

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»

Факультет комп'ютерних наук і технологій

(повне найменування факультету)

Кафедра програмних засобів

(повне найменування кафедри)

Пояснювальна записка

до дипломного проекту (роботи)

бакалавр

(ступінь вищої освіти)

на тему ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ ОНЛАЙН МАГАЗИНУ
СУЧАСНОГО ОДЯГУ
SOFTWARE OF A MODERN CLOTHING ONLINE STORE

Виконав(ла): студент(ка) 4 курсу, групи КНТ-151

Спеціальності 121 Інженерія програмного

(код і найменування спеціальності)

забезпечення

Освітня програма (спеціалізація)

Інженерія програмного забезпечення

ОСТАПЕНКО М.В.

(ПРИЗВИЩЕ та ініціали)

Керівник ОЛІЙНИК А.О.

(ПРИЗВИЩЕ та ініціали)

Рецензент СКРУПСЬКИЙ С.Ю.

(ПРИЗВИЩЕ та ініціали)

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»
 (повне найменування закладу вищої освіти)

Факультет КНТ
 Кафедра програмних засобів
 Ступінь вищої освіти бакалавр
 Спеціальність 121 Інженерія програмного забезпечення
(код і найменування)
 Освітня програма (спеціалізація) Інженерія програмного забезпечення
(назва освітньої програми (спеціалізації))

ЗАТВЕРДЖУЮ

Завідувач кафедри ПЗ, д.т.н, проф.
Сергій СУББОТІН
 “ ” 2025 року

З А В Д А Н Н Я

НА ДИПЛОМНИЙ ПРОЄКТ (РОБОТУ) СТУДЕНТА(КИ)

ОСТАПЕНКА Максима Вадимовича
(ПРІЗВИЩЕ, ім'я, по батькові)

1. Тема проєкту (роботи) Програмне забезпечення онлайн магазину сучасного одягу. Software of a Modern Clothing Online Store

керівник проєкту (роботи) д.т.н., професор, ОЛІЙНИК Андрій Олександрович,
(науковий ступінь, вчене звання, ПРІЗВИЩЕ, ім'я, по батькові)

затверджені наказом закладу вищої освіти від “28” квітня 2025 року № 209

2. Строк подання студентом проєкту (роботи) 02 червня 2025 року

3. Вихідні дані до проєкту (роботи) рекомендована література, технічне завдання

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити) 1. Аналіз предметної області. 2. Вибір та обґрунтування структури системи, яка проєктується. 3. Основні рішення щодо реалізації компонентів системи. 4. Експлуатація програми. 5. Тестування та результати.

5. Перелік графічного матеріалу (з з точним зазначенням обов'язкових креслень, кількість слайдів, плакатів)

Слайди презентації

6. Консультанти розділів проєкту (роботи)

Розділ	ПРИЗВИЩЕ, ініціали та посада консультанта	Підпис, дата	
		завдання видав	прийняв виконане завдання
1-5 Основна частина	ОЛІЙНИК А.О., професор		
Нормоконтроль	КАЛІНІНА М.В., асистент		

7. Дата видачі завдання « 14 » квітня 2025 року.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломного проєкту (роботи)	Строк виконання етапів проєкту (роботи)	Примітка
1	Постановка завдання роботи.	1 тиждень	Завдання, ТЗ
2	Аналіз предметної області.	1 тиждень	Розділ 1
3	Вибір мови програмування та інших технологій	2 тиждень	Розділ 2
	розробки.		
4	Розробка архітектури програми.	2 тиждень	Розділ 3
5	Розробка програми.	3-4 тижні	Розділ 3, 4
6	Тестування та експериментальне дослідження	5 тиждень	Розділ 4
	програмного забезпечення.		
7	Оформлення пояснювальної записки та документів	6 тиждень	Додатки
	до неї.		
8	Нормоконтроль та рецензування.	7 тиждень	
9	Захист роботи.	8 тиждень	

Студент(ка)

(підпис) Максим ОСТАПЕНКО
(Ім'я ПРИЗВИЩЕ)

Керівник проєкту (роботи)

(підпис) Андрій ОЛІЙНИК
(Ім'я ПРИЗВИЩЕ)

РЕФЕРАТ

Пояснювальна записка до дипломної кваліфікаційної роботи бакалавра: 94 с., 16 табл., 66 рис., 3 дод., 17 джерел.

ПЛАТФОРМА ДЛЯ ОНЛАЙН-ТОРГІВЛІ, ІНТЕРНЕТ-МАГАЗИН, ОДЯГ, ВЕБЗАСТОСУНОК, TYPESCRIPT, JAVASCRIPT, NODE.JS, REACT.JS, NEXT.JS, NEST.JS, POSTGRESQL.

Об'єкт дослідження – процес розробки програмного забезпечення інтернет-магазину одягу.

Предмет дослідження – програмні засоби для створення інтернет-магазину з максимальною конверсією.

Мета роботи – створення вебзастосунку інтернет-магазину для продажу одягу

Методи дослідження. Вебтехнології та сучасні фреймворки (React.js, Next.js) для SPA-застосунків, серверний рендеринг (SSR), серверні технології (Node.js, Nest.js), бази даних (PostgreSQL), хмарні сервіси для розгортання (AWS, Vercel).

Результати. Створено вебзастосунок із використанням фреймворків Next.js для фронтенд-частини та Nest.js для бекенд-частини. Проведено тестування ефективності роботи інтернет-магазину в умовах високого навантаження.

Висновки. Реалізовано оптимізований та ефективний вебзастосунок інтернет-магазину із використанням сучасних технологій, що дозволило зменшити час завантаження сторінок, підвищивши тим самим конверсію.

ABSTRACT

Explanatory note to the diploma qualifying work of the bachelor: 94 pages, 16 tables, 66 figures, 3 appendixes, 17 sources.

PLATFORM FOR ONLINE TRADING, E-COMMERCE STORE, CLOTHING, WEB APPLICATION, TYPESCRIPT, JAVASCRIPT, NODE.JS, REACT.JS, NEXT.JS, NEST.JS, POSTGRESQL.

Object of research the architecture and tools of e-commerce stores.

Subject of research software tools for creating an online store with maximum conversion.

The goal of the thesis is to develop a web application for an online clothing store.

Research methods. Web technologies and modern frameworks (React.js, Next.js) for SPA applications, server-side rendering (SSR), server technologies (Node.js, Nest.js), databases (PostgreSQL), and cloud services for deployment (AWS, Vercel).

Results. A web application was developed using the Next.js framework for the frontend and the Nest.js framework for the backend. Performance testing was conducted to evaluate the efficiency of the online store under high load conditions.

Conclusions. An optimized and efficient e-commerce web application was implemented using modern technologies, which reduced page load time and thereby increased conversion rates.

ЗМІСТ

	С.
Перелік скорочень та умовних познач.....	8
Вступ.....	9
1 Аналіз предметної області.....	10
1.1 Актуальність предметної області	10
1.2 Огляд існуючих методів вирішення завдання.....	11
1.3 Огляд існуючих програмних засобів, що вирішують аналогічні завдання	12
1.4 Постановка завдання роботи.....	16
1.5 Висновки за розділом 1	16
2 Вибір та обґрунтування структури системи, яка проєктується.....	18
2.1 Вибір інструментарію розробки програмного забезпечення	18
2.2 Структурна схема програми.....	21
2.3 Опис модулів розроблювальної системи	22
2.4 Висновки за розділом 2	23
3 Основні рішення щодо реалізації компонентів системи.....	24
3.1 Алгоритм функціонування розроблювальної системи	24
3.2 Вибір шаблону проєктування	41
3.3 Опис модулів програми	42
3.4 Проєктування бази даних	43
3.5 Висновки за розділом 3	48
4 Експлуатація програми	49
4.1 Призначення та умови застосування програми	49
4.2 Характеристики програми.....	50
4.3 Звертання до програми	51
4.4 Вхідні й вихідні дані	51
4.5 Повідомлення	52
4.6 Повідомлення	53
5 Тестування та результати	62

5.1 Опис методики тестування.....	62
5.2 Результати використання розробленого вебзастосунку	62
Висновки	65
Перелік джерел посилання	66
Додаток А Технічне завдання	67
Додаток Б Текст програми	75
Додаток В Слайди презентації.....	85

ПЕРЕЛІК СКОРОЧЕНЬ ТА УМОВНИХ ПОЗНАК

API	– Application Programming Interface;
APP	– Application;
IDE	– Integrated Development Environment;
SaaS	– Software as a Service;
SEO	– Search Engine Optimization;
SPA	– Single Page Application;
SSG	– Static Site Generation;
SSR	– Server-Side Rendering;
BC	– вбудована система;
ПЗ	– програмне забезпечення.

ВСТУП

Розквіт електронної торгівлі та збільшення попиту на онлайн-шопінг обумовлюють потребу у швидких, зручних та продуктивних вебзастосунків. Сучасні технології дозволяють розробляти масштабовані рішення, що гарантують високу швидкодію, стабільність та комфорт для кінцевих користувачів.

Проблема розробки оптимізованого онлайн магазину полягає у виборі правильних архітектурних рішень, використанні сучасних технологій та забезпеченні високої конверсії. Зважаючи на зростаючі вимоги до швидкості завантаження сторінок, безпеки даних та масштабованості, ключовим викликом стає впровадження ефективної архітектури для обробки великої кількості користувачів та транзакцій.

Метою даної роботи є розробка сучасного вебзастосунку для онлайн магазину одягу, який використовує новітні технології, зокрема React.js, Next.js, Node.js, Nest.js та PostgreSQL. Це дозволяє реалізувати оптимізований механізм рендерингу (SSR), покращити продуктивність та створити зручний інтерфейс для користувачів.

Результатом роботи є повноцінний вебзастосунок, який дозволяє здійснювати покупки онлайн, керувати асортиментом товарів, обробляти замовлення та забезпечувати безперебійну роботу сервісу навіть під великим навантаженням. Проведені тести підтверджують, що використані технології сприяють зменшенню часу завантаження сторінок, підвищенню швидкості роботи онлайн магазину та покращенню загального користувацького досвіду.

Завдяки впровадженню сучасних рішень у сфері веброзробки, створене застосування дозволяє оптимізувати процес онлайн-продажів та підвищити конверсію магазину, що є ключовою метою будь-якого e-commerce бізнесу.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Актуальність предметної області

Інтернет-покупки набирають популярності завдяки своїй швидкості та зручності – тепер не потрібно виходити з дому, щоб купити потрібні речі. Магазини одягу онлайн пропонують клієнтам можливість ознайомитися з каталогом, вибрати найкращі варіанти, оформити замовлення та отримати його, економлячи час.

Головною проблемою, яку вирішує сучасний інтернет-магазин, є створення зручного, швидкого та функціонального вебзастосунку, що забезпечує комфортний процес вибору та купівлі товарів. Це особливо важливо для користувачів, які очікують інтуїтивно зрозумілий інтерфейс, швидке завантаження сторінок і надійність у роботі.

На ринку є багато сервісів для продажу одягу, таких як Shopify [1] та WooCommerce [2]. Хоча ці платформи мають свої сильні та слабкі сторони, наш вебзастосунок пропонує низку унікальних функцій, що забезпечують йому конкурентні переваги.

Основні проблеми, з якими стикаються користувачі:

- повільне завантаження сторінок, що призводить до втрати потенційних клієнтів. Незручний перегляд оголошень;
- незручна система пошуку та фільтрації, що ускладнює вибір товару;
- обмежений функціонал персонального кабінету, що ускладнює управління замовленнями;
- проблеми з обробкою платежів та безпекою персональних даних;
- відсутність ефективного механізму рекомендацій, що знижує конверсію;
- низька швидкість обробки замовлень, що негативно впливає на лояльність покупців.

У наступних розділах буде детально розглянуто, як наше програмне забезпечення зможе вирішити ці проблеми та запропонувати кращі рішення у порівнянні з аналогами.

1.2 Огляд існуючих методів вирішення завдання

Сьогодні на ринку онлайн-торгівлі одягом існує кілька провідних платформ, серед яких можна виділити ASOS [3] та Zalando [4]. Ці інтернет-магазини є лідерами в галузі модного ритейлу, пропонуючи широкий асортимент товарів, зручний користувацький досвід та інноваційні технології для підвищення ефективності продажів.

ASOS [3] є одним із найпопулярніших онлайн-магазинів одягу, орієнтованим переважно на молодіжну аудиторію. Його головна перевага – це великий вибір брендів, включаючи як доступні мас-маркетові варіанти, так і преміальні колекції.

Основні особливості платформи ASOS [3]:

- персоналізовані рекомендації: алгоритми аналізують вподобання користувачів і пропонують товари, які можуть їх зацікавити;
- програма ASOS Premier: надає можливість безкоштовної доставки за передплатою;
- функція "See My Fit": дозволяє користувачам побачити, як одяг виглядає на моделях різної комплекції;
- легка система повернень, що підвищує довіру покупців.

Zalando [4] – це європейський гігант онлайн-шопінгу, який пропонує різноманітні бренди одягу, взуття та аксесуарів. Його головна особливість – акцент на локальні ринки та адаптацію під регіональних покупців.

Основні особливості Zalando [4]:

- програма Zalando Plus, яка пропонує швидку доставку та ексклюзивні знижки;

- розширена система пошуку, що дозволяє фільтрувати товари за різними параметрами, включаючи екологічність;
- інтеграція з Instagram для натхнення та пошуку трендів;
- послуга "Try Before You Buy", що дає змогу приміряти товари перед оплатою;
- система оцінки розміру, що використовує дані користувачів для підбору ідеального розміру.

Обидва інтернет-магазини мають сильні сторони та відмінні стратегії залучення користувачів, проте також стикаються з певними викликами. Аналізуючи їхні підходи, ми можемо впровадити оптимізовану систему пошуку, покращену систему персоналізації товарів та вдосконалену логістику у нашому вебзастосунку, що дозволить зробити шопінг ще більш швидким, зручним та ефективним.

1.3 Огляд існуючих програмних засобів, що вирішують аналогічні завдання

1.3.1 ASOS

ASOS [3] – це глобальний інтернет-магазин, який пропонує великий асортимент одягу, взуття та аксесуарів від відомих брендів і власних ліній. Сайт надає доступ до актуальних трендів моди, дозволяючи клієнтам робити покупки онлайн з будь-якого пристрою.

Основні переваги платформи:

- широкий асортимент. ASOS [3] пропонує тисячі товарів від різних брендів, охоплюючи як мас-маркет, так і преміальний сегмент. Це дозволяє споживачам знайти продукцію відповідно до своїх вподобань, бюджету та стилю;
- зручний інтерфейс. Платформа характеризується інтуїтивно зрозумілою структурою, розширеними фільтрами пошуку та адаптивним дизайном, що спрощує процес вибору та замовлення товарів;

- гнучкість у виборі. Користувачі можуть здійснювати пошук продукції за різними критеріями, включаючи розмір, колір, бренд, матеріал та інші параметри, що сприяє персоналізованому підходу до шопінгу;
- система відгуків та рейтингів. ASOS [3] надає можливість перегляду оцінок та коментарів інших покупців, що допомагає користувачам ухвалювати більш обґрунтовані рішення під час купівлі;
- безпека та гарантії. Платформа забезпечує безпечні методи оплати, оперативну доставку та зручні умови повернення товарів, що мінімізує ризики та підвищує рівень довіри споживачів.

На рисунку 1.1 зображено головний екран вебсайту ASOS [3].

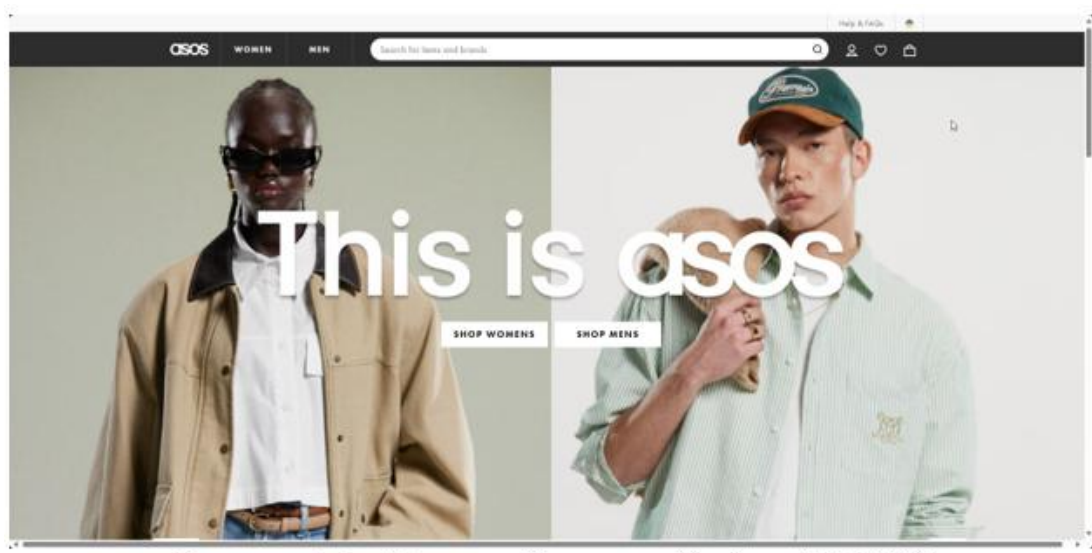


Рисунок 1.1 – Головний екран вебсайту ASOS [3]

1.3.2 Zalando

Zalando [4] є однією з найбільших онлайн-платформ у Європі, що спеціалізується на продажу одягу, взуття та аксесуарів. Завдяки широкому асортименту товарів від відомих брендів та високому рівню сервісу, компанія забезпечує зручний та сучасний підхід до здійснення покупок.

Основні переваги платформи:

- різноманітний асортимент. Zalando [4] пропонує продукцію понад 2000 міжнародних брендів, охоплюючи як бюджетний сегмент, так і преміальні колекції. Це дозволяє споживачам обирати товари відповідно до власних вподобань та фінансових можливостей;
- зручний користувацький інтерфейс. Платформа має інтуїтивно зрозумілий дизайн, що включає розширені фільтри, рекомендаційні алгоритми та персоналізовані пропозиції, сприяючи комфортному процесу вибору товарів;
- гнучка система пошуку. Zalando [4] дозволяє користувачам налаштовувати пошук за такими критеріями, як бренд, ціна, матеріал, стиль, екологічність виробництва тощо. Це забезпечує персоналізований підхід до здійснення покупок;
- система рейтингів та відгуків. Користувачі можуть переглядати відгуки інших покупців, що сприяє ухваленню обґрунтованих рішень під час купівлі;
- надійність та безпека. Zalando [4] гарантує безпечні методи оплати, швидку доставку та зручну систему повернення товарів протягом 100 днів, що підвищує рівень довіри до платформи.

На рисунку 1.2 зображено головний екран вебсайту Zalando [4].

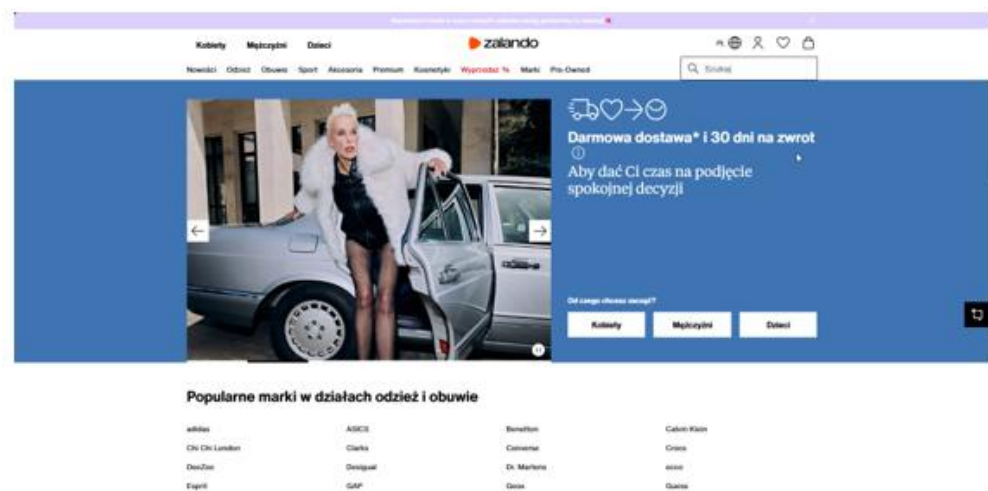


Рисунок 1.2 – Головний екран вебсайту Zalando [4]

1.3.3 Порівняльна таблиця існуючих та розроблюваного програмних продуктів

ASOS [3] та Zalando [4] – провідні інтернет-магазини одягу, взуття та аксесуарів. Проте, окрім переваг, ці платформи мають і деякі обмеження, що можуть позначитися на комфорті покупців.

Наприклад, деякі товари можуть мати обмежену доступність у певних регіонах, а умови доставки та повернення варіюються залежно від країни. Крім того, не всі бренди представлені на обох платформах, що може впливати на вибір покупців.

Водночас, обидва сервіси надають ряд спільних переваг, таких як розширені можливості фільтрації товарів, зручний користувацький інтерфейс, безпечні методи оплати та швидку доставку.

У таблиці 1.1 наведено порівняльну характеристику платформ ASOS [3] та Zalando [4], що дозволяє оцінити їх ключові особливості.

Таблиця 1.1 – Порівняння програмних продуктів

Назва	ASOS	Zalando
Глобальна доставка	+	-
Широкий асортимент товарів	+	+
Велика кількість брендів	+	+
Розширена система фільтрації	-	+
Зручний інтерфейс	+	+
Тривалий термін повернення	-	+
Підписка на преміальні послуги	+	+
Гнучка система знижок	+	+
Різноманітність способів оплати	-	+
Оцінки продавців і товарів	-	+

1.4 Постановка завдання роботи

Метою проєкту є створення вебзастосунка для онлайн магазину одягу, який забезпечить користувачам можливість переглядати, купувати товари, а адміністраторам – керувати контентом магазину.

Вебзастосунок повинен включати наступний функціонал, який був визначений на основі аналізу конкурентів для забезпечення конкурентоздатності та задоволення потреб користувачів:

- а) реєстрація та авторизація користувачів;
- б) перегляд каталогу товарів з можливістю фільтрації та сортування;
- в) оформлення замовлення;
- г) додавання товарів до обраного;
- г) особистий кабінет користувача (редагування профілю, перегляд історії замовлень);
- д) адміністративна панель для керування контентом магазину, яка включає:
 - 1) створення, редагування та видалення категорій товарів;
 - 2) додавання, редагування та видалення товарів;
 - 3) управління замовленнями;
 - 4) керування користувачами.

1.5 Висновки за розділом 1

У першому розділі було проведено дослідження та аналіз вебзастосунків популярних інтернет-магазинів одягу.

Під час оцінки були визначені основні переваги та недоліки кожного з розглянутих сервісів. Результатом цього аналізу став розділ 1.4, у якому сформовано перелік функцій, необхідних для впровадження у розроблюваний вебзастосунок.

Актуальність створення даного вебзастосунку підтверджена необхідністю більш зручного процесу управління товарами, замовленнями та користувачами, а також підвищенням зручності онлайн-шопінгу для клієнтів.

2 ВИБІР ТА ОБҐРУНТУВАННЯ СТРУКТУРИ СИСТЕМИ, ЯКА ПРОЄКТУЄТЬСЯ

2.1 Вибір інструментарію розробки програмного забезпечення

Для розробки вебзастосунку було обрано сучасні технології, які дають високу продуктивність, масштабованість та зручність.

Основний стек складається з наступних компонентів:

а) Frontend на Next.js [5] з TypeScript [6]:

1) Next.js [5] забезпечує швидку серверну та статичну генерацію сторінок, що покращує продуктивність;

2) TypeScript [6] додає статичну типізацію, що сприяє кращій структурованості коду та зменшенню помилок;

б) Backend на Nest.js [7] з Node.js [8]:

1) Nest.js [7] побудований на основі TypeScript [6] та використовує модульну архітектуру, що спрощує розширюваність і підтримку проєкту;

2) Node.js [8] забезпечує високу продуктивність сервера завдяки асинхронній обробці запитів;

в) база даних:

1) для збереження даних буде використовуватись реляційна база даних PostgreSQL [9], яка добре інтегрується з Nest.js [7].

2.1.1 Огляд особливостей мов програмування

Особливості мов програмування:

а) Next.js [5] з TypeScript [6]:

1) серверний рендеринг та статична генерація: Next.js [5] підтримує SSR [10] (Server-Side Rendering) та SSG [11] (Static Site Generation), що покращує продуктивність і SEO [12];

2) гнучка маршрутизація: Вбудована система маршрутизації Next.js [5] спрощує створення динамічних сторінок;

3) статична типізація з TypeScript [6]: Використання TypeScript [6] забезпечує контроль типів, що зменшує кількість помилок у коді;

б) Nest.js [7] з Node.js [8]:

1) модульна архітектура: Nest.js [7] побудований за принципами SOLID [13], що полегшує розширюваність і підтримку коду;

2) підтримка TypeScript [6]: Nest.js [7] повністю написаний на TypeScript [6], що сприяє узгодженості між фронтендом і бекендом;

3) ефективна робота з API: Використання REST [14] або GraphQL [15] дозволяє гнучко взаємодіяти з клієнтською частиною.

У таблиці 2.1 наведено порівняння мов програмування.

Таблиця 2.1 – Порівняння мов програмування

Назва	JavaScript	TypeScript	PHP
Тип мови	Динамічно типізована	Статично типізована	Динамічно типізована
Строга перевірка типів	Ні	Так	Ні
Сумісність з ECMAScript	+	+	-
Швидкодія	Висока	Висока	Висока
Об'єктно-орієнтований підхід	+	+	+

2.1.2 Огляд особливостей обраного середовища розробки

Як середовище розробки було обрано VS Code [16], через ряд його переваг, таких як:

Легкість використання. VS Code [16] має простий та інтуїтивно зрозумілий інтерфейс.

Розширюваність та плагіни. VS Code [16] має широкий вибір розширень та плагінів, які дозволяють налаштовувати середовище розробки під власні потреби.

Інтеграція з Git [17]. Вбудована підтримка Git [17] у VS Code [16] дає можливість легко працювати з репозиторіями, робити коміти, переглядати відгалуження та виконувати інші операції безпосередньо з інтерфейсу.

Кросплатформеність. VS Code [16] підтримується на різних операційних системах, таких як Windows, macOS та Linux.

У таблиці 2.2 наведено порівняння середовищ розробки.

Таблиця 2.2 – Порівняння середовищ розробки

Назва	VS Code	Visual Studio	WebStorm
Багатоплатформність	+	-	+
Швидкість модернізації	+	+	+
Наявність візуального конструктора	-	+	+
Підтримка репозиторіїв	+	+	+

2.1.3 Огляд особливостей обраної системи управління базами даних

Збереження та обробка даних у вебзастосунку буде з допомогою PostgreSQL [9]. PostgreSQL [9] – це об'єктно-реляційна система управління базами даних (СУБД) з відкритим кодом, яка забезпечує високу продуктивність, масштабованість та безпеку.

Основні переваги PostgreSQL [9]:

- гнучкість: Підтримка складних типів даних, JSON, функцій та розширень дозволяє ефективно працювати з великими та складними структурами даних;
- масштабованість: PostgreSQL [9] може обробляти великі обсяги даних, а також підтримує горизонтальне та вертикальне масштабування;
- безпека: Вбудовані механізми аутентифікації, шифрування та контроль доступу забезпечують надійний захист даних;
- підтримка транзакцій: Відповідність ACID-принципам гарантує надійність та цілісність даних;
- кросплатформеність: PostgreSQL [9] працює на всіх популярних операційних системах, таких як Windows, macOS і Linux.

Завдяки PostgreSQL [9] у зв'язці з Nest.js [7] та TypeORM/Prisma виходить зручна взаємодія між бекендом і базою даних, що в цілому спрощує розробку та підтримку вебзастосунку з бекенд сторони.

У таблиці 2.3 наведено порівняння баз даних.

Таблиця 2.3 – Порівняння баз даних

Назва	Microsoft SQL Server	MySQL	PostgreSQL
Сумісність з різними операційними системами	+	+	+
Підтримка розподілених операційних систем	+	+	+
Масштабованість	+	-	+
Спільнота та підтримка	-	+	+
Швидкодія	+	-	+

2.2 Структурна схема програми

Перед розробкою вебзастосунку була розроблена структурна схема програми.

Основні модулі вебзастосунку:

- Frontend (Next.js [5], TypeScript [6]) – відповідає за інтерфейс користувача, обробку візуальних даних та взаємодію з API;
- Backend (Nest.js [7], Node.js [8]) – виконує основну логіку, яка скрита під копотом, обробляє запити від фронтенду та керує доступом до даних;
- база даних (PostgreSQL [9]) – зберігає інформацію про користувачів, товари, замовлення та інші сутності;
- API Gateway – завдяки ендпоінтам дає можливість обмінюватися даними між фронтендом і бекендом;
- Auth Service – відповідає за автентифікацію та авторизацію користувачів, обробку JWT-токенів та керування ролями;
- Admin Panel – інтерфейс для адміністрування категорій, товарів, користувачів та замовлень.

На рис. 2.1 наведено структурну схему програми.

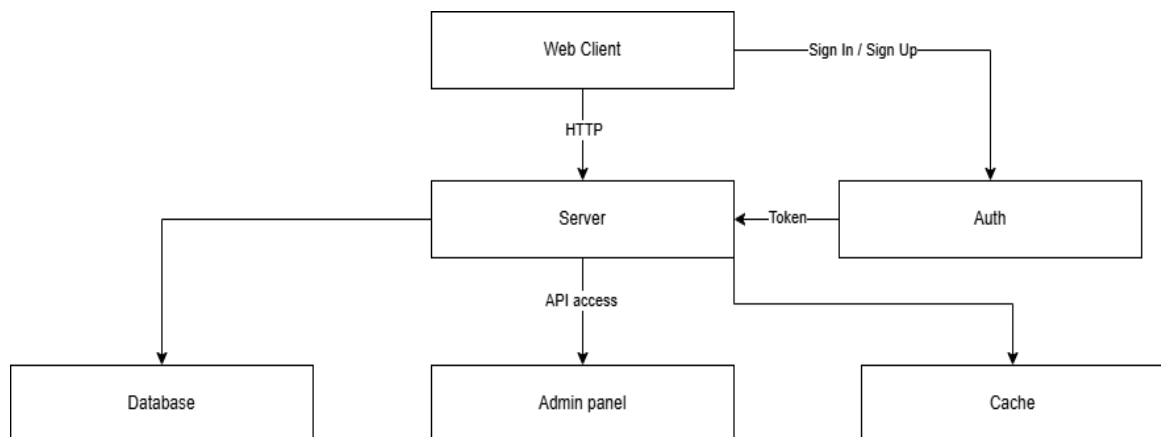


Рисунок 2.1 – Структурна схема програми

2.3 Опис модулів розроблювальної системи

У таблиці 2.4 наведено опис модулів вебзастосунку з визначенням їх основних функцій.

Таблиця 2.4 – Опис модулів вебзастосунку з визначенням основних функцій

Модуль	Опис
Frontend (Next.js, TypeScript)	Відповідає за користувацький інтерфейс, обробку введених даних та взаємодію з API. Забезпечує навігацію та динамічний рендеринг контенту.
Backend (NestJS, Node.js)	Виконує бізнес-логіку, обробляє запити від фронтенду, взаємодіє з базою даних та зовнішніми сервісами.
База даних (PostgreSQL)	Зберігає інформацію про користувачів, товари, замовлення, категорії та інші сутності. Забезпечує цілісність і швидкий доступ до даних.
API Gateway	Слугує єдиною точкою входу для клієнтських запитів, забезпечує маршрутизацію, балансування навантаження та безпеку.
Auth Service	Відповідає за автентифікацію та авторизацію користувачів, обробку JWT-токенів, керування ролями та доступами.
Admin Panel	Інтерфейс для адміністрування категорій, товарів, користувачів та замовлень. Дозволяє створювати, редагувати та видаляти записи.

2.4 Висновки за розділом 2

У другому розділі проєкту було проведено детальний аналіз і вибір ключових компонентів для розробки вебзастосунку. Були обрані інструменти, фреймворки та мова програмування, які оптимально відповідають вимогам проєкту. Крім того, були визначені основні модулі системи, які визначають логіку функціонування програми, а також загальна структура вебзастосунку, яка відображає взаємозв'язок між цими модулями.

3 ОСНОВНІ РІШЕННЯ ЩОДО РЕАЛІЗАЦІЇ КОМПОНЕНТІВ СИСТЕМИ

3.1 Алгоритм функціонування розроблювальної системи

Алгоритми які використовуються при роботі із розробленим вебзастосунком зображені на рисунку 3.1.

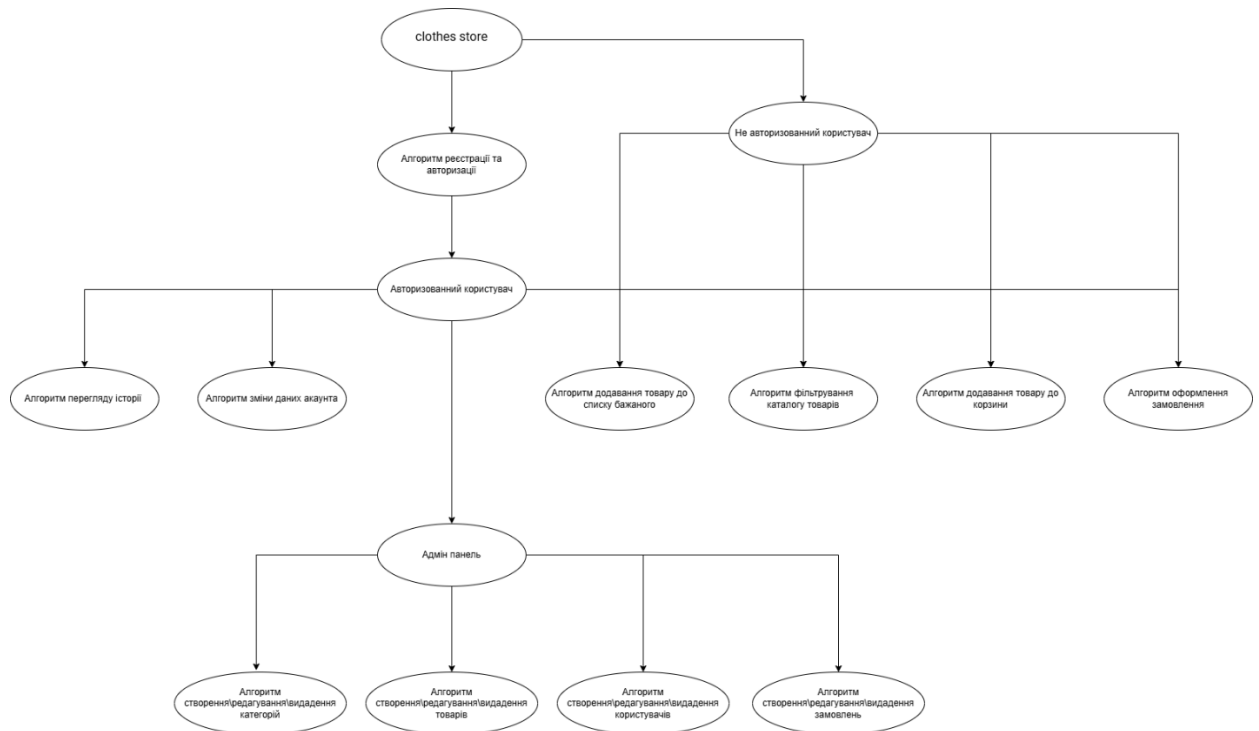


Рисунок 3.1 – Алгоритми, що використовуються у вебзастосунку

Як видно зі схеми під час розробки вебзастосунку, були розроблені наступні алгоритми:

- реєстрації та авторизації;
- додавання товару до списку бажаного;
- фільтрування каталогу товарів;
- додавання товару до кошика;
- оформлення замовлення;
- перегляду історії;
- зміни даних акаунта;

- створення\редагування\видалення категорій;
- створення\редагування\видалення товарів;
- створення\редагування\видалення користувачів;
- створення\редагування\видалення замовлень.

3.1.1 Алгоритм авторизації та реєстрації

Для реєстрації в вебзастосунку користувачу потрібно ввести в форму реєстрації наступні данні:

- First name;
- Last name;
- Email;
- Password;
- Confirm password;

Після того як користувач ввів усі необхідні дані, він може натиснути кнопку «Sign Up». Після цього його дані відправляються на сервер для створення нового аккаунту користувача. Якщо реєстрація успішна, сервер створює сесію для користувача та зберігає ідентифікатор сесії в базі даних. Користувач автоматично авторизується й може повноцінно користуватися вебзастосунком. Блок-схема алгоритму реєстрації наведена на рисунку 3.3.

Під час авторизації користувачу треба ввести усі необхідні облікові дані, такі як електронна пошта та пароль. Після введення цих даних сервер перевіряє їх правильність. Якщо вони вірні, сервер створює нову сесію та зберігає її у базі даних. Сесійний ідентифікатор надсилається користувачеві у вигляді cookie, який браузер автоматично додає до всіх подальших запитів.

Завдяки використанню сесій забезпечується безпека та контроль доступу до вебзастосунку. Оскільки інформація про авторизацію зберігається на сервері, це ускладнює несанкціонований доступ. Також, сесії мають обмежений час дії, що зменшує ризик компрометації даних у разі втрати

доступу до пристрою користувача. Блок-схема алгоритму авторизації наведена на рисунку 3.2.

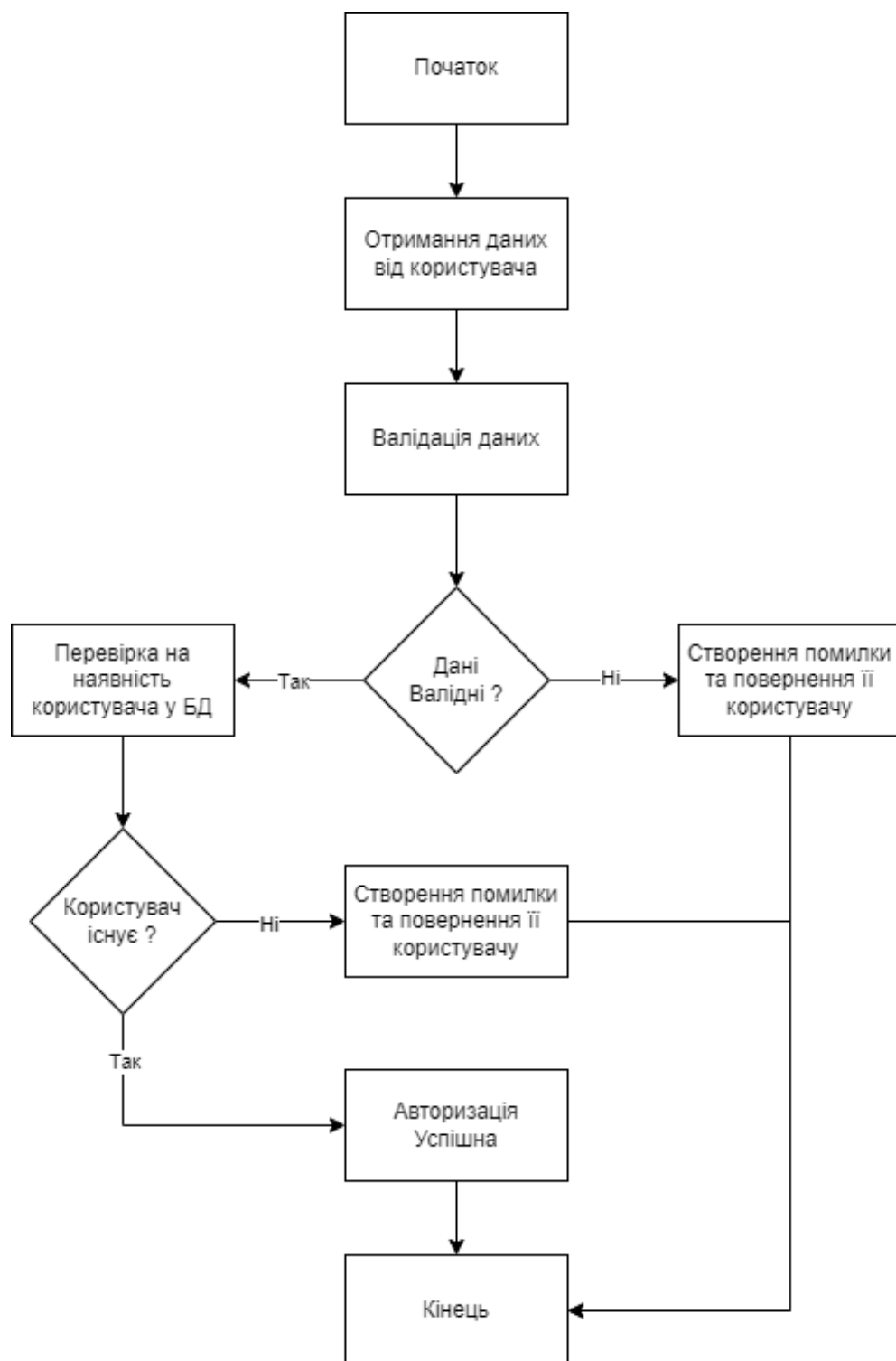


Рисунок 3.2 – Блох-схема алгоритму авторизації

Блок-схема алгоритму реєстрації наведена на рисунку 3.3.

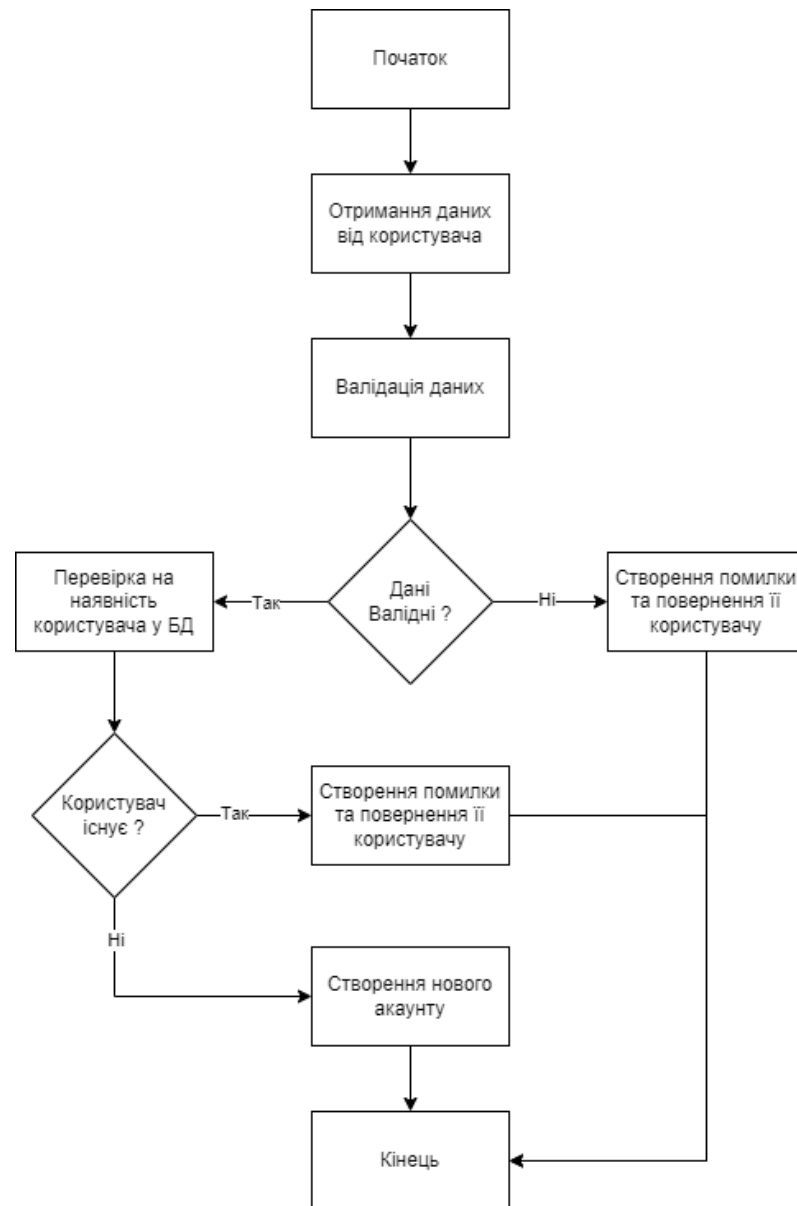


Рисунок 3.3 – Блок-схема алгоритму реєстрації

3.1.2 Алгоритм додавання товару до списку бажаного

Коли користувач знаходить товар, який хоче зберегти для подальшого перегляду, він може натиснути кнопку з іконкою серця.

Після натискання кнопки ідентифікатор вибраного товару зберігається в localStorage, але перед цим виконується перевірка, чи вже є цей товар у списку бажаного. Якщо товар ще не доданий, його ідентифікатор

зберігається, якщо ж він уже є у списку, товар навпаки видаляється (принцип toggle).

Усі додані товари зберігаються в localStorage, і користувач може переглянути їх у відповідному розділі вебзастосунку.

Блок-схема алгоритму додавання товару до списку бажаного наведена на рисунку 3.4.

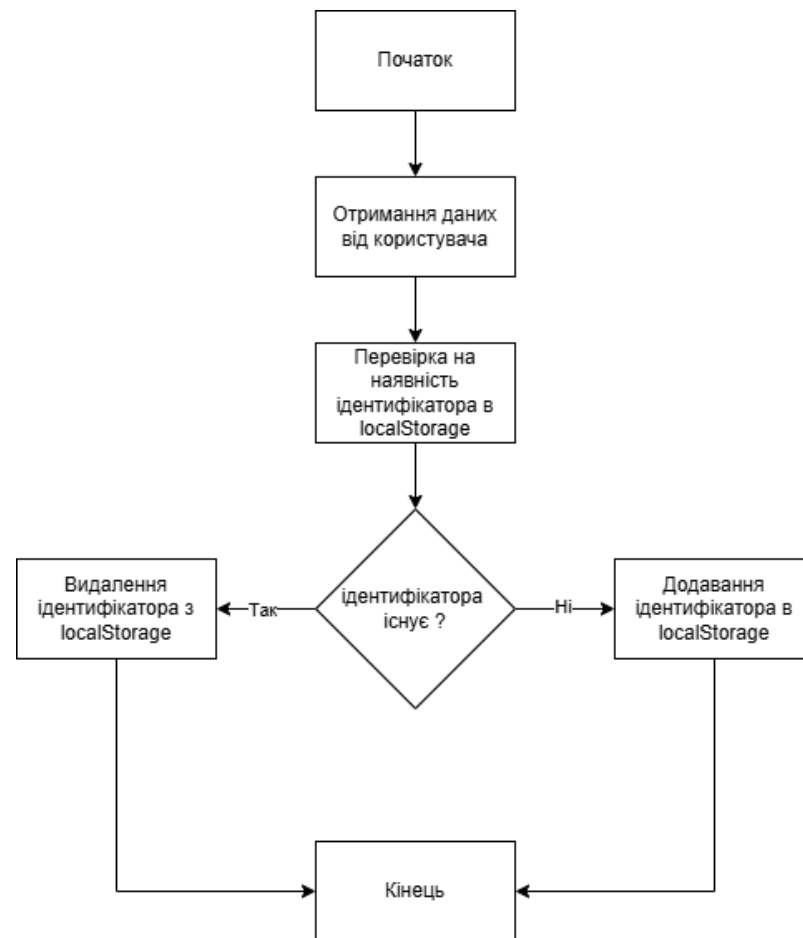


Рисунок 3.4 – Блок-схема алгоритму додавання товару до списку бажаного

3.1.3 Алгоритм фільтрування каталогу товарів

У каталозі товарів реалізовано два типи фільтрації: миттєва та фільтрація з підтвердженням.

Миттєва фільтрація. Це тип фільтрації застосовується для вибору за основними параметрами, такими як дата додавання товару, сортування за ціною (від найнижчої до найвищої або навпаки) та сортування за алфавітом.

При виборі одного з параметрів фільтрації відповідне значення додається як query-параметр у запит. Сервер отримує запит, обробляє параметри, фільтрує товари відповідно до обраного критерію та повертає відсортований масив даних.

Фільтрація з підтвердженням. Цей вид фільтрації використовується для складніших запитів, таких як вибір категорій товарів або фільтрація за діапазоном цін.

Користувач обирає необхідні параметри, після чого натискає кнопку «Apply». Обрані значення передаються в тілі запиту (body), сервер обробляє ці дані, застосовує відповідні фільтри та повертає відфільтрований список товарів.

Застосування цього підходу зумовлене тим, що кожна категорія товарів має унікальний UUID-ідентифікатор довжиною 36 символів. Використання таких ідентифікаторів у вигляді query-параметрів могло б призвести до помилок, оскільки максимальна довжина URL-запиту обмежена 2000 символами. У випадку вибору великої кількості категорій запит міг би перевищити цей ліміт і оброблятися не коректно. Передача параметрів у тілі запиту дозволяє уникнути таких проблем і забезпечує стабільну роботу фільтрації.

Завдяки такій реалізації фільтрації також покращилась зручність користування інтерфейсом – користувачі мають змогу спокійно обирати потрібні параметри, не турбуючись про негайне оновлення сторінки при кожній дії. Це знижує навантаження на сервер.

Блок-схема алгоритму фільтрування каталогу товарів наведена на рисунку 3.5.



Рисунок 3.5 – Блох-схема алгоритму фільтрування каталогу товарів

3.1.4 Алгоритм додавання товару до кошика

При додаванні товару до кошика використовується `localStorage`, де зберігаються ідентифікатори товарів та їх кількість.

Коли користувач знаходиться на сторінці товару, він може встановити бажану кількість одиниць за допомогою лічильника та натиснути кнопку "Add to cart". Після цього відбувається перевірка, чи цей товар вже є у кошику. Якщо товар ще не додано, його айді та вибрана кількість зберігаються в `localStorage`. Якщо товар вже є в кошику, система просто оновлює кількість відповідно до вибору користувача.

На сторінці кошика виконується запит на отримання інформації про товари за їхніми ідентифікаторами. Сервер повертає всі відповідні товари, після чого клієнтська частина вебзастосунку відображає їх разом із вибраною кількістю. Користувач може змінювати кількість товару або видаляти його з кошика. Усі ці зміни також оновлюються в `localStorage`, забезпечуючи збереження даних між сесіями роботи.

Блок-схема алгоритму додавання товару до корзини наведена на рисунку 3.6.

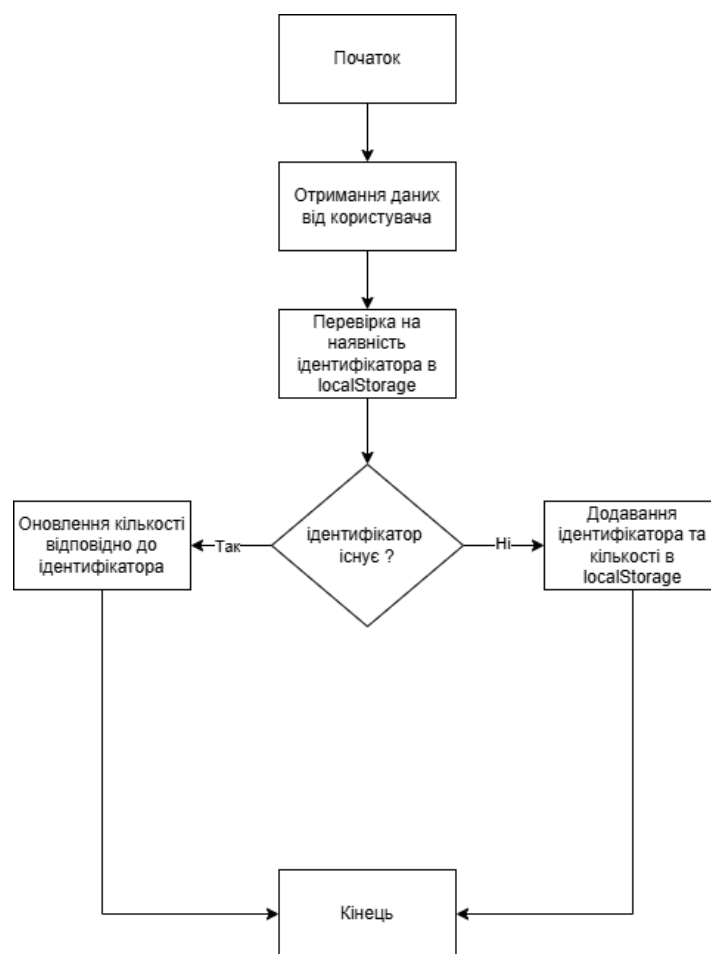


Рисунок 3.6 – Блок-схема алгоритму додавання товару до корзини

3.1.5 Алгоритм оформлення замовлення

Процес оформлення замовлення починається після того, як користувач додав товари до кошика та перейшов до етапу оформлення. На

цій сторінці він повинен заповнити всі необхідні поля, такі як контактна інформація, адреса. Якщо користувач пропускає будь-яке з обов'язкових полів або вводить некоректні дані, форма виводить повідомлення про помилку валідації, не дозволяючи продовжити оформлення.

Після успішного заповнення всіх полів відбувається відправка запиту на сервер, де створюється новий запис замовлення в базі даних. Сервер зберігає всю інформацію про замовлення, включаючи товари, їх кількість, загальну суму та дані користувача.

Після успішного створення замовлення користувач може переглянути його деталі у своєму особистому кабінеті в розділі "History", де відображається замовлення та його зміст. Водночас адміністратор отримує доступ до цього замовлення у панелі керування, у вкладці "Orders", де він може обробляти замовлення.

Блок-схема алгоритму оформлення замовлення наведена на рис. 3.7.

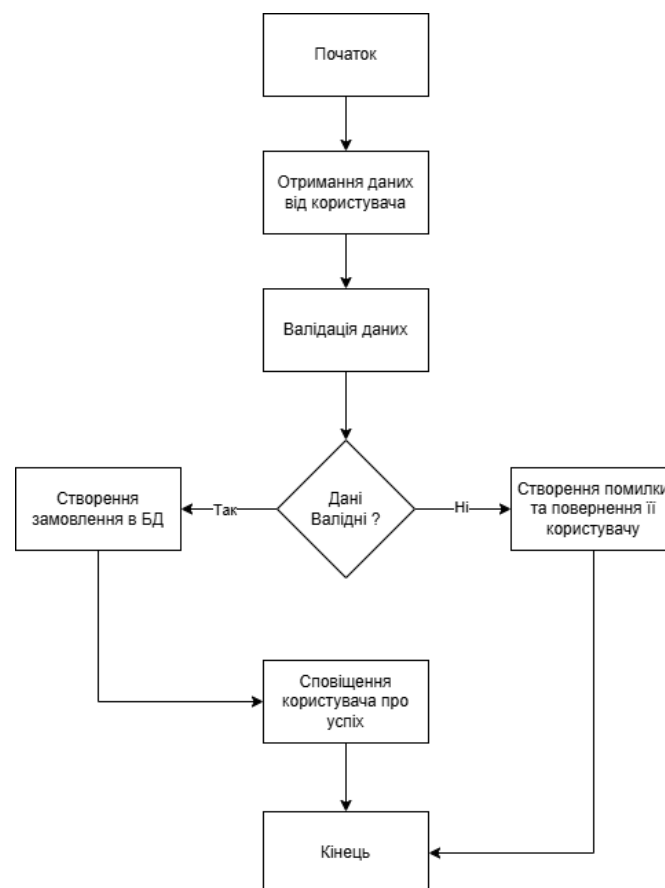


Рисунок 3.7 – Блох-схема алгоритму оформлення замовлення

3.1.6 Алгоритм перегляду історії

Після оформлення замовлення користувач має можливість переглянути свою історію покупок у особистому кабінеті. Для цього він переходить до вкладки "History", після чого вебзастосунок відправляє запит на сервер для отримання всіх його замовлень.

Сервер виконує пошук замовлень у базі даних за унікальним ідентифікатором користувача та формує відповідь, що містить список усіх оформлених замовлень. Після отримання цієї інформації клієнтська частина вебзастосунку відображає список замовлень.

Користувач може вибрати будь-яке замовлення зі списку, щоб переглянути його деталі. У детальному перегляді відображається список придбаних товарів, їх кількість, загальна сума, адреса доставки. Це дозволяє користувачеві відстежувати свої покупки та контролювати виконання замовлень.

Блок-схема алгоритму перегляду історії наведена на рисунку 3.8.

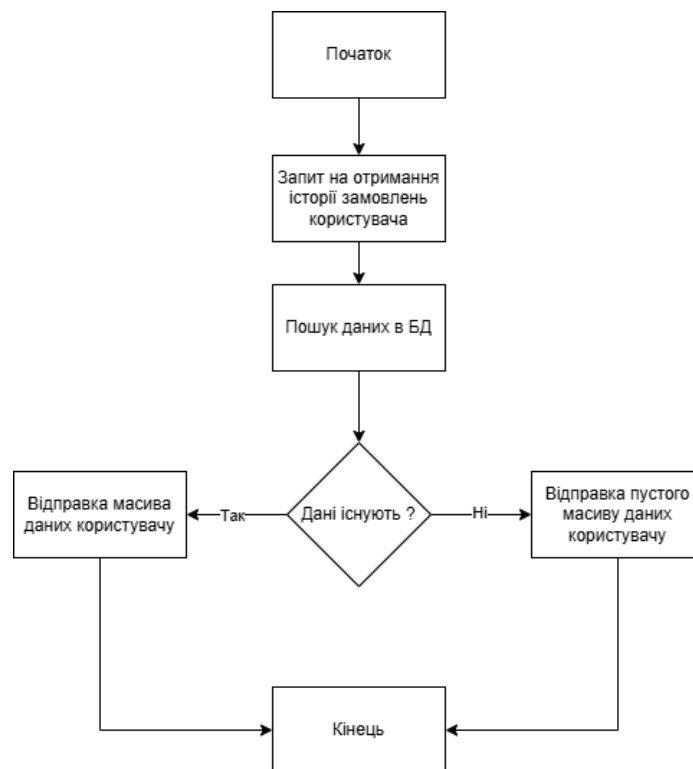


Рисунок 3.8 – Блок-схема алгоритму перегляду історії

3.1.7 Алгоритм зміни даних акаунта

Для зміни персональних даних користувач повинен увійти до особистого кабінету та перейти у відповідний розділ налаштувань акаунта. У цьому розділі він має можливість редагувати основні дані, такі як ім'я (first name), прізвище (last name), електронна пошта (email) та номер телефону (phone).

Окрім цього, користувач може змінити свою адресу, включаючи країну (country), місто (city), область/штат (state), вулицю (street), номер будинку (house), квартиру (flat), поверх (floor) та поштовий індекс (zip). Також передбачена можливість зміни пароля для забезпечення безпеки облікового запису.

Після внесення змін користувач надсилає їх на сервер, де відбувається перевірка коректності введених даних. Якщо дані відповідають вимогам, сервер оновлює відповідні поля у базі даних та надсилає підтвердження успішного збереження змін. Після успішного оновлення інформації користувач може продовжити користування вебзастосунком із актуальними даними.

Блок-схема алгоритму зміни даних акаунта наведена на рисунку 3.9.

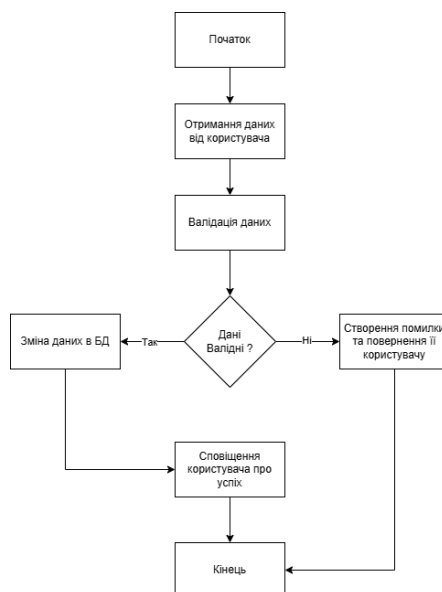


Рисунок 3.9 – Блок-схема алгоритму зміни даних акаунта

3.1.8 Алгоритм створення\редагування\видалення категорій

Адміністратор має можливість створювати, редагувати та видаляти категорії товарів у вебзастосунку. Для цього він повинен увійти в адміністративну панель та перейти у вкладку "Categories", де відображається таблиця зі списком усіх існуючих категорій. У цій таблиці адміністратор може виконувати різні дії, такі як створення нової категорії, редагування, видалення, активація та деактивація категорій.

Щоб створити нову категорію, адміністратор натискає кнопку "Create", після чого відкривається форма, у якій потрібно заповнити всі обов'язкові поля. Після внесення даних категорія зберігається та додається до загального списку.

Для редагування категорії адміністратор обирає необхідний елемент у таблиці, натискає на іконку з трьома крапками та вибирає опцію "Edit". Після цього відкривається форма редагування, де можна змінити потрібні параметри. Після внесення змін адміністратор натискає кнопку "Save", і оновлені дані зберігаються.

Система підтримує два види видалення:

- **одиначне видалення** – адміністратор вибирає необхідну категорію, натискає іконку з трьома крапками та обирає "Delete", після чого підтверджує дію;

- **групове видалення** – для цього адміністратор відзначає декілька категорій за допомогою чекбоксів у таблиці, після чого натискає кнопку "Delete" у блоці групових дій та підтверджує видалення.

Зміни, внесені адміністратором, негайно застосовуються та синхронізуються із загальною базою даних.

Блок-схема алгоритму створення\редагування\видалення категорій наведена на рисунку 3.10.

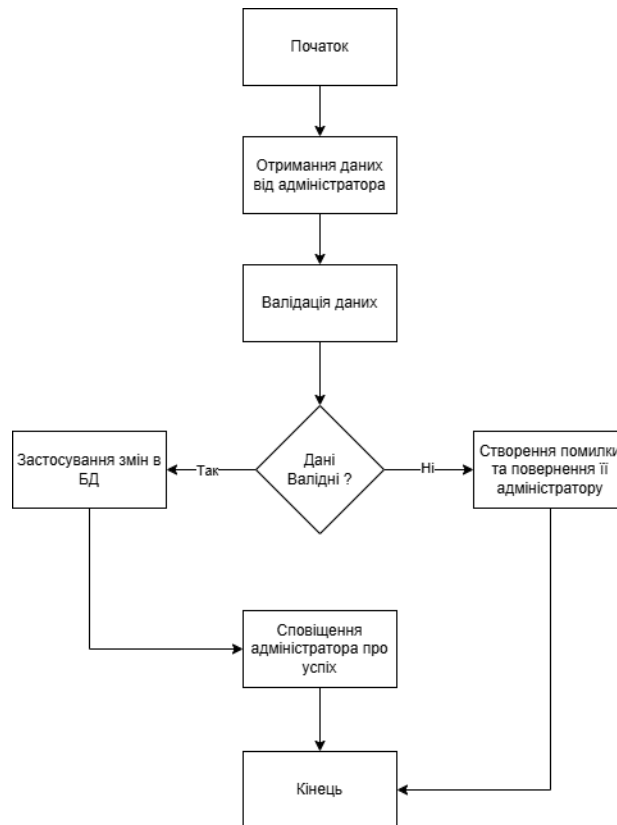


Рисунок 3.10 – Блох-схема алгоритму створення\редагування\видалення категорій

3.1.9 Алгоритм створення\редагування\видалення товарів

Адміністратор має можливість створювати, редагувати та видаляти товари у вебзастосунку. Для цього він повинен увійти в адміністративну панель та перейти у вкладку "Products", де відображається таблиця зі списком усіх існуючих товарів. У цій таблиці адміністратор може виконувати різні дії, такі як створення нового товару, редагування, видалення, активація та деактивація товарів.

Щоб створити новий товар, адміністратор натискає кнопку "Create", після чого відкривається форма, у якій потрібно заповнити всі обов'язкові поля, такі як назва, опис, категорія, ціна, зображення. Після внесення даних товар зберігається та додається до загального списку.

Для редагування товару адміністратор обирає необхідний елемент у таблиці, натискає на іконку з трьома крапками та вибирає опцію "Edit". Після

цього відкривається форма редагування, де можна змінити будь-які параметри товару. Після внесення змін адміністратор натискає кнопку "Save", і оновлені дані зберігаються.

Система підтримує два види видалення:

- одиночне видалення – адміністратор вибирає необхідний товар, натискає іконку з трьома крапками та обирає "Delete", після чого підтверджує дію;
- групове видалення – для цього адміністратор відзначає декілька товарів за допомогою чекбоксів у таблиці, після чого натискає кнопку "Delete" у блоці групових дій та підтверджує видалення.

Зміни, внесені адміністратором, негайно застосовуються та синхронізуються із загальною базою даних.

Блок-схема алгоритму створення\редагування\видалення товарів наведена на рисунку 3.11.

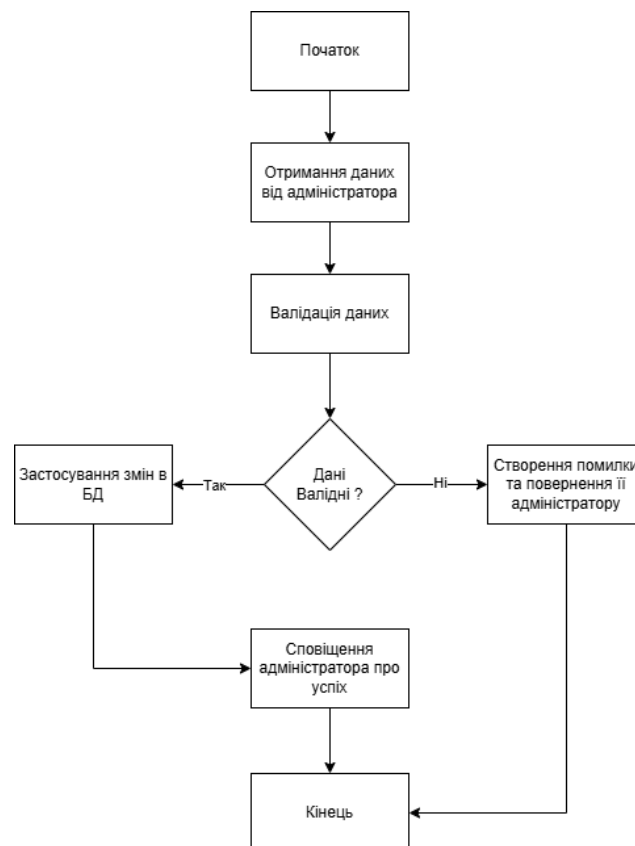


Рисунок 3.11 – Блок-схема алгоритму створення\редагування\видалення товарів

3.1.10 Алгоритм створення\редагування\видалення користувачів

Адміністратор має можливість створювати, редагувати та видаляти користувачів у вебзастосунку. Для цього він повинен увійти в адміністративну панель та перейти у вкладку "Users", де відображається таблиця зі списком усіх зареєстрованих користувачів. У цій таблиці адміністратор може виконувати різні дії, такі як створення нового користувача, редагування, видалення облікових записів.

Щоб створити нового користувача, адміністратор натискає кнопку "Create", після чого відкривається форма, у якій потрібно заповнити всі обов'язкові поля, такі як ім'я, прізвище, email, номер телефону, адресу, пароль. Після внесення даних обліковий запис зберігається та додається до загального списку користувачів.

Для редагування користувача адміністратор обирає необхідного користувача у таблиці, натискає на іконку з трьома крапками та вибирає опцію "Edit". Після цього відкривається форма редагування, де можна змінити будь-які дані користувача. Після внесення змін адміністратор натискає кнопку "Save", і оновлені дані зберігаються.

Система підтримує два види видалення:

- **одиначне видалення** – адміністратор вибирає необхідного користувача, натискає іконку з трьома крапками та обирає "Delete", після чого підтверджує дію;

- **групове видалення** – для цього адміністратор відзначає декілька користувачів за допомогою чекбоксів у таблиці, після чого натискає кнопку "Delete" у блоці групових дій та підтверджує видалення.

Зміни, внесені адміністратором, негайно застосовуються та синхронізуються із загальною базою даних.

Блок-схема алгоритму створення\редагування\видалення користувачів наведена на рисунку 3.12.

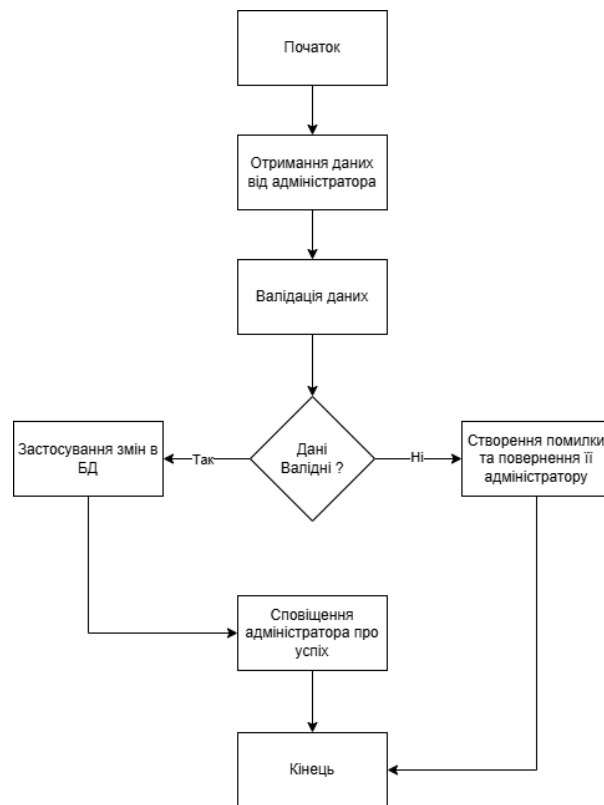


Рисунок 3.12 – Блох-схема алгоритму створення\редагування\видалення користувачів

3.1.11 Алгоритм створення\редагування\видалення замовлень

Адміністратор має можливість створювати, редагувати та видаляти замовлення у вебзастосунку. Для цього він повинен увійти в адміністративну панель та перейти у вкладку "Orders", де відображається таблиця з усіма замовленнями користувачів. У цій таблиці адміністратор може виконувати різні дії, такі як створення нового замовлення, редагування, видалення.

Щоб створити нове замовлення, адміністратор натискає кнопку "Create", після чого відкривається форма, у якій потрібно заповнити всі обов'язкові поля, такі як ідентифікатор користувача, дані та адресу, список товарів. Після заповнення інформації замовлення додається в систему та відображається у загальному списку.

Для редагування замовлення адміністратор обирає необхідне замовлення у таблиці, натискає іконку з трьома крапками та вибирає опцію

"Edit". У формі редагування він може змінити деталі замовлення. Після внесення змін натискається кнопка "Save", і оновлені дані зберігаються у базі даних.

Система підтримує два види видалення:

- **одиначне видалення** – адміністратор вибирає необхідне замовлення, натискає іконку з трьома крапками та обирає "Delete", після чого підтверджує дію;

- **групове видалення** – для цього адміністратор відзначає декілька замовлень за допомогою чекбоксів у таблиці, після чого натискає кнопку "Delete" у блоці групових дій та підтверджує видалення.

Зміни, внесені адміністратором, негайно застосовуються та синхронізуються із загальною базою даних.

Блок-схема алгоритму створення\редагування\видалення замовлень наведена на рисунку 3.13.

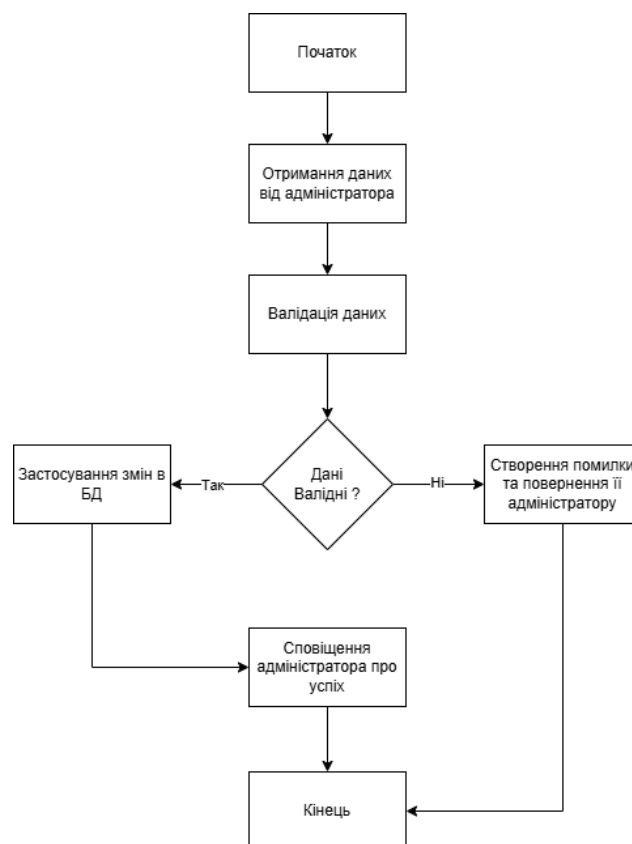


Рисунок 3.13 – Блок-схема алгоритму створення\редагування\видалення замовлень

3.2 Вибір шаблону проєктування

3.2.1 Шаблону проєктування бекенду

Бекенд частина проєкту побудована за модульною архітектурою, завдяки цьому забезпечується висока гнучкість, масштабованість та підтримуваність вебзастосунку. Кожен модуль відповідає за конкретну функціональність, що спрощує розширення та внесення змін без значного впливу на інші частини проєкту.

Основні компоненти бекенду:

- контролери – обробляють запити від клієнта та викликають відповідні сервіси;
- сервіси – містять бізнес-логіку та реалізують основні операції з даними;
- репозиторії – забезпечують доступ до бази даних та виконання CRUD-операцій;
- модулі безпеки – відповідають за аутентифікацію, авторизацію та шифрування даних;
- черги та фонові процеси – забезпечують асинхронну обробку задач та оптимізацію навантаження.

Завдяки модульному підходу забезпечується розділення обов’язків, яке полегшує тестування, підтримку та подальший розвиток системи.

3.2.2 Шаблону проєктування фронтенду

Фронтенд частина побудована за принципами Component-Based Architecture. У проєкті реалізована структурна організація коду, що передбачає розподіл на наступні основні рівні:

- Ui – містить користувацький інтерфейс, що відповідає за відображення інформації;

- Components – незалежні елементи інтерфейсу, які можуть використовуватися повторно в різних частинах застосунку;
- Widgets – складніші інтерактивні блоки, що поєднують кілька компонентів та забезпечують розширений функціонал;
- Api – модуль для комунікації з бекендом, що обробляє HTTP-запити та відповіді;
- Services – містить бізнес-логіку на фронтенді, включаючи управління станом додатка та роботу з даними.

Дана архітектурна організація дає змогу забезпечити високу гнучкість розробки, повторне використання компонентів та зручне тестування окремих частин застосунку. Завдяки цьому застосунок легше підтримувати і адаптувати до змін у вимогах проекту.

3.3 Опис модулів програми

Опис модулів вебзастосунку наведено у табл. 3.1.

Таблиця 3.1 – Опис модулів реалізованого вебзастосунку з визначенням основних функцій

Модуль	Опис
Auth	Відповідає за аутентифікацію та авторизацію користувачів.
Config	Управління конфігураційними параметрами системи.
Categories	Обробка категорій, їх створення, редагування та видалення.
Orders	Управління замовленнями, включаючи створення, зміну статусу та обробку платежів.
Products	Обробка товарів, їх характеристик, цін та наявності.

Продовження таблиці 3.1

Модуль	Опис
Sessions	Робота з користувацькими сесіями, збереження стану автентифікації.
Users	Управління користувачами, їхніми даними та правами доступу.

3.4 Проєктування бази даних

Основні сутності бази даних наведено на рис. 3.14.

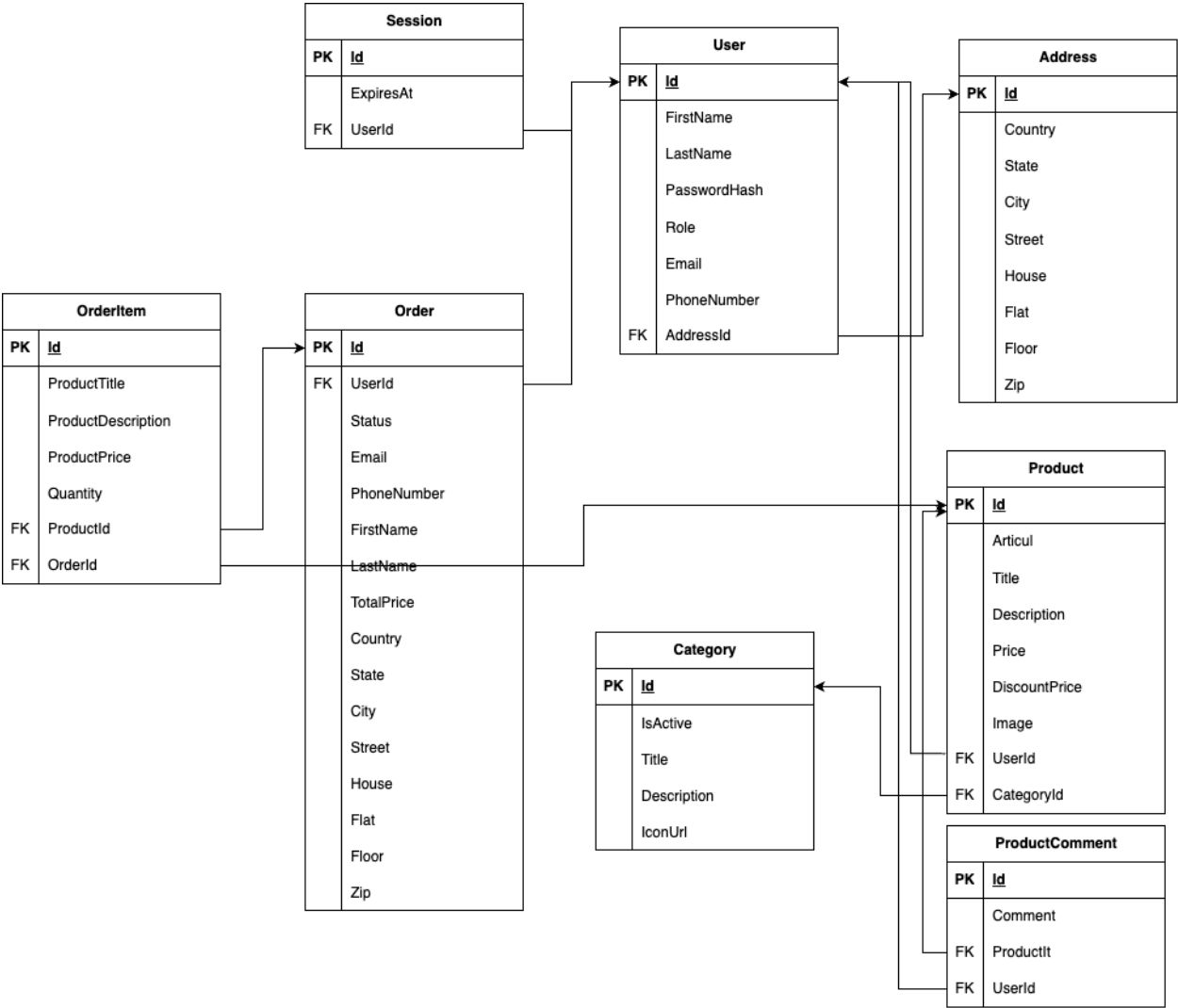


Рисунок 3.14 – Концептуальна модель бази даних

Сутність User (Користувач) (Id, FirstName, LastName, PasswordHash, Role, Email, PhoneNumber, AddressId). Опис полів сутності та їх типу наведено у таблиці 3.2.

Таблиця 3.2 – Опис полів сутності User

№	Назва поля	Тип даних	Розмір
1	Id	String	36
2	FirstName	String	100
3	LastName	String	100
4	PasswordHash	String	255
5	PhoneNumber	String	15
6	Email	String	255
7	Role	String	100
8	AddressId	String	36

Сутність Session (Сесія) (Id, ExpiresAt, UserId). Опис полів сутності та їх типу наведено у таблиці 3.3.

Таблиця 3.3 – Опис полів сутності Session

№	Назва поля	Тип даних	Розмір
1	Id	String	36
2	ExpiresAt	DATETIME	Null
3	UserId	String	36

Опис полів сутності Address та їх типу наведено у таблиці 3.4.

Таблиця 3.4 – Опис полів сутності Address

№	Назва поля	Тип даних	Розмір
1	Id	String	36
2	Country	String	255

Продовження таблиці 3.4

№	Назва поля	Тип даних	Розмір
3	State	String	255
4	City	String	255
5	Street	String	255
6	House	String	50
7	Flat	Number	10
8	Floor	Number	10
9	Zip	Number	10

Сутність Order (Замовлення) (Id, UserId, Status, Email, PhoneNumber, FirstName, LastName, TotalPrice, Country, State, City, Street, House, Flat, Floor, Zip). Опис полів сутності та їх типу наведено у таблиці 3.5.

Таблиця 3.5 – Опис полів сутності Order

№	Назва поля	Тип даних	Розмір
1	Id	String	36
2	UserId	String	36
3	Status	String	50
4	Email	String	255
5	PhoneNumber	String	15
6	FirstName	String	100
7	LastName	String	100
8	TotalPrice	Number	100
9	Country	String	255
10	State	String	255
11	City	String	255
12	Street	String	255

Продовження таблиці 3.5

№	Назва поля	Тип даних	Розмір
13	House	String	50
14	Flat	Number	10
15	Floor	Number	10
16	Zip	Number	10

Сутність OrderItem (Id, ProductTitle, ProductDescription, ProductPrice, Quantity, ProductId, OrderId). Опис полів сутності та їх типу наведено у таблиці 3.6.

Таблиця 3.6 – Опис полів сутності OrderItem

№	Назва поля	Тип даних	Розмір
1	Id	String	36
2	ProductTitle	String	100
3	ProductDescription	String	255
4	ProductPrice	Number	100
5	Quantity	Number	100
6	ProductId	String	36
7	OrderId	String	36

Сутність Product (Id, Articul, Title, Description, Price, DiscountPrice, Image, UserId, CategoryId). Опис полів сутності та їх типу наведено у таблиці 3.7.

Таблиця 3.7 – Опис полів сутності Product

№	Назва поля	Тип даних	Розмір
1	Id	String	36
2	Articul	String	100

Продовження таблиці 3.7

№	Назва поля	Тип даних	Розмір
3	Title	String	255
4	Description	String	255
5	Price	Number	100
6	DiscountPrice	Number	100
7	Image	String	255
8	UserId	String	36
9	CategoryId	String	36

Сутність Category (Id, IsActive, Title, Description, IconUrl). Опис полів сутності та їх типу наведено у таблиці 3.8.

Таблиця 3.8 – Опис полів сутності Category

№	Назва поля	Тип даних	Розмір
1	Id	String	36
2	IsActive	Boolean	TINYINT(1)
3	Title	String	255
4	Description	String	255
5	IconUrl	String	255

Сутність ProductComment (Id, Comment, ProductId, UserId). Опис полів сутності та їх типу наведено у таблиці 3.9.

Таблиця 3.9 – Опис полів сутності ProductComment

№	Назва поля	Тип даних	Розмір
1	Id	String	36
2	Comment	String	255
3	ProductId	String	36

Продовження таблиці 3.9

№	Назва поля	Тип даних	Розмір
4	UserId	String	36

3.5 Висновки за розділом 3

У цьому розділі розглядаються алгоритми, що розроблені для забезпечення роботи вебзастосунку. Кожен алгоритм докладно описується з урахуванням його призначення, принципів функціонування та використання у системі.

Після аналізу функціональних вимог вебзастосунку, визначено, що шаблон проєктування, який відповідає усім потребам системи. На основі цього шаблону описано необхідні модулі. З урахуванням обраних модулів розроблена концептуальна модель бази даних. У цій моделі описано сутності, які включаються до системи, а також їх поля та взаємозв'язки. Кожна сутність була проаналізована для визначення її ролі у системі та необхідності збереження відповідної інформації.

4 ЕКСПЛУАТАЦІЯ ПРОГРАМИ

4.1 Призначення та умови застосування програми

Вебзастосунок призначений для онлайн-торгівлі одягом, забезпечуючи зручність та ефективність процесу вибору та купівлі товарів для користувачів.

Вебзастосунок виконує наступні функції:

- а) реєстрація та авторизація користувачів;
- б) перегляд каталогу товарів з можливістю фільтрації та сортування;
- в) оформлення замовлення;
- г) додавання товарів до обраного;
- г) особистий кабінет користувача (редагування профілю, перегляд історії замовлень);
- д) адміністративна панель для керування контентом магазину, яка включає:

- 1) створення, редагування та видалення категорій товарів;
- 2) додавання, редагування та видалення товарів;
- 3) управління замовленнями;
- 4) керування користувачами.

Для коректного функціонування вебзастосунку, повинні виконуватися наступні умови до апаратної частини сервера:

- стабільне і швидке з'єднання з інтернетом для забезпечення безперервної доступності сервісу;
- надійне сховище даних для збереження інформації про користувачів, товари та замовлення.

Умови до програмної частини, для нормального функціонування вебзастосунку:

- розроблений інтуїтивно зрозумілий інтерфейс користувача для зручного використання сервісу;

- захист персональних даних користувачів та конфіденційності інформації;
- постійне оновлення вебзастосунку для виправлення помилок і вдосконалення функціоналу.

Системні вимоги до апаратного забезпечення:

- процесор: Ryzen 3 1300x або вище;
- оперативна пам'ять: Мінімум 2 ГБ;
- версія ОС: Windows 10 або новіша, Linux, MacOS 10 або новіша;
- інтернет-підключення: Для стабільного підключення до вебзастосунку.

4.2 Характеристики програми

Основні характеристики вебзастосунку:

а) користувацька аутентифікація та особистий кабінет: користувачі мають можливість зареєструвати новий обліковий запис. Авторизовані користувачі отримують доступ до особистого кабінету, де можуть зберігати та редагувати особисту інформацію, а також переглядати історію своїх замовлень;

б) перегляд та управління товарами: усі користувачі можуть переглядати каталог з можливістю фільтрації. Детальна інформація про кожен товар включає фотографії, опис, артикуль та ціну;

в) кошик та оформлення замовлення: користувачі можуть додавати обрані товари до кошика та оформлювати замовлення, вказуючи необхідні дані для доставки та оплати;

г) обрані товари: користувачі мають можливість додавати товари до списку обраного (фаворитів) для швидкого доступу та перегляду у будь-який момент;

г) адмін-панель для адміністратора: адміністратор має окремий доступ до панелі управління, де може:

- 1) додавати, редагувати та видаляти категорії товарів;
 - 2) додавати нові товари, редагувати їх опис, ціни та доступність;
 - 3) управляти обліковими записами користувачів;
 - 4) переглядати та обробляти замовлення клієнтів;
- д) налаштування облікового запису: користувачі можуть редагувати свої особисті дані, контактну інформацію та змінювати пароль у налаштуваннях.

4.3 Звертання до програми

Для доступу до онлайн магазину відкрийте сайт у будь-якому зручному браузері. Введіть адресу магазину у адресній строці та натисніть клавішу "Enter" або перейдіть за відповідним посиланням. Для забезпечення стабільної та коректної роботи сайту необхідне надійне підключення до мережі Інтернет.

4.4 Вхідні й вихідні дані

Початкові дані:

- електронна пошта: для реєстрації користувач повинен вказати свою електронну адресу, яка використовується для входу в систему;
- пароль: користувач встановлює пароль для захисту свого облікового запису;
- ім'я та прізвище: потрібно ім'я та прізвище користувача для створення повноцінного профілю.

Вихідні дані:

- профільна Інформація: після реєстрації користувач може додати або змінити інформацію в своєму профілі, таку як номер телефону, адрес тощо;

- історія замовлень: система зберігає інформацію про всі оформлені замовлення користувача, що доступна для перегляду у кабінеті;
- список обраного: користувач має можливість додавати товари до списку фаворитів для подальшого перегляду та зручного доступу.

4.5 Повідомлення

4.5.1 Мета вебзастосунку

Вебзастосунок призначений для зручного та ефективного процесу онлайн-купівлі одягу. Користувачі можуть переглядати товари, додавати їх до кошика, оформлювати замовлення, а також зберігати улюблені позиції.

4.5.2 Основні функції

Основні функції вебзастосунку:

- реєстрація та вхід в обліковий запис користувача;
- перегляд та фільтрація товарів за різними категоріями та параметрами;
- додавання товарів до кошика та оформлення замовлення;
- збереження улюблених товарів;
- перегляд історії замовлень;
- налаштування профілю користувача;
- адміністративна панель для управління товарами, категоріями, користувачами та замовленнями.

4.5.3 Технічні вимоги

Технічні вимоги до вебзастосунку:

- вебзастосунок розроблений з використанням технологій: Nest.js [7], Node [8] для серверної частини, Next.js [5] та TypeScript [6] для клієнтської частини;

- для коректної роботи вебзастосунку необхідне стабільне підключення до Інтернету.

4.5.4 Безпека та конфіденційність

Вебзастосунок забезпечує захист персональних даних користувачів відповідно до вимог законодавства. Використовується шифрування даних та сучасні методи захисту від кібератак і несанкціонованого доступу.

4.6 Виконання програми

Для успішного функціонування вебзастосунку необхідно:

- хостинг та сервер: надійний хостинг для розміщення сайту, а також налаштований вебсервер для обробки запитів від користувачів (перегляд товарів, оформлення замовлень, управління профілем);
- безпека даних: впровадження механізмів захисту від атак (наприклад, DDoS), шифрування персональних даних користувачів, а також забезпечення конфіденційності та цілісності інформації;
- база даних: база даних використовується для зберігання інформації про користувачів, товари, замовлення, історію покупок, обране тощо;
- інтерфейс користувача: розробка зручного, інтуїтивно зрозумілого та адаптивного інтерфейсу для комфортного перегляду товарів, фільтрації, додавання у кошик та оформлення замовлень;
- тестування та налагодження: проведення комплексного тестування сайту, включаючи функціональне, модульне та навантажувальне тестування, з подальшим налагодженням перед запуском нових версій;
- стабільність та продуктивність: Забезпечення стабільної роботи вебзастосунку навіть за високого навантаження, під час акцій, розпродажів та великої кількості одночасних замовлень.

Для доступу до нашого сайту достатньо відкрити будь-який сучасний веббраузер, ввести адресу сайту у рядок пошуку та натиснути "Enter". Важливо: для коректної роботи вебзастосунку необхідне стабільне підключення до Інтернету.

На рисунках 4.1, 4.2, 4.3, 4.4 зображено сторінки вебзастосунку.

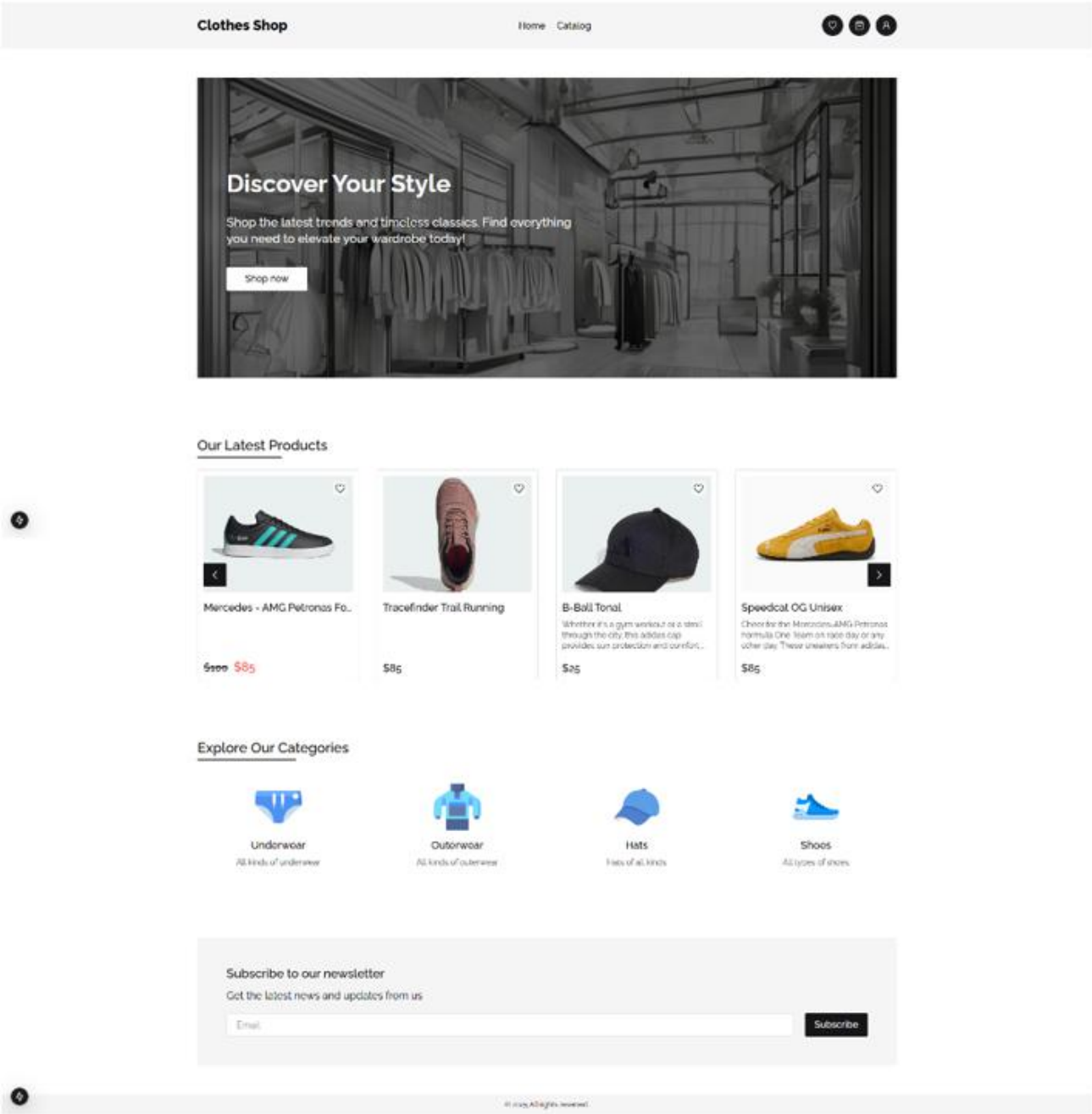


Рисунок 4.1 – Головна сторінка вебзастосунку

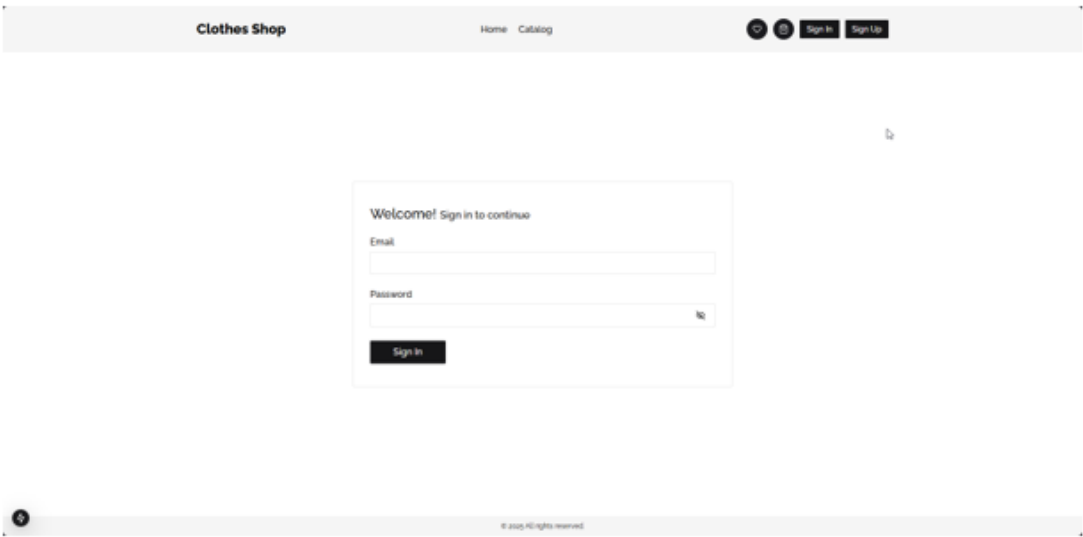


Рисунок 4.2 – Сторінка авторизації

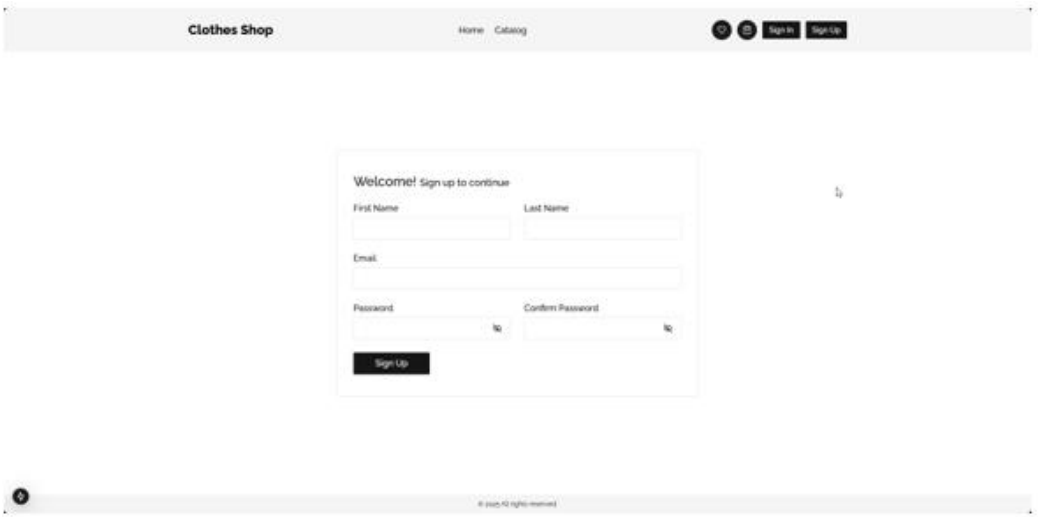


Рисунок 4.3 – Сторінка реєстрації

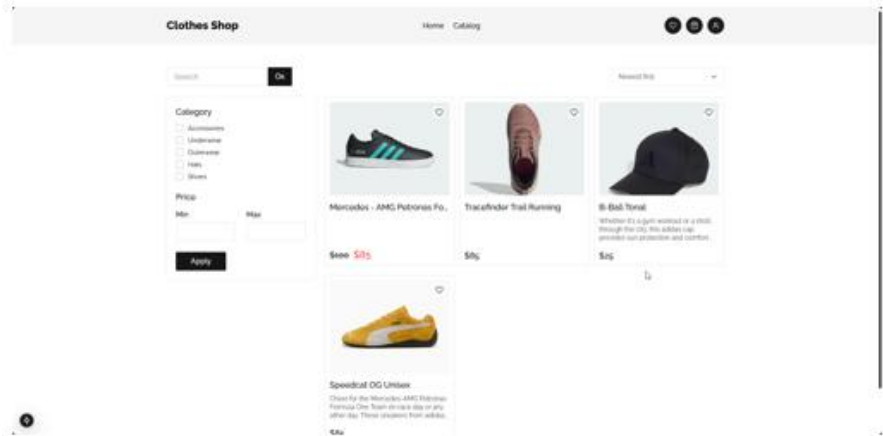


Рисунок 4.4 – Сторінка каталогу

На рисунках 4.5, 4.6, 4.7, 4.8, 4.9, 4.10 зображено сторінки вебзастосунку.

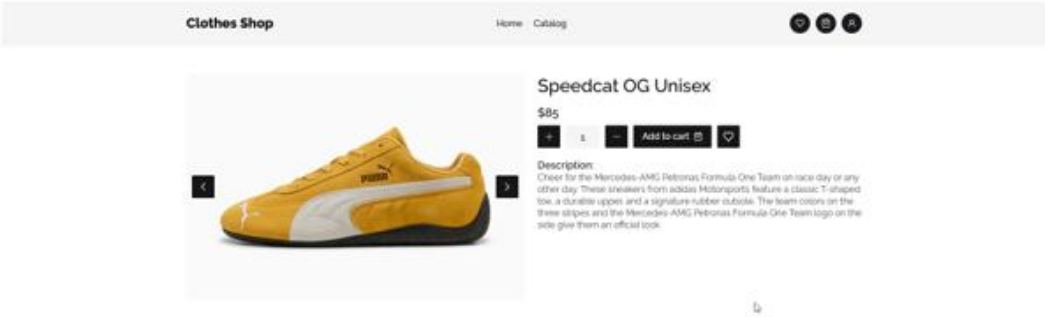


Рисунок 4.5 – Сторінка товару

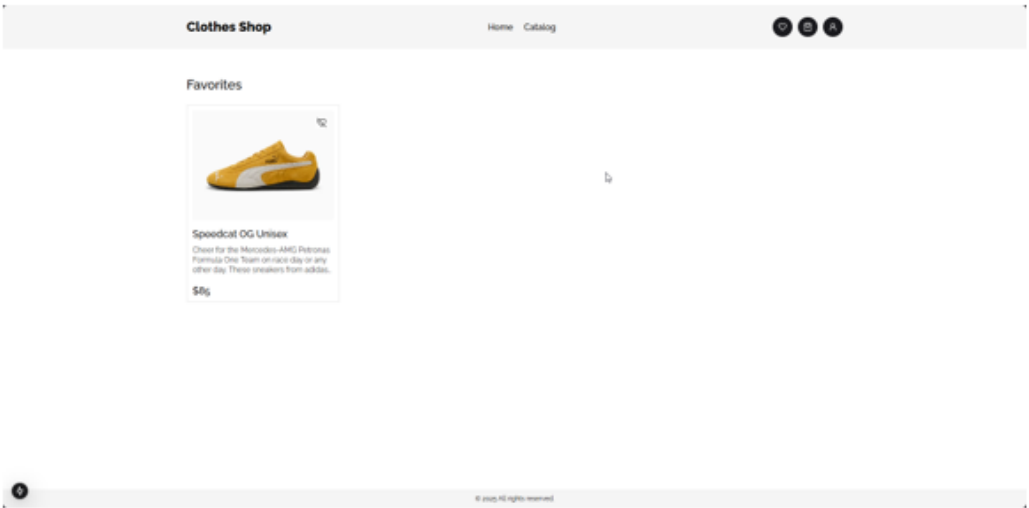


Рисунок 4.6 – Сторінка збережених товарів

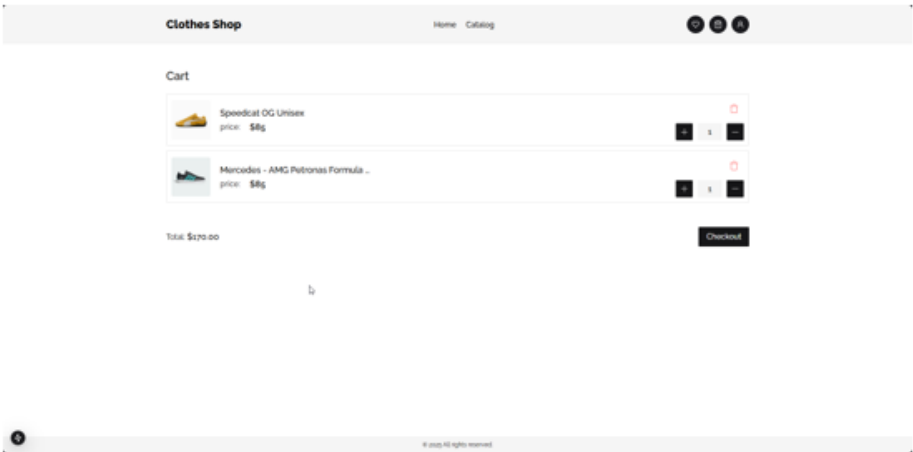


Рисунок 4.7 – Сторінка кошику товарів

Clothes Shop Home Catalog

Checkout

First name: Admin Last name: Admin Email: admin@gmail.com Phone number:
 Country: State: City:
 Street: House: Flat: Floor: Zip:
 Total: \$670.00

Products

Speedcat OG Unisex	\$61 x 1
Mercedes - AMG Petronas F1	\$6

Confirm Order

© 2020 All rights reserved

Рисунок 4.8 – Сторінка оформлення замовлення

Clothes Shop Home Catalog

Profile

My account

Name: Admin Email: Admin Phone number:
Save

My address

Country: State: City:
 Street: House: Flat: Floor: Zip:
Save

My password

Logout

© 2020 All rights reserved

Рисунок 4.9 – Сторінка аккаунту

Clothes Shop Home Catalog

History

Your checkout

First name: Admin Last name: Admin Email: admin@gmail.com Phone number: 63 543 5435
 Country: Ukraine State: Zaporizha obl City: Zaporizha
 Street: Dnipro Mygorodskogo Zip: 6700 House: 21
 Total: \$ 670

Mercedes - AMG Petronas Formula One Team V1 Court Base	\$ 1 X \$6
Speedcat OG Unisex	\$ 1 X \$6

Logout

© 2020 All rights reserved

Рисунок 4.10 – Сторінка історії замовлень

На рисунках 4.11, 4.12, 4.13, 4.14, 4.15, зображено сторінки вебзастосунку.

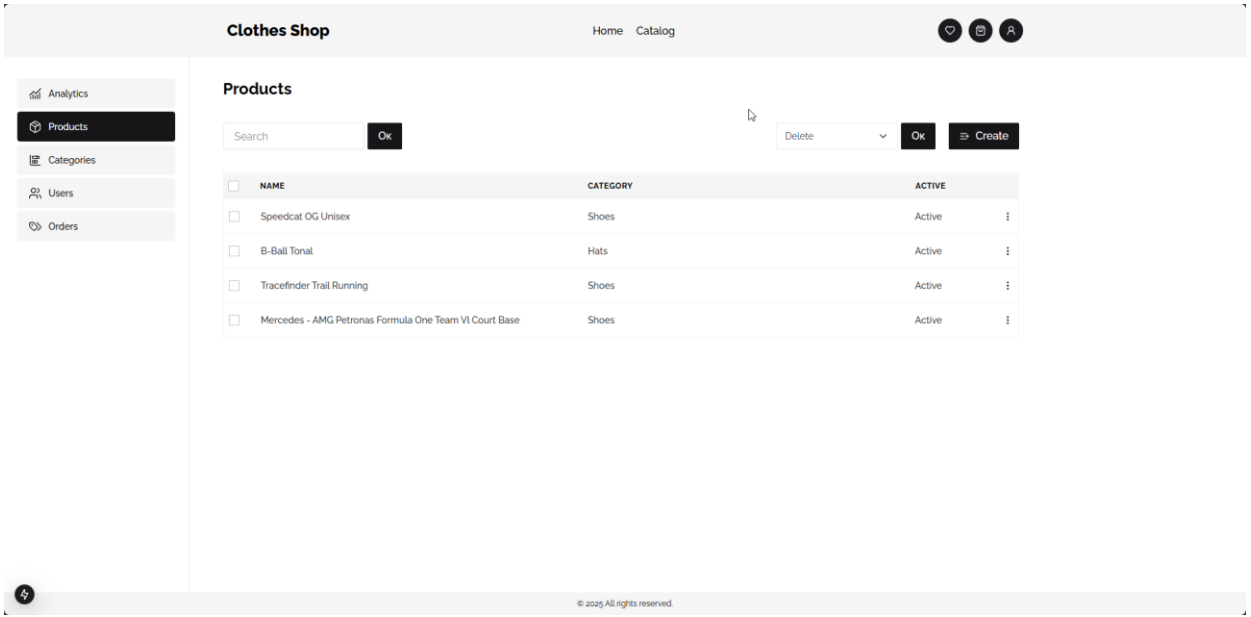


Рисунок 4.11 – Сторінка продуктів в адмін панелі

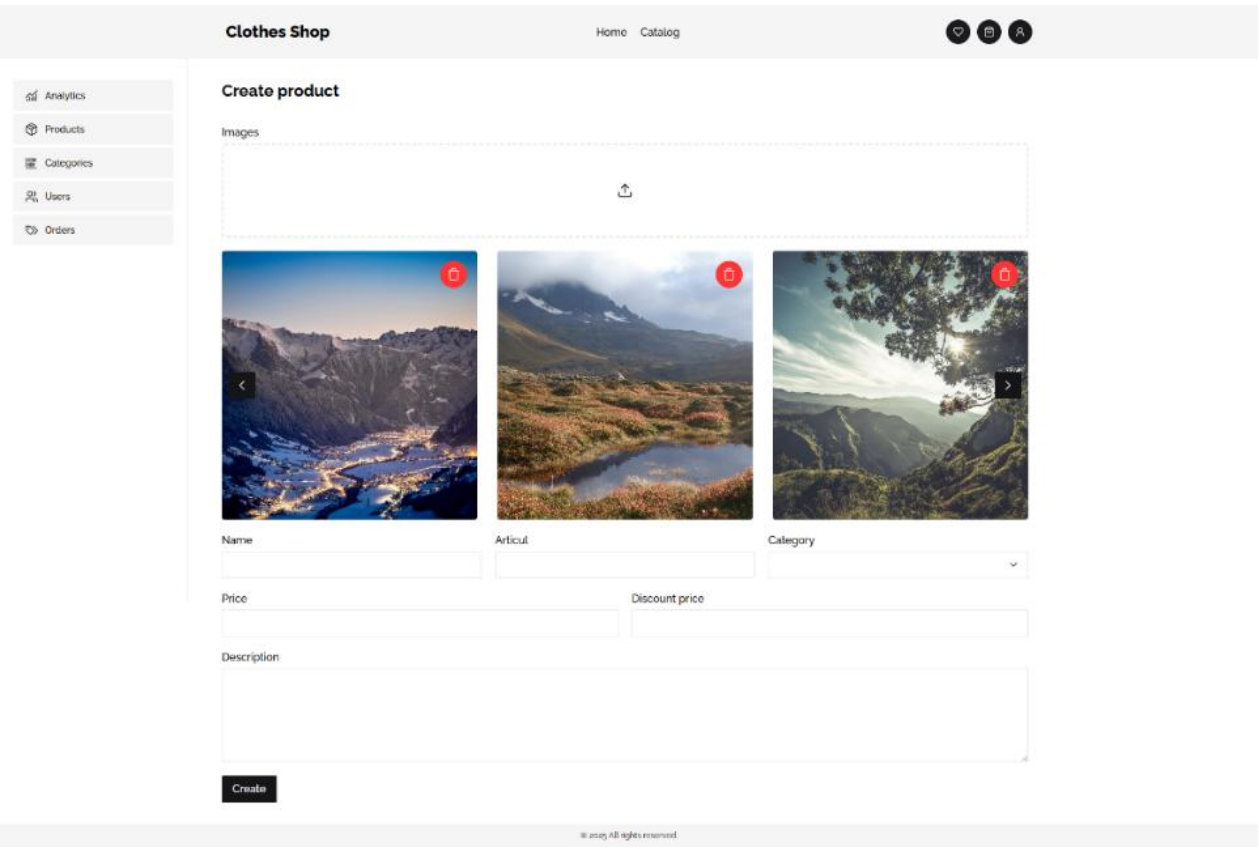


Рисунок 4.12 – Сторінка створення продукту в адмін панелі

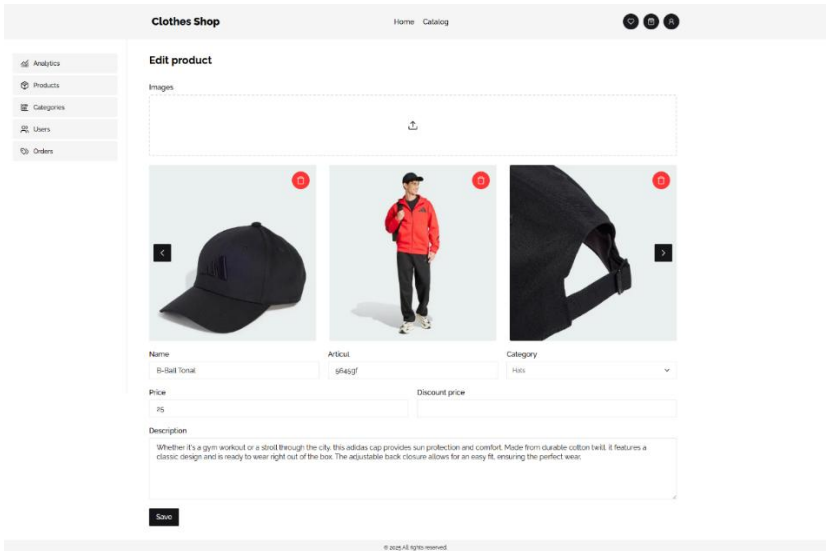


Рисунок 4.13 – Сторінка редагування продукту в адмін панелі

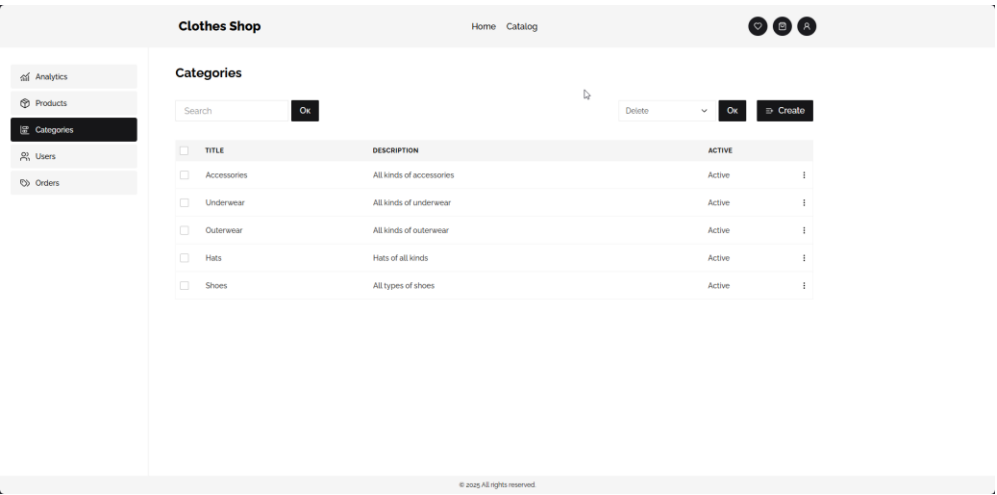


Рисунок 4.14 – Сторінка категорій в адмін панелі

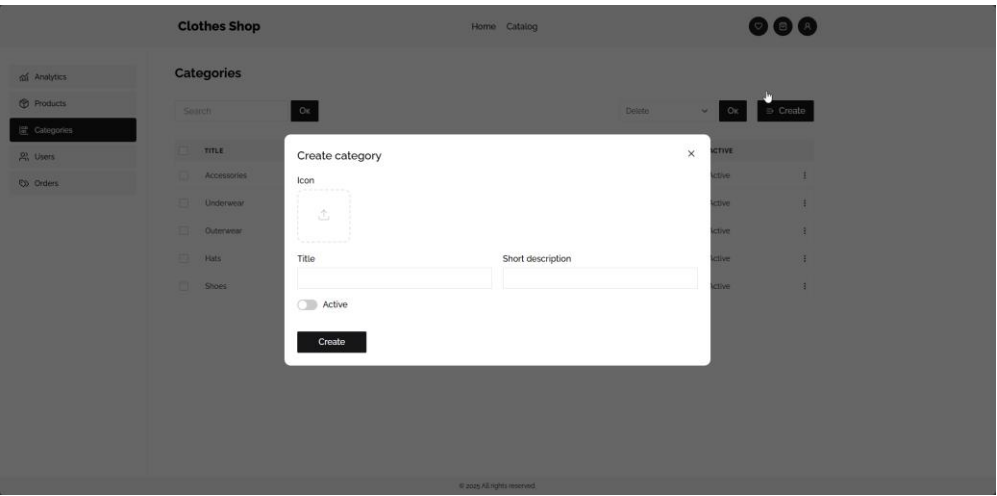


Рисунок 4.15 – Попап створення категорії в адмін панелі

На рисунках 4.16, 4.17, 4.18, 4.19, 4.20 зображено сторінки вебзастосунку.

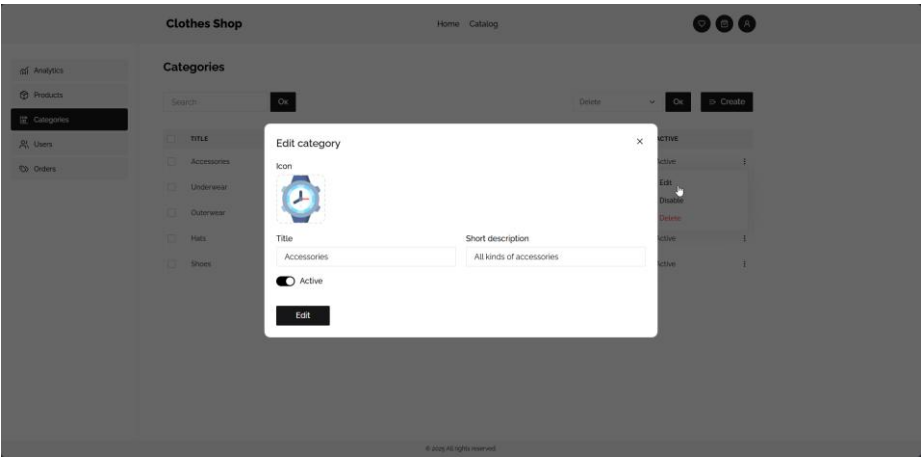


Рисунок 4.16 – Попап редагування категорії в адмін панелі

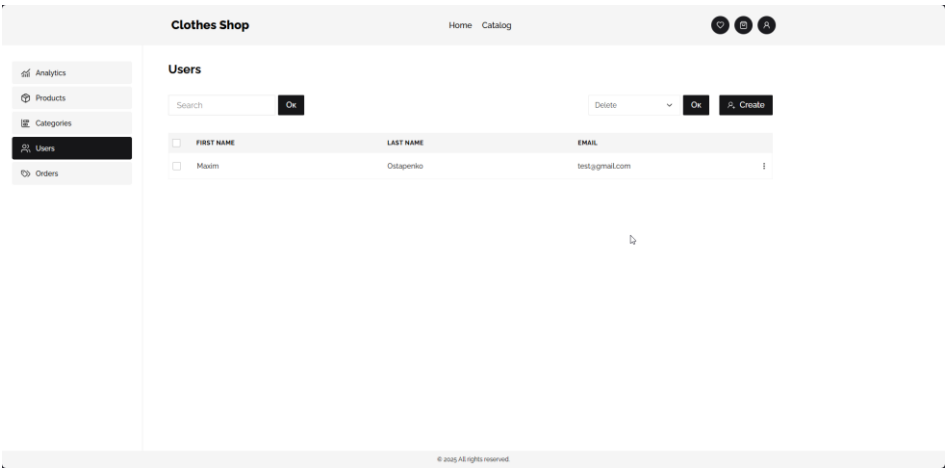


Рисунок 4.17 – Сторінка користувачів в адмін панелі

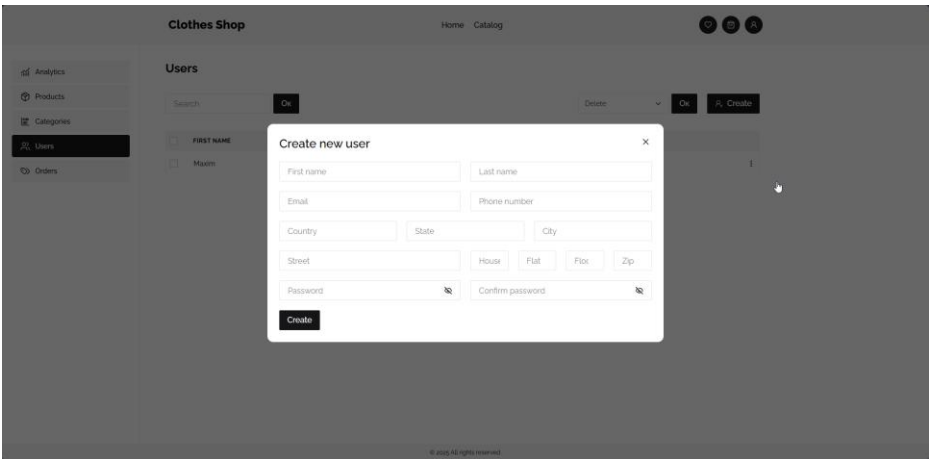


Рисунок 4.18 – Попап редагування користувача в адмін панелі

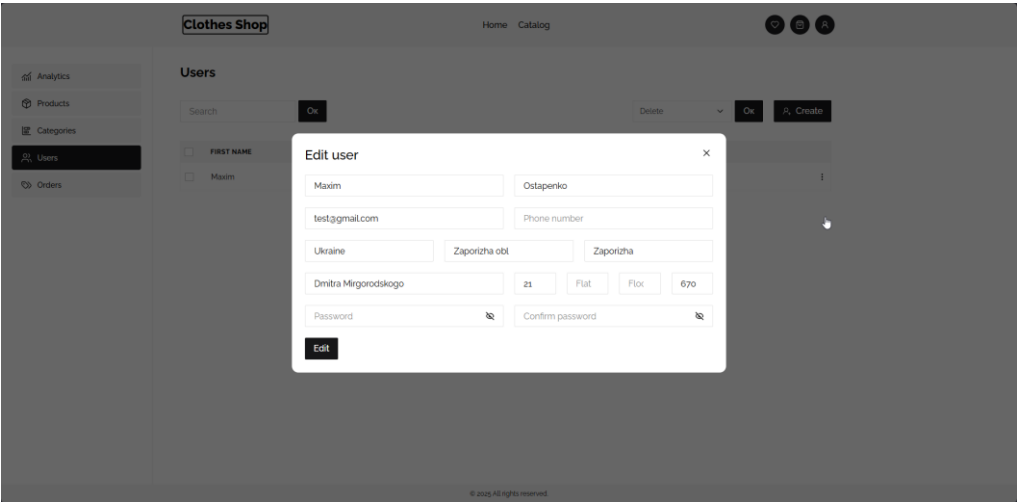


Рисунок 4.19 – Попап редагування користувача в адмін панелі

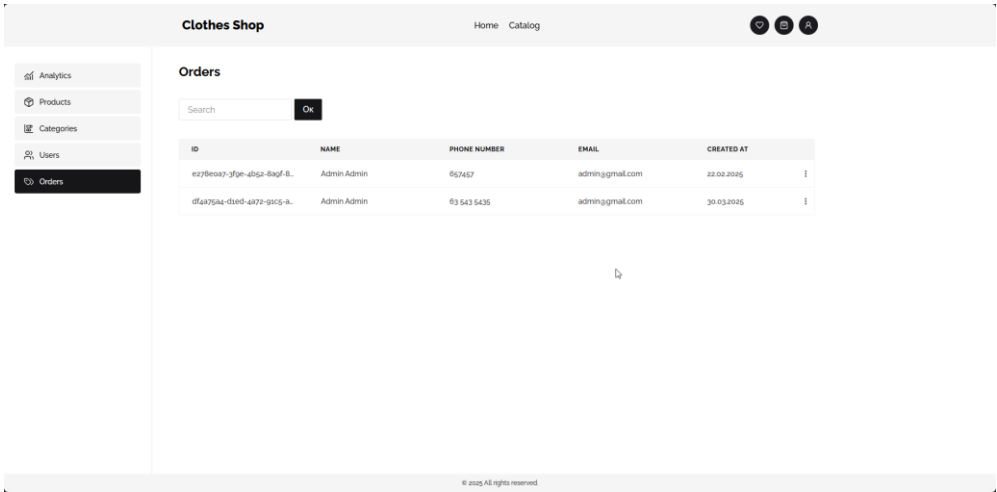


Рисунок 4.20 – Сторінка замовлень в адмін панелі

5 ТЕСТУВАННЯ ТА РЕЗУЛЬТАТИ

5.1 Опис методики тестування

Під час процесу розробки вебзастосунку було проведено систематичне тестування, включаючи unit-тестування всього коду. Це сприяло виявленню і усуненню значної кількості багів і помилок. Такий підхід дозволив досягти мінімального рівня виявлених дефектів та помилок на момент випуску програмного продукту.

5.2 Результати використання розробленого вебзастосунку

Наведено проведені тести у таблиці 5.1.

Таблиця 5.1 – Результати тестування вебзастосунку

Назва тесту	Статус	Результат	Очікуваний результат
Реєстрація користувача	Успішно	Користувач зареєстрований.	Новий користувач
Авторизація користувача	Успішно	Користувач авторизований	Авторизований користувач
Додавання в кошик товару	Успішно	Товар додався в кошик	Доданий товарн в кошику
Видалення товару з кошику	Успішно	Товар видалився з кошику	Видалення товару з кошику
Додавання товару до списку бажаного	Успішно	Товар додався до списку бажаного	Товар додано до списку бажаного
Видалення користувача через адмін панель	Успішно	Створено нового користувача	Користувача створено

Продовження таблиці 5.1

Назва тесту	Статус	Результат	Очікуваний результат
Оформлення замовлення	Успішно	Замовлення було успішно створене	Замовлення оформлене
Зміна паролю користувача	Успішно	Користувач має новий пароль	Користувач має новий пароль
Зміна ім'я користувача	Успішно	Нове ім'я користувача	Користувач має нове ім'я
Зміна адреси користувача	Успішно	Користувач має нову адресу	Користувач має нову адресу
Створення категорії товарів	Успішно	Створена нова категорія товарів	Створена нова категорія товарів
Редагування категорії товарів	Успішно	Категорія товарів була відредагована	Відредагована категорія товарів
Видалення категорії товарів	Успішно	Категорія товарів була видалена, товари відв'язались від цієї категорії	Категорія товарів видалена
Створення товару	Успішно	Товар успішно створено	Створено новий товар
Редагування товару	Успішно	Товар успішно відредаговано	Товар відредаговано
Видалення товару	Успішно	Товар успішно видалено	Товар видалено
Редагування користувача через адмінку	Успішно	Дані користувача змінились	Користувач відредагований

Після проведення тестування вебзастосунку було успішно підтверджено, що всі юніт-тести пройшли без проблем. Усі компоненти програми працювали правильно та згідно з очікуваннями. Результати тестування підтвердили високу якість розробленого вебзастосунку. Це свідчить про те, що ключовий функціонал додатку функціонує бездоганно.

ВИСНОВКИ

Під час виконання кваліфікаційної роботи було проведено детальне дослідження предметної області – програмного забезпечення для онлайн магазину сучасного одягу. У процесі роботи над першим розділом здійснено аналіз цієї області з метою подальшої розробки вебзастосунку для онлайн магазину. Під час аналізу були вивчені існуючі аналоги вебзастосунків, їх переваги та недоліки, а також проведено порівняння. На основі отриманих результатів було визначено функціональні вимоги до нового вебзастосунку.

У другому розділі було обрано інструменти для розробки застосунку. Було порівняно актуальні мови програмування та середовища розробки, наведено аргументи на користь вибору конкретних засобів. Також була створена структурна схема застосунку, а також описано розроблені модулі.

Після аналізу можливих інструментів для розробки, вебзастосунок було реалізовано з використанням Nest.js [7] і Node.js [8] для бекенду, Next.js [5] з TypeScript [6] для створення графічного інтерфейсу, а для роботи з базою даних було обрано PostgreSQL [9].

У третьому розділі було наведено схему та детально описано розроблені алгоритми. Після проведеного аналізу був визначений шаблон проєктування вебзастосунку та визначено сутності з полями для створюваної бази даних.

В результаті було розроблено вебзастосунок, який виконує всі поставлені завдання відповідно до технічного завдання, є відкритим для розширення функціональності та забезпечує безпеку даних і високу швидкодію всіх операцій. Тестування підтвердило стабільність програмного продукту, точне й швидке виконання всіх завдань та підтвердило необхідність його подальшої підтримки та розвитку.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Shopify [Electronic resource]. – Access mode: <https://www.shopify.com/>.
2. WooCommerce [Electronic resource]. – Access mode: <https://woocommerce.com/>.
3. ASOS [Electronic resource]. – Access mode: <https://www.asos.com/>.
4. Zalando [Electronic resource]. – Access mode: <https://zalando.com/>.
5. Next.js [Electronic resource]. – Access mode: <https://nextjs.org/docs>.
6. TypeScript [Electronic resource]. – Access mode: <https://www.typescriptlang.org/>.
7. Nest.js [Electronic resource]. – Access mode: <https://docs.nestjs.com/>.
8. Node [Electronic resource]. – Access mode: <https://nodejs.org/docs/latest/api/>.
9. PostgreSQL [Electronic resource]. – Access mode: <https://www.postgresql.org/docs/>.
10. SSR [Electronic resource]. – Access mode: https://cio-wiki.org/wiki/Server-Side_Rendering.
11. SSG [Electronic resource]. – Access mode: https://en.wikipedia.org/wiki/Static_site_generator.
12. Search engine optimization (SEO) [Electronic resource]. – Access mode: https://en.wikipedia.org/wiki/Search_engine_optimization.
13. SOLID [Electronic resource]. – Access mode: <https://en.wikipedia.org/wiki/SOLID>.
14. REST [Electronic resource]. – Access mode: <https://en.wikipedia.org/wiki/REST>.
15. GraphQL [Electronic resource]. – Access mode: <https://graphql.org/>.
16. VS Code [Electronic resource]. – Access mode: <https://code.visualstudio.com/>.
17. Git [Electronic resource]. – Access mode: <https://git-scm.com/>.

ДОДАТОК А
Технічне завдання

Вступ

Найменування: Програмне забезпечення онлайн магазину сучасного одягу.

Розвиток електронної комерції та зростання популярності онлайн-покупок вимагають створення швидких, зручних та ефективних вебзастосунків. Сучасні вебтехнології дають змогу будувати масштабовані рішення, які забезпечують високу продуктивність, надійність та зручність для користувачів.

Проблема розробки оптимізованого онлайн магазину полягає у виборі правильних архітектурних рішень, використанні сучасних технологій та забезпеченні високої конверсії. Зважаючи на зростаючі вимоги до швидкості завантаження сторінок, безпеки даних та масштабованості, ключовим викликом стає впровадження ефективної архітектури для обробки великої кількості користувачів та транзакцій.

Метою даної роботи є розробка сучасного вебзастосунку для онлайн магазину одягу, який використовує новітні технології, зокрема React.js, Next.js, Node.js, Nest.js та PostgreSQL. Це дозволяє реалізувати оптимізований механізм рендерингу (SSR), покращити продуктивність та створити зручний інтерфейс для користувачів.

Результатом роботи є повноцінний вебзастосунок, який дозволяє здійснювати покупки онлайн, керувати асортиментом товарів, обробляти замовлення та забезпечувати безперебійну роботу сервісу навіть під великим навантаженням. Проведені тести підтверджують, що використані технології сприяють зменшенню часу завантаження сторінок, підвищенню швидкості роботи онлайн магазину та покращенню загального користувацького досвіду.

Завдяки впровадженню сучасних рішень у сфері веброзробки, створене застосування дозволяє оптимізувати процес онлайн-продажів та підвищити конверсію магазину, що є ключовою метою будь-якого e-commerce бізнесу.

A.1 Назва програми

Назва додатка: Clothes shop.

Вибір назви "Clothes Shop" зумовлений кількома факторами. По-перше, слово "Clothes" чітко вказує на основний напрям застосунку – можливість купувати одяг. Це допомагає користувачам зрозуміти призначення програми ще до її відкриття.

По-друге, додавання слова "Shop" підкреслює, що йдеться саме про торговий майданчик. Такий підхід робить назву простою, зрозумілою та легкою для запам'ятовування, водночас акцентуючи увагу на головному функціоналі додатка.

A.2 Призначення та область застосування

Вебзастосунок "Clothes Shop" призначений для зручного та ефективного онлайн-шопінгу. Використовуючи цей застосунок, користувачі можуть легко знаходити та купувати стильний одяг, що відповідає їхнім уподобанням та бюджету.

Цільова аудиторія – широкий спектр покупців: модники, шукачі трендових речей, люди, які оновлюють свій гардероб, а також ті, хто шукає якісний одяг за вигідними цінами.

"Clothes Shop" пропонує користувачам великий вибір одягу для різних стилів і сезонів, дозволяючи швидко знаходити необхідні речі завдяки зручній навігації та фільтрам. Додаток забезпечує простий процес оформлення замовлення, створюючи комфортний і приємний досвід покупок.

A.3 Підстави для розробки

Основою для проведення розробки є виконання кваліфікаційної роботи для спеціальності "Інженерія програмного забезпечення". Завдання

узгоджено з професором кафедри програмних засобів А.О.Олійником, далі іменованим замовником, і затверджений студентом групи КНТ-151 М.В.Остапенко, далі іменованим виконавцем.

А.4 Призначення розробки

Призначення розробки полягає у створенні цифрової платформи, яка спростить та оптимізує процес вибору й придбання одягу в інтернет-магазині. Головною метою є забезпечення зручності та ефективності для двох основних категорій користувачів:

- покупці, які шукають стильний і якісний одяг;
- продавці, які розміщують свої товари.

Розробка "Clothes Shop" спрямована на полегшення взаємодії між продавцями та покупцями, забезпечуючи зручний, надійний і безпечний процес онлайн-шопінгу.

А.5 Вимоги до програми

А.5.1 Вимоги до функціональних характеристик

Функціональні вимоги до вебзастосунку включають:

- а) реєстрація та авторизація користувачів;
- б) перегляд каталогу товарів з можливістю фільтрації та сортування;
- в) оформлення замовлення;
- г) додавання товарів до обраного;
- г) особистий кабінет користувача (редагування профілю, перегляд історії замовлень);
- д) адміністративна панель для керування контентом магазину, яка включає:

- 1) створення, редагування та видалення категорій товарів;
- 2) додавання, редагування та видалення товарів;

- 3) управління замовленнями;
- 4) керування користувачами.

A.5.2 Вимоги до надійності

Вебзастосунок повинен забезпечувати стабільну та безперебійну роботу навіть в умовах високого навантаження або обмеженої мережевої доступності. Це означає, що система повинна мати високу стійкість до витоків пам'яті, помилок програмного забезпечення та відмов апаратного забезпечення.

Повинна бути реалізована ефективна система резервного копіювання та відновлення даних, щоб уникнути втрати інформації у випадку аварій або виходу з ладу обладнання.

A.5.3 Вимоги до технічних та програмних засобів

Вебзастосунок має вимоги до технічних та програмних засобів, щоб забезпечити ефективну та стабільну роботу. Він потребує сучасних вебсерверів і баз даних, які здатні обробляти великий потік запитів від користувачів.

Для забезпечення безпеки даних і захисту від зловживань програма потребує використання сучасних методів шифрування і аутентифікації. Вона також повинна мати механізми контролю доступу до даних і захисту від вразливостей у програмному забезпеченні.

Крім того, програма повинна бути сумісною з різними операційними системами і веббраузерами, щоб забезпечити максимальну доступність для користувачів. Наявність резервних копій даних і механізмів відновлення в разі виникнення проблем також є необхідною складовою для забезпечення надійності програми.

Системні вимоги:

- процесор: Ryzen 3 1300x або вище;
- оперативна пам'ять: Мінімум 2 ГБ;
- версія ОС: Windows 10 або новіша, Linux, MacOS 10 або новіша;
- інтернет-підключення: Для стабільного підключення до вебзастосунку.

A.6 Вимоги до програмної документації

Програмна документація повина включати в себе:

- титульний лист;
- завдання;
- реферат;
- зміст;
- перелік умовних позначень, символів, одиниць, скорочень і термінів;
- вступ;
- основна частина;
- висновки;
- перелік посилань;
- додатки.

A.7 Стадії і етапи розробки

A.7.1 Стадії розробки

Стадії розробки включають наступні етапи:

- аналіз вимог: Збирання та уточнення вимог до вебзастосунку, визначення функціональних та нефункціональних вимог, а також аналіз потреб користувачів;
- проєктування: Розроблення архітектури вебзастосунку, вибір технологій та інструментів розробки, створення технічних специфікацій;

- розробка: Написання коду вебзастосунку відповідно до вимог та специфікацій, розробка функціональних модулів та інтерфейсів користувача;
- тестування: Проведення тестування вебзастосунку для перевірки відповідності вимогам, виявлення та виправлення помилок;
- впровадження: Підготовка вебзастосунку до впровадження в робоче середовище, налаштування та конфігурація програми;
- підтримка: Надання підтримки та обслуговування після впровадження вебзастосунку, включаючи виправлення помилок та підтримку користувачів.

A.7.2 Етапи розробки

Розробка буде включати наступні етапи:

- Frontend розробка;
- Backend розробка;
- Unit-тестування;
- UI-тестування.

A.7.3 Зміст робіт по етапах

Зміст робіт по етапах буде включати наступні пункти:

а) аналіз вимог:

- 1) збір та аналіз вимог користувачів;
- 2) визначення функціональних та нефункціональних вимог;
- 3) створення документації, що описує вимоги;

б) проєктування:

- 1) розробка архітектури програмного забезпечення;
- 2) проєктування інтерфейсів користувача;
- 3) визначення структури бази даних;

в) розробка:

- 1) написання програмного коду відповідно до вимог та проєктної документації;

- 2) тестування окремих компонентів та модулів;

г) тестування:

- 1) виконання функціональних та нефункціональних тестів;

- 2) виявлення та виправлення помилок (багів);

- 3) верифікація відповідності програми вимогам;

г) впровадження:

- 1) встановлення та налаштування програмного забезпечення на середовище реальної експлуатації;

- 2) підготовка користувачів до роботи з програмним продуктом;

- 3) запуск продукту в експлуатацію;

д) підтримка та розвиток:

- 1) надання технічної підтримки користувачам;

- 2) виявлення та виправлення помилок після впровадження;

- 3) вдосконалення та розвиток програмного забезпечення на основі отриманих відгуків користувачів.

ДОДАТОК Б
Текст програми

Б.1 Файл AdminProducts.tsx

```

"use client";

import { useState } from "react";
import { useRouter } from "next/navigation";
import { useQueryClient } from "@tanstack/react-query";
import { toast } from "react-toastify";
import { ListPlus } from "lucide-react";

import { AdminProductsTable } from "@widgets";
import { GroupActionsSelect, SearchInput } from "@components";
import { Button, Pagination } from "@ui";

import { useDeleteProducts, useGetProducts, useToggleProducts } from
"@/apis";
import {
  ADMIN_CREATE_PRODUCTS_ROUTE,
  paginationLimitsList,
  productsGroupActionsList,
  SUCCESS_MESSAGE
} from "@shared";

import styles from "../AdminProducts.module.scss";

const AdminProducts = () => {
  const [activePage, setActivePage] = useState<number>(1);
  const [perPage, setPerPage] = useState<number>(paginationLimitsList
[0]);

  const [search, setSearch] = useState<string>("");

```

```

const [groupAction, setGroupAction] = useState<string>(
  productsGroupActionsList [0]?.value
);
const [selectedItems, setSelectedItems] = useState<string []>( []);

const router = useRouter();

const queryClient = useQueryClient();

const { data: productsData, isLoading: productsIsLoading } =
useGetProducts(
  activePage,
  perPage,
  search
);

const { mutate: deleteProducts } = useDeleteProducts({
  onSuccess: () => {
    toast.success("Product deleted successfully");
    queryClient.invalidateQueries({ queryKey: ["products"] });
  }
});

const { mutate: toggleProducts } = useToggleProducts({
  onSuccess: () => {
    toast.success(SUCCESS_MESSAGE);
    queryClient.invalidateQueries({ queryKey: ["products"] });
  }
});

const handleSelectAll = (isChecked: boolean) => {

```

```

    if (productsData) {
      if (isChecked) {
        setSelectedItems(productsData?.data.map((item) => item.id));
      } else {
        setSelectedItems( []);
      }
    }
  }
};

```

```

const handleSelectItem = (isChecked: boolean, id: string) => {
  setSelectedItems((prev) =>
    isChecked ? [...prev, id] : prev.filter((item) => item !== id)
  );
};

```

```

const handleApplyGroupAction = () => {
  if (groupAction === "delete") {
    deleteProducts(selectedItems);
  } else if (groupAction === "active") {
    toggleProducts({ isActive: true, ids: selectedItems });
  } else if (groupAction === "disable") {
    toggleProducts({ isActive: false, ids: selectedItems });
  }
};

```

```

const goCreate = () => {
  router.push(ADMIN_CREATE_PRODUCTS_ROUTE);
};

```

```

return (

```

```

<section className={styles.admin}>
  <div className="container">
    <div className={styles.admin__inner}>
      <h2 className={styles.admin__title}>Products</h2>
      <div className={styles.admin__row}>
        <div className={styles.admin__search}>
          <SearchInput onChange={setSearch} />
        </div>
        <div className={styles.admin__actions}>
          <div className={styles.admin__actions_select}>
            <GroupActionsSelect
              value={groupAction}
              onChange={setGroupAction}
              options={productsGroupActionsList}
              onSubmit={handleApplyGroupAction}
            />
          </div>
          <Button
            leftIcon={
              <ListPlus
                size={20}
                strokeWidth={1.5}
                absoluteStrokeWidth
              />
            }
            onClick={goCreate}
          >
            Create
          </Button>
        </div>
      </div>
    </div>
  </div>

```

```

</div>
{!productsIsLoading ? (
  <>
    <div className={styles.admin__table}>
      <AdminProductsTable
        data={productsData?.data || []}
        selectedItems={selectedItems}
        onSelect={handleSelectItem}
        onSelectAll={handleSelectAll}
        onDelete={deleteProducts}
        onToggle={toggleProducts}
      />
    </div>
    <div className={styles.admin__pagination}>
      <Pagination
        currentPage={activePage}
        onChange={setActivePage}
        perPage={perPage}
        onChangePerPage={setPerPage}
        limitsList={paginationLimitsList}
        totalItems={productsData?.totalItems || 0}
      />
    </div>
  </>
) : null}
</div>
</div>
</section>
);
};

```

```
export default AdminProducts;
```

Б.2 Файл AdminProductsTable.tsx

```
"use client";
```

```
import { useState } from "react";
```

```
import {
  createColumnHelper,
  flexRender,
  getCoreRowModel,
  useReactTable
} from "@tanstack/react-table";
```

```
import { AdminProductsTableRow } from "@widgets";
```

```
import { Checkbox } from "@ui";
```

```
import { IProduct, useMounted } from "@shared";
```

```
import { AdminProductsTableProps } from "../AdminProductsTable.types";
```

```
import styles from "../AdminProductsTable.module.scss";
```

```
const columnHelper = createColumnHelper<IProduct>();
```

```
const columns = [
  columnHelper.accessor("title", {
    header: () => <span>Name</span>,
    cell: (info) => info.getValue()
  }),
```

```

columnHelper.accessor("category.title", {
  header: () => <span>Category</span>,
  cell: (info) => info.getValue()
}),
columnHelper.accessor("isActive", {
  header: () => <span>Active</span>,
  cell: (info) => (info.getValue() ? "Active" : "Inactive")
})
];

const AdminProductsTable = ({
  data,
  selectedItems,
  onSelect,
  onSelectAll,
  onDelete,
  onToggle
}: AdminProductsTableProps) => {
  const [isAllSelected, setIsAllSelected] = useState<boolean>(false);

  const { mounted } = useMounted();

  const table = useReactTable({
    data: data,
    columns,
    getCoreRowModel: getCoreRowModel()
  });

  if (!mounted) return null;

```

```

const handleSelectAll = (isChecked: boolean) => {
  setIsAllSelected(isChecked);
  onSelectAll(isChecked);
};

const handleSelectItem = (isChecked: boolean, id: string) => {
  onSelect(isChecked, id);
  setIsAllSelected(data.length === selectedItems.length + 1 &&
isChecked);
};

return (
  <div className={styles.table}>
    <div className={styles.table__head}>
      <div className={styles.table__head_checkbox}>
        <Checkbox
          checked={isAllSelected}
          onChange={(e) => handleSelectAll(e.target.checked)}
        />
      </div>

      {table.getHeaderGroups().map((headerGroup) => (
        <div
          key={headerGroup.id}
          className={styles.table__head_row}
        >
          {headerGroup.headers.map((header) => (
            <p
              key={header.id}
              className={styles.table__head_item}

```

```

        >
        {flexRender(
            header.column.columnDef.header,
            header.getContext()
        )}
    </p>
    )}
</div>
)}

<div className={styles.table__head_dots} />
</div>
<div className={styles.table__body}>
    {table.getRowModel().rows.map((row) => (
        <AdminProductsTableRow
            key={row.original.id}
            row={row}
            selectedItems={selectedItems}
            handleSelectItem={handleSelectItem}
            onDelete={onDelete}
            onToggle={onToggle}
        />
    )}
</div>
</div>
);
};

export default AdminProductsTable;

```

ДОДАТОК В
Слайди презентації

Програмне забезпечення онлайн магазину сучасного одягу

Виконав:
ст. групи КНТ-151

Максим ОСТАПЕНКО

Керівник:
професор

Андрій ОЛІЙНИК

Рисунок В.1 – Слайд 1

Мета роботи, Об'єкт дослідження , Предмет дослідження

Метою даної роботи є розробка сучасного вебзастосунку для онлайн магазину одягу, який використовує новітні технології, зокрема React.js, Next.js, Node.js, Nest.js та PostgreSQL. Це дозволяє реалізувати оптимізований механізм рендерингу (SSR), покращити продуктивність та створити зручний інтерфейс для користувачів.

Об'єкт дослідження – Процес розробки програмного забезпечення інтернет-магазину одягу.

Предмет дослідження – програмні засоби для створення інтернет-магазину з максимальною конверсією.

2

Рисунок В.2 – Слайд 2

Задачі проєкту

- ◆ Аналіз ринку та цільової аудиторії
- ◆ Розробка структури та дизайну вебзастосунку
- ◆ Програмна розробка вебзастосунку
- ◆ Тестування платформи (функціональність, UX/UI, безпека)

3

Рисунок В.3 – Слайд 3

Функції системи

- ◆ реєстрація та авторизація користувачів
 - ◆ перегляд каталогу товарів з можливістю фільтрації та сортування
 - ◆ оформлення замовлення
 - ◆ додавання товарів до обраного
 - ◆ особистий кабінет користувача (редагування профілю, перегляд історії замовлень);
- адміністративна панель для керування контентом магазину, яка включає:
- ◆ створення, редагування та видалення категорій товарів
 - ◆ додавання, редагування та видалення товарів
 - ◆ управління замовленнями
 - ◆ керування користувачами

4

Рисунок В.4 – Слайд 4

Порівняння існуючих аналогів

Назва	ASOS	Zalando
Глобальна доставка	+	-
Широкий асортимент товарів	+	+
Велика кількість брендів	+	+
Розширена система фільтрації	-	+
Зручний інтерфейс	+	+
Тривалий термін повернення	-	+
Підписка на преміальні послуги	+	+
Гнучка система знижок	+	+
Різноманітність способів оплати	-	+
Оцінки продавців і товарів	-	+

5

Рисунок В.5 – Слайд 5

Порівняння мов програмування

Назва	JavaScript	TypeScript	PHP
Тип мови	Динамічно типізована	Статично типізована	Динамічно типізована
Строга перевірка типів	Ні	Так	Ні
Сумісність з ECMAScript	+	+	-
Швидкодія	Висока	Висока	Висока
Об'єктно-орієнтований підхід	+	+	+

6

Рисунок В.6 – Слайд 6

Порівняння середовищ розробки

Назва	VS Code	Visual Studio	WebStorm
Багатоплатформність	+	-	+
Швидкість модернізації	+	+	+
Наявність візуального конструктора	-	+	+
Підтримка репозиторіїв	+	+	+

7

Рисунок В.7 – Слайд 7

Порівняння баз даних

Назва	Microsoft SQL Server	MySQL	PostgreSQL
Сумісність з різними операційними системами	+	+	+
Підтримка розподілених операційних систем	+	+	+
Масштабованість	+	-	+
Спільнота та підтримка	-	+	+
Швидкодія	+	-	+

8

Рисунок В.8 – Слайд 8

Orders:

Управління замовленнями, включаючи створення, зміну статусу та обробку платежів.

Products:

Обробка товарів, їх характеристик, цін та наявності.

Sessions:

Робота з користувацькими сесіями, збереження стану автентифікації.

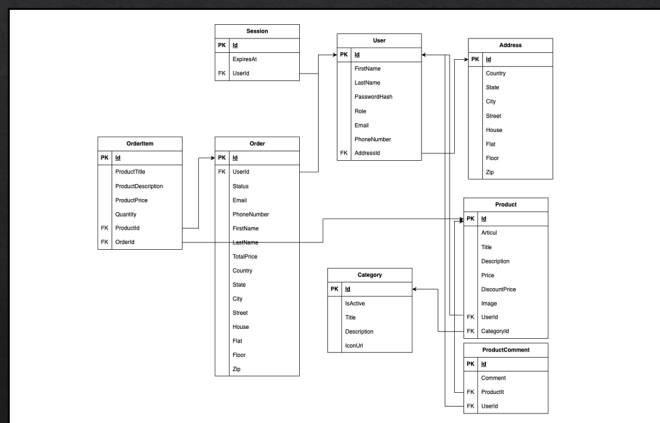
Users:

Управління користувачами, їхніми даними та правами доступу.

11

Рисунок В.11 – Слайд 11

Основні сутності бази даних



12

Рисунок В.12 – Слайд 12

Сторінки вебзастосунку

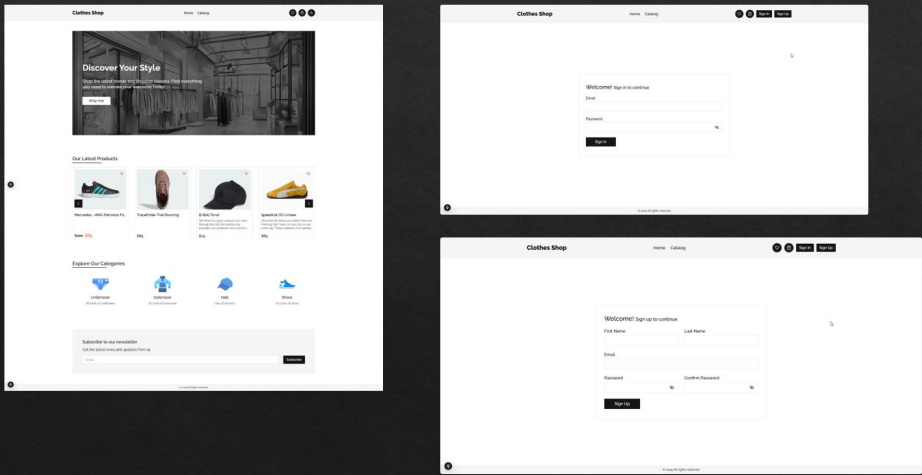


Рисунок В.13 – Слайд 13

Сторінки вебзастосунку

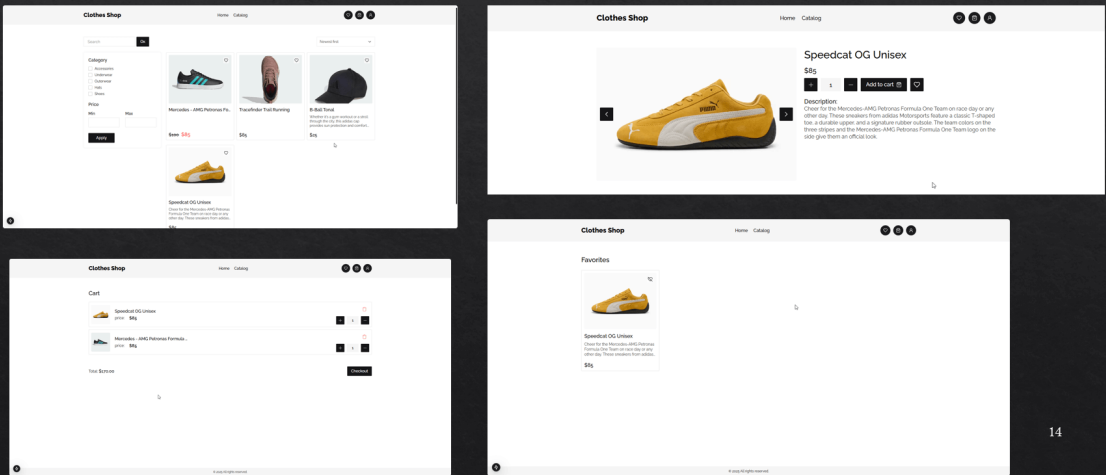


Рисунок В.14 – Слайд 14

Сторінки вебзастосунку

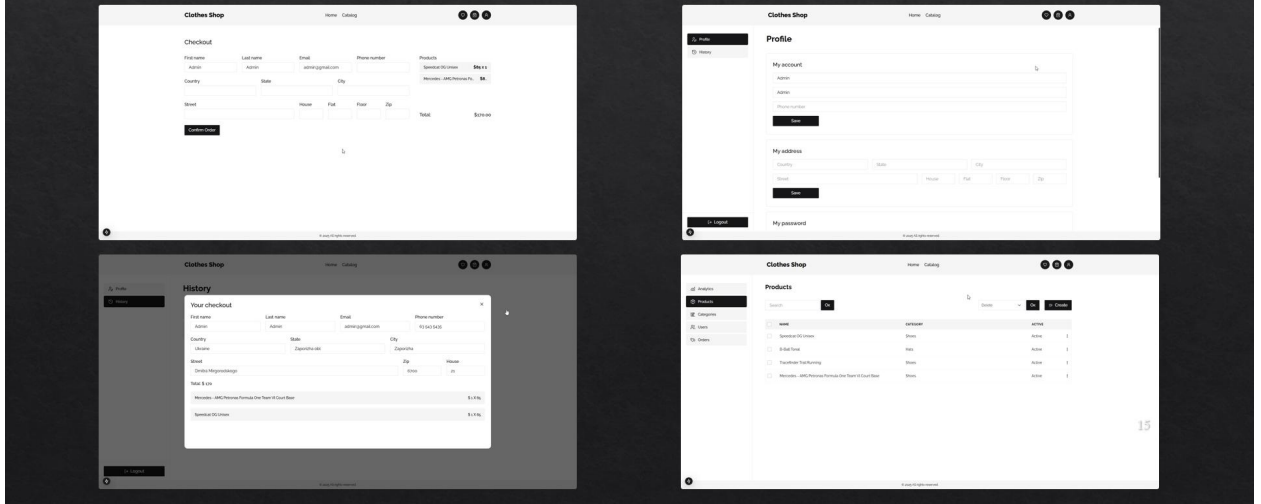


Рисунок В.15 – Слайд 15

Сторінки вебзастосунку

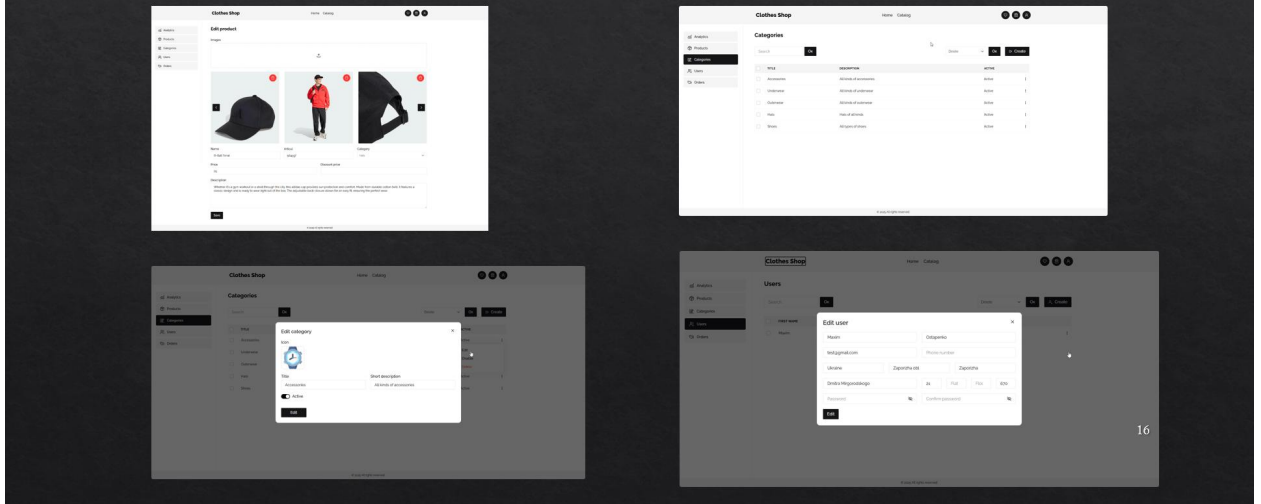


Рисунок В.16 – Слайд 16

Висновки

В результаті було розроблено вебзастосунок, який виконує всі поставлені завдання відповідно до технічного завдання, є відкритим для розширення функціональності та забезпечує безпеку даних і високу швидкодію всіх операцій. Тестування підтвердило стабільність програмного продукту, точне й швидке виконання всіх завдань та підтвердило необхідність його подальшої підтримки та розвитку.

17

Рисунок В.17 – Слайд 17