + shippingPrice -
```

```
discountPrice);
  })
);
```

Для перевірки сценарію онлайн-оплати передбачено використання @paypal/react-paypal-js і тестового середовища PayPal Sandbox [32], [33]. Це дає змогу відтворити процес оплати без виконання реальних фінансових операцій і перевірити взаємодію між сторінкою замовлення, платіжним компонентом та подальшим оновленням статусу оплати.

Контроль доступу передбачено на двох рівнях. Перший рівень охоплює автентифікацію користувача та поділ ролей за допомогою JWT-токена, проміжного обробника `isAuth` і перевірки адміністративних прав через `isAdmin`. JWT використовується для передавання підтвердженої інформації про користувача між клієнтською та серверною частинами без необхідності зберігати стан сесії на сервері. Після входу користувача сервер формує токен, який надалі додається до захищених запитів і дає змогу визначити, від імені якого облікового запису виконується дія [34].

Окремим елементом безпеки є зберігання паролів у хешованому вигляді. Для цього використовується `bcryptjs`, який дає змогу не зберігати відкритий пароль у базі даних. Під час реєстрації пароль перетворюється на хеш, а під час входу система порівнює введене значення з уже збереженим хешем. Такий підхід відповідає загальним рекомендаціям щодо безпечного зберігання паролів, за якими пароль не повинен зберігатися як відкритий текст, а має оброблятися спеціалізованим алгоритмом хешування [35]. Це знижує ризики, пов’язані з можливим несанкціонованим доступом до бази даних користувачів.

Другий рівень контролю доступу стосується перевірки доступу до конкретного ресурсу, наприклад замовлення. Навіть якщо користувач успішно пройшов автентифікацію, система має додатково перевірити, чи належить запитуване замовлення саме цьому користувачеві або чи має він права адміністратора. Це дає змогу розмежовувати ролі користувачів і водночас захищати персональні дані від несанкціонованого перегляду.

2.6 Проєктування користувацького інтерфейсу та засобів підвищення зручності використання

Під час проєктування користувацького інтерфейсу за основу взято принципи мінімалізму, логічної навігації, послідовності сценаріїв і зменшення когнітивного навантаження. Для інтернет-магазину це означає, що шлях від ознайомлення з товаром до завершення замовлення має бути коротким, передбачуваним і зрозумілим, що узгоджується з сучасними підходами до UX-дизайну цифрових сервісів [1], [3].

Основний сценарій неавторизованого користувача охоплює перехід від головної сторінки до каталогу або тематичної добірки, пошук і фільтрацію, перегляд сторінки товару та додавання його в кошик. Для авторизованого користувача цей шлях доповнюється введенням адреси доставки, вибором способу оплати, переглядом підсумку замовлення та історії покупок. Для адміністратора передбачено окремий сценарій з панеллю керування, редагуванням сутностей і переглядом аналітики.

Навігаційну структуру спроєктовано так, щоб користувач швидко переходив до основних дій: перегляду товарів, пошуку, кошика, входу в обліковий запис і зміни мови. Верхня навігація виконує роль постійної точки орієнтації, а тематичні блоки на головній сторінці створюють додаткові входи в каталог. Це важливо для інтернет-магазину одягу, де пошук може здійснюватися не лише за назвою, а й за матеріалом, стилем, культурним маркером або ціннісною ознакою.

Окрему увагу приділено сторінці пошуку. Вона поєднує текстове поле, багатовибіркові фільтри, перемикачі наявності, вибір цінового діапазону та блок активних параметрів. Таке компонування спрощує уточнення запиту й дає користувачеві змогу контролювати умови, за якими здійснюється відбір товарів.

Важливим засобом підвищення зручності використання є двомовність. Перемикання мови відбувається без перезавантаження сторінки, а предметні терміни додатково опрацьовуються словниковим механізмом. Для українського

інтернет-магазину це пов’язано як із доступністю, так і з можливістю представлення каталогу ширшій аудиторії.

Візуальну подачу інтерфейсу спроєктовано з урахуванням українського контексту та ціннісної концепції магазину. Атмосфера сайту має бути емоційно стриманою, теплою й довірчою, без агресивного продажного тону [36]. Тому доцільними є спокійна палітра, м’які контрасти, достатня кількість вільного простору та зрозуміла типографіка. Таке рішення підтримує людяний тон комунікації й не відволікає користувача від товарів, матеріалів і змістових акцентів.

Колірна логіка інтерфейсу має підкреслювати природність, локальність і надійність. Нейтральні світлі відтінки можуть використовуватися як основа для сторінок і карток товарів, теплі акцентні кольори — для кнопок, тематичних блоків і важливих дій, а стримані темні відтінки — для тексту та навігації. Така палітра не створює враження надмірної комерційності й водночас підтримує асоціації з натуральними тканинами, ремісничістю, українською ідентичністю та відповідальним споживанням.

Головна сторінка виконує вітринну та смислову функції [37]. У ній передбачено тематичні блоки для переходу до натуральних тканин, добірок з вишивкою, екоорієнтованих товарів, а також комунікаційний акцент на інклюзивності, підтримці України та соціальній відповідальності бренду. У такий спосіб інтерфейс продовжує гіпотезу першого розділу щодо значущості національного контексту, емоційної ідентифікації користувача з брендом і демонстрації чіткої соціальної позиції.

Тон інтерфейсної комунікації має бути простим, доброзичливим і зрозумілим. Для такого магазину недоцільні агресивні заклики до покупки; натомість доречні формулювання, які допомагають користувачеві орієнтуватися, пояснюють зміст добірок і підкреслюють цінність матеріалів, локального виробництва та соціальної позиції бренду.

До засобів підвищення зручності використання належать адаптивна побудова сторінок, послідовна структура екранних блоків, індикатори етапів

оформлення замовлення, зрозумілі статусні повідомлення, розмежування користувацьких і адміністративних сценаріїв та тематичні точки входу в каталог. У сукупності ці рішення формують UX/UI-модель інтернет-магазину, у якій користувач бачить не лише перелік товарів, а зрозумілий шлях взаємодії з системою [3].

Отже, під час проєктування інтерфейсу враховано логіку користувацьких сценаріїв, візуальну подачу, емоційний тон комунікації та український контекст бренду. Інтерфейс підтримує шлях покупця від головної сторінки до оформлення замовлення, окремий сценарій адміністратора та наскрізну двомовність предметних даних. Це узгоджує систему з концепцією магазину, побудованою навколо змістовної навігації, локального контексту та соціально відповідальної комунікації [2].

2.7 Висновок до другого розділу

У другому розділі було обґрунтовано вибір технологій і спроєктовано архітектуру інтернет-магазину одягу. Для клієнтської частини обрано React, для серверної логіки — Node.js та Express, для зберігання даних — MongoDB і Mongoose. Це дає змогу побудувати клієнт-серверний застосунок із можливістю подальшого розширення.

Архітектура системи передбачає поділ на клієнтський інтерфейс, серверну логіку та базу даних. База даних охоплює користувачів, товари, замовлення й переклади. Особливістю моделі є авторська система атрибутів товару в колекції Product: тканина, колір, бренд, категорія, теги, двомовні назви й описи.

Функціональну структуру побудовано за модульним принципом. Виокремлено модулі каталогу, пошуку й фільтрації, авторизації та профілю, оформлення замовлення, адміністрування та двомовності. Такий підхід спрощує супровід системи й дає змогу розширювати окремі частини без повного перегляду архітектури.

РОЗДІЛ 3. ПРОГРАМНА РЕАЛІЗАЦІЯ ТА АНАЛІЗ РЕЗУЛЬТАТІВ РОБОТИ

У третьому розділі подано практичну реалізацію інтернет-магазину та результати перевірки його основних функціональних підсистем. Побудову матеріалу узгоджено з реальним сценарієм роботи системи: від клієнтської частини, серверної логіки й взаємодії з базою даних до пошуку, двомовності, оформлення замовлення, адміністративного керування та аналізу працездатності.

3.1 Реалізація клієнтської частини інтернет-магазину

Клієнтську частину інтернет-магазину побудовано як цілісний користувацький простір, у межах якого покупець послідовно переходить від ознайомлення з брендом до вибору товару та оформлення замовлення. Система функціонує як SPA, тому основні переходи між екранами виконуються без повного перезавантаження сторінки. Центральним елементом є файл App.js, у якому зосереджено маршрути, верхню навігацію, бічне меню категорій, індикатор кошика та перемикач мови. Завдяки цьому структура переходів залишається передбачуваною, а користувачеві не доводиться щоразу заново орієнтуватися в логіці сторінок.

Стан авторизованого користувача, кошика, адреси доставки та способу оплати підтримується через контекст Store та локальне сховище браузера [38]. Це означає, що інтернет-магазин не втрачає поточний сценарій після оновлення сторінки: товари в кошику, вибрана мова й етап оформлення замовлення залишаються доступними під час подальшої роботи. Такий підхід безпосередньо впливає на зручність використання і знижує кількість перерваних сценаріїв покупки.

На рисунку 3.1 видно, що головна сторінка будується як поєднання презентаційного блоку, тематичних переходів і вітринного каталогу. Саме тут

задається перше враження про інтернет-магазин і формується контекст подальшого пошуку. Унаслідок цього клієнтська частина не зводиться лише до показу товарів, а формує впізнавану комунікацію бренду ще до переходу на сторінку окремої позиції.

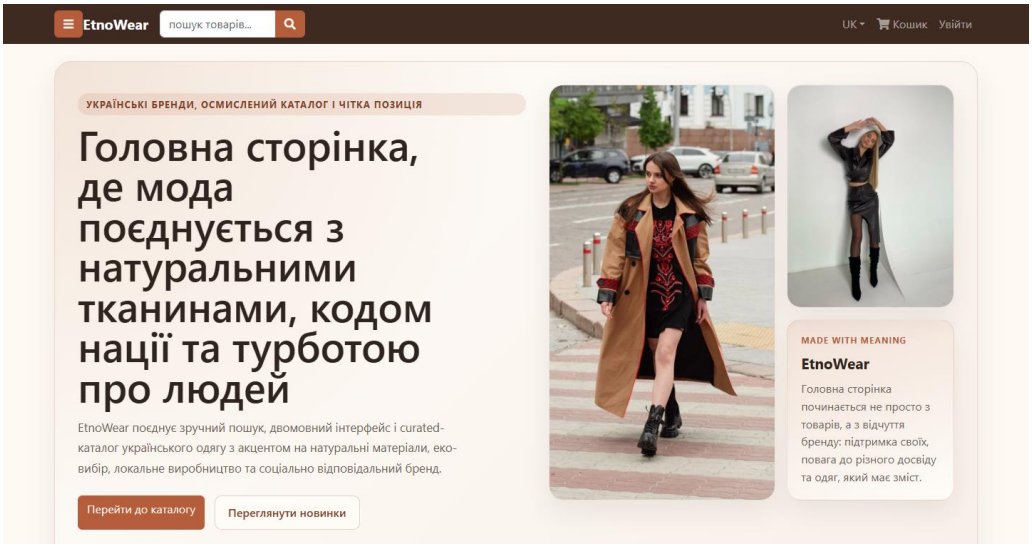

Рисунок 3.1 – Головна сторінка інтернет-магазину

Сторінка товару, представлена на рисунку 3.2 концентрує всю інформацію, потрібну для прийняття рішення про покупку: зображення, назву, марку, тканину, теги, статус наявності та кнопку додавання в кошик.

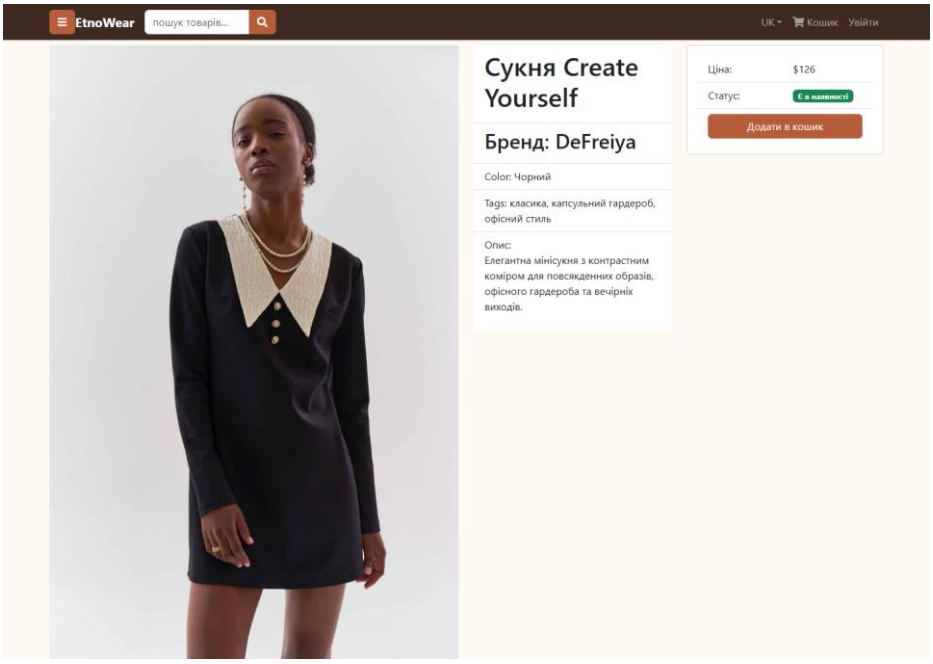

Рисунок 3.2 – Сторінка окремого товару

Кошик, зображений на рисунку 3.3, є проміжною контрольної точкою, у якій користувач може перевірити склад покупки, скоригувати кількість позицій і перейти до оформлення замовлення.

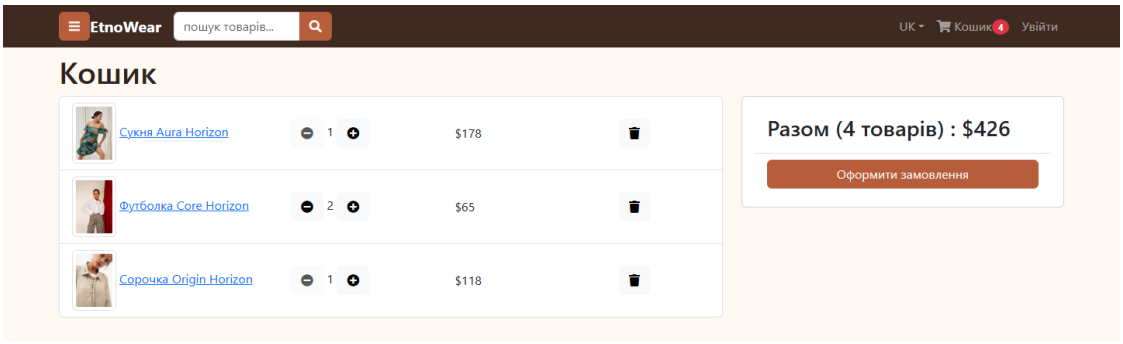

Рисунок 3.3 – Кошик із вибраними товарами

Показ кошика з кількома товарними позиціями дає змогу простежити не лише факт додавання товару, а й логіку повторного перерахунку вартості, зміну кількості, відображення окремих карток та узгодження підсумкової суми перед переходом до оформлення замовлення.

Для персоналізованої роботи з інтернет-магазином передбачено реєстрацію та вхід користувача. Вікно реєстрації, подане на рисунку 3.4, містить поля для введення імені, електронної пошти, пароля та повторного підтвердження пароля.

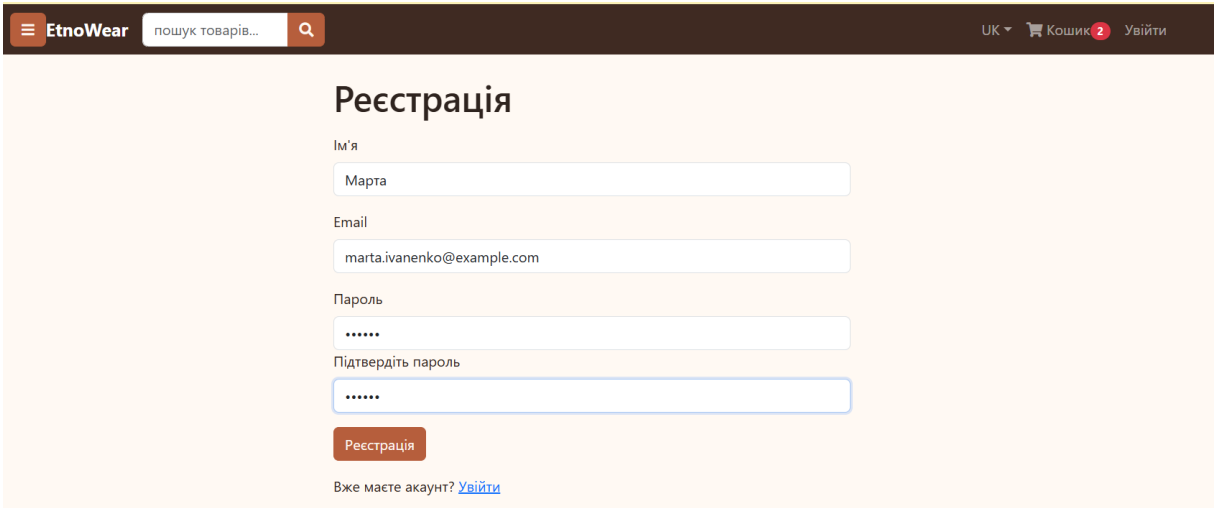

Рисунок 3.4 – Вікно реєстрації користувача

Після створення облікового запису користувач може виконати вхід у систему. На рисунку 3.5 показано форму входу, у якій використовуються електронна пошта та пароль. Після успішної авторизації користувач отримує доступ до особистого профілю, історії замовлень і подальшого оформлення покупок від свого імені.

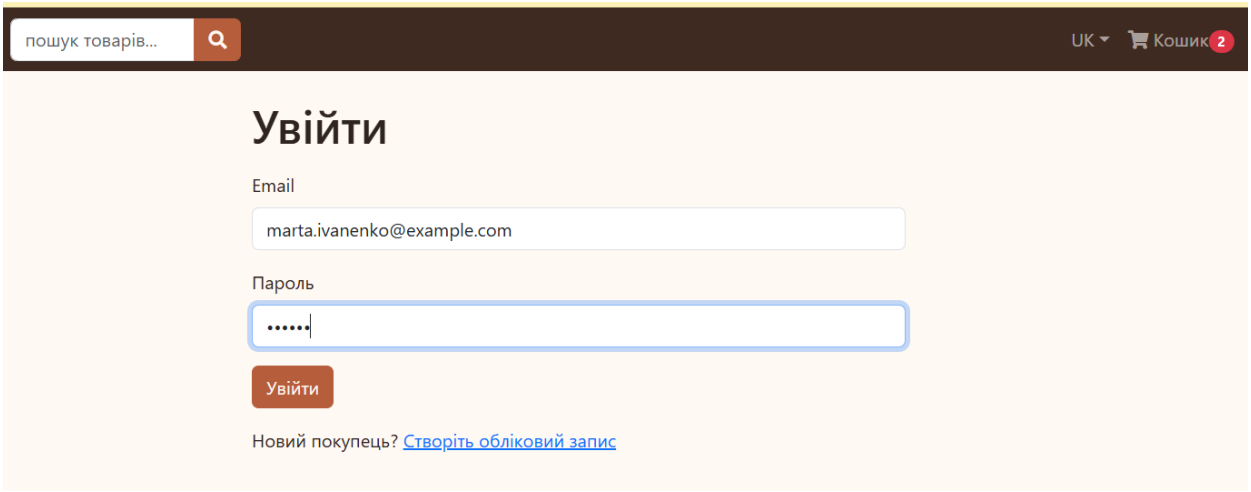

Рисунок 3.5 – Вікно входу користувача

Особистий профіль користувача показано на рисунку 3.6. У цьому вікні відображаються ім’я, електронна пошта та пароль, а також передбачено можливість оновити власні дані. Наявність такого розділу дає користувачеві базовий контроль над обліковим записом без звернення до адміністратора.

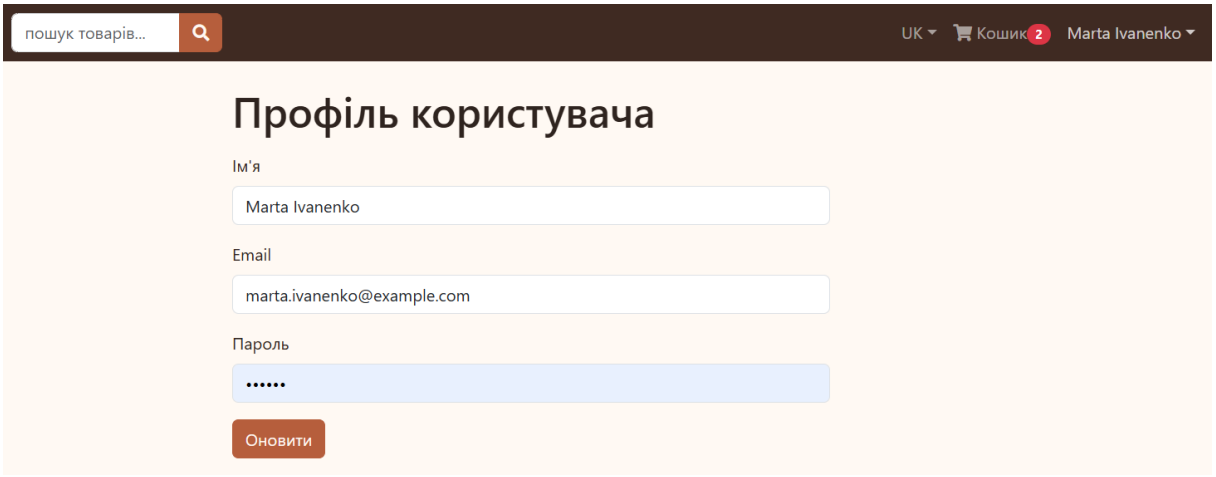

Рисунок 3.6 – Вікно профілю користувача

Окремим елементом особистого кабінету є історія замовлень. На рисунку 3.7 наведено вікно, у якому користувач бачить перелік власних замовлень. Це дозволяє перевірити попередні покупки, їхній склад, вартість і поточний статус.

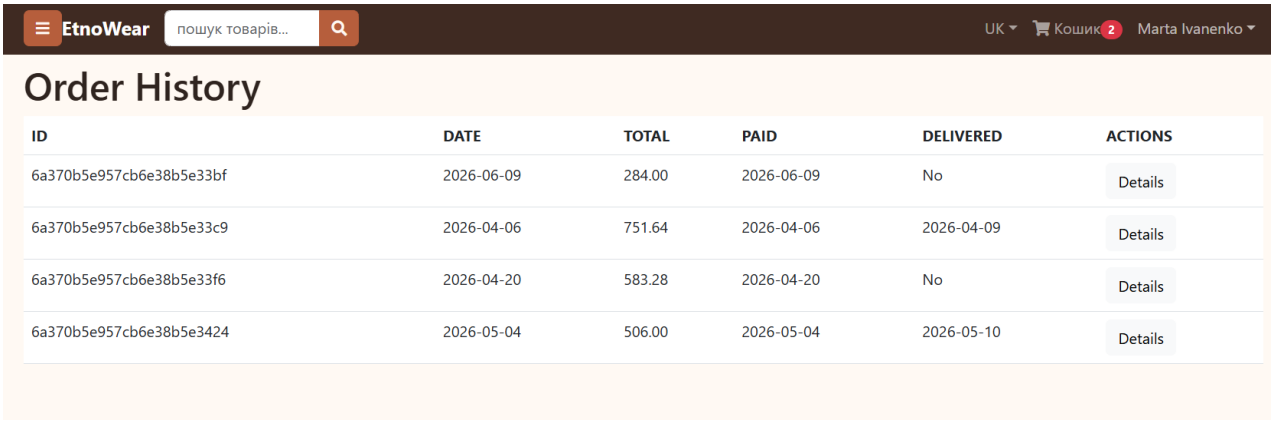

Рисунок 3.7 – Вікно історії замовлень користувача

Реєстрація, вхід, профіль і перегляд історії замовлень завершують персональний сценарій роботи з інтернет-магазином. Користувач не лише переглядає товари й формує кошик, а й отримує власний простір для керування даними та відстеження покупок.

3.2 Реалізація серверної частини та взаємодії з базою даних

Серверну частину реалізовано на основі Express. У файлі server.js, код лістингу якого поданий у додатку Б, налаштовано підключення до MongoDB, обробку JSON-даних та маршрути для товарів, користувачів, замовлень, перекладів, завантаження зображень і службових даних. Завдяки цьому код для різних частин системи розділено за призначенням.

Робота з даними побудована через моделі Mongoose. Для товарів використовуються структуровані поля назви двома мовами, символьного ідентифікатора, категорії, марки, тканини, кольору, тегів, ціни та залишку на складі. Ці дані використовуються на сторінці товару, у пошуку, фільтрації та під час адміністративного редагування.

Окрім роботи з товарами, серверна частина підтримує сценарії, описані в клієнтській частині. Для цього використовуються окремі маршрути користувачів і замовлень, які працюють із моделями User та Order. Після входу користувача система зберігає його дані й токен авторизації, а подальші запити до профілю або історії замовлень виконуються вже від імені конкретного облікового запису.

Модель користувача містить базові дані для роботи з особистим кабінетом. Ці дані використовуються у формах реєстрації, входу та оновлення профілю. Це пов’язує інтерфейсні екрани користувача із серверною логікою автентифікації. Модель замовлення зберігає перелік товарів, дані доставки, спосіб оплати, загальну суму та посилання на користувача, який створив замовлення.

Зв’язок між інтерфейсом, серверною логікою і базою даних можна простежити на прикладі оновлення товару, що схематично зображено на рисунку 3.8. У цьому сценарії адміністратор змінює дані товару в інтерфейсі, клієнтська частина передає оновлені значення на сервер, а серверний маршрут зберігає зміни у відповідному документі колекції Product.

На рисунку послідовно показано основні етапи цього процесу. Спочатку адміністратор працює з формою редагування товару, де вносить зміни до його назви, опису, ціни, категорії або інших характеристик. Після підтвердження редагування клієнтська частина формує HTTP-запит методом PUT до маршруту /api/products/:id, у якому ідентифікатор id визначає конкретний товар, що має бути оновлений.

Перед внесенням змін сервер перевіряє права доступу користувача. Це потрібно для того, щоб змінювати товари могли лише авторизовані користувачі з відповідною роллю адміністратора. Якщо перевірка проходить успішно, сервер звертається до моделі Product, оновлює потрібні поля та зберігає результат у колекції products. Після цього оновлені дані можуть бути повторно відображені в каталозі, тому користувач бачить уже актуальну інформацію про товар.

Рисунок 3.8 – Послідовність оновлення товару

Адміністратор відкриває форму редагування, змінює атрибути товару, після чого клієнтська частина передає нові значення до маршруту ‘PUT /api/products/:id’. Код маршруту представлено у лістингу 3.1.

Лістинг 3.1 – Фрагмент серверного маршруту оновлення товару

```
productRouter.put(
  '/:id',
  isAuth,
  isAdmin,
  expressAsyncHandler(async (req, res) => {
    const product = await Product.findById(req.params.id);
    if (product) {
      product.name = req.body.name;
      product.nameUk = req.body.nameUk || '';
      product.slug = req.body.slug;
      product.price = req.body.price;
      product.category = req.body.category;
      product.tags = Array.isArray(req.body.tags) ? req.body.tags : [];
      product.brand = req.body.brand;
      product.fabric = req.body.fabric;
      product.countInStock = req.body.countInStock;
      await product.save();
      res.send({ message: 'Product Updated' });
    }
  })
);
```

Наведений фрагмент показує, що оновлення товару виконується лише після перевірки автентифікації та ролі адміністратора. Після цього сервер

знаходить документ товару за ідентифікатором, записує нові значення в модель Product і зберігає зміни в колекції products. Сервер перевіряє права доступу, оновлює документ у колекції products, а результат одразу відображається в каталозі та адміністративному списку.

Подана схема (див. рис. 3.8) показує, що зміна товару проходить повний цикл: інтерфейсна форма, серверний маршрут, модель, документ у колекції та подальше відображення в публічній частині інтернет-магазину.

Лістинг 3.2 показує структуру реального документа товару, яка використовується під час подальшого формування карток у каталозі, сторінки окремого товару та списку позицій в адміністративному модулі.

Лістинг 3.2 – Приклад документа товару після збереження в колекції `products`

```
{
  "_id": "6a331dc40c36f3eef44291ae",
  "name": "Create Yourself Dress",
  "nameUk": "Сукня Create Yourself",
  "slug": "defreiya-create-yourself-dress",
  "brand": "DeFreiya",
  "category": "Dresses",
  "color": "Black",
  "tags": ["classic", "capsule", "office"],
  "price": 126,
  "countInStock": 6,
  "fabric": "Viscose blend",
  "descriptionUk": "Елегантна мінісукня з контрастним коміром..."
}
```

На рисунку 3.9 представлено форму оновлення товару для адміністратора на сайті.

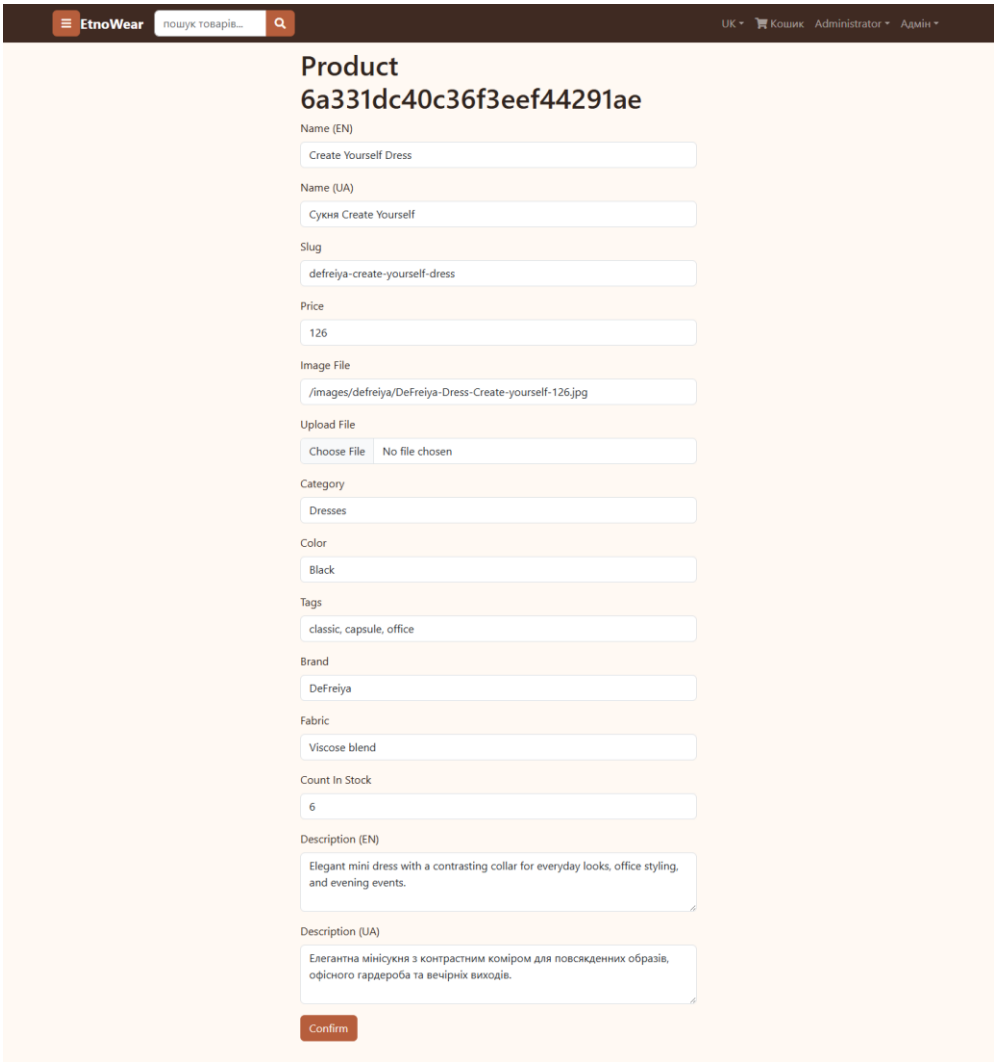

Рисунок 3.9 – Форма редагування товару в адміністративному модулі

Форма редагування безпосередньо відтворює структуру документа (лістинг 3.2) товару в базі даних, тому адміністративний модуль працює не з умовними полями, а з реальними атрибутами каталогу.

Серверна частина забезпечує не лише адміністративне редагування товарів, а й повний користувацький сценарій: створення облікового запису, вхід, роботу з профілем, формування замовлення та перегляд історії покупок. Це узгоджує серверну логіку з клієнтськими екранами, описаними в попередньому підрозділі, і показує, як дії користувача в інтерфейсі переходять у запити до відповідних маршрутів API.

3.3 Реалізація функціоналу пошуку, фільтрації та двомовності

Пошук у каталозі реалізовано як багатокритеріальний механізм, що поєднує текстовий запит, категорію, марку, тканину, колір, теги, стан наявності, порядок сортування та ціновий діапазон. На клієнтському рівні стан вибраних параметрів перетворюється на адресний рядок за допомогою ‘URLSearchParams’. Унаслідок цього параметри категорії, марки, тканини, кольору, тегів, наявності, сортування та цінового діапазону фіксуються у посиланні й можуть бути відтворені після повторного відкриття сторінки [21].

На серверному рівні маршрут ‘/api/products/search’ об'єднує текстовий пошук і багатокритеріальну фільтрацію. Для текстового запиту використовуються регулярні вирази за кількома полями товару, а для категорії, марки, тканини, кольору та тегів формуються окремі умови відбору. Така схема забезпечує придатність каталогу до складних сценаріїв пошуку, коли користувач поєднує кілька ознак одночасно.

Під час зміни параметрів фільтрації клієнтська частина оновлює запит до сервера та отримує лише ті товари, які відповідають заданим умовам. Це зменшує кількість нерелевантних позицій у каталозі й робить взаємодію з інтернет-магазином більш цілеспрямованою. Наприклад, користувач може спочатку вибрати категорію товару, потім уточнити бренд, матеріал або колір, а після цього відсортувати результати за ціною чи іншою ознакою.

Практичне значення такого підходу полягає в тому, що користувач не просто бачить каталог, а може швидко звузити асортимент до логічно близької підбірки. Наприклад, поєднання ознак “Shirts”, “PoliVik”, “embroidered” та “in stock” дає невеликий, але змістовний перелік релевантних позицій, який зручно використовувати для цілеспрямованого вибору. На рисунку 3.10 подано сторінку пошуку з базовим набором фільтрів.

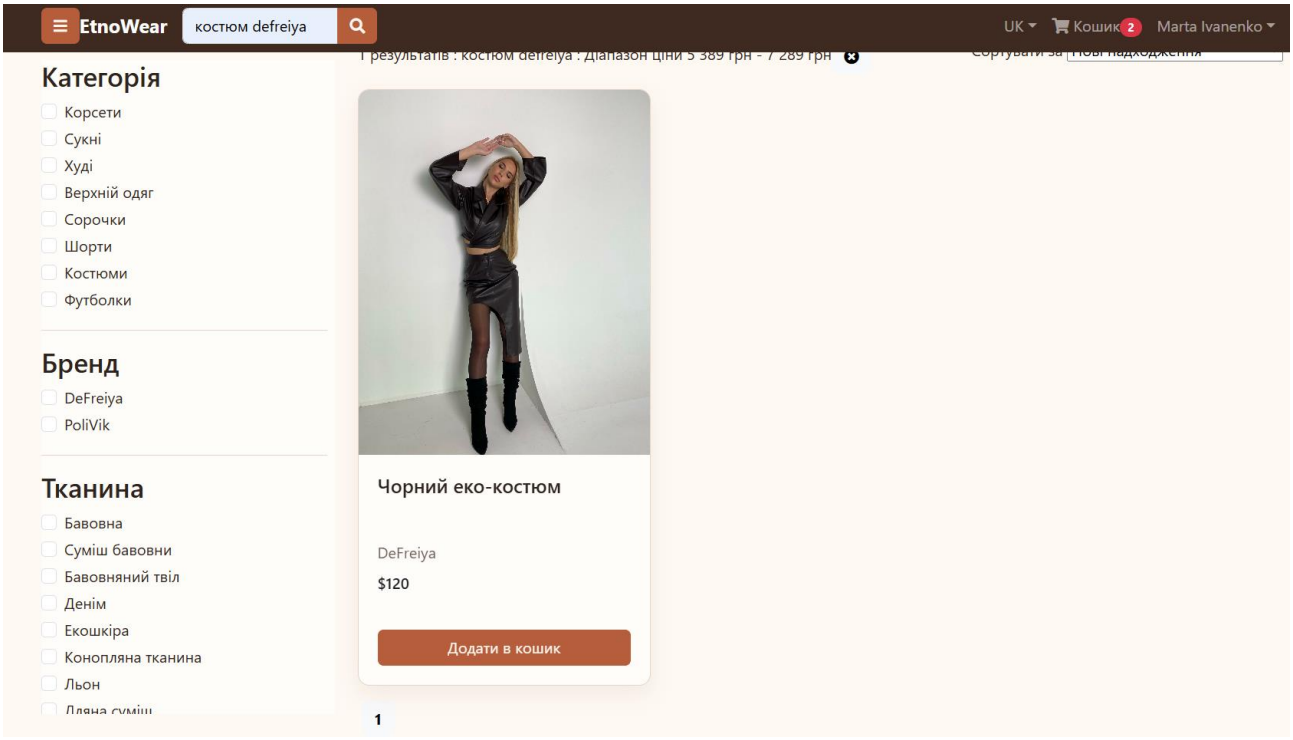

Рисунок 3.10 – Сторінка пошуку з базовим набором фільтрів

Фільтраційна панель і результати відбору розміщено на одному екрані (див. рис. 3.10), що спрощує контроль над поточним станом пошуку.

Лістинг 3.3 показує, як поточний стан фільтрації перетворюється на адресу сторінки. Це рішення робить результати пошуку відтворюваними, оскільки вибрані параметри не губляться після оновлення сторінки або повторного відкриття посилання.

Лістинг 3.3 – Фрагмент формування параметрів пошуку

```
const buildSearchParams = (filters) => {
  const params = new URLSearchParams();
  if (filters.query && filters.query !== 'all') {
    params.set('query', filters.query);
  }
  if (filters.category?.length) {
    params.set('category', filters.category.join(','));
  }
  if (filters.brand?.length) {
    params.set('brand', filters.brand.join(','));
  }
```

```
  if (filters.fabric?.length) {
    params.set('fabric', filters.fabric.join(','));
  }
  return params;
};
```

Маршрут ‘/api/products/search’ забезпечує перетворення параметрів адресного рядка на умови вибірки в MongoDB. На клієнтському рівні сторінка пошуку формує URL із параметрами “query”, “category”, “brand”, “fabric”, “color”, “tag”, “stock”, “minPrice”, “maxPrice”, “page” та “order”, після чого ці значення передаються на сервер. У серверному маршруті виконується нормалізація параметрів, розбиття текстового запиту на окремі слова, побудова багатозначних фільтрів, формування обмежень за ціною та наявністю товару. Якщо текстовий запит не порожній, пошук виконується з використанням агрегування та обчислення релевантності; в іншому випадку застосовується звичайна вибірка find() із сортуванням і посторінковим виведенням.

Лістинг 3.4 – Фрагмент серверного маршруту багатокритеріального пошуку товарів

```
const buildTokenFilter = (token) => {
const escapedToken = escapeRegExp(token);
return {
$or: [
{ name: { $regex: escapedToken, $options: 'i' } },
{ nameUk: { $regex: escapedToken, $options: 'i' } },
{ category: { $regex: escapedToken, $options: 'i' } },
{ brand: { $regex: escapedToken, $options: 'i' } },
{ fabric: { $regex: escapedToken, $options: 'i' } },
{ color: { $regex: escapedToken, $options: 'i' } },
{
tags: {
$elemMatch: {
$regex: escapedToken,
```

```
$options: 'i',
},
},
},
{ description: { $regex: escapedToken, $options: 'i' } },
{ descriptionUk: { $regex: escapedToken, $options: 'i' } },
],
};
};
```

Поданий фрагмент (див. лістинг 3.4) підтверджує, що механізм пошуку не обмежується простим порівнянням одного поля. Сервер одночасно враховує текстовий запит, множинні ознаки каталогу, наявність товару, ціновий діапазон, сортування та посторінковий перегляд. Також пошук підтримує як англомовні, так і україномовні значення, оскільки перевіряються поля “name”, “nameUk”, “description” та “descriptionUk”. На рисунку 3.11 наведено приклад багатокритеріальної фільтрації товарів.

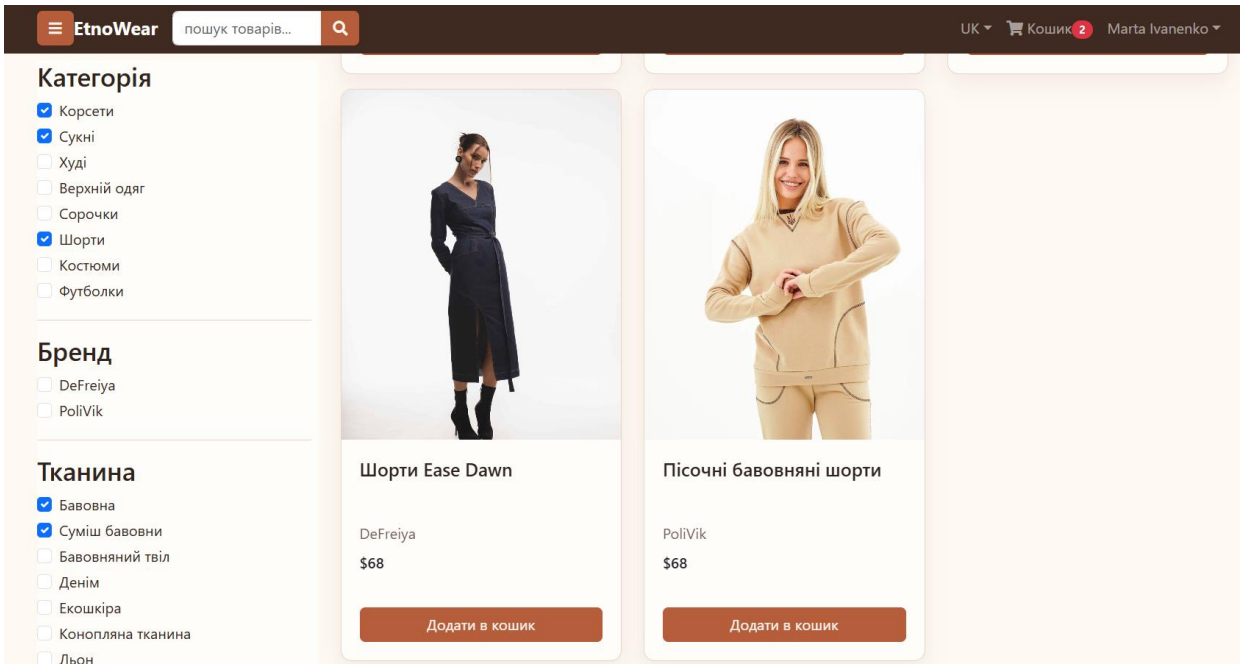

Рисунок 3.11 – Приклад багатокритеріальної фільтрації товарів

Сторінка показує режим, у якому пошук уже звужено за кількома параметрами одночасно. Така форма подання особливо корисна для каталогу з великою кількістю товарних позицій.

Двомовність реалізовано не як формальне дублювання написів, а як повноцінний робочий механізм. Після перемикання мови змінюються не лише кнопки й заголовки, а й категорії, тканини, теги, описи товарів та частина назв. Для цього використовуються локальний словник інтерфейсу та окремий словниковий маршрут перекладів.

Лістинг 3.5 – Фрагмент вибору локалізованих даних і перекладу словникових значень

```
const localizedFieldName = `${fieldName}Uk`;
if (language === 'uk' && entity[localizedFieldName]) {
return entity[localizedFieldName];
}
const baseValue = entity[fieldName] || entity[localizedFieldName] || '';
if (language === 'uk' && fieldName === 'name') {
return translateProductNameWithDictionary(baseValue, dictionaryEntries);
}
return baseValue;
};
export function LanguageProvider({ children }) {
const [language, setLanguageState] = useState(
localStorage.getItem('language') || 'en'
);
const [dictionaryEntries, setDictionaryEntries] = useState([]);
const setLanguage = (nextLanguage) => {
setLanguageState(nextLanguage);
localStorage.setItem('language', nextLanguage);
};
```

У лістингу 3.5 показано, як система вибирає локалізовані поля товару та перекладає словникові значення. Якщо користувач обирає українську мову і для товару передбачено українську назву або опис, інтерфейс відображає саме ці значення. Якщо локалізоване поле відсутнє, система використовує базове значення. На рисунку 3.12 наведено порівняння сторінки товару в українській та англійській версіях.

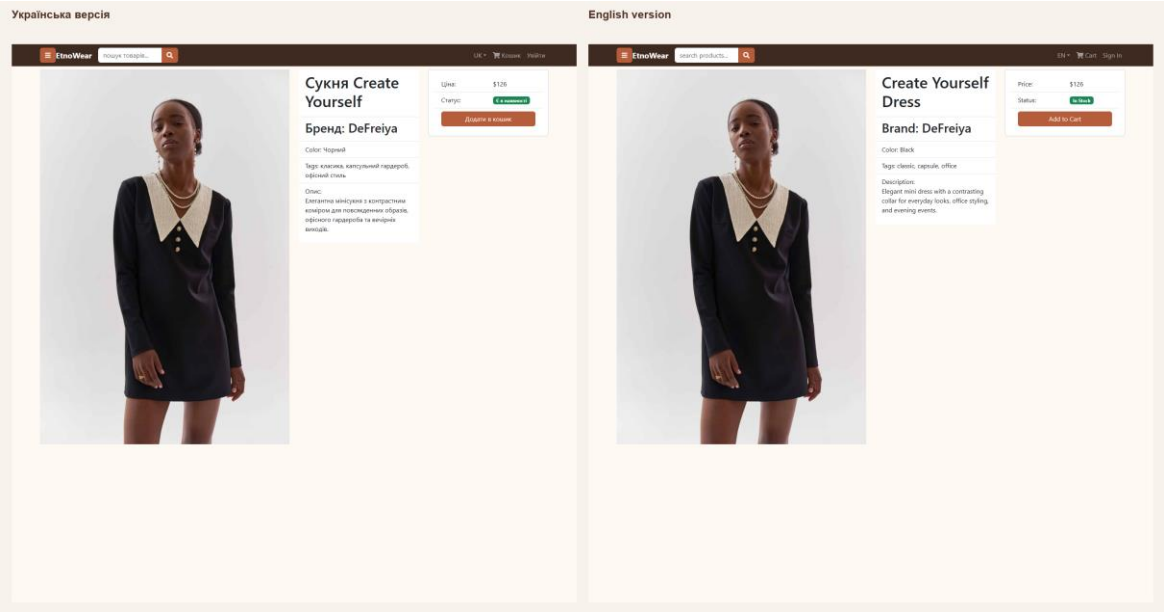

Рисунок 3.12 – Відображення сторінки товару в різних локалізаціях

Порівняння двох мовних режимів демонструє, що локалізація поширюється і на інтерфейсні елементи, і на предметний зміст сторінки. Отже, пошук, фільтрація та двомовність у системі працюють як пов’язані частини одного сценарію: користувач задає параметри добору в інтерфейсі, клієнтська частина передає їх через URL, сервер формує запит до бази даних, а результат повертається у вигляді локалізованого каталогу товарів.

Подальша апробація реалізованого функціоналу пошуку, фільтрації та двомовності може бути пов’язана з оцінюванням того, які саме параметри відбору є найціннішими для користувача під час вибору одягу. Окремий інтерес становить питання, чи впливає двомовність інтерфейсу на сприйняття каталогу та зручність пошуку для різних груп користувачів.

Перспективним напрямом подальшого дослідження є також перевірка доцільності введення окремого фільтра для адаптивного або інклюзивного одягу. З огляду на загальну концепцію інтернет-магазину, орієнтовану на інклюзивність, натуральні матеріали та соціальну чутливість бренду, такий параметр може мати не лише функціональне, а й ціннісне значення.

3.4 Реалізація оформлення замовлення

Оформлення замовлення побудовано як послідовний сценарій із кількох етапів: кошик, адреса доставки, спосіб оплати, перевірка замовлення та сторінка вже створеного документа. На клієнтському рівні користувач бачить зрозумілий рух по кроках, а на серверному рівні кожен створений документ проходить повторну перевірку складу замовлення, наявності товарів і підсумкової суми.

На сторінці PlaceOrderScreen формується остаточний перегляд замовлення: товари, адреса доставки, спосіб оплати, вартість товарів, доставка, знижка та загальна сума. Після підтвердження ці дані надсилаються до маршруту ‘POST /api/orders, а після створення документа користувач переходить на сторінку OrderScreen, де може відстежити статуси оплати та доставки. На рисунку 3.13 подано екран перевірки замовлення перед остаточним підтвердженням.

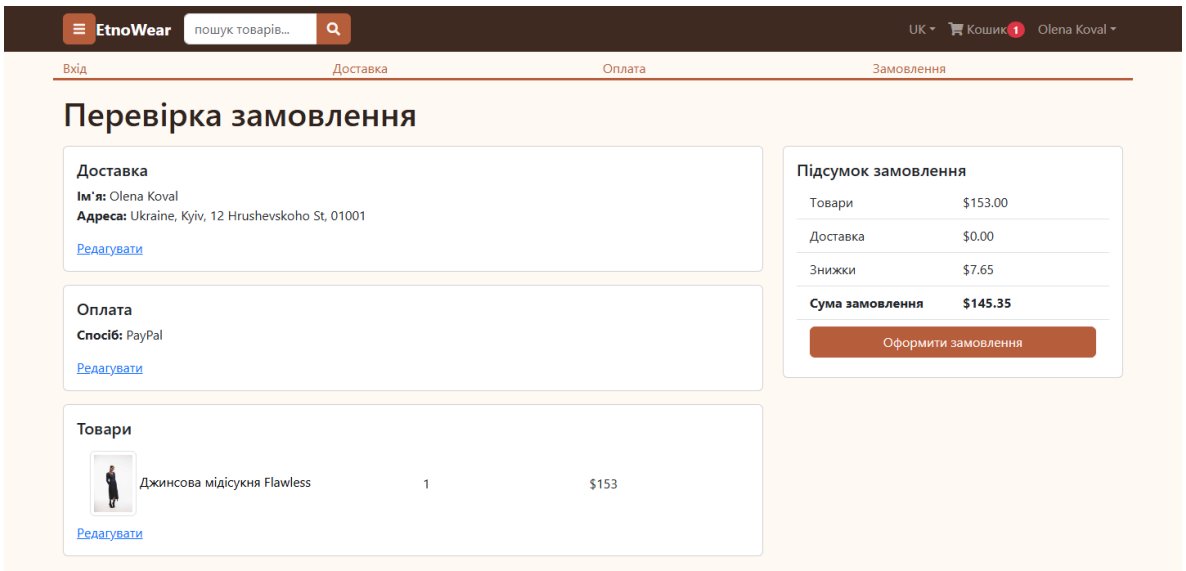

Рисунок 3.13 – Екран перевірки замовлення перед остаточним підтвердженням

Саме на цьому етапі користувач бачить повну картину майбутньої покупки й може скоригувати дані до створення замовлення. Це зменшує ризик помилкового оформлення, оскільки всі ключові дані зібрані на одному екрані.

Клієнтська частина передає на сервер не лише перелік товарів, а й адресу доставки, спосіб оплати та розраховані суми. Фрагмент створення замовлення на сторінці PlaceOrderScreen подано у лістингу 3.6.

Лістинг 3.6 – Фрагмент створення замовлення на клієнтському рівні

```
const placeOrderHandler = async () => {
  try {
    dispatch({ type: 'CREATE_REQUEST' });
    const { data } = await axios.post(
      '/api/orders',
      {
        orderItems: cart.cartItems,
        shippingAddress: cart.shippingAddress,
        paymentMethod: cart.paymentMethod,
        itemsPrice: cart.itemsPrice,
        shippingPrice: cart.shippingPrice,
        discountPrice: cart.discountPrice,
        totalPrice: cart.totalPrice,
      },
      {
        headers: {
          authorization: `Bearer ${userInfo.token}`,
        },
      }
    );
    dispatch({ type: 'CREATE_SUCCESS' });
    ctxDispatch({ type: 'CART_CLEAR' });
    localStorage.removeItem('cartItems');
    navigate(`/order/${data.order._id}`);
  } catch (err) {
    dispatch({ type: 'CREATE_FAIL' });
  }
};
```

У лістингу 3.6 показано, що створення замовлення виконується від імені авторизованого користувача. Після успішної відповіді сервера кошик очищується, а користувач переходить на сторінку створеного замовлення. Це завершує сценарій покупки на клієнтському рівні.

На серверному рівні маршрут ‘POST /api/orders’ приймає дані замовлення, перевіряє товари та створює документ у колекції orders. Фрагмент маршруту наведено у лістингу 3.7.

Лістинг 3.7 – Фрагмент серверного маршруту створення замовлення

```
orderRouter.post(
  '/',
  isAuth,
  expressAsyncHandler(async (req, res) => {
    const newOrder = new Order({
      orderItems: req.body.orderItems.map((item) => ({
        ...item,
        product: item._id,
      })),
      shippingAddress: req.body.shippingAddress,
      paymentMethod: req.body.paymentMethod,
      itemsPrice: req.body.itemsPrice,
      shippingPrice: req.body.shippingPrice,
      discountPrice: req.body.discountPrice,
      totalPrice: req.body.totalPrice,
      user: req.user._id,
    });

    const order = await newOrder.save();

    res.status(201).send({
      message: 'New Order Created',
      order,
    });
  })
);
```

Наведений фрагмент демонструє зв’язок між замовленням і конкретним користувачем через поле user. Крім того, кожна позиція замовлення отримує посилання на відповідний товар і надалі дасть можливість аналізувати замовлення та використовувати ці дані в адміністративному модулі.

На рисунку 3.14 подано блок-схему оформлення замовлення в інтернет-магазині. Схема стисло показує, як поєднуються користувацькі кроки й серверна обробка даних під час створення замовлення. Спочатку користувач перевіряє кошик і вводить дані доставки, після чого система формує запит на створення замовлення, зберігає його в БД.

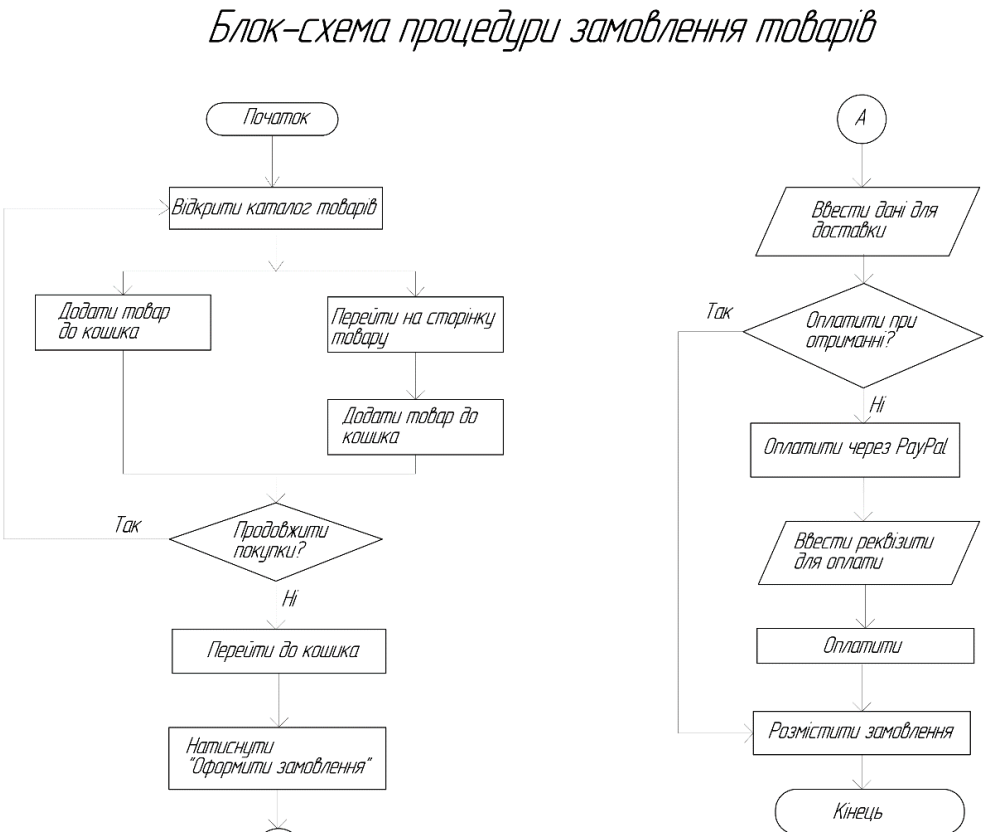

Рисунок 3.14 – Блок-схема оформлення замовлення в інтернет-магазині

На рисунку 3.15 представлено сторінку замовлення зі статусами оплати та доставки.

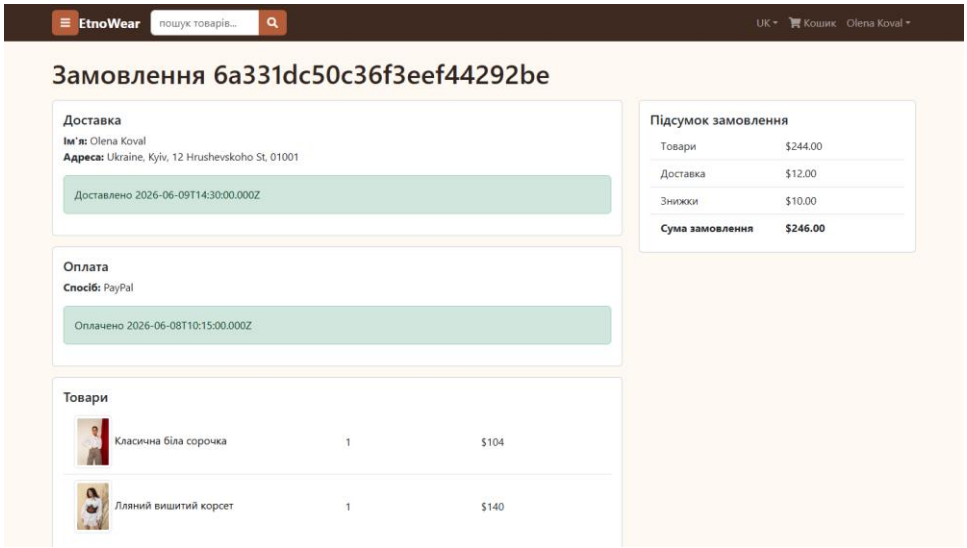

Рисунок 3.15 – Сторінка замовлення зі статусами оплати та доставки

Сторінка замовлення завершує основний користувацький сценарій і водночас слугує вихідною точкою для подальшого адміністративного супроводу. На ній користувач бачить склад покупки, адресу доставки, спосіб оплати, загальну суму та поточний стан виконання замовлення. Реалізований механізм оформлення замовлення поєднує клієнтський сценарій покупки із серверним створенням документа в базі даних. Завдяки цьому замовлення перетворюється на повноцінний запис, пов’язаний із користувачем, товарами, оплатою та доставкою.

3.5 Реалізація адміністративного модуля та аналітики

Адміністративна частина інтернет-магазину охоплює панель керування, список товарів, замовлень і користувачів, а також форми редагування сутностей. Вона доступна лише після перевірки прав адміністратора, тому публічний і службовий режими роботи системи чітко розмежовано.

Адміністративний доступ у системі визначається через службову ознаку isAdmin у даних користувача. Звичайний користувач після реєстрації не отримує доступу до службових сторінок автоматично. Надати права адміністратора можна через редагування облікового запису користувача, де для нього встановлюється відповідна позначка. Приклад такого вікна наведено на рисунку 3.16.

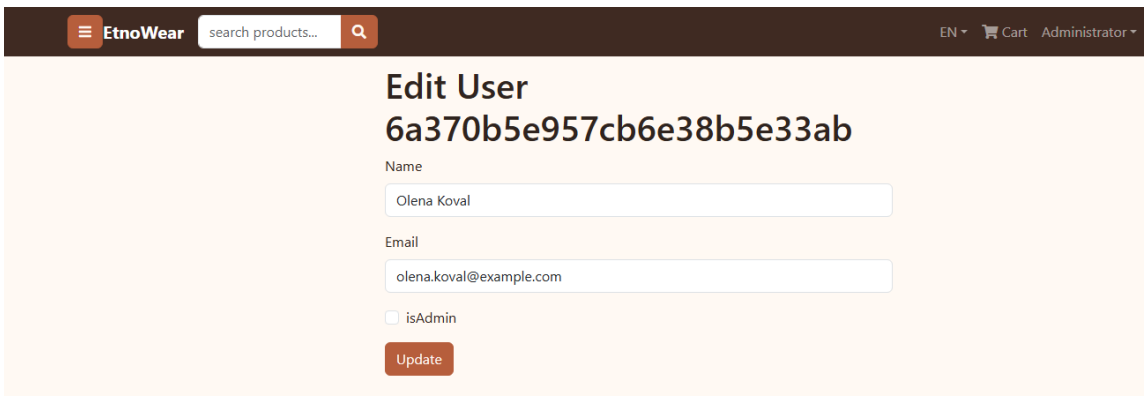

Рисунок 3.16 – Форма редагування користувача з можливістю надання прав адміністратора

Наявність поля isAdmin дозволяє системі перевіряти роль користувача перед відкриттям адміністративних маршрутів. У клієнтській частині це впливає на відображення службових пунктів меню, а на серверному рівні — на доступ до маршрутів редагування товарів, користувачів, замовлень і перегляду аналітики. Тому навіть за прямого переходу за адресою адміністративної сторінки користувач без відповідних прав не повинен отримати доступ до службових дій.

Перевірка доступу виконується за допомогою проміжних обробників isAuth та isAdmin. Перший перевіряє, чи користувач увійшов у систему, а другий — чи має він адміністративні права. Приклад використання цих перевірок у серверному маршруті наведено у лістингу 3.8.

Лістинг 3.8 – Приклад захисту адміністративного маршруту

```
productRouter.post(
  '/',
  isAuth,
  isAdmin,
  expressAsyncHandler(async (req, res) => {
    const newProduct = new Product({
      name: 'sample name',
      slug: 'sample-slug-' + Date.now(),
      price: 0,
      category: 'sample category',
      brand: 'sample brand',
      countInStock: 0,
      description: 'sample description',
    });

    const product = await newProduct.save();
    res.send({ message: 'Product Created', product });
  })
);
```

У лістингу 3.8 видно, що створення нового товару можливе лише після проходження двох перевірок. Це відокремлює звичайні користувацькі сценарії від службових операцій і зменшує ризик несанкціонованої зміни каталогу.

Панель керування DashboardScreen отримує з маршруту ‘/api/orders/summary’ агреговані дані про кількість користувачів, замовлень, суму продажів і категорійну структуру каталогу. На їх основі будуються аналітичні

блоки та графіки, які дозволяють оцінювати поточний стан асортименту й динаміку замовлень. Аналітика в цьому випадку не є окремим зовнішнім інструментом: вона формується на основі даних, які вже зберігаються в базі даних інтернет-магазину.

Фрагмент серверного маршруту, який готує дані для аналітичної панелі, наведено у лістингу 3.9.

Лістинг 3.9 – Фрагмент маршруту отримання аналітичних даних

```
orderRouter.get(
  '/summary',
  isAuth,
  isAdmin,
  expressAsyncHandler(async (req, res) => {
    const orders = await Order.aggregate([
      {
        $group: {
          _id: null,
          numOrders: { $sum: 1 },
          totalSales: { $sum: '$totalPrice' },
        },
      },
    ]);

    const users = await User.aggregate([
      {
        $group: {
          _id: null,
          numUsers: { $sum: 1 },
        },
      },
    ]);

    const dailyOrders = await Order.aggregate([
      {
        $group: {
          _id: { $dateToString: { format: '%Y-%m-%d', date:
'$createdAt' } },
          orders: { $sum: 1 },
          sales: { $sum: '$totalPrice' },
        },
      },
      { $sort: { _id: 1 } },
    ]);

    const productCategories = await Product.aggregate([
      {
        $group: {
```

```
            _id: '$category',
            count: { $sum: 1 },
          },
        },
      ]);

      res.send({ users, orders, dailyOrders, productCategories });
    })
);
```

Наведений фрагмент показує, що аналітична панель працює з узагальненими даними. Сервер окремо підраховує кількість замовлень, суму продажів, кількість користувачів, динаміку замовлень за датами та розподіл товарів за категоріями. Завдяки цьому адміністратор бачить не лише окремі записи, а й загальну картину роботи інтернет-магазину.

На рисунку 3.17 показано аналітичний рівень адміністративної частини: картки з ключовими показниками, графік продажів і кругову діаграму категорійної структури каталогу.

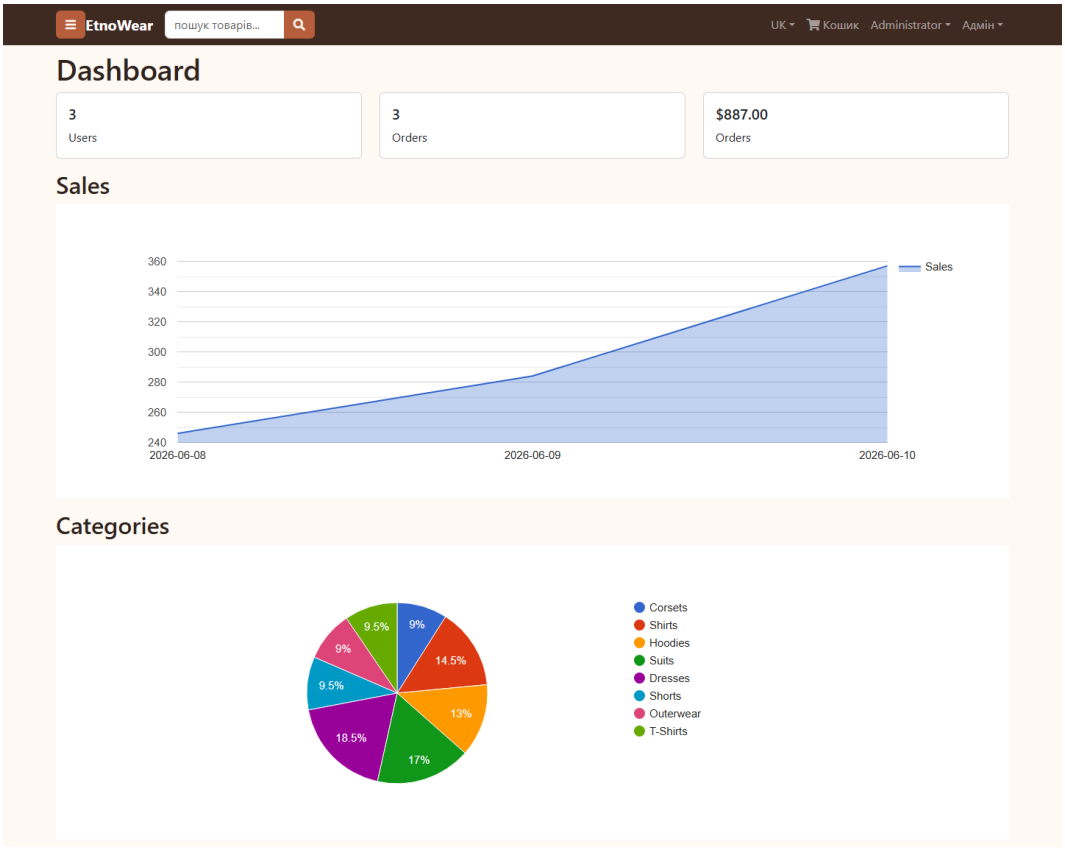

Рисунок 3.17 – Адміністративна панель з аналітичними блоками та графіками

Для адміністратора практично важливими є не тільки графіки, а й робочі списки. Сторінки ProductListScreen, OrderListScreen і UserListScreen реалізують операції перегляду, переходу до детального редагування та видалення сутностей. Для товарів додатково передбачено створення нової позиції з автоматичним переходом до форми редагування. Унаслідок цього адміністративний модуль охоплює як оглядову аналітику, так і повсякденне керування каталогом, замовленнями та обліковими записами. Адміністративні вікна керування товарами, замовленнями та користувачами винесено в додаток В.

Робочі вікна адміністратора відображають той набір операцій, який потрібен для щоденного супроводу інтернет-магазину без звернення до програмного коду. Через них можна переглядати створені замовлення, змінювати інформацію про товари, контролювати користувачів і за потреби надавати окремим обліковим записам адміністративні права.

Адміністративний модуль поєднує керування даними й базову аналітику. Він забезпечує доступ до службових операцій лише для користувачів із правами адміністратора, дає змогу підтримувати актуальний стан каталогу, переглядати замовлення й користувачів, а також отримувати узагальнені показники роботи інтернет-магазину.

Практична цінність аналітичного блоку полягає в тому, що він слугує основою для ухвалення управлінських рішень у роботі інтернет-магазину. Узагальнені відомості про кількість замовлень, суму продажів, активність користувачів і категорійну структуру асортименту дають змогу оцінювати поточний стан системи не інтуїтивно, а на підставі фактичних даних. Це дозволяє своєчасно виявляти найбільш затребувані категорії товарів, контролювати зміну інтенсивності продажів і приймати рішення щодо оновлення асортименту, акцентів на головній сторінці, коригування залишків або подальшого розширення окремих груп товарів.

Що свідчить про те що, аналітика в адміністративному модулі виконує не допоміжну, а прикладну функцію. Вона переводить керування інтернет-

магазином із рівня простого перегляду списків на рівень обґрунтованого контролю та планування подальших дій.

3.6 Аналіз користувацького інтерфейсу

З погляду зручності використання інтерфейс побудовано послідовно: головна сторінка виконує роль змістового входу, сторінка пошуку дає прозорий контроль над параметрами відбору, сторінка товару концентрує ключові відомості для прийняття рішення, а оформлення замовлення розбито на логічні кроки. Така структура не перевантажує користувача й підтримує природний рух від зацікавлення до покупки.

Важливою рисою інтерфейсу є узгодженість між концепцією інтернет-магазину та її практичною реалізацією. Комунікація про натуральні тканини, культурні маркери, екологічні позиції, інклюзивність і підтримку України не винесена за межі системи, а інтегрована в головну сторінку та тематичні переходи каталогу. Завдяки цьому інтернет-магазин сприймається не лише як вітрина товарів, а як простір із чіткою ціннісною позицією.

Під час аналізу інтерфейсу враховувалися кілька практичних критеріїв: зрозумілість навігації, доступність основних дій, послідовність сценарію покупки, коректність відображення даних українською та англійською мовами, а також відповідність екранів ролі користувача. Для звичайного покупця ключовими є пошук, перегляд товару, кошик і замовлення. Для адміністратора — доступ до керування товарами, замовленнями, користувачами та аналітикою.

Перевірка працездатності виконувалася у двох формах. Перша форма – функціональне відтворення основних сценаріїв звичайного користувача й адміністратора. Друга форма – автоматизоване тестування інтерфейсних сценаріїв за допомогою Selenium [39]. Такий підхід дав змогу не лише описати очікувану поведінку системи, а й зафіксувати фактичний результат виконання низки сценаріїв.

Окрім Selenium, для подальшого розширення тестового контуру доцільно застосовувати й інші інструменти. Для перевірки HTTP-маршрутів можна використовувати Postman або Newman, для ізольованого тестування клієнтських компонентів – React Testing Library, для перевірки серверної логіки – Supertest, а для навантажувального аналізу – k6 або JMeter. Однак саме Selenium є найбільш наочним для демонстрації повного сценарію взаємодії з інтерфейсом.

Для підтвердження зручності інтерфейсу перевірялися не лише окремі кнопки або сторінки, а повні сценарії роботи. Такий підхід дозволяє оцінити, чи може користувач без зайвих перешкод пройти шлях від головної сторінки до створення замовлення, а адміністратор — відкрити службові розділи й виконати базові дії з керування системою.

Основні сценарії, які перевірялися під час функціонального аналізу інтернет-магазину, наведено в таблиці 3.1. Вони охоплюють дії звичайного користувача та адміністратора: від перегляду головної сторінки й пошуку товарів до створення замовлення та відкриття службових розділів системи.

Таблиця 3.1 – Основні функціональні сценарії перевірки інтернет-магазину

Роль	**Сценарій**	**Очікуваний результат**	**Фактичний результат**
1	**2**	**3**	**4**
Користувач	Відкриття головної сторінки	Головний екран завантажується з hero-блоком і картками товарів	Підтверджено
Користувач	Перехід до каталогу з головної сторінки	Відкривається список товарів з фільтрами	Підтверджено
Користувач	Пошук за категорією, маркою і тегом	Формується релевантна вибірка товарів	Підтверджено
Користувач	Відкриття сторінки товару	Відображаються локалізовані атрибути товару та блок покупки	Підтверджено
Користувач	Додавання товару в кошик	Позиція з'являється у кошику	Підтверджено
Користувач	Зміна кількості в кошику	Підсумок кошика оновлюється	Підтверджено

Продовження таблиці 3.1

1	2	3	4
Користувач	Перемикання мови	Інтерфейс і предметні терміни змінюють мову	Підтверджено
Користувач	Перехід до перевірки замовлення	Відображаються адреса, оплата, товари й підсумок	Підтверджено
Користувач	Створення замовлення	Формується документ замовлення	Підтверджено
Адміністратор	Відкриття панелі керування	Показано аналітичні блоки та графіки	Підтверджено
Адміністратор	Відкриття списку товарів	Доступне створення й редагування позицій	Підтверджено
Адміністратор	Відкриття списку замовлень	Доступний перегляд замовлень і переходи до деталей	Підтверджено
Адміністратор	Відкриття списку користувачів	Відображаються облікові записи та права	Підтверджено

Перевірка охоплює не окремі екрани ізольовано (див. табл. 3.1), а повний шлях користувача в системі. Для покупця важливими є перегляд товарів, пошук, кошик, мовне перемикання та створення замовлення. Для адміністратора ключовими є доступ до аналітики, списків товарів, замовлень і користувачів. Це дає змогу оцінити, чи працюють основні частини системи як єдиний сценарій.

Для додаткової перевірки інтерфейсних сценаріїв було виконано автоматизований прогін за допомогою Selenium. Результати цього прогону наведено в таблиці 3.2.

Таблиця 3.2 – Результати автоматизованого виконання Selenium-сценаріїв

Сценарій	**Зміст перевірки**	**Тривалість**	**Статус**
1	**2**	**3**	**4**
home_page	Знайдено заголовок головної сторінки та 15 карток товарів.	6.49 с	Успішно
search_filters	Фільтрований пошук повернув 11 карток товарів.	5.89 с	Успішно

Продовження таблиці 3.2

1	2	3	4
bilingual_product	Англомовний режим активовано; кнопка покупки відображається як: Add to Cart.	6.45 с	Успішно
place_order	Сторінку перевірки замовлення відкрито з адресою доставки, товарами та способом оплати.	5.78 с	Успішно
admin_dashboard	Адміністративну панель відкрито; блоки користувачів, замовлень і продажів відображаються коректно.	6.43 с	Успішно
admin_lists	продуктів: 15; замовлень: 35; користувачів: 15	8.07 с	Успішно

Дані показують (див. табл. 3.2), що автоматизовані сценарії охопили як публічну, так і адміністративну частину інтернет-магазину. Перевірено завантаження головної сторінки, багатокритеріальний пошук, локалізацію сторінки товару, сторінку перевірки замовлення, адміністративну аналітику та робочі списки адміністратора.

За підсумками автоматизованого прогону виконано шість Selenium-сценаріїв, і всі вони завершилися успішно. Повний JSON-звіт, скрипт автоматизації та скріншоти сценаріїв подано в додатку Д.

Окремо варто оцінити інтерфейс із погляду практичного використання системи продавцем-адміністратором. Для такого користувача важливо не лише мати доступ до списків товарів, замовлень і користувачів, а й запускати систему без роботи з командним рядком. Тому для локального використання інтернет-магазину розроблено допоміжні скрипти run_store.bat та stop_store.bat, які запускають і зупиняють систему через подвійний клік.

Скрипт run_store.bat спрощує запуск клієнтської та серверної частин, а stop_store.bat дає змогу швидко завершити роботу локального середовища. Це особливо важливо для сценарію, у якому інтернет-магазин може бути адаптований для невеликого локального бренду або магазину одягу. У такому

випадку адміністратор не обов’язково має технічну підготовку, тому запуск, зупинка та базове керування системою повинні бути максимально зрозумілими.

Покрокову інструкцію для продавця-адміністратора представлено у додатку Е, У ній окремо описано запуск системи, вхід під обліковим записом адміністратора, додавання товару, редагування залишків, перегляд замовлень і завершення роботи. Інструкції з користування підсилюють практичну цінність роботи, оскільки показують, що система може використовуватися як основа для реального локального інтернет-магазину.

Аналіз користувацького інтерфейсу показує, що система підтримує різноманітні сценарії, такі як перегляд товарів і оформлення замовлення, й окремо інструменти для роботи адміністратора. Інтерфейс побудовано послідовно: покупець переходить від ознайомлення з брендом до покупки, а адміністратор отримує доступ до керування товарами, замовленнями, користувачами й аналітикою.

3.7 Висновок до третього розділу

У третьому розділі було описано практичну реалізацію інтернет-магазину одягу, зокрема клієнтську частину, серверну логіку, механізми пошуку й фільтрації, двомовність, оформлення замовлення, адміністративний модуль та базову аналітику. Реалізовані модулі забезпечують повний користувацький сценарій: від ознайомлення з головною сторінкою та перегляду каталогу до створення замовлення й перегляду його статусу. Для адміністратора передбачено керування товарами, користувачами, замовленнями та перегляд узагальнених показників роботи магазину.

Проведена перевірка працездатності показала, що основні сценарії покупця й адміністратора виконуються коректно. Selenium-сценарії підтвердили роботу головної сторінки, пошуку, двомовного режиму, перевірки замовлення, адміністративної панелі та службових списків. Це дає підстави розглядати

створену систему не лише як навчальний вебзастосунок, а і як основу для подальшої адаптації під реальний локальний магазин одягу.

Практична цінність реалізованої системи полягає в тому, що вона придатна не лише для демонстрації окремих програмних рішень, а й для використання як основа цифрового інструменту локального бренду одягу. Інтернет-магазин підтримує повний базовий цикл роботи з каталогом, пошуком, оформленням замовлень, обліковими записами користувачів і адміністративним супроводом. Для продавця-адміністратора це означає можливість працювати з товарами, замовленнями, користувачами та аналітичними даними через готовий інтерфейс без безпосереднього втручання в програмний код.

Не менш важливими є можливості подальшого масштабування системи. Реалізована архітектура допускає розширення асортименту, поглиблення аналітичного блоку, підключення додаткових платіжних і логістичних сервісів, розвиток системи ролей доступу, а також введення нових параметрів пошуку й фільтрації, зокрема для інклюзивного або адаптивного одягу.

Одним з напрямів подальшого розвитку є використання даних фільтрації для аналітики користувацьких уподобань. У межах реалізованої системи вже передбачено розширену систему атрибутів товару: категорію, бренд, тканину, колір, теги, наявність і ціновий діапазон. Надалі ці параметри можуть використовуватися не лише для навігації в каталозі, а й для дослідження того, які характеристики товарів найчастіше впливають на вибір користувача. Такі дані можна зіставляти з додаванням товарів у кошик, створенням замовлень і переглядом окремих категорій, що дасть змогу обґрунтованіше оновлювати асортимент, тематичні добірки та структуру фільтрів.

Окремо слід зазначити, що реалізована система враховує не лише технічні сценарії електронної торгівлі, а й змістову концепцію магазину. У клієнтській частині передбачено тематичні точки входу до каталогу, двомовне відображення предметних даних, стриману й довірчу візуальну подачу та комунікаційні акценти, пов’язані з українським контекстом, інклюзивністю і соціальною відповідальністю бренду.

РОЗДІЛ 4. БЕЗПЕКА ЖИТТЄДІЯЛЬНОСТІ ТА ОСНОВИ ОХОРОНИ ПРАЦІ

У розділі розглянуто два взаємопов'язані аспекти. Перший стосується психології безпеки праці як складової загальної проблематики психології з урахуванням сучасних умов війни, тривалого стресу та соціальної ізольованості. Другий аспект охоплює гігієнічні вимоги до організації та обладнання робочих місць з відеодисплейними терміналами, оскільки саме такі умови праці є типовими для фахівців, які виконують програмну реалізацію, інформаційну підтримку та технічний супровід інтернет-магазину.

4.1 Психологія безпеки праці в загальній проблемі психології

Психологія безпеки праці посідає важливе місце в системі наук про людину та її трудову діяльність. Вона досліджує психічні процеси, стани й властивості особистості, від яких залежить безпечна поведінка працівника у виробничому середовищі. У центрі уваги перебувають увага, пам'ять, мислення, емоційна стійкість, рівень самоконтролю, мотивація, здатність швидко орієнтуватися в ситуації та приймати адекватні рішення. Саме ці характеристики значною мірою визначають, чи буде людина дотримуватися вимог безпеки, своєчасно реагувати на зміни умов праці та уникати помилкових дій [40].

У загальній проблематиці психології безпека праці розглядається через взаємозв'язок свідомості, поведінки та зовнішніх умов діяльності. Порушення вимог безпеки не завжди є наслідком незнання правил. Досить часто їх причиною стають перевтома, монотонність роботи, зниження концентрації уваги, емоційне виснаження, дефіцит часу, звикання до ризику або надмірна самовпевненість. За таких умов навіть формально підготовлений працівник може припускатися дій, що підвищують імовірність помилки або нещасного випадку [40].

Особливого значення психологія безпеки праці набуває в інтелектуальних видах діяльності, де зовнішні ризики не завжди є очевидними, проте значною є

роль нервово-емоційного та когнітивного перевантаження. Під час роботи за персональним комп'ютером накопичуються втома, напруження зору, дратівливість, знижується точність сприйняття інформації та сповільнюється реакція. Унаслідок цього підвищується ймовірність технічних помилок, ігнорування регламентованих перерв, порушення режиму праці та відпочинку. Отже, психологічна безпека праці стосується не лише небезпечних виробничих ситуацій у традиційному розумінні, а й підтримання належного психофізіологічного стану працівника.

Психологічний аспект безпеки праці тісно пов'язаний з організацією трудового процесу. Чіткість завдань, передбачуваність режиму роботи, відсутність постійного інформаційного хаосу, раціональний розподіл навантаження та сприятливий морально-психологічний клімат знижують ризик небезпечної поведінки. Натомість хронічний поспіх, невизначеність, надлишок повідомлень, багатозадачність і тривале перебування в стані напруження погіршують здатність до самоконтролю та підвищують імовірність помилкових рішень [40].

У сучасному українському контексті психологія безпеки праці потребує розгляду з урахуванням впливу війни. Воєнні події формують тривале психоемоційне напруження, породжують фонову тривожність, порушення сну, інформаційне перевантаження, відчуття нестабільності та постійну готовність до реагування на зовнішні загрози. Такі чинники знижують концентрацію уваги, підвищують виснаження, погіршують здатність до тривалого зосередження та можуть безпосередньо впливати на якість професійної діяльності.

Не менш суттєвим чинником є ізольованість, яка в сучасних умовах може виникати внаслідок дистанційного формату роботи, вимушеного переміщення, скорочення звичних соціальних контактів або психологічного замикання людини в собі. Ізольованість послаблює емоційну підтримку, ускладнює своєчасне виявлення перевтоми, сприяє емоційному вигоранню та знижує відчуття включеності в колектив. Для працівника, діяльність якого пов'язана з

тривалою взаємодією з цифровими системами, це означає підвищення ризику неуважності, затримки у виконанні завдань і порушення самодисципліни.

Таким чином, психологія безпеки праці має розглядатися як складова загальної психології, що пояснює закономірності безпечної поведінки людини в трудовій діяльності. У сучасних умовах її зміст розширюється за рахунок чинників воєнного стресу, емоційної виснаженості та соціальної ізольованості, які потребують урахування під час організації праці, регламентування навантаження та підтримання працездатності [40].

4.2 Гігієнічні вимоги до організації та обладнання робочих місць з ВДТ

Робоче місце з відеодисплейним терміналом має забезпечувати безпечні та зручні умови праці протягом усього робочого часу. Його організація передбачає раціональне розміщення технічних засобів, меблів та допоміжного обладнання, створення сприятливого мікроклімату, належного освітлення і дотримання ергономічних параметрів. Для діяльності, пов'язаної з роботою за персональним комп'ютером, ці вимоги мають принципове значення, оскільки впливають на стан зору, опорно-рухового апарату, нервової системи та загальну працездатність працівника [41].

Площа й об'єм приміщення повинні бути достатніми для безпечного розміщення обладнання та вільного виконання трудових операцій. На одне робоче місце з ВДТ доцільно передбачати не менше 6,0 м2 площі та не менше 20,0 м3 об'єму приміщення. Такі параметри сприяють підтриманню прийнятного повітрообміну, зменшують відчуття скупченості та створюють умови для раціонального розташування меблів і техніки.

Під час розміщення кількох робочих місць необхідно дотримуватися безпечних відстаней між моніторами та столами. Відстань між тильною поверхнею одного монітора та екраном іншого рекомендується встановлювати не менше 2,0 м, а між бічними поверхнями моніторів - не менше 1,2 м. Якщо робота потребує значної концентрації уваги, окремі місця доцільно

відокремлювати перегородками висотою 1,5-2,0 м, що зменшує відволікання та поліпшує психогігієнічні умови праці.

Освітлення робочого місця має виключати появу відблисків на екрані та надмірне контрастне навантаження на зір. Для цього монітор слід розміщувати з урахуванням природного й штучного освітлення, використовувати засоби регулювання світлового потоку та уникати прямого потрапляння сонячних променів на екран. Доцільним є застосування світлих спокійних тонів в оздобленні приміщення, оскільки вони сприяють зменшенню зорової втоми та формують комфортне робоче середовище [41].

Важливою умовою є правильне формування робочої пози. Сидіння повинно забезпечувати стійке положення тіла, ступні мають спиратися на підлогу або підставку для ніг, стегна розташовуватися близько до горизонтального положення, а кут у ліктьових суглобах становити приблизно 70-90°. Нахил голови слід обмежувати, щоб не створювати надмірного статичного навантаження на м'язи шиї та плечового пояса. Дотримання таких вимог дає змогу зменшити втому та профілактувати професійно зумовлені розлади опорно-рухового апарату.

Робочий стіл має забезпечувати достатній простір для ніг і зручне розміщення технічних засобів. Робоче крісло доцільно використовувати підйомно-поворотного типу з регулюванням висоти сидіння, кута нахилу спинки та можливістю надійної фіксації обраного положення. Поверхня сидіння і спинки повинна бути напівм'якою, неслизькою, повітропроникною та такою, що легко очищується. Підставка для ніг має відповідати антропометричним особливостям користувача, а її опорна поверхня повинна бути рифленою й мати бортик по передньому краю.

Клавіатуру рекомендується розміщувати на відстані приблизно 100-300 мм від краю столу, зверненого до працівника, або на спеціальній висувній поверхні. Таке розташування дає змогу зменшити напруження кистей і передпліч. Маніпулятор типу миша слід розміщувати в межах зручної досяжності, не допускаючи постійного перенапруження м'язів руки. Монітор повинен бути

встановлений так, щоб верхня межа екрана не перевищувала рівень очей, а відстань від очей до екрана забезпечувала комфортне сприйняття інформації.

Гігієнічні вимоги поширюються також на технічний стан обладнання та санітарне утримання робочого місця. Необхідно регулярно очищувати поверхні обладнання від пилу, стежити за справністю електропроводки, розеток і з'єднувальних шнурів, не закривати вентиляційні отвори техніки та не розміщувати сторонні предмети на системному блоці чи моніторі. Самостійне втручання у внутрішні вузли обладнання без відповідної підготовки не допускається. Дотримання цих вимог знижує ризик перегрівання апаратури, ураження електричним струмом та передчасного виходу техніки з ладу [41].

Отже, гігієнічні вимоги до організації та обладнання робочих місць з ВДТ охоплюють просторові характеристики приміщення, освітлення, ергономіку меблів, правильне розміщення технічних засобів, санітарне утримання обладнання та дотримання правил експлуатації. Для працівників, чия діяльність пов'язана з програмною реалізацією та супроводом інтернет-магазину, виконання цих вимог є необхідною умовою підтримання працездатності, профілактики перевтоми та збереження здоров'я [41].

4.3 Висновок до четвертого розділу

У розділі розглянуто психологічні та гігієнічні аспекти безпеки життєдіяльності й охорони праці, актуальні для діяльності, пов'язаної з тривалою роботою за персональним комп'ютером. Установлено, що психологія безпеки праці охоплює увагу, самоконтроль, емоційну стійкість і здатність працівника діяти безпечно в умовах навантаження. У сучасному українському контексті на цей аспект додатково впливають воєнний стрес, тривожність та соціальна ізольованість. Також визначено, що належна організація робочого місця з ВДТ передбачає дотримання вимог до площі приміщення, розміщення обладнання, освітлення, робочої пози, санітарного стану та технічної справності апаратури.

ВИСНОВКИ

У кваліфікаційній роботі розв'язано завдання розробки інтернет-магазину з функціональними можливостями аналітики, авторизації та оптимізації інтерфейсу. Створена система поєднує двомовний каталог товарів, гнучкий пошук, багатокритеріальну фільтрацію, оформлення замовлень, адміністративну панель та засоби аналітичного супроводу.

У першому розділі проаналізовано сучасний стан інтернет-магазинів одягу, досліджено функціональні можливості аналогів, розглянуто значення авторизації, аналітики та інтерфейсної взаємодії, а також сформовано вимоги до розроблюваної системи. Це створило підставу для подальшого обґрунтування проєктних рішень.

У другому розділі обґрунтовано вибір технологічного стеку, спроєктовано клієнт-серверну архітектуру, визначено структуру бази даних і функціональну організацію системи. Подано опис основних механізмів обробки даних, зокрема пошуку, фільтрації, серверної перевірки замовлення та контролю доступу.

У третьому розділі реалізовано клієнтську частину інтернет-магазину, серверну логіку, взаємодію з базою даних, оформлення замовлення, адміністративний модуль та аналітичні елементи. Проведено перевірку працездатності системи на рівні основних різних сценаріїв.

У четвертому розділі розглянуто питання безпеки життєдіяльності та охорони праці, пов'язані з роботою за персональним комп'ютером, зокрема психологічні чинники безпечної праці та гігієнічні вимоги до організації робочого місця з відеодисплейним терміналом.

Практичне значення одержаних результатів полягає у створенні програмного рішення, яке може бути використане як основа для реального інтернет-магазину локального бренду одягу. Подальший розвиток системи може бути пов'язаний із розширенням аналітики, інтеграцією зовнішніх сервісів, поглибленням тематичної каталогізації та впровадженням персоналізованих механізмів взаємодії з користувачем.

ПЕРЕЛІК ДЖЕРЕЛ

1. Липак, Т. А., & Липак, Г. І. (2025). Вплив новітніх технологій на методологію дослідження досвіду користувача. У Теорія модернізації в контексті сучасної світової науки: збірник наукових праць з матеріалами IV Міжнародної наукової конференції (с. 318). Вінниця: ТОВ «УКРЛОГОС Груп». https://doi.org/10.62731/mcnd-24.01.2025

2. Липак, Т. А., & Липак, Г. І. (2025). Оцінка користувацького досвіду (UX Evaluation) при розробці інтерфейсів. Scientific Collection InterConf, 235, 236–239.

3. Cuesta-Valiño, P., Gutiérrez-Rodríguez, P., Loranca-Valle, C., & Catalá-Pérez, D. (2026). Influence of E-Commerce Usability, Consumer Happiness, and Satisfaction on Purchase Intentions in Fashion Retail. Journal of Consumer Behaviour. https://doi.org/10.1002/cb.70134

4. Phamthi, V. A., Nagy, Á., & Ngo, T. M. (2024). The Influence of Perceived Risk on Purchase Intention in E-Commerce: A Systematic Review and Research Agenda. International Journal of Consumer Studies. https://doi.org/10.1111/ijcs.13067

5. McKee, S., Sands, S., Pallant, J. I., & Cohen, J. (2023). The Evolving Direct-to-Consumer Retail Model: A Review and Research Agenda. International Journal of Consumer Studies, 47(6), 2816–2842. https://doi.org/10.1111/ijcs.12972

6. Липак, Г., Кунанець, Н., Дуда, О., & Липак, Т. (2025). Побудова інтерфейсів користувача вебсайту бібліотеки на засадах UX-дизайну. Цифрова платформа: інформаційні технології в соціокультурній сфері, 8(1), 172–192. https://doi.org/10.31866/2617-796X.8.1.2025.335539

7. Kantar Ukraine. (2023, June 2). Українські споживачі очікують емпатії від брендів. Kantar. https://www.kantar.com/ua/inspiration/brands/the-importance-of-a-brand-being-empathetic

8. Лісовий, М., & Липак, Г. (2025). Проєктування еластичних хмарних архітектур для цифрових сервісів громадської взаємодії. У Актуальні задачі

сучасних технологій: збірник тез доповідей XIV міжнародної науково-технічної конференції молодих учених та студентів (с. 292). Тернопіль.

9. Липак, Г., & Ющенко, О. (2025). Інтеграція інтелектуальних рішень в екосистему WordPress для e-commerce платформ. У Матеріали XIII науково-технічної конференції «Інформаційні моделі, системи та технології» (с. 159). Тернопіль.

10. Zhou, Z., Zheng, F., Lin, J., & Zhou, N. (2021). The interplay among green brand knowledge, expected eudaimonic well-being and environmental consciousness on green brand purchase intention. Corporate Social Responsibility and Environmental Management, 28(2), 630–639. https://doi.org/10.1002/csr.2075

11. Кліщ, М., Липак, Г., Кунанець, Н., Пасічник, С., & Липак, Т. (2025). Структура інформаційної системи передбачення та інтерпретації зміни стану користувача сервісу. Вісник Національного університету «Львівська політехніка». Інформаційні системи та мережі, 17, 226–238. https://doi.org/10.23939/sisn2025.17.226

12. Kasta. (n.d.). Kasta SuperApp — найбільший вибір товарів онлайн. https://kasta.ua/

13. ANSWEAR.ua. (n.d.). ANSWEAR.ua – інтернет-магазин жіночого, чоловічого та дитячого одягу, взуття та аксесуарів. https://answear.ua/

14. MustHave. (n.d.). MustHave online store. https://musthave.ua/

15. ZARA. (n.d.). ZARA Ukraine | New collection online. https://www.zara.com/ua/

16. OVERTHESEA. (n.d.). OVERTHESEA. https://overtheseadress.com/

17. Dodo Socks. (n.d.). Main. https://dodosocks.com/en/

18. SVARGA. (n.d.). Українські вишиванки в інтернет-магазині SVARGA. https://svarga.ua/

19. Готович, В. А., Дуда, О. М., & Никитюк, В. В. (укладачі). (2024). Методичні вказівки до виконання кваліфікаційної роботи ОР бакалавр для студентів спеціальності 122 – Комп'ютерні науки всіх форм навчання. Тернопіль: Тернопільський національний технічний університет імені Івана Пулюя.

20. React. (n.d.). Quick Start. https://react.dev/learn

21. React Router. (n.d.). BrowserRouter. https://reactrouter.com/api/declarative-routers/BrowserRouter

22. React Router. (n.d.). Address Book. https://reactrouter.com/tutorials/address-book

23. Express.js. (n.d.). Routing. https://expressjs.com/en/guide/routing.html

24. Django Software Foundation. (n.d.). Django documentation. https://docs.djangoproject.com/

25. Laravel. (n.d.). Laravel documentation. https://laravel.com/docs

26. Spring. (n.d.). Spring Boot documentation overview. https://docs.spring.io/spring-boot/documentation.html

27. MongoDB. (n.d.). What is MongoDB? https://www.mongodb.com/docs/manual/

28. Mongoose. (n.d.). Schemas. https://mongoosejs.com/docs/guide.html

29. MDN Web Docs. (n.d.). Overview of HTTP. https://developer.mozilla.org/en-US/docs/Web/HTTP/Guides/Overview

30. Cloudinary. (n.d.). Upload API reference. https://cloudinary.com/documentation/image_upload_api_reference

31. MDN Web Docs. (n.d.). URLSearchParams. https://developer.mozilla.org/en-US/docs/Web/API/URLSearchParams

32. PayPal Developer. (n.d.). JavaScript SDK. https://developer.paypal.com/sdk/js/

33. PayPal Developer. (n.d.). PayPal sandbox testing guide. https://developer.paypal.com/tools/sandbox/

34. Jones, M., Bradley, J., & Sakimura, N. (2015). JSON Web Token (JWT) (RFC 7519). RFC Editor. https://www.rfc-editor.org/info/rfc7519/

35. OWASP Foundation. (n.d.). Password storage cheat sheet. OWASP Cheat Sheet Series. https://cheatsheetseries.owasp.org/cheatsheets/Password_Storage_Cheat_Sheet.html

36. Norman, D. A. (2004). Emotional design: Why we love (or hate) everyday things. Basic Books.

37. Sharp, H., Rogers, Y., & Preece, J. (2019). Interaction design: Beyond human-computer interaction (5th ed.). Wiley.

38. MDN Web Docs. (n.d.). Window: localStorage property. https://developer.mozilla.org/en-US/docs/Web/API/Window/localStorage

39. Selenium. (n.d.). WebDriver documentation. https://www.selenium.dev/documentation/webdriver/

40. Зеркалов, Д. В. (2011). Безпека життєдіяльності. Основа.

41. Гандзюк, М. П., Желібо, Є. П., & Халімовський, М. О. (2007). Основи охорони праці. Каравела.

ДОДАТКИ

Додаток А

Розширена база даних

Таблиця А.1 – Колекція Product

Поле	Тип	Призначення
name	String	Назва товару англійською мовою.
nameUk	String	Назва товару українською мовою.
slug	String	Унікальний символьний ідентифікатор товару в URL.
image	String	Шлях або URL зображення товару.
brand	String	Бренд виробу.
category	String	Категорія товару.
color	String	Колір товару.
tags	String	Тематичні теги для фільтрації та пошуку.
description	String	Опис товару англійською мовою.
descriptionUk	String	Опис товару українською мовою.
price	Number	Ціна товару.
countInStock	Number	Кількість одиниць на складі.
fabric	String	Тип тканини.
createdAt / updatedAt	Date	Службові часові позначки Mongoose.

Таблиця А.2 – Колекція Order

Поле	Тип	Призначення
orderItems	Array	Склад замовлення з копіями даних товарів.
shippingAddress	Object	Адреса доставки: ПІБ, адреса, місто, індекс, країна.
paymentMethod	String	Обраний спосіб оплати.
paymentResult	Object	Службові дані підтвердження оплати.
itemsPrice	Number	Сума товарів без доставки.
shippingPrice	Number	Вартість доставки.
discountPrice	Number	Сума знижки.
totalPrice	Number	Підсумкова вартість замовлення.
user	ObjectId	Посилання на користувача-замовника.
isPaid / paidAt	Boolean / Date	Стан та дата оплати.
isDelivered / deliveredAt	Boolean / Date	Стан та дата доставки.
createdAt / updatedAt	Date	Службові часові позначки.

Таблиця А.3 – Колекція User

Поле	Тип	Призначення
name	String	Ім'я користувача.

email	String	Унікальна електронна адреса.
password	String	Хешований пароль.
isAdmin	Boolean	Ознака адміністративних прав.
createdAt / updatedAt	Date	Службові часові позначки.

Таблиця А.4 – Колекція Translation

Поле	**Тип**	**Призначення**
source	String	Вихідний термін або фраза.
target	String	Перекладений термін або фраза.
from	String	Мова джерела.
to	String	Мова перекладу.
priority	Number	Пріоритет використання перекладу.
domain	String	Предметна область перекладу.
kind	String	Тип перекладної одиниці.
createdAt / updatedAt	Date	Службові часові позначки.

Додаток Б

Лістинг коду server.js

```
import express from 'express';
import path from 'path';
import mongoose from 'mongoose';
import dotenv from 'dotenv';
import exchangeRouter from './routes/exchangeRoutes.js';
import seedRouter from './routes/seedRoutes.js';
import productRouter from './routes/productRoutes.js';
import translationRouter from './routes/translationRoutes.js';
import userRouter from './routes/userRoutes.js';
import orderRouter from './routes/orderRoutes.js';
import uploadRouter from './routes/uploadRoutes.js';
dotenv.config();
mongoose
  .connect(process.env.MONGODB_URI)
  .then(() => {
    console.log('connected to db');
  })
  .catch((err) => {
    console.log(err.message);
  });
const app = express();
pp.use(express.json());
app.use(express.urlencoded({ extended: true }));
app.get('/api/keys/paypal', (req, res) => {
  res.send(process.env.PAYPAL_CLIENT_ID || 'sb');
});
app.use('/api/exchange', exchangeRouter);
app.use('/api/upload', uploadRouter);
app.use('/api/seed', seedRouter);
app.use('/api/products', productRouter);
app.use('/api/translations', translationRouter);
app.use('/api/users', userRouter);
```

```
app.use('/api/orders', orderRouter);
const __dirname = path.resolve();
app.use(express.static(path.join(__dirname, '/frontend/build')));
app.get('*', (req, res) =>
  res.sendFile(path.join(__dirname, '/frontend/build/index.html'))
);
app.use((err, req, res, next) => {
  res.status(500).send({ message: err.message });
});
const port = process.env.PORT || 5000;
app.listen(port, () => {
  console.log(`serve at http://localhost:${port}`);
});
```

Додаток В

Знімки екрана адміністраторських сторінок

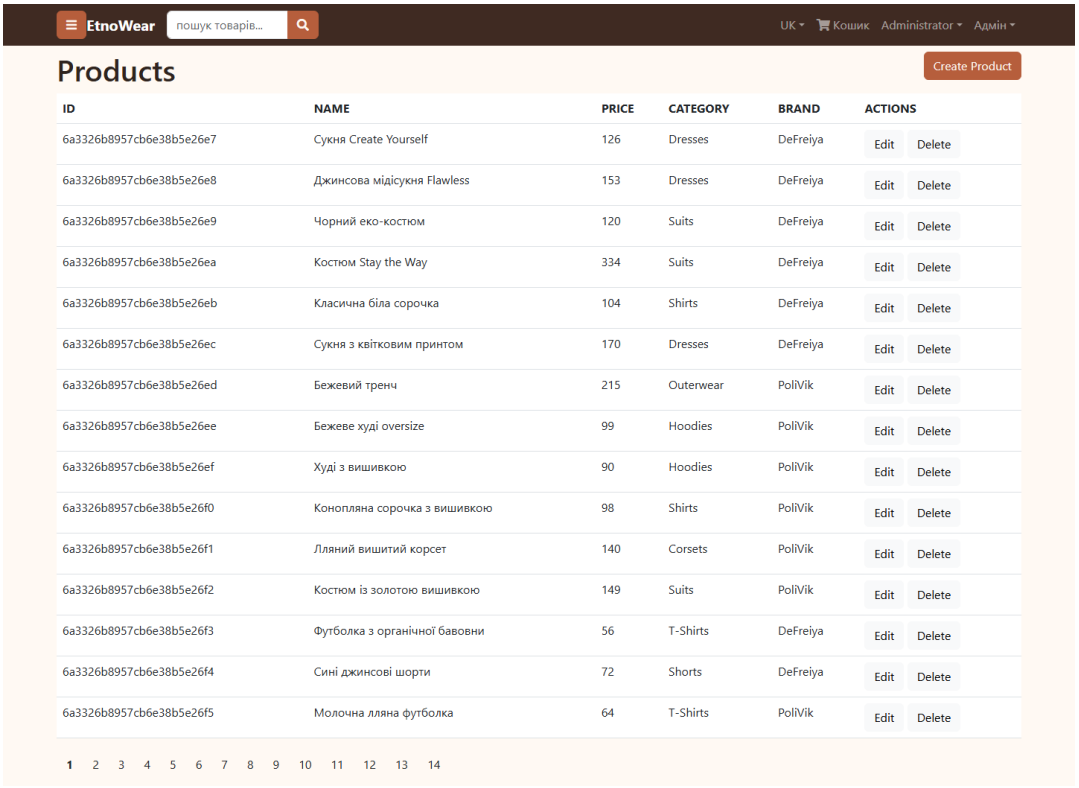

Рисунок В.1 – Сторінка ProductListScreen

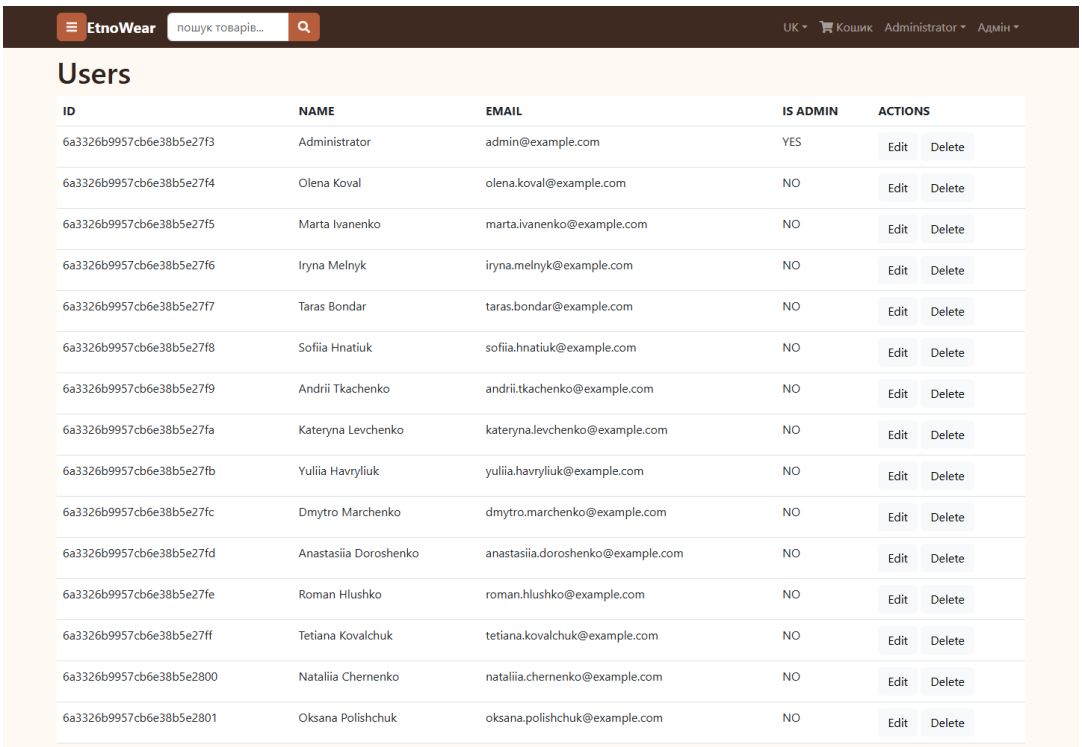

Рисунок В.2 – Сторінка UserListScreen

Рисунок В.3 – Сторінка OrderListScreen

Додаток Д

JSON-звіт автоматизованого прогону Selenium


```
{
  "executed_at": "2026-06-18 02:05:22",
  "results": [
    {
      "name": "home_page",
      "status": "passed",
      "duration_sec": 6.49,
      "details": "Знайдено заголовок головної сторінки та 15
карток товарів.",
      "screenshot":
"text\\generated_assets\\selenium\\home_page.png"
    },
    {
      "name": "search_filters",
      "status": "passed",
      "duration_sec": 5.89,
      "details": "Фільтрований пошук повернув 11 карток товарів.",
      "screenshot":
"text\\generated_assets\\selenium\\search_filters.png"
    },
    {
      "name": "bilingual_product",
      "status": "passed",
      "duration_sec": 6.45,
      "details": "Англомовний режим активовано; кнопка покупки
відображається як: Add to Cart.",
      "screenshot":
"text\\generated_assets\\selenium\\bilingual_product.png"
    },
    {
      "name": "place_order",
      "status": "passed",
      "duration_sec": 5.78,
      "details": "Сторінку перевірки замовлення відкрито з адресою
доставки, товарами та способом оплати.",
      "screenshot":
"text\\generated_assets\\selenium\\place_order.png"
    },
    {
      "name": "admin_dashboard",
      "status": "passed",
      "duration_sec": 6.43,
      "details": "Адміністративну панель відкрито; блоки
користувачів, замовлень і продажів відображаються коректно.",
      "screenshot":
"text\\generated_assets\\selenium\\admin_dashboard.png"
    },
    {
```

```
      "name": "admin_lists",
      "status": "passed",
      "duration_sec": 8.07,
      "details": "/admin/products: 15 ; /admin/orders: 35 ;
/admin/users: 15",
      "screenshot":
"text\\generated_assets\\selenium\\admin_lists.png"
    }
  ]
}
```

Додаток Е

Інструкція з використання інтернет-магазину для продавця-адміністратора

Подана інструкція описує спрощений порядок роботи з інтернет-магазином для продавця-адміністратора. Сценарій розраховано на користувача, який не працює з командним рядком і не запускає окремо фронтенд та бекенд вручну.

Е.1 Підготовка до запуску

Для локального використання достатньо двічі клацнути файл ‘run_store.bat’ у кореневій папці проєкту. Скрипт автоматично перевіряє наявність зібраної клієнтської частини, запускає сервер на http://localhost:5000 та відкриває сторінку входу в браузері. У такому режимі користувачеві не потрібно вводити npm-команди.

Якщо система вже запущена, файл ‘run_store.bat’ просто відкриє браузер на сторінці входу. Для завершення локальної роботи можна використати файл ‘stop_store.bat’, який зупиняє серверний процес.

Е.2 Вхід до адміністративної частини

Після запуску слід перейти на сторінку входу та ввести логін і пароль адміністратора. Після успішної авторизації у верхній навігації стає доступним випадаюче меню адміністратора з переходами до панелі керування, списку товарів, замовлень і користувачів.

Е.3 Робота з товарами

Для перегляду каталогу з боку адміністратора слід відкрити пункт `Products`. У списку відображаються ідентифікатор, назва, ціна, категорія та марка товару. За допомогою кнопки `Create Product` можна створити нову позицію. Після цього відкривається форма редагування, у якій заповнюються назва англійською й українською мовами, символьний ідентифікатор, ціна, категорія, колір, теги, марка, тканина, кількість на складі, зображення та описи.

Для оновлення наявного товару слід натиснути кнопку `Edit`, змінити потрібні поля та підтвердити збереження. Для видалення використовується кнопка `Delete`. Такі дії достатні для повсякденного супроводу асортименту без звернення до програмного коду або бази даних.

Е.4 Робота із замовленнями

Пункт ‘Orders’ відкриває список усіх замовлень із зазначенням користувача, дати, суми, статусу оплати та статусу доставки. Кнопка ‘Details’ переводить на сторінку конкретного замовлення, де можна переглянути адресу доставки, склад товарів та фінансовий підсумок. Якщо замовлення ще не доставлене, адміністратор може позначити його як доставлене.

Е.5 Робота з користувачами

У розділі ‘Users’ відображаються всі облікові записи, їх електронні адреси та ознака адміністративних прав. Звідси можна перейти до редагування користувача або видалити зайвий запис. Цей розділ доцільно використовувати для базового супроводу доступу до системи.

Е.6 Аналітична панель

Панель керування ‘Dashboard’ дає змогу швидко оцінити кількість користувачів, замовлень, суму продажів і категорійну структуру каталогу. Цей розділ зручний для щоденного перегляду загального стану інтернет-магазину та контролю поточної активності.

Е.7 Практична схема використання

У типовому сценарії продавець-адміністратор виконує лише кілька дій: запускає ‘run_store.bat’, входить у систему, перевіряє нові замовлення, за потреби редагує товари та наприкінці роботи закриває браузер або використовує ‘stop_store.bat’. Таким чином, локальна робота з інтернет-магазином наближається до звичного використання звичайної програми.