

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ДЕРЖАВНИЙ УНІВЕРСИТЕТ «ЖИТОМИРСЬКА ПОЛІТЕХНІКА»
ФАКУЛЬТЕТ ІНФОРМАЦІЙНО-КОМП'ЮТЕРНИХ ТЕХНОЛОГІЙ
КАФЕДРА ІНЖЕНЕРІЇ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

ПОЯСНЮВАЛЬНА ЗАПИСКА

до кваліфікаційної роботи освітнього ступеня «бакалавр»
за спеціальністю 121 «Інженерія програмного забезпечення»
(освітня програма «Інженерія програмного забезпечення»)

на тему:

«Вебплатформа для управління продажами нішевої парфумерії»

Виконав студент групи ІПЗ-22-2

КРАВЧЕНКО Катерина Максимівна

Керівник роботи:

ЄФРЕМОВ Юрій Миколайович

Рецензент:

ЧИЖМОТРЯ Олексій Володимирович

Житомир – 2026

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ДЕРЖАВНИЙ УНІВЕРСИТЕТ «ЖИТОМИРСЬКА ПОЛІТЕХНІКА»
ФАКУЛЬТЕТ ІНФОРМАЦІЙНО-КОМП'ЮТЕРНИХ ТЕХНОЛОГІЙ
КАФЕДРА ІНЖЕНЕРІЇ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

«ЗАТВЕРДЖУЮ»

Зав. каф. інженерії
програмного забезпечення
Тетяна ВАКАЛЮК
«28» лютого 2026 р.

ЗАВДАННЯ
на кваліфікаційну роботу

Здобувач вищої освіти: **КРАВЧЕНКО Катерина Максимівна**

Керівник роботи: **ЄФРЕМОВ Юрій Миколайович**

Тема роботи: **«Вебплатформа для управління продажами нішевої парфумерії»**

затверджена Наказом закладу вищої освіти від «27» лютого 2026 р., №92/с

Вихідні дані для роботи: **розробка системи управління продажами нішевої парфумерії, що оптимізує роботу його адміністрації і дозволить провести автоматизацію рутинних завдань та цифровізувати облік продукції.**

Консультанти з бакалаврської кваліфікаційної роботи із зазначенням розділів, що їх стосуються:

Розділ	Консультант	Завдання видав	Завдання прийняв
1	Єфремов Ю.М.	01.03.2026р.	01.03.2026р.
2	Єфремов Ю.М.	01.03.2026р.	01.03.2026р.
3	Єфремов Ю.М.	01.03.2026р.	01.03.2026р.

РЕФЕРАТ

Випускна кваліфікаційна робота бакалавра складається з програмного комплексу платформи для управління продажами нішевої парфумерії та пояснювальної записки. Пояснювальна записка до випускної роботи містить 72 сторінки, 23 ілюстрацій та 6 таблиць.

Головною метою є створення інформаційної системи, що оптимізує роботу персоналу та впровадить зручний інтерфейс користувачів, що полегшить процес користування та пошуку відповідних нішевих категорій продукції.

У ході виконання випускної кваліфікаційної роботи бакалавра були визначені основні завдання для розробки модулю, проведений аналіз аналогів інформаційної системи та обрано технології, що дозволять реалізувати низку вимог до програмного забезпечення.

КЛЮЧОВІ СЛОВА: ІНФОРМАЦІЙНА СИСТЕМА, ПАРФУМЕРІЯ, ОБЛІК, BACKEND, PYTHON, REACT.

					ІПЗ.КР.Б – 121 – 26 – ПЗ			
Змн.	Арк.	№ докум.	Підпис	Дата				
Розроб.		Кравченко К.М.			Вебплатформа для управління продажами нішевої парфумерії Пояснювальна записка	Літ.	Арк.	Аркушів
Керівник		Єфремов Ю.М.					3	75
						Житомирська політехніка, група ІПЗ-22-2		
Н. контр.								
Зав. каф.		Вакалюк Т.А.						

ABSTRACT

The final qualification work consists of a software complex of a platform for managing sales of niche perfumes and an explanatory note. The explanatory note to the graduation work contains 72 pages, 23 illustrations and 6 tables.

The main goal is to create an information system that optimizes the work of personnel and implements a convenient user interface that will facilitate the process of using and searching for relevant niche product categories.

During the final qualification work, the main tasks for developing the module were identified, an analysis of analogues of the information system was conducted, and technologies were selected that will allow implementing a number of requirements for software.

KEYWORDS: INFORMATION SYSTEM, PERFUMERY, ACCOUNTING, BACKEND, PYTHON, REACT.

		Кравченко К.М.			ІПЗ.КР.Б – 121 – 26– ІЗ	Арк.
		Єфремов Ю.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		4

ЗМІСТ

РЕФЕРАТ	3
ABSTRACT	4
ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ	7
ВСТУП.....	8
РОЗДІЛ 1. АНАЛІЗ ВИМОГ КОРИСТУВАЧА ТА КОНЦЕПТУАЛЬНЕ МОДЕЛЮВАННЯ.....	10
1.1. Постановка задачі	10
1.2. Аналіз аналогів програмного продукту.....	12
1.3. Вибір архітектури системи	16
1.4. Обґрунтування вибору інструментальних засобів та вимоги до апаратного забезпечення.....	20
Висновки до першого розділу	23
РОЗДІЛ 2. ПРОЕКТУВАННЯ ПЛАТФОРМИ ДЛЯ УПРАВЛІННЯ ПРОДАЖАМИ НІШЕВОЇ ПАРФУМЕРІЇ	25
2.1. Визначення варіантів використання та структури системи	25
2.2. Розробка бази даних системи	29
2.3. Проектування та реалізація алгоритмів роботи системи.....	34
2.4. Реалізація серверної та клієнтської логіки системи.....	40
Висновки до другого розділу	50
РОЗДІЛ 3. ІНТЕРФЕЙС ТА ПОРЯДОК РОБОТИ З ПЛАТФОРМОЮ ДЛЯ УПРАВЛІННЯ ПРОДАЖАМИ НІШЕВОЇ ПАРФУМЕРІЇ	52
3.1. Порядок встановлення та налаштування параметрів додатку	52
3.2. Структура інтерфейсу додатку.....	55
3.3. Тестування роботи розробленого додатку	64
Висновки до третього розділу	67

		Кравченко К.М.			ІПЗ.КР.Б – 121 – 26– ПЗ	Арк.
		Єфремов Ю.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		5

ВИСНОВКИ	69
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ.....	71
ДОДАТКИ	73
Додаток А	74
Додаток Б.....	74
Додаток В	76
Додаток Г	78
Додаток Д	80

		Кравченко К.М.			ІПЗ.КР.Б – 121 – 26– ІЗ	Арк.
		Єфремов Ю.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		6

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

БД – База даних

CRUD – Create-Read-Update-Delete

СУБД – Система управління базою даних

DOM – Document Object Model

ACID – Atomicity, Consistency, Isolation, Durability

XSS – Cross-Site Scripting

CSRF – Cross-Site Request Forgery

		Кравченко К.М.			ІПЗ.КР.Б – 121 – 26– ІЗ	Арк.
		Єфремов Ю.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		7

ВСТУП

Актуальність теми. В умовах стрімкого розвитку електронної комерції спостерігається чітка тенденція до переходу від масових ринкових рішень до високоперсоналізованих цифрових послуг. Нішева парфумерія становить особливий сегмент, якому притаманні ексклюзивність асортименту, висока вартість продукції та специфічні вимоги до взаємодії з клієнтською базою. Використання стандартних архітектур інтернет-магазинів у цій галузі часто виявляється неефективним, оскільки вони не здатні забезпечити гнучкий облік нестандартних товарних одиниць та не пропонують належного інструментарію для розширеної категоризації ароматів за ольфакторними пірамідами. Отже, розробка спеціалізованої вебплатформи є необхідним та обґрунтованим кроком. Такий програмний продукт дозволить цифровізувати рутинні адміністративні завдання, автоматизувати процеси обробки замовлень і гарантуватиме прозору аналітику продажів, що є критично важливим для оптимізації бізнесу.

Метою роботи є проєктування та програмна реалізація вебплатформи для управління продажами нішевої парфумерії, яка створить єдине цифрове середовище для оптимізації процесів обліку продукції та покращення загального клієнтського досвіду.

Завданням на роботу є аналіз предметної області та розгляд існуючих аналогів на ринку, визначення основних вимог до майбутньої системи й вибір інструментальних засобів для її розробки, проєктування архітектури програмного продукту та структури бази даних, безпосередня розробка серверної та клієнтської частин вебдодатка, а також тестування створеної платформи для перевірки її працездатності та виявлення можливих помилок.

Об'єктом дослідження виступає процес організації електронної комерції та управління життєвим циклом онлайн-продажів у сегменті ексклюзивної парфумерії.

Предметом дослідження є моделі, алгоритми обробки даних та інструментальні вебтехнології, що застосовуються для проєктування і створення архітектури торгової онлайн-платформи.

		Кравченко К.М.			ІПЗ.КР.Б – 121 – 26– ПЗ	Арк.
		Єфремов Ю.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		8

У процесі виконання випускної кваліфікаційної роботи застосовувалися такі методи дослідження, як системний аналіз для виявлення архітектурних потреб майбутнього продукту, об'єктно-орієнтоване проєктування для побудови взаємозв'язків між програмними компонентами, а також методи ручного та автоматизованого тестування з метою підтвердження відповідності розробленої системи заявленим вимогам.

		Кравченко К.М.			ІПЗ.КР.Б – 121 – 26– ІЗ	Арк.
		Єфремов Ю.М.				9
Змн.	Арк.	№ докум.	Підпис	Дата		

РОЗДІЛ 1. АНАЛІЗ ВИМОГ КОРИСТУВАЧА ТА КОНЦЕПТУАЛЬНЕ МОДЕЛЮВАННЯ

1.1. Постановка задачі

Розвиток сучасних систем електронної комерції диктує необхідність переходу від універсальних торгових майданчиків до спеціалізованих вебрішень, адаптованих під конкретну предметну область. Управління продажами у сегменті нішевої парфумерії характеризується високим рівнем складності через специфіку самого продукту. На відміну від масового ринку, селективні аромати вимагають деталізованої категоризації, що враховує архітектуру ольфакторної піраміди, унікальні інгредієнти та специфічні формати випуску. Традиційні платформи часто виявляються неефективними, оскільки не здатні забезпечити гнучкість в управлінні таким контентом, що призводить до зниження якості обслуговування клієнтів та ускладнення процесів адміністрування персоналом.

Виходячи з цього, першочерговим завданням є розробка цифрової платформи, яка забезпечить комплексне управління асортиментом та логістикою продажів. Система повинна реалізовувати розширений механізм формування карток товару, що дозволить адміністраторам гнучко додавати та редагувати верхні, середні та базові ноти ароматів, створюючи повноцінний ольфакторний профіль. Окрім цього, критично важливою є підтримка багатоваріантності товарних позицій. Оскільки нішева парфумерія реалізується не лише у повнорозмірних флаконах, а й у форматах пробників та відливантів різного мілілітражу, архітектура бази даних має передбачати точний облік кожної варіації об'єму для однієї позиції. Це дозволить вести автоматизований моніторинг залишків на складі в режимі реального часу: після кожної успішної транзакції система повинна автоматично віднімати відповідну кількість мілілітрів або флаконів, мінімізуючи ризики продажу відсутнього товару та усуваючи людський фактор.

Іншим вагомим аспектом розробки є проєктування інтерфейсу для покращення користувацького досвіду. Для забезпечення високої конверсії та

		Кравченко К.М.			ІПЗ.КР.Б – 121 – 26– ПЗ	Арк.
		Єфремов Ю.М.				10
Змн.	Арк.	№ докум.	Підпис	Дата		

зниження відсотка покинутих кошиків, платформа має підтримувати безперешкодне оформлення замовлень як для авторизованих клієнтів, так і для неавторизованих відвідувачів. Проходження реєстрації повинно пропонувати додаткові переваги, такі як збереження історії транзакцій, відстеження статусів обробки та швидке автозаповнення даних доставки, проте не має бути жорсткою вимогою для здійснення покупки.

Візуальний інтерфейс розроблюваного вебдодатка повинен відповідати преміальності сегмента: бути мінімалістичним, інтуїтивно зрозумілим та повністю адаптивним, щоб гарантувати коректне відображення та зручність взаємодії на пристроях із різною роздільною здатністю екрана. Наявність продуманої системи навігації та гнучкої фільтрації за брендами, цінovими категоріями чи сімействами ароматів дозволить клієнтам швидко знаходити релевантну продукцію, що значно покращить загальне враження від сервісу.

Невід'ємною складовою успішного функціонування платформи є створення повноцінної панелі адміністрування. Для ефективної роботи персоналу необхідно реалізувати чіткий алгоритм обробки замовлень із використанням системи статусів, що відображають кожен етап логістичного ланцюжка — від підтвердження покупки до її кінцевої доставки клієнту. Окрім управління замовленнями та контентом, система повинна надавати керівництву інтернет-магазину інструменти для моніторингу бізнес-показників. Розробка аналітичного модуля, який автоматично збирає дані та візуалізує статистику продажів, визначає найпопулярніші товари, ноти та бренди, дозволить адміністрації своєчасно реагувати на зміни попиту та приймати обґрунтовані рішення для розширення асортименту.

Таким чином, розроблюваний програмний комплекс має стати єдиним централізованим середовищем, що поєднує зручний клієнтський сервіс із потужними інструментами адміністрування. Автоматизація рутинних операцій, строгий контроль складських запасів за об'ємами та розширена робота з парфумерним контентом дозволять не лише підвищити ефективність бізнес

		Кравченко К.М.			ІПЗ.КР.Б – 121 – 26– ПЗ	Арк.
		Єфремов Ю.М.				11
Змн.	Арк.	№ докум.	Підпис	Дата		

процесів підприємства, а й забезпечити високий рівень цифрової взаємодії з вимогливою цільовою аудиторією.

1.2. Аналіз аналогів програмного продукту

Для створення конкурентоспроможної та функціональної платформи необхідно провести ретельний аналіз існуючих рішень на ринку електронної комерції у сфері парфумерії. Проведене дослідження аналогічних веб ресурсів дозволяє глибше зрозуміти поточний стан ринку, виявити архітектурні прогалини наявних систем та обґрунтувати шляхи їх удосконалення. У якості об'єктів для порівняння обрано три популярні торгові майданчики: FragranceX, Perfume.com та The Perfume Spot.

FragranceX — це онлайн-магазин парфумерії, який спеціалізується на реалізації оригінальних ароматів за зниженими цінами. Ця платформа орієнтована переважно на масовий сегмент споживачів і функціонує на ринку як дисконтний ритейлер. Вигляд сайту зображено на рис.1.1.

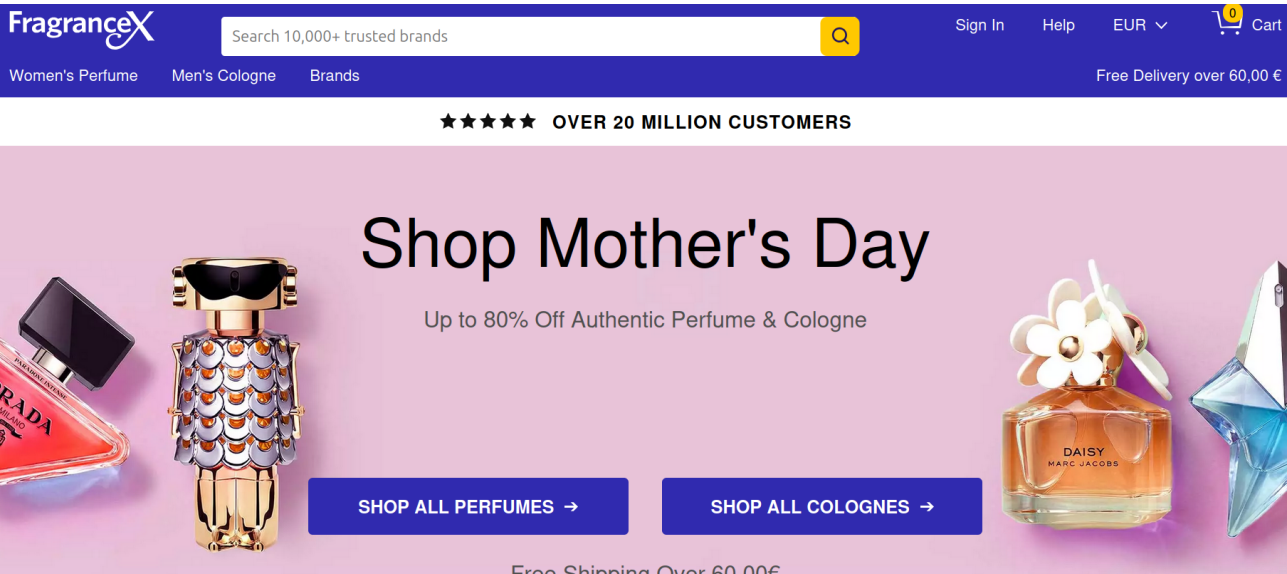


Рисунок 1.1 – FragranceX

До сильних сторін даного ресурсу належить наявність широкого асортименту брендової продукції. Завдяки орієнтації на масовий ринок платформа пропонує висококонкурентну цінову політику, яка постійно підтримується регулярними акційними пропозиціями та розпродажами.

		Кравченко К.М.			ІПЗ.КР.Б – 121 – 26– ПЗ	Арк.
		Єфремов Ю.М.				12
Змн.	Арк.	№ докум.	Підпис	Дата		

З технічної та користувацької точок зору система має суттєві вразливості, серед яких найбільш критичним є застарілий візуальний інтерфейс, що ускладнює взаємодію з клієнтом. Обмежені інструменти персоналізації роблять процес вибору менш індивідуальним, а недосконала система навігації та фільтрації товарів не дозволяє користувачам здійснювати гнучкий пошук специфічних ароматів за нотами чи концентрацією.

Perfume.com виступає одним із класичних інтернет-магазинів парфумерії, що функціонує на ринку тривалий час. Ресурс пропонує об'ємний вибір ароматів, охоплюючи різноманітні цінові категорії від мас-маркету до люксових брендів. Вигляд сайту зображено на рис.1.2.

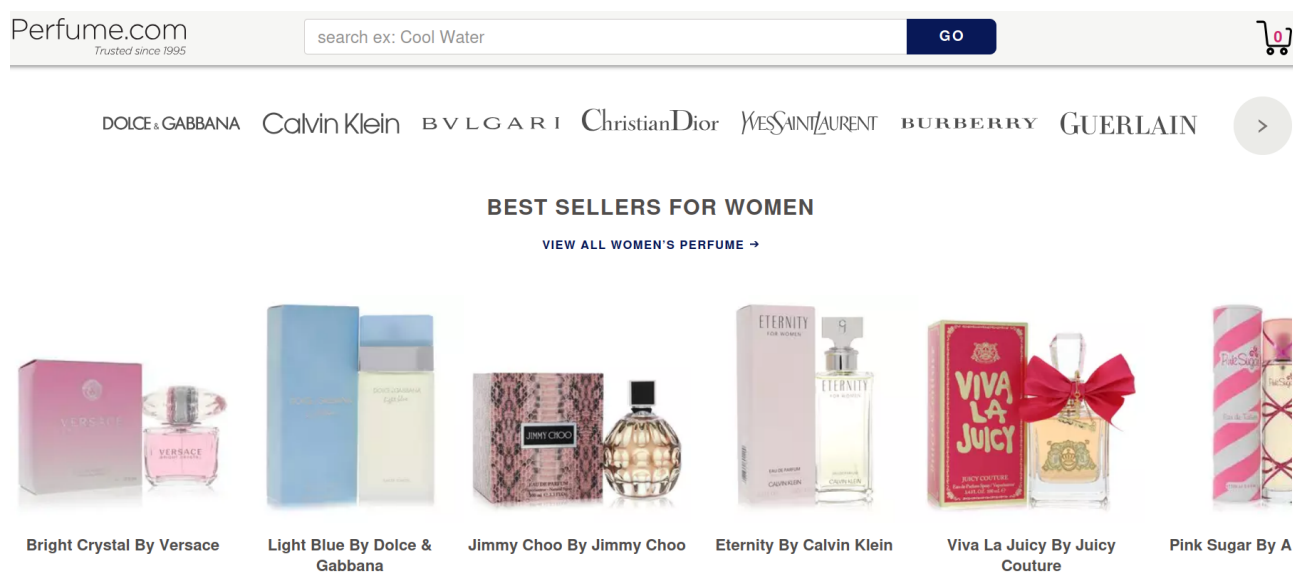


Рисунок 1.2 – Perfume.com

Головною перевагою платформи є великий та структурований каталог товарів, що дозволяє охопити широку цільову аудиторію. Крім того, архітектура клієнтської частини забезпечує максимально простий і логічно зрозумілий процес оформлення замовлення, що позитивно впливає на показники утримання клієнтів під час здійснення покупки та можливості з легкістю знайти бажаний товар за допомогою гнучкої системи фільтрів.

Незважаючи на стабільну роботу, платформа демонструє слабкий рівень сучасного UX/UI дизайну, який не відповідає актуальним вимогам електронної

		Кравченко К.М.			ІПЗ.КР.Б – 121 – 26– ПЗ	Арк.
		Єфремов Ю.М.				13
Змн.	Арк.	№ докум.	Підпис	Дата		

комерції преміум-сегмента. Відсутність інтелектуальних алгоритмів рекомендацій товарів та недостатньо розвинена система аналітики унеможливають глибоку персоналізацію взаємодії з покупцем, що є вкрай важливим для реалізації нішевої продукції.

The Perfume Spot — це цифровий торговий майданчик, який фокусується на швидкій реалізації оригінальної парфумерії зі знижками та забезпеченні оперативної логістики замовлень. Вигляд головної сторінки сайту зображено на рис.1.3.

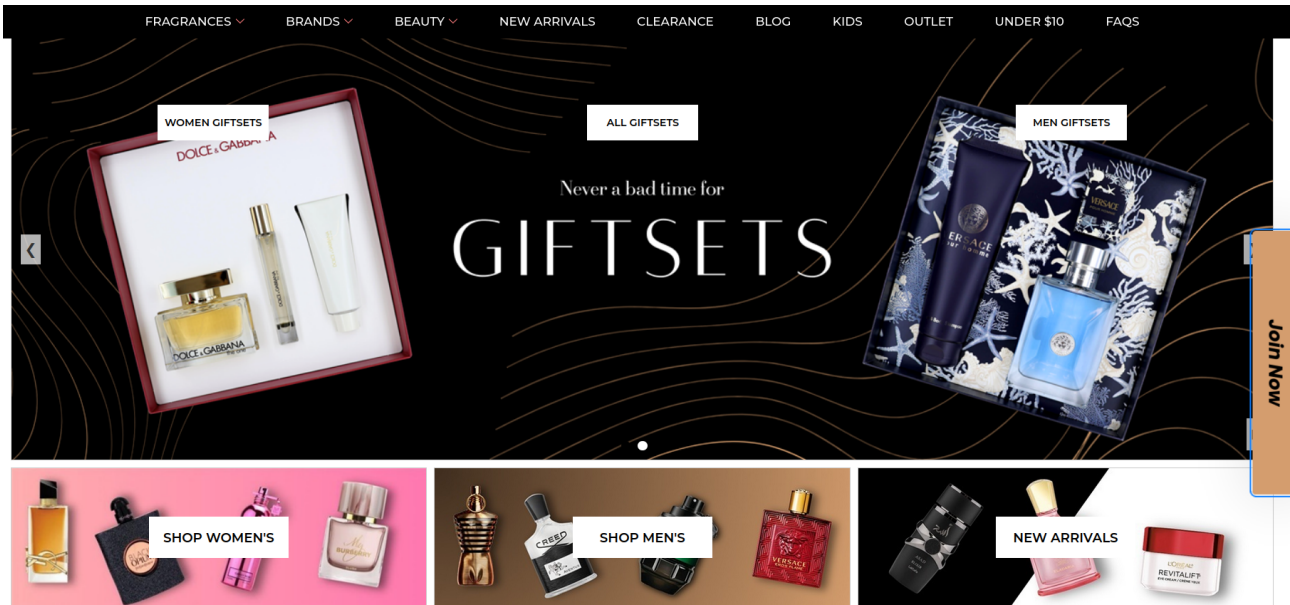


Рисунок 1.3 – The Perfume Spot

Сильними сторонами цього аналогу є привабливе ціноутворення на популярні аромати та висока швидкість обробки транзакцій. Завдяки надійній роботі внутрішнього логістичного модуля платформа здобула достатньо стабільну репутацію серед постійних покупців.

Основним слабким місцем ресурсу є застаріле оформлення користувацького інтерфейсу, яке абсолютно не адаптовано під сучасні мобільні пристрої з різними видами роздільної здатності. Разом із цим, на платформі спостерігається критична нестача додаткових клієнтських сервісів, таких як професійний підбір ароматів, а також простежується слабка інтеграція

		Кравченко К.М.			ІПЗ.КР.Б – 121 – 26– ПЗ	Арк.
		Єфремов Ю.М.				14
Змн.	Арк.	№ докум.	Підпис	Дата		

актуальних механік електронної комерції, що суттєво обмежує функціональні можливості продукту порівняно з сучасними стандартами розробки.

Після огляду існуючих рішень на ринку нішевих парфумерій було прийнято рішення підбити підсумки загального аналізу. Для візуального відтворення отриманих даних було виділено ключові критерії оцінки та сформовано зведену порівняльну таблицю (табл. 1.1), де існуючі аналоги зіставляються з концепцією проєктованої веб-платформи. Порівняння здійснювалось за критеріями, як обсяг товарів, UI/UX, адаптивність, а також зручність пошуку по сайту за допомогою фільтрів.

Таблиця 1.1

Порівняння аналогів системи

Характеристика	FragranceX	Perfume.com	The Perfume Spot
Наявність широкого каталогу товарів	+	+	+
Сучасний та адаптивний UI/UX дизайн	-	-	-
Гнучка фільтрація (за нотами, об'ємами, відливантами)	-	+	-
Порівняння товарів	-	-	-
Розширена аналітика для адміністратора	-	-	-
Мобільна адаптивність	+	+-	-
Розширена система фільтрів	-	+	-

Підсумовуючи результати аналізу, можна зробити висновок, що розглянуті цифрові ресурси орієнтовані виключно на продаж товарів мас-маркету за рахунок великих обсягів та знижок. Їхня архітектура не здатна задовольнити специфічні потреби нішевої парфумерії, де ключову роль відіграють деталізований опис ольфакторних пірамід та гнучкий облік різних форматів

продукції. Проектована платформа має вирішити ці проблеми шляхом впровадження сучасних архітектурних рішень.

1.3. Вибір архітектури системи

У контексті стрімкого розвитку електронної комерції та зростаючих вимог до надійності вебдодатків, фундаментальним етапом проєктування є вибір архітектури програмного забезпечення. Архітектура визначає високорівневу структуру системи, принципи взаємодії її компонентів, стратегії розгортання, а також механізми управління даними. Вірно підібрана архітектурна модель не лише забезпечує виконання поточних бізнес-вимог, але й гарантує можливість подальшого розширення, інтеграції нових модулів та безперебійного функціонування в умовах пікових навантажень. На етапі проєктування системи розробник повинен провести ретельний аналіз альтернативних рішень та обрати найоптимальніше, зваживши всі переваги та недоліки. Сучасна інженерія програмного забезпечення пропонує низку підходів до побудови серверної частини, серед яких найбільш поширеними та обговорюваними є монолітна та мікросервісна архітектури.

Традиційним та історично першим підходом до розробки вебдодатків є монолітна архітектура. Вона передбачає побудову програмного продукту як єдиної, уніфікованої та нерозривної системи. У такій моделі всі ключові компоненти, включаючи модулі обробки клієнтських запитів, бізнес-логіку, підсистему авторизації, логіку взаємодії з базою даних та генерацію користувацького інтерфейсу, розміщуються в єдиній кодовій базі та виконуються в межах одного процесу. Для монолітних систем характерним є використання єдиної централізованої реляційної бази даних, до якої звертаються всі модулі програми.

Сильними сторонами монолітної архітектури є її концептуальна простота на початкових етапах життєвого циклу розробки. Завдяки відсутності мережових бар'єрів між внутрішніми компонентами, монолітні системи демонструють високу швидкість обміну даними всередині програми, оскільки виклик будь-якої

		Кравченко К.М.			ІПЗ.КР.Б – 121 – 26– ПЗ	Арк.
		Єфремов Ю.М.				16
Змн.	Арк.	№ докум.	Підпис	Дата		

функції відбувається безпосередньо в оперативній пам'яті, без затримок на мережеву маршрутизацію. Розгортання такого продукту є тривіальним процесом, оскільки вимагає копіювання лише одного виконуваного артефакту на сервер. Крім того, наявність централізованої бази даних суттєво спрощує забезпечення транзакційної цілісності та узгодженості даних, що досягається механізмами ACID. Моноліт значно легше піддається наскрізному тестуванню та відлагодженню, оскільки весь потік виконання запиту можна простежити в одному середовищі.

Проте з часом, у міру зростання бізнесу та додавання нового функціоналу, монолітний додаток неминуче перетворюється на надзвичайно складну і громіздку структуру, яку важко підтримувати. Тісна зв'язність компонентів призводить до того, що внесення змін в один модуль може викликати непередбачувані помилки в абсолютно інших частинах системи.

Масштабування моноліту можливе виключно шляхом дублювання всього додатка на нових серверах, що є вкрай неефективним використанням апаратних ресурсів. Наприклад, якщо певний функціональний блок потребує великих обчислювальних потужностей, системному адміністратору доведеться масштабувати всю програму, навіть якщо інші її модулі цього зовсім не потребують. Також монолітна архітектура створює ефект жорсткого технологічного блокування: перехід на новіші версії мов програмування чи використання альтернативних технологій для вирішення специфічних задач стає практично неможливим без повного переписування всього продукту.

На противагу монолітному підходу виступає мікросервісна архітектура. Це сучасний метод розробки програмного забезпечення, за якого складна система декомпонується на набір невеликих, слабо зв'язаних та автономних сервісів. Кожен мікросервіс відповідає за чітко визначену частину бізнес-логіки, функціонує у власному ізольованому процесі та спілкується з іншими компонентами за допомогою легковагових протоколів, найчастіше через RESTful API або за допомогою брокерів повідомлень для асинхронної взаємодії [21].

		Кравченко К.М.			ІПЗ.КР.Б – 121 – 26– ПЗ	Арк.
		Єфремов Ю.М.				17
Змн.	Арк.	№ докум.	Підпис	Дата		

Головною перевагою мікросервісної архітектури є безпрецедентна гнучкість у масштабуванні та розгортанні. Кожен сервіс може бути масштабований ізолювано, залежно від його специфічного навантаження, що дозволяє раціонально використовувати хмарні ресурси. Надійність та відмовостійкість системи загалом суттєво підвищується: критичний збій в одному мікросервісі не призводить до каскадного падіння всього застосунку, оскільки інші сервіси продовжують функціонувати в штатному режимі. Розробка в мікросервісному середовищі стимулює децентралізацію управління даними — кожен сервіс керує власною, оптимізованою під його задачі базою даних. Це унеможливорює несанкціоноване втручання інших компонентів у його структуру та дозволяє застосовувати підхід поліглотного програмування, коли для кожного конкретного сервісу обирається найбільш підходящий технологічний стек.

Разом з тим, впровадження мікросервісів супроводжується значним ускладненням системної інфраструктури. Наявність великої кількості незалежних вузлів вимагає впровадження складних механізмів оркестрації, розподіленого логування та моніторингу. Мережева взаємодія між сервісами створює додаткові затримки, яких не існує в монолітах, а забезпечення узгодженості даних між різними базами вимагає застосування складних архітектурних патернів [10] [22]. Таким чином, розгортання та адміністрування мікросервісної системи вимагає високої кваліфікації інженерної команди.

Аналізуючи специфіку розроблюваної платформи для управління продажами нішевої парфумерії, було прийнято обґрунтоване рішення на користь мікросервісної архітектури. Цей вибір продиктований низкою об'єктивних факторів, пов'язаних із бізнес-моделлю нішевого сегмента та технічними викликами електронної комерції.

По-перше, предметна область електронної торгівлі парфумерією характеризується яскраво вираженою асиметрією навантаження. Переважна більшість запитів користувачів спрямована на перегляд каталогу: пошук ексклюзивних ароматів, застосування складних фільтрів за ольфакторною пірамідою (ноти серця, базові ноти тощо), читання історії бренду та перегляд

		Кравченко К.М.			ІПЗ.КР.Б – 121 – 26– ІЗ	Арк.
		Єфремов Ю.М.				18
Змн.	Арк.	№ докум.	Підпис	Дата		

якісних медіафайлів. Операції ж оформлення замовлення, реєстрації клієнта або оновлення складських залишків відбуваються значно рідше. У випадку використання моноліту, під час сезонних розпродажів або рекламних кампаній, довелося б масштабувати всю систему лише заради того, щоб впоратися з потоком запитів на читання каталогу. Мікросервісний підхід дозволяє виділити «Сервіс каталогу» в окремий незалежний вузол, який можна багаторазово масштабувати горизонтально, не зачіпаючи при цьому «Сервіс замовлень» чи «Сервіс обліку складу».

По-друге, специфіка продукції вимагає надзвичайно гнучкого підходу до управління даними. Нішева парфумерія реалізується не лише у форматі класичних повнорозмірних флаконів, але й у вигляді тестерів, мініатюр та відливаних точного мілілітражу. Це означає, що логіка складського обліку є вкрай складною: продаж одного відливанту на 5 мл має автоматично перераховувати загальний об'єм доступної рідини у відкритому флаконі-донорі. Виокремлення цієї логіки в незалежний мікросервіс управління складом дозволяє ізолювати складні транзакційні операції від вітрини магазину. Спираючись на принципи доменно-орієнтованого проєктування [17], ми розділяємо складні бізнес-процеси на обмежені контексти, що робить систему передбачуваною та надійною.

По-третє, мікросервісна архітектура забезпечує необхідний рівень відмовостійкості. Для комерційної платформи критично важливо не втрачати клієнтів через технічні збої. Якщо, наприклад, сервіс розсилки електронних листів або сервіс аналітики тимчасово вийде з ладу через перевантаження, ядро платформи — каталог ароматів та корзина покупок — залишатимуться повністю працездатними. Користувач зможе безперешкодно вибрати товар і зробити замовлення, а повідомлення про успішну покупку буде надіслано системою пізніше, коли допоміжний сервіс відновить свою роботу.

Хоча система в цілому базується на мікросервісній парадигмі, внутрішня будова кожного окремого мікросервісу розроблятиметься на засадах «чистої архітектури» [15]. Такий симбіоз дозволить жорстко відокремити доменну логіку

		Кравченко К.М.			ІПЗ.КР.Б – 121 – 26– ПЗ	Арк.
		Єфремов Ю.М.				19
Змн.	Арк.	№ докум.	Підпис	Дата		

роботи з ароматами та транзакціями від зовнішніх інфраструктурних деталей, таких як механізми маршрутизації чи специфіка бази даних.

Таким чином, вибір мікросервісної архітектури, попри вищі початкові витрати на проектування інфраструктури, є стратегічно доцільним та перспективним. Він створює міцний технологічний фундамент для платформи нішевої парфумерії, гарантуючи їй високу швидкість реакції, бездоганний користувацький досвід, надійність при збереженні комерційних даних та широкі можливості для подальшого бізнес-масштабування.

1.4. Обґрунтування вибору інструментальних засобів та вимоги до апаратного забезпечення

Успішна реалізація цифрової платформи для управління продажами нішевої парфумерії безпосередньо залежить від правильного вибору інструментальних засобів та технологічного стеку. Оскільки на попередніх етапах проектування було прийнято рішення про використання мікросервісного підходу та розділення програмного продукту на незалежні клієнтську і серверну частини, обрані технології повинні забезпечувати високу продуктивність, надійність при обробці транзакцій, гнучкість у розгортанні та високий рівень безпеки комерційних даних.

Для розробки серверної частини було обрано мову програмування Python [1] та її високорівневий вебфреймворк Django [4]. У сучасній парадигмі веб-розробки використання Django еволюціонувало: великі компанії все частіше застосовують його не як монолітне рішення для створення повноцінних сайтів, а виключно як потужний інструмент для побудови серверного API. Такий підхід ідеально відповідає нашій меті — створити незалежну серверну логіку, яка комунікуватиме з клієнтським додатком через RESTful інтерфейс [27]. Фреймворк Django успадкував від мови Python філософію, що передбачає наявність готових модулів для вирішення типових завдань електронної комерції. Це включає вбудовану систему автентифікації користувачів, панель адміністратора, зручну маршрутизацію та надійні механізми безпеки, що

		Кравченко К.М.			ІПЗ.КР.Б – 121 – 26– ПЗ	Арк.
		Єфремов Ю.М.				20
Змн.	Арк.	№ докум.	Підпис	Дата		

захищають від поширених вразливостей, таких як XSS або CSRF [29]. Враховуючи, що система управління продажами парфумерії вимагає складної бізнес-логіки для обліку специфічних товарних одиниць у вигляді флаконів з різним об'ємом, використання перевіреного часом фреймворку, який має понад десятирічну історію активного розвитку, дозволяє уникнути фундаментальних архітектурних помилок.

Забезпечення бездоганного користувацького досвіду є критичним фактором у преміальному сегменті нішевої парфумерії. Саме тому для реалізації клієнтської частини було обрано JavaScript-бібліотеку React [5]. Традиційні підходи з рендерингом сторінок на сервері є занадто повільними для сучасних інтернет-магазинів, оскільки вимагають повного перезавантаження вікна браузера при кожній дії. React вирішує цю проблему шляхом створення односторінкових додатків. Бібліотека не взаємодіє безпосередньо з DOM браузера, а використовує концепцію віртуального DOM. Це означає, що під час застосування складних фільтрів, наприклад, пошук ароматів за базовими нотами сандалу та мускусу в певному ціновому діапазоні, React обчислює мінімально необхідний набір інструкцій для оновлення інтерфейсу і миттєво відображає результати без затримок. Крім того, компонентна архітектура React дозволяє розділити інтерфейс на незалежні модулі, які можна повторно використовувати в різних частинах додатку, що суттєво пришвидшує розробку та полегшує подальшу підтримку коду. Гнучка взаємодія реактивних компонентів із серверним API [9] дозволяє асинхронно отримувати дані каталогу, забезпечуючи високу швидкість відгуку навіть при слабкому інтернет-з'єднанні.

Управління фінансовими транзакціями, збереженням профілів користувачів та точним складським обліком вимагає використання надійної реляційної моделі даних. Для цих цілей найкращим вибором є система управління базами даних MySQL [11]. Вона є однією з найпоширеніших та найстабільніших систем у світі електронної комерції, що гарантує суворе дотримання принципів ACID при виконанні транзакцій. Реляційна структура MySQL ідеально підходить для опису складної предметної області

		Кравченко К.М.			ІПЗ.КР.Б – 121 – 26– ПЗ	Арк.
		Єфремов Ю.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		21

парфумерного магазину, дозволяючи встановлювати чіткі зв'язки між сутностями: брендами, ароматами, категоріями, замовленнями та історією покупок клієнта. Окрім цього, база даних має відмінну сумісність із серверними застосунками та інструментами ORM, що дозволяє безпечно і швидко виконувати складні SQL-запити.

Специфіка продажу нішевої парфумерії передбачає наявність великого обсягу високоякісного візуального контенту: деталізованих фотографій флаконів, відеоматеріалів брендів та промо-банерів. Зберігати такі медіафайли безпосередньо на основному сервері додатку або в реляційній базі даних є вкрай неефективним рішенням, яке призводить до швидкого вичерпання апаратних ресурсів та уповільнення роботи системи. Для вирішення цієї проблеми у проєкті застосовується MinIO [17] — високопродуктивне сховище об'єктів, сумісне зі стандартом Amazon S3. Подібно до хмарних сервісів AWS, MinIO дозволяє ізольовано зберігати файли і надавати доступ до них через мережу Інтернет за допомогою згенерованих посилань. Використання об'єктного сховища знімає навантаження з основного Django-сервера [2], оптимізує процеси кешування медіаконтенту та гарантує надійність зберігання графічних активів.

Для об'єднання всіх перелічених технологій у єдину, керовану інфраструктуру використовується платформа контейнеризації Docker [7]. Сучасна інженерна практика вимагає, щоб програмне забезпечення працювало абсолютно однаково як на локальному комп'ютері розробника, так і на бойовому сервері. Docker вирішує цю проблему шляхом пакування Django-бекенду, React-фронтенду [8], бази даних MySQL та сховища MinIO в ізольовані контейнери, які містять усі необхідні залежності та бібліотеки. Оскільки наш додаток складається з кількох незалежних сервісів, для їхньої оркестрації застосовується інструмент Docker Compose. Він дозволяє описати конфігурацію всієї інфраструктури в одному YAML-файлі та запустити багатоконтейнерну систему єдиною командою. Такий підхід гарантує, що можливі збої або критичні помилки в одному мікросервісі не вплинуть на працездатність ізольованого контейнера з базою даних, підвищуючи загальну відмовостійкість платформи.

		Кравченко К.М.			ІПЗ.КР.Б – 121 – 26– ІЗ	Арк.
		Єфремов Ю.М.				22
Змн.	Арк.	№ докум.	Підпис	Дата		

Навіть за умови глибокої оптимізації програмного коду, стабільне функціонування системи неможливе без відповідного апаратного забезпечення. Розгортання контейнеризованої інфраструктури висуває специфічні вимоги до сервера. Для безперебійної паралельної обробки клієнтських запитів та управління кількома Docker-контейнерами сервер повинен бути оснащений щонайменше чотириядерним процесором із тактовою частотою від 3.0 ГГц. Оскільки запуск мікросервісів та операції бази даних вимагають значного простору для тимчасового зберігання обчислень, мінімальний обсяг оперативної пам'яті становить 8 ГБ. Надзвичайно важливою вимогою є використання твердотільного накопичувача SSD замість класичних жорстких дисків HDD. Відсутність рухомих механічних частин у SSD-дисках усуває затримки при зчитуванні та записі інформації, що є критичним для забезпечення миттєвого доступу до реляційних баз даних та файлового сховища [19]. Обсяг дискового простору повинен складати не менше 100 ГБ для вільного розміщення операційної системи Linux, образів Docker, томів бази даних та медіафайлів.

Зі сторони кінцевого користувача вимоги до обладнання є мінімальними. Завдяки тому, що основні обчислювальні процеси відбуваються на сервері, клієнту достатньо мати базовий персональний комп'ютер або мобільний пристрій із доступом до мережі Інтернет. Головною і єдиною програмною вимогою для роботи з платформою є наявність сучасного веб-браузера, який підтримує виконання сучасних стандартів JavaScript.

Висновки до першого розділу

У першому розділі було проведено комплексний аналіз предметної області та визначено ключові вимоги до проєктування платформи для управління продажами нішевої парфумерії.

Дослідження існуючих аналогів на ринку виявило, що популярні торгові майданчики орієнтовані переважно на мас-маркет, а їхня монолітна структура не здатна надати достатній рівень персоналізації та складської гнучкості для преміум-сегмента. Виявлені архітектурні та функціональні прогалини

		Кравченко К.М.			ІПЗ.КР.Б – 121 – 26– ПЗ	Арк.
		Єфремов Ю.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		23

конкурентів логічно обґрунтували потребу у створенні власного цифрового рішення.

Вирішення сформульованих завдань зумовило вибір мікросервісної архітектури як концептуальної основи майбутньої платформи. Такий підхід гарантує високу відмовостійкість електронного магазину та дозволяє ізольовано масштабувати найбільш навантажені вузли, наприклад, сервіс пошуку та каталогу товарів, не впливаючи на роботу модуля обробки замовлень.

Для реалізації обраної архітектурної моделі було визначено оптимальний технологічний стек, що відповідає сучасним критеріям надійності комерційних систем. Розділення додатку на незалежну серверну частину на базі високорівневого фреймворку Django та клієнтський додаток за допомогою бібліотеки React забезпечить миттєвий відгук інтерфейсу без перезавантаження сторінок. Як сховище транзакційних даних обрано реляційну СУБД MySQL, що гарантує цілісність фінансових операцій, тоді як для оптимізації роботи з важким медіаконтентом застосовано об'єктне сховище MinIO. Впровадження платформи контейнеризації Docker гарантуватиме ідентичність середовищ розробки та розгортання, спрощуючи адміністрування інфраструктури.

Виконаний системний аналіз, визначення вимог та обґрунтування інструментальних засобів формують надійний технологічний і концептуальний фундамент. Це дозволяє завершити етап аналітики та перейти до безпосереднього об'єктно-орієнтованого проєктування системи і розробки її бази даних у наступному розділі.

		Кравченко К.М.			ІПЗ.КР.Б – 121 – 26– ПЗ	Арк.
		Єфремов Ю.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		24

РОЗДІЛ 2. ПРОЕКТУВАННЯ ПЛАТФОРМИ ДЛЯ УПРАВЛІННЯ ПРОДАЖАМИ НІШЕВОЇ ПАРФУМЕРІЇ

2.1. Визначення варіантів використання та структури системи

Після обґрунтування мікросервісної архітектури та вибору технологічного стеку у попередньому розділі, наступним етапом розробки є концептуальне проектування поведінки системи. Для цього необхідно чітко ідентифікувати цільову аудиторію програмного продукту, визначити її функціональні потреби та побудувати логічну структуру взаємодії за допомогою методології об'єктно-орієнтованого аналізу.

Аналіз предметної області електронної торгівлі парфумерією дозволив виділити три ключові актори авторизованих користувачів платформи: клієнт, менеджер та адміністратор, а також окрему сутність неавторизованого відвідувача. Розмежування прав доступу між цими акторами є критично важливим завданням для забезпечення безпеки комерційних даних та зручності адміністрування. Гість системи має обмежені права, які зводяться до перегляду вітрини, використання повнотекстового пошуку та оформлення так званого гостьового замовлення без обов'язкового створення облікового запису.

Реєстрація суттєво розширює можливості клієнта, надаючи йому доступ до персонального профілю, де зберігається історія власних транзакцій з деталізацією за статусами, адреси доставки, а також формується список бажаного. Менеджер виступає сполучною ланкою між електронним магазином та покупцем; його основна компетенція полягає в оперативній обробці створених замовлень, зміні їх статусів та редагуванні позицій з урахуванням фактичних складських залишків. Адміністратор наділений найвищим рівнем привілеїв. Окрім повного доступу до управління користувачами, включно з можливістю блокування підозрілих акаунтів, адміністратор повністю керує каталогом товарів, наповнює систему брендами та нотами, а також має доступ до розширеної аналітичної панелі.

Для задоволення потреб перелічених акторів було сформовано комплекс функціональних вимог. Ядро розроблюваної системи має забезпечувати гнучку

		Кравченко К.М.			ІПЗ.КР.Б – 121 – 26– ІЗ	Арк.
		Єфремов Ю.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		25

роботу з каталогом парфумерії. Оскільки селективні аромати характеризуються складною ольфакторною пірамідою, платформа повинна підтримувати глибоку систему фільтрації за багатокритеріальною логікою: пошук одночасно за статтю, конкретним брендом, діапазоном ціни та специфічними парфумерними нотами. Функціонал вітрини має бути оптимізований механізмами пагінації та розумним повнотекстовим пошуком, який аналізуватиме як назви, так і розгорнуті описи композицій. Окремою вагомою вимогою є реалізація алгоритмів генерації рекомендацій, яка пропонуватиме клієнту схожі товари на основі виявлення спільних парфумерних нот в обраних ароматах.

Управління складським обліком вимагає впровадження спеціалізованої сутності товарних варіацій, що дозволить адміністраторам прив'язувати до одного базового парфуму кілька об'ємів з індивідуальними цінами, унікальними складськими ідентифікаторами та показниками залишку. З боку адміністративної панелі вимагається реалізація інтуїтивного інтерфейсу для роботи з медіаконтентом, зокрема забезпечення можливості обрізки зображень безпосередньо перед завантаженням на сервер та зміни порядку фотографій флаконів методом перетягування.

Бізнес-вимоги до логістичного модуля передбачають інтеграцію зі сторонніми сервісами доставки, зокрема автоматизовану взаємодію з API компанії «Нова Пошта» для вибору актуальних відділень, а також надання альтернативної можливості оформлення адресної кур'єрської доставки. Для підвищення соціальної довіри покупців платформа повинна підтримувати систему коментарів та оцінювання товарів, алгоритм якої гарантуватиме, що один клієнт зможе залишити лише один унікальний відгук до конкретного придбаного парфуму, а адміністратор матиме змогу опублікувати офіційну відповідь бренду.

Нефункціональні вимоги до програмного комплексу фокусуються на забезпеченні безперебійної швидкодії та інтеграційній гнучкості. Швидкість відповіді інтерфейсу при застосуванні багаторівневих фільтрів каталогу не повинна спричиняти затримок. Це вимагає впровадження глибоких оптимізацій

		Кравченко К.М.			ІПЗ.КР.Б – 121 – 26– ПЗ	Арк.
		Єфремов Ю.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		26

на рівні об'єктно-реляційного відображення серверної частини для запобігання проблемі надлишкових запитів шляхом попереднього завантаження зв'язаних даних про ноти, варіанти об'ємів та коментарі. Інтерфейс ідентифікації має підтримувати інтеграцію з протоколом Google OAuth для забезпечення безпечного входу в систему в один клік. Нарешті, адміністративний модуль аналітики повинен у режимі реального часу агрегувати метрики продажів, візуалізуючи динаміку транзакцій, топ-позиції, частку ринку окремих брендів та популярність специфічних ольфакторних нот за допомогою інтерактивних графічних компонентів.

Моделювання взаємодії описаних акторів із функціоналом платформи формалізується за допомогою діаграми варіантів використання системи. Головні прецеденти відображають ключові сценарії життєвого циклу електронної комерції: ідентифікація особи у системі, навігація багаторівневим парфумерним каталогом, процес конвертації товарів у кошику на підтверджене замовлення з обов'язковою фіксацією ціни, а також процеси адміністрування, які включають комплексний моніторинг статистичних даних і безперервне оновлення реляційної бази ароматів та брендів.

Після визначення ключових акторів та їх дій всередині майбутньої платформи було спроектовано діаграму акторів системи (рис. 2.1).

Головними користувачами розроблюваної платформи є чотири типи акторів: неавторизований відвідувач, зареєстрований клієнт, менеджер та адміністратор системи. Кожна з перелічених ролей наділена специфічним набором прав, що забезпечує інформаційну безпеку комерційних даних та зручність цільового використання функціоналу згідно з посадовими або клієнтськими потребами.

Неавторизований відвідувач або гість виступає первинним споживачем, головна мета якого полягає в ознайомленні з асортиментом селективної парфумерії. Для зниження порогу входу та стимулювання імпульсивних покупок цей актор має відкритий доступ до перегляду вітрини, застосування багатокритеріальної фільтрації ароматів та оформлення швидкого замовлення

		Кравченко К.М.			ІПЗ.КР.Б – 121 – 26– ПЗ	Арк.
		Єфремов Ю.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		27

без обов'язкового створення облікового запису. За бажанням отримати розширені можливості гість ініціює процес реєстрації та ідентифікації.

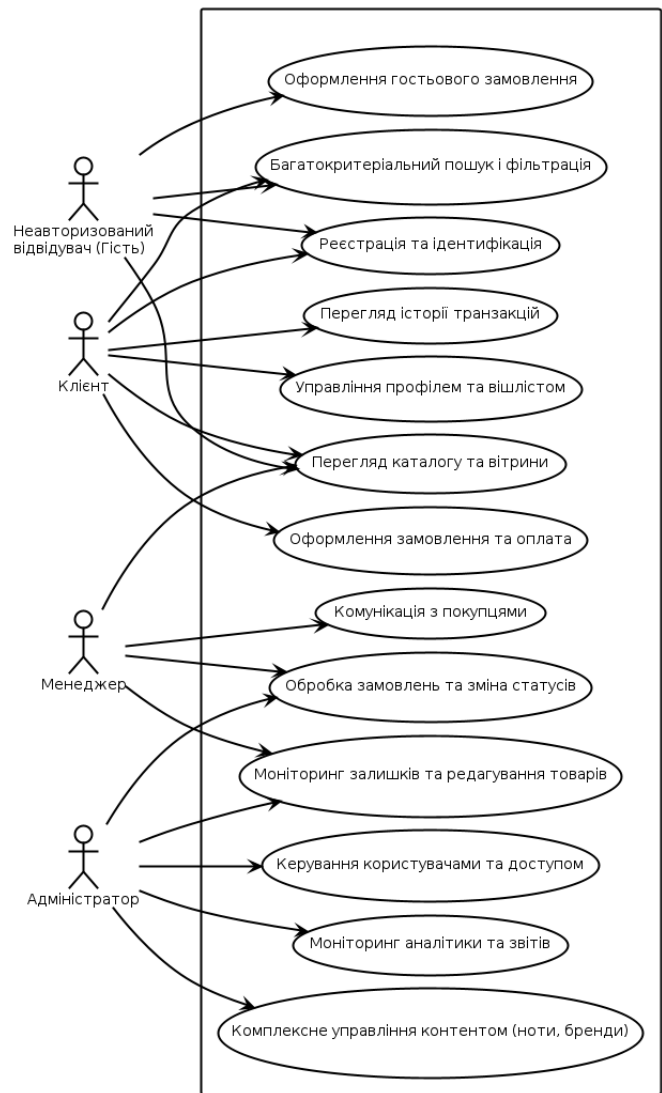


Рисунок 2.1 – Діаграма акторів системи

Зареєстрований клієнт є ключовим споживачем послуг, який оперує персоналізованим середовищем платформи. Окрім базового функціоналу пошуку та купівлі товарів, цей користувач повноцінно взаємодіє з модулем управління особистим профілем. Клієнт формує список бажаного для планування майбутніх покупок, відстежує історію власних транзакцій та оформлює нові замовлення з використанням збережених раніше адресних даних. Такий підхід суттєво пришвидшує процес оформлення покупки та стимулює клієнтську лояльність.

Менеджер відіграє роль сполучної ланки в операційній діяльності електронного магазину. Його головна відповідальність зводиться до

забезпечення безперебійного циклу виконання заявок. Для досягнення цієї мети фахівець використовує набір інструментів, які дозволяють переглядати перелік покупок, змінювати їхні статуси на різних етапах логістики, проводити ручний моніторинг складських залишків та редагувати товарні позиції. Також на рівні менеджера відбувається пряма комунікація з покупцями для уточнення деталей оплати чи доставки.

Адміністратор є вищою керуючою ланкою і зосереджується на стратегічному налаштуванні платформи. На відміну від операційного персоналу, цей актор наділений повними привілеями для комплексного управління специфічним парфумерним контентом. Він відповідає за розширення бази брендів, конструювання нових ольфакторних нот та додавання нестандартних товарних варіацій. Важливою функцією адміністратора є глобальний контроль за системою доступу: він керує профілями інших працівників та клієнтів, блокує порушників і призначає ролі. Крім того, саме адміністратор працює з модулем аналітики для відстеження динаміки продажів та формування підсумкових фінансових звітів.

2.2. Розробка бази даних системи

Основою надійного функціонування будь-якої платформи електронної комерції є правильно спроектована база даних, яка забезпечує швидкий доступ до інформації, підтримує цілісність транзакцій та дозволяє масштабувати цифрову систему в майбутньому. Для реалізації збереження даних розробленого сервісу використовується реляційна база даних MySQL.

Фундаментальним архітектурним рішенням під час проєктування схеми даних стало використання механізму логічного видалення записів для переважної більшості таблиць. Замість фізичного знищення рядків із бази даних застосовується спеціальна базова модель, яка додає до кожної сутності набір обов'язкових службових полів: унікальний ідентифікатор, час створення, час останнього оновлення, час видалення та логічний прапорець статусу видалення [12]. При ініціалізації команди видалення система лише змінює значення

		Кравченко К.М.			ІПЗ.КР.Б – 121 – 26– ПЗ	Арк.
		Єфремов Ю.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		29

відповідного прапорця на позитивне і гарантує збереження історії фінансових транзакцій та запобігає порушенню зв'язків між даними.

Управління доступом та ідентифікація користувачів реалізовані через таблицю users. Ця сутність розширює стандартну модель безпеки базового фреймворку та містить електронну адресу, яка виступає унікальним логіном, повне ім'я, статус активності акаунта, а також рольовий ідентифікатор. Завдяки наявності поля статусу система суворо розмежовує права доступу між звичайними клієнтами, менеджерами з продажу та системними адміністраторами. Детальний опис полів цієї таблиці наведено у таблиці 2.1.

Таблиця 2.1

Опис таблиці users

Назва	Тип даних	Опис
id	uuid	Унікальний ідентифікатор користувача
email	varchar	Адреса електронної пошти користувача
full_name	varchar	Повне ім'я особи
status	varchar	Роль у системі
is_active	boolean	Прапорець дозволу на авторизацію
is_staff	boolean	Прапорець доступу до адміністративної панелі
is_deleted	boolean	Прапорець логічного видалення профілю
created_at	timestamp	Точний час створення облікового запису

Ядром розробленої бази даних є підсистема каталогу парфумерії, яка складається з кількох взаємопов'язаних таблиць. Інформація про виробників зберігається у таблиці brands, що містить унікальну назву бренду, країну походження, детальний текстовий опис та посилання на логотип. Для запобігання втраті даних налаштовано обмеження на рівні бази: видалення бренду блокується, якщо в системі існують прив'язані до нього активні аромати.

Оскільки нішева продукція вимагає детального опису ольфакторної піраміди, було створено таблицю notes. Вона зберігає назви ароматичних

		Кравченко К.М.			ІПЗ.КР.Б – 121 – 26– ПЗ	Арк.
		Єфремов Ю.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		30

інгредієнтів та класифікує їх за типом на верхні, серцеві або базові ноти. Головна інформація про товар акумулюється в таблиці perfumes. Тут фіксуються назва, текстовий опис, цільова стать та рівень концентрації аромату. Ця таблиця пов'язана з брендами через зовнішній ключ, а з нотами — через проміжну таблицю perfume_notes, що реалізує зв'язок типу багато-до-багатьох (табл. 2.2).

Таблиця 2.2

Опис таблиці notes

Назва	Тип даних	Опис
id	uuid	Унікальний ідентифікатор парфуму
brand_id	uuid	Ідентифікатор парфумерного будинку
name	varchar	Комерційна назва аромату
description	text	Розгорнутий опис парфумерної композиції
gender	varchar	Цільова стать аромату
concentration	varchar	Клас концентрації ароматичних речовин
is_deleted	boolean	Прапорець логічного видалення товару

Ключовою особливістю бізнес-моделі є реалізація одного аромату в різних об'ємах, тому облік залишків та ціноутворення винесено в окрему спеціалізовану таблицю product_variants. Ця сутність зберігає інформацію про конкретний варіант випуску, включаючи точний об'єм у мілілітрах, актуальну ціну, фактичний залишок на складі та унікальний складський артикул. Розділення дозволяє клієнтам обирати між повнорозмірним флаконом і пробником у межах однієї сторінки вітрини, а адміністраторам — вести точний облік (табл. 2.3).

Логістичний модуль бази даних спроектовано з урахуванням необхідності мінімізації мережевих запитів до зовнішніх логістичних сервісів. Таблиці cities та nova_poshta_branches виконують роль внутрішнього кешу довідників служби доставки. Вони зберігають назви населених пунктів, повні текстові назви відділень, їхні порядкові номери та унікальні ідентифікатори транспортної компанії.

Таблиця 2.3

		Кравченко К.М.			ІПЗ.КР.Б – 121 – 26– ПЗ	Арк.
		Єфремов Ю.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		31

Опис таблиці product_variants

Назва	Тип даних	Опис
id	uuid	Унікальний ідентифікатор товарної варіації
perfume_id	uuid	Ідентифікатор базового аромату з каталогу
volume_ml	integer	Номінальний об'єм у мілілітрах
price	decimal	Вартість конкретної одиниці товару
quantity	integer	Кількість доступних одиниць на складі
sku	varchar	Унікальний ідентифікатор для складського обліку
is_deleted	boolean	Прапорець логічного зняття варіації з продажу

Безпосередній процес оформлення замовлення обслуговується комплексом із кількох таблиць. Таблиця customers зберігає контактні дані одержувача, маючи необов'язковий зв'язок з обліковим записом авторизованого користувача. Це дозволяє реалізувати бізнес-вимогу щодо підтримки гостьових покупок без примусової реєстрації профілю. Головна інформація про транзакцію записується у таблицю orders, де фіксуються обраний логістичний метод, посилення на відділення або адреса кур'єра, загальна сума до сплати та поточний статус обробки заявки. Схема таблиці відображена у таблиці 2.4.

Таблиця 2.4

Опис таблиці orders

Назва	Тип даних	Опис
id	uuid	Унікальний ідентифікатор замовлення
delivery_method	varchar	Обраний покупцем спосіб доставки
nova_poshta_branch_id	uuid	Ідентифікатор обраного відділення логістичної служби

Назва	Тип даних	Опис
courier_address	text	Текстова адреса для здійснення кур'єрської доставки
customer_id	uuid	Ідентифікатор картки одержувача посилки
total_price	decimal	Загальна розрахована сума транзакції
status	varchar	Поточний стан виконання замовлення персоналом
created_at	timestamp	Час успішної реєстрації заявки системою

Деталізація кожної покупки забезпечується таблицею order_items, яка пов'язує конкретне замовлення з обраними товарними варіаціями та їхньою кількістю. Критично важливим архітектурним рішенням у цій сутності є збереження ціни товару статично на момент укладання угоди. Це створює надійний фінансовий знімок транзакції та гарантує, що майбутня зміна вартості парфуму в каталозі адміністратором не призведе до автоматичного хибного перерахунку сум в історичних чеках клієнтів. Детальний опис полів таблиці відображено у таблиці 2.5.

Таблиця 2.5

Опис таблиці order_items

Назва	Тип даних	Опис
id	uuid	Унікальний ідентифікатор товарної позиції в чеку
order_id	uuid	Ідентифікатор відповідного замовлення
product_variant_id	uuid	Ідентифікатор придбаної товарної варіації
quantity	integer	Кількість замовлених одиниць даної варіації
price	decimal	Зафіксована вартість на момент укладання угоди

Зворотний зв'язок від покупців акумулюється в таблиці comments. Вона містить текстовий відгук про придбаний товар, числову оцінку задоволеності та поле для офіційної відповіді від імені адміністрації платформи. Для забезпечення об'єктивності та боротьби з накрутками рейтингу товарів на рівні бази даних встановлено жорстке обмеження унікальності, яке дозволяє одному ідентифікованому користувачу залишити лише один валідний відгук до конкретної позиції каталогу.

2.3. Проєктування та реалізація алгоритмів роботи системи

Під час попереднього планування та проєктування можливостей системи було зосереджено увагу на ключових алгоритмах платформи. Одним із таких аспектів є процес обробки клієнтських запитів, для детального розуміння якого побудовано діаграму послідовності, що візуалізує оформлення замовлення з резервуванням складських залишків (рис. 2.2).

Сценарій ініціюється відправленням запиту на створення замовлення. На першому етапі відбувається строга валідація вхідних даних через об'єкт серіалізації, де перевіряється правильність формату контактної інформації та наявність мінімальної кількості обраних позицій. Після успішного проходження цієї перевірки контролер делегує виконання основної бізнес-логіки відповідному сервісному модулю.

Центральним елементом архітектурного рішення є застосування блоку атомарної транзакції. Усередині цього програмного контуру послідовно виконуються три критичні кроки. Спочатку система ідентифікує особу покупця шляхом розгалуження алгоритму на три можливі гілки: оформлення менеджером від імені клієнта, виконання запиту від авторизованого профілю або купівля анонімним відвідувачем. Далі генерується базовий запис замовлення з початковим статусом та ініціюється цикл резервування товарних одиниць.

Найбільш вагомим та технічно складним етапом є безпосереднє резервування залишків. Для кожної товарної позиції система виконує запит на читання з накладанням ексклюзивного блокування рядка на рівні бази даних.

		Кравченко К.М.			ІПЗ.КР.Б – 121 – 26– ІЗ	Арк.
		Єфремов Ю.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		34

Такий підхід унеможливлює виникнення стану гонки ресурсів [23] при паралельному оформленні покупок одного й того ж варіанту парфуму різними користувачами одночасно. Блокування діє до моменту повного фінального завершення транзакції. У випадку виявлення недостатньої кількості одиниць товару на складі генерується виключення, яке миттєво ініціює повний відкат усього транзакційного блоку, скасовуючи неповні зміни та звільняючи заблоковані ресурси БД [16].

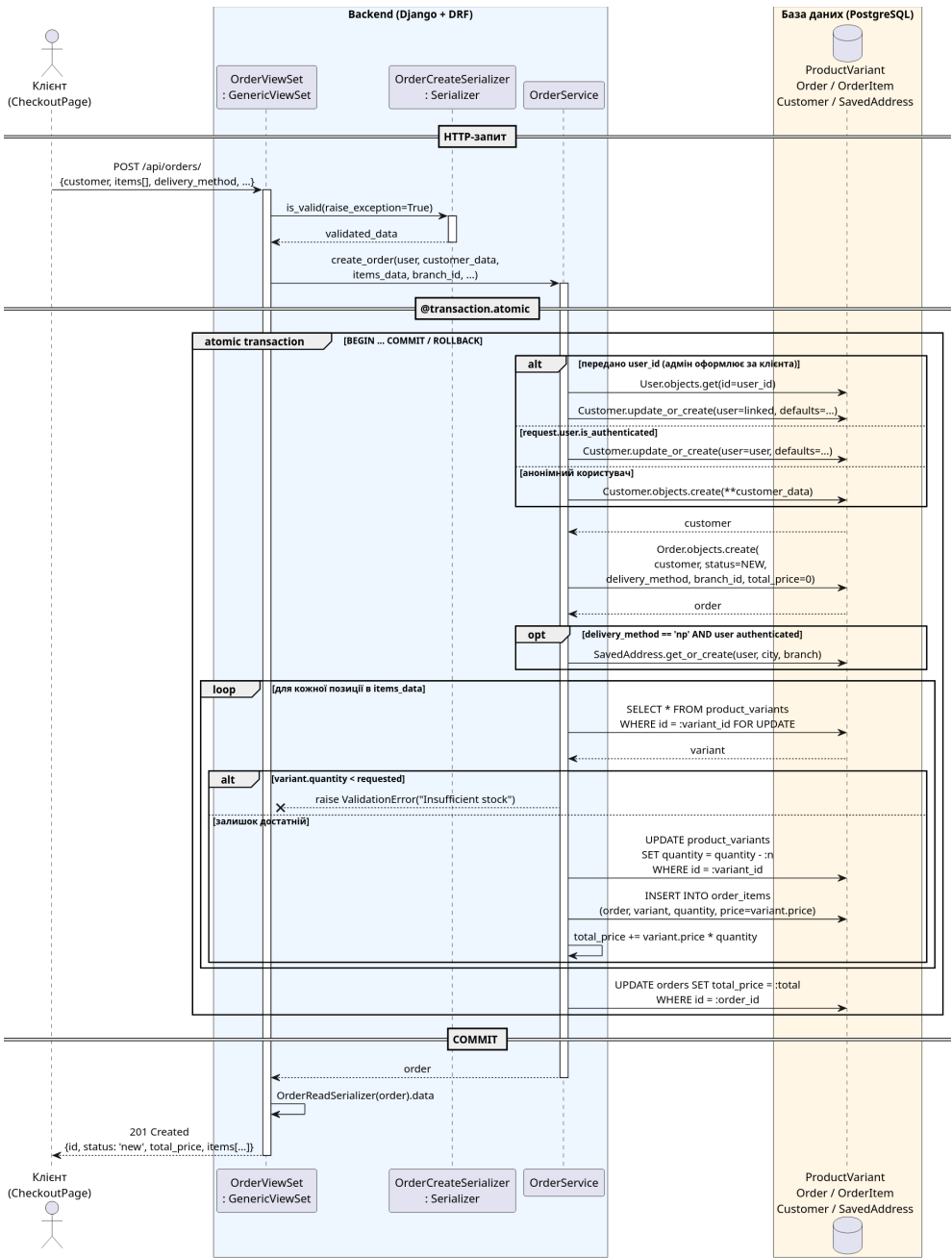


Рисунок 2.2 – Діаграма оформлення замовлення

Якщо ж перевірка наявності проходить успішно, виконується безпечне віднімання від доступної кількості товару за допомогою механізмів оновлення. Паралельно з цим генерується запис про конкретну позицію чека із жорсткою фіксацією ціни, що була актуальною саме на момент укладання угоди, що захищає історичні фінансові дані від змін у випадку майбутнього редагування вартості товару адміністратором.

Після успішного завершення циклу резервування система розраховує загальну вартість сформованого кошика, транзакція остаточно фіксується, а клієнтський додаток отримує код успішного створення ресурсу разом із деталями сформованого замовлення.

Для детального моделювання життєвого циклу сутності замовлення від моменту його створення до завершення спроектовано UML-діаграму станів, що візуалізує п'ять ключових етапів. Діаграму зображено на рис. 2.3.

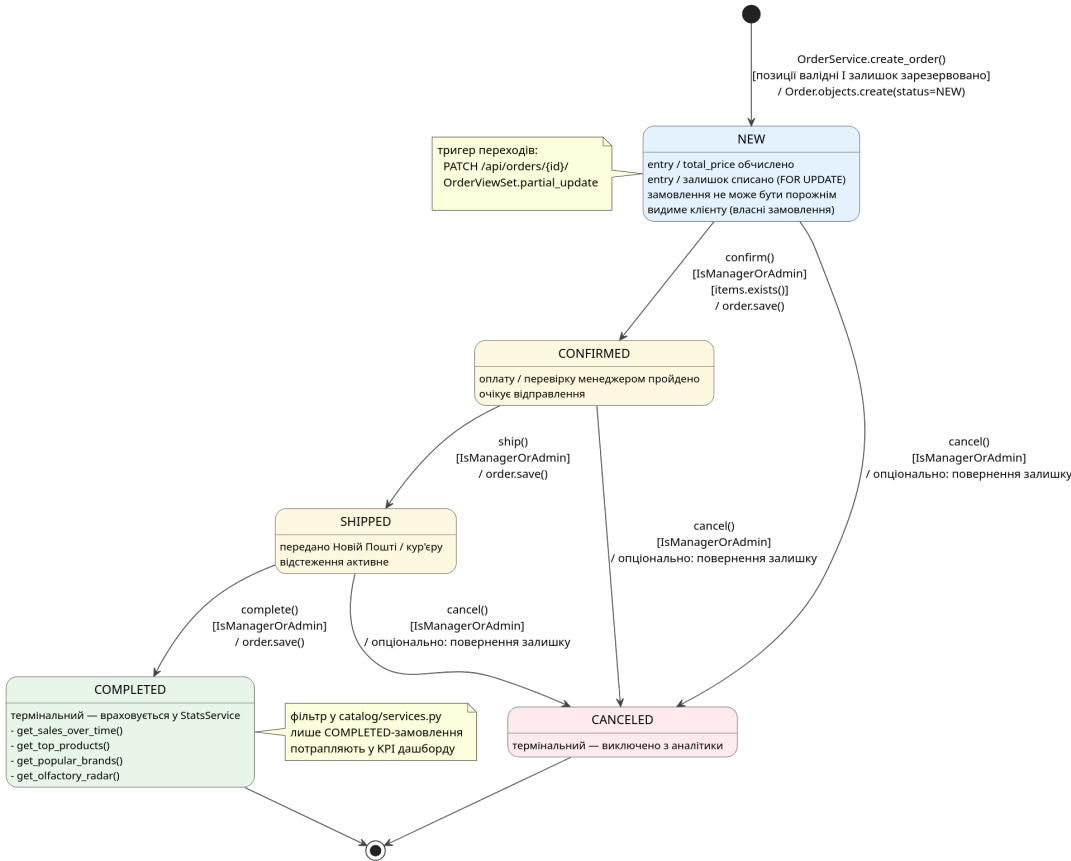


Рисунок 2.3 – Діаграма станів замовлення

Початковий перехід системи у стан нового замовлення [18] ініціюється виключно після успішного виконання програмного методу створення.

Беззаперечною умовою цього переходу є успішна валідація обраних товарних позицій та безпомилкове резервування складських залишків на рівні БД. Об'єкт потрапляє у систему вже з остаточно обчисленою загальною вартістю та гарантовано заблокованим об'ємом товару на складі.

Основна гілка життєвого циклу транзакції представляє собою ланцюжок змін статусів від нового замовлення до підтвердженого, потім відправленого і, зрештою, виконаного. Усі переходи між цими етапами логістичної обробки ініціюються операційним персоналом через відповідний інтерфейс. Для забезпечення безпеки кожен такий перехід суворо захищений авторизаційним бар'єром, який гарантує виконання дій виключно користувачами з рівнем доступу менеджера або адміністратора. Окремої уваги заслуговує перехід до статусу підтвердження, для якого на рівні бізнес-логіки діє строгий інваріант непорожнього кошика. Якщо в процесі обробки менеджер з певних причин видалив усі позиції з чека, система заблокує подальшу зміну статусу з генерацією відповідної помилки.

Паралельно з основним логістичним сценарієм передбачено альтернативну гілку скасування. Переведення замовлення в анульований стан можливе з будь-якого попереднього активного етапу обробки. Важливим архітектурним наслідком такої дії є запуск компенсаційного механізму, який передбачає обов'язкове повернення зарезервованих мілілітрів чи флаконів назад на віртуальний баланс складу для забезпечення подальшого продажу.

Вищерозглянута архітектура містить два фінальні стани: успішно завершено та скасовано. Вихід із цих станів заблоковано з метою захисту комерційної аналітики від несанкціонованої модифікації історичних даних. Відповідно до спроектованої логіки, виключно успішно завершені замовлення враховуються сервісом статистики для формування ключових показників ефективності платформи. Саме на основі цих транзакцій агрегуються дані для розрахунку загальної динаміки продажів, виявлення найбільш затребуваних парфумерних брендів, визначення топових товарних позицій та побудови ольфакторного радара [20] вподобань цільової аудиторії.

		Кравченко К.М.			ІПЗ.КР.Б – 121 – 26– ПЗ	Арк.
		Єфремов Ю.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		37

Для забезпечення безперервного користувацького досвіду та підтримання високого рівня інформаційної безпеки у системі реалізовано механізм автоматичного оновлення сесії [30]. Візуалізація цього процесу представлена на відповідній діаграмі послідовності, що описує складну взаємодію між реактивним компонентом, менеджером глобального стану та сервером під час ротації tokenів доступу.

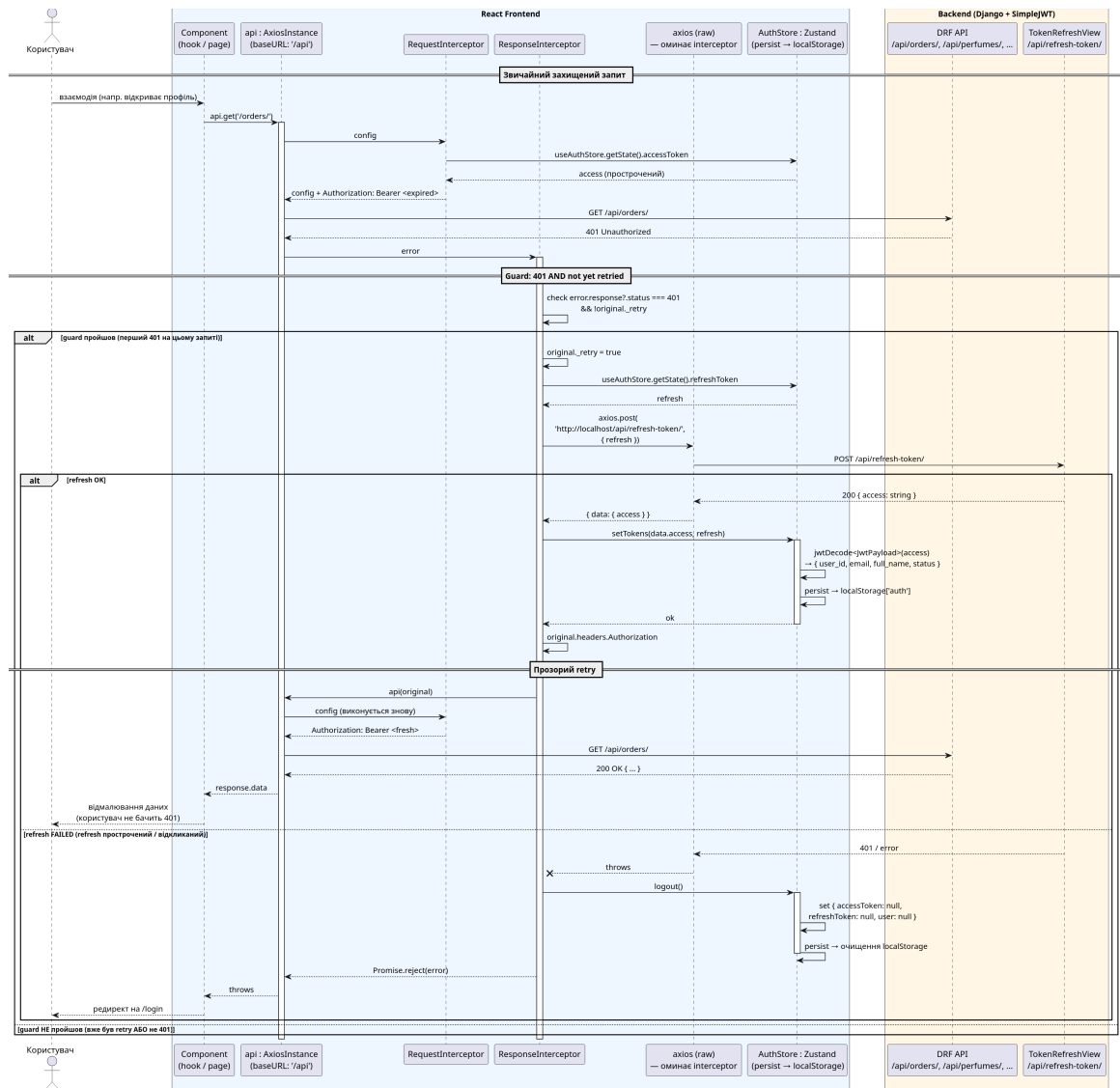


Рисунок 2.4 – Діаграма послідовності оновлення сесії

Сценарій розпочинається зі стандартного звернення клієнтського компонента до захищеного ресурсу через екземпляр мережевого клієнта. Перед безпосередньою відправкою даних спрацьовує перехоплювач запиту, який автоматично додає авторизаційний заголовок із поточним токеном, отриманим

зі сховища. У випадку, коли час життя цього ідентифікатора вичерпано, серверна частина правомірно відхиляє запит із кодом помилки про відсутність доступу.

Ключовим архітектурним елементом алгоритму є перехоплювач відповідей, який діє як єдиний центр координації сесій. Отримавши помилку авторизації, він виконує строгу перевірку умов: підтверджує відповідність статусу коду та перевіряє спеціальний прапорець повторної спроби. Наявність такого прапорця є критично важливою вимогою для захисту програмного комплексу від потрапляння у нескінченний цикл мережових запитів. Якщо умови виконуються, перехоплювач маркує поточний запит як такий, що потребує повторення, вилучає токен оновлення з локального сховища та ініціює звернення до відповідного серверного маршруту.

Важливою технічною деталлю є те, що запит на оновлення сесії виконується через окремий базовий екземпляр мережевого клієнта, який не містить підключених перехоплювачів. Таке архітектурне рішення аргументується необхідністю запобігання рекурсивного перехоплення самого запиту оновлення у випадку його невдачі.

Подальший розвиток подій розділяється на два можливі логічні шляхи. У разі успішної валідації токена оновлення сервер повертає нові дані для доступу. Перехоплювач декодує їхню структуру, синхронізує оновлену інформацію з менеджером глобального стану та локальним сховищем браузерa, після чого прозоро для користувача повторює початковий відхилений запит із вже актуальним заголовком. Якщо ж токен оновлення також виявляється простроченим або примусово відкликаним адміністратором, алгоритм ініціює процедуру повного виходу з системи. Сховище повністю очищується від персональних даних, а помилка передається на рівень відображення для перенаправлення відвідувача на сторінку ідентифікації.

Окремо передбачено механізм захисту для ситуацій, коли повторний запит із новим токеном знову завершується помилкою доступу. У такому разі перевірка прапорця повторної спроби блокує подальші дії та миттєво повертає помилку ініціатору, унеможливлуючи зациклення алгоритму.

		Кравченко К.М.			ІПЗ.КР.Б – 121 – 26– ПЗ	Арк.
		Єфремов Ю.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		39

Вищерозглянуті алгоритми будуть вкладені в основу при побудові платформи нішевої парфумерії та допоможуть попередити помилки, що виникають при подальшому архітектурному плануванні та розширенні можливостей системи.

2.4. Реалізація серверної та клієнтської логіки системи

Серверна частина розроблюваної платформи побудована на базі високорівневого вебфреймворку Django із застосуванням спеціалізованого інструментарію Django REST Framework [3] [26] для конструювання програмного інтерфейсу застосунків. Альтернативним підходом до побудови серверної логіки могло б стати використання легковагових мікрофреймворків, проте такий вибір вимагав би самостійної ручної реалізації базових механізмів безпеки, валідації та маршрутизації. Натомість інтеграція комплексного фреймворку дозволяє уникнути написання шаблонного коду та зосередитися безпосередньо на бізнес-логіці електронної комерції нішевої парфумерії.

Фундаментом організації програмного коду виступає концепція ізольованих додатків, яка передбачає розбиття системи на самодостатні модулі. Кожен такий модуль відповідає виключно за власну предметну область і має максимально низький рівень зв'язності з іншими компонентами системи. У межах програмного комплексу платформи виокремлено п'ять ключових модулів. Модуль каталогу відповідає за комплексне управління парфумерною продукцією, брендами, ароматичними нотами, варіантами об'ємів та збереженими товарами користувачів. Модуль замовлень відокремлено обслуговує процеси створення покупок, деталізацію позицій у чеку, адресну інформацію та логістичні розрахунки вартості доставки. Модуль користувачів інкапсулює моделі профілів та реалізує механізми ідентифікації і перевірки прав доступу. Модуль коментарів забезпечує незалежне функціонування системи зворотного зв'язку у вигляді відгуків покупців на придбану продукцію.

Окрему стратегічну роль відіграє технічний модуль ядра системи [28]. Цей компонент не містить специфічної комерційної логіки, натомість він акумулює

		Кравченко К.М.			ІПЗ.КР.Б – 121 – 26– ПЗ	Арк.
		Єфремов Ю.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		40

загальні абстракції та базові класи, які багаторазово наслідуються та використовуються в усіх інших частинах програми. Виокремлення загального функціоналу в незалежний структурний елемент є критично важливим архітектурним рішенням. У разі відмови від такого розділення та побудови системи як єдиного тісно зв'язаного цілого, внесення змін до логіки одного бізнес-процесу могло б призвести до каскадних помилок у суміжних підсистемах. Обраний модульний принцип гарантує високу згуртованість коду, суттєво спрощує процедуру модульного тестування та створює надійні передумови для подальшого масштабування платформи, дозволяючи за потреби трансформувати будь-який модуль в окремий мікросервіс.

Задля можливості змінювати конфігурацію проекту ключові змінні було винесено в .env файл, який доданий в .gitignore в цілях безпеки. Нижче наведено довільний вміст .env файлу:

```
NGINX_HOST=localhost
DJANGO_SECRET_KEY=dev-secret
MYSQL_DATABASE=app_db
MYSQL_USER=app_user
MYSQL_PASSWORD=app_pass
MYSQL_ROOT_PASSWORD=root_pass
MYSQL_HOST=db
MINIO_ROOT_USER=minioadmin
MINIO_ROOT_PASSWORD=minioadmin123
MINIO_BUCKET=perfume-media
```

Спроектowana архітектура платформи є результатом цілеспрямованого застосування набору класичних шаблонів проектування [24], які мають визначальний вплив на надійність та гнучкість системи. Впроваджені рішення формують цілісну інженерну конструкцію, де кожен механізм відповідає за виконання конкретних вимог щодо фінансової цілісності, безпечної обробки конкурентного доступу, захисту сесій та високої продуктивності інтерфейсу [19].

Важливу роль в організації доступу до даних відіграє шаблон репозиторію, функцію якого в обраному фреймворку виконує спеціалізований менеджер

		Кравченко К.М.			ІПЗ.КР.Б – 121 – 26– ПЗ	Арк.
		Єфремов Ю.М.				41
Змн.	Арк.	№ докум.	Підпис	Дата		

управління моделями. Цей підхід найяскравіше розкривається під час реалізації механізму логічного видалення записів. Менеджер автоматично прикріплює до кожного запиту умову фільтрації активних елементів, унаслідок чого викликаючий програмний код працює з даними абсолютно прозоро і не містить інформації про існування поля статусу видалення. Застосування цього патерну слугує класичною ілюстрацією принципу інверсії залежностей, за якого складна низькорівнева логіка фільтрації надійно інкапсулюється у виділеному об'єкті, дозволяючи зберегти чистоту та незалежність основних алгоритмів системи.

Для забезпечення стабільності програмного інтерфейсу та безпечної передачі інформації між клієнтською і серверною частинами інтегровано патерн об'єкта передачі даних [6] [25]. Роль таких об'єктів виконують серверні серіалізатори, які формують строгий контракт доступу, що функціонує абсолютно незалежно від внутрішньої реляційної схеми бази даних. Практична цінність даного підходу полягає у гарантуванні стабільності зовнішнього мережевого інтерфейсу під час етапів глибокого внутрішнього рефакторингу. Розробники отримують змогу безперешкодно змінювати назви полів чи структуру зберігання у моделях бази даних, зберігаючи при цьому незмінну та передбачувану структуру мережеских запитів і відповідей.

Розширення базового функціоналу програмних компонентів без втручання в їхній початковий вихідний код реалізовано за допомогою патерну декоратора. У межах розробленої платформи ця архітектурна концепція застосовується на двох взаємодоповнюючих рівнях. На рівні базового синтаксису мови програмування використовуються вбудовані механізми обгортання функцій, які ізольовано додають необхідну поведінку управління транзакціями бази даних та строгу перевірку прав доступу користувачів. Зазначені механізми формують послідовний ланцюжок обробників, де кожен окремий вузол наділяє поточний процес додатковими властивостями, серед яких системне логування подій, автоматичне збереження персистентних даних та глобальне перехоплення помилок виконання.

		Кравченко К.М.			ІПЗ.КР.Б – 121 – 26– ПЗ	Арк.
		Єфремов Ю.М.				42
Змн.	Арк.	№ докум.	Підпис	Дата		

Однією з фундаментальних інженерних задач під час розробки платформ електронної комерції є забезпечення цілісності даних із високою історичною та аналітичною цінністю. У процесі функціонування торгового майданчика постійно виникають типові операційні ситуації, серед яких зняття певного парфуму з продажу, припинення співпраці з конкретним брендом або деактивація облікового запису користувача. Застосування класичного підходу фізичного знищення інформації з бази даних у таких випадках є критичною архітектурною помилкою, оскільки це неминуче спричинить каскадне руйнування пов'язаної історії замовлень, викривлення фінансових звітів та порушення роботи модулів аналітичних агрегацій.

Для запобігання описаним ризикам у системі запроваджено строгий архітектурний інваріант логічного видалення, який діє на рівні базової програмної абстракції. В ядрі програмного комплексу створено абстрактну модель, яка успадковується іншими сутностями та автоматично додає до них логічний маркер статусу видалення разом із міткою точного часу здійснення цієї операції. Відмова від безповоротного знищення рядків таблиць гарантує надійне збереження комерційної інформації та створює можливість відновлення випадково прихованих даних.

Прозорість цього процесу для клієнтської частини досягається шляхом перевизначення стандартного менеджера управління базою даних, який автоматично відфільтровує неактивні записи ще на етапі побудови запитів. Завдяки цьому інженерному рішенню зняті з продажу товари миттєво приховуються від відвідувачів у публічному каталозі, проте залишаються повністю доступними для адміністративного персоналу через розширений інтерфейс доступу до всіх об'єктів системи. Додатковим рівнем захисту виступає механізм перевірки цілісності зв'язків, який ініціює помилку валідації та блокує спроби логічного приховування базових класифікаторів, таких як парфумерні бренди чи ароматичні ноти. Лістинг абстрактної моделі логічного видалення та її кастомного менеджера наведено у Додатку А.

		Кравченко К.М.			ІПЗ.КР.Б – 121 – 26– ПЗ	Арк.
		Єфремов Ю.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		43

Створення замовлення є найскладнішою серверною операцією в системі, оскільки вона охоплює одночасну взаємодію з кількома таблицями та повинна гарантовано виконуватися в умовах конкурентного доступу. У реальному середовищі експлуатації електронної комерції виникає технічна проблема стану гонитви. Це вразливість, за якої два незалежні покупці майже одночасно намагаються придбати останню доступну одиницю специфічного товару. За умови наївної лінійної реалізації алгоритму обидва запити можуть успішно пройти валідацію залишків, що призведе до підтвердження обох покупок та формування від'ємного балансу на складі.

Для усунення цього класу проблем логіку оформлення замовлення повністю інкапсульовано у спеціалізований сервісний об'єкт та реалізовано як єдину атомарну операцію з використанням ексклюзивного блокування рядків бази даних. На початковому етапі весь алгоритм програмно обгортається у транзакційний контекст. Це змушує систему управління базами даних відкрити транзакцію перед виконанням першого звернення та автоматично закрити її фіксацією змін у разі успішного проходження всіх кроків. При виникненні будь-якої помилки валідації ініціюється повне скасування транзакції. Застосований принцип все або нічого гарантує, що неможливе виникнення часткового стану системи, за якого замовлення створюється без відповідного декременту залишків або без запису окремих позицій чека.

Програмна реалізація починається з отримання валідованих даних та генерації базового об'єкта замовлення з початковим статусом обробки. На цьому кроці система лише резервує унікальний ідентифікатор транзакції без фіксації підсумкової суми та переліку товарів.

Найбільш критичним етапом є ітерація за списком обраних клієнтом товарних позицій. Для кожного варіанту парфуму виконується спеціалізований блокувальний запит до бази даних. Семантика цього механізму полягає в тому, що потрібний рядок товару зчитується із одночасним накладанням жорсткого ексклюзивного блокування на рівні рушія бази даних. Будь-який інший паралельний запит, спрямований на модифікацію або ексклюзивне читання

		Кравченко К.М.			ІПЗ.КР.Б – 121 – 26– ПЗ	Арк.
		Єфремов Ю.М.				44
Змн.	Арк.	№ докум.	Підпис	Дата		

цього ж рядка, буде безумовно переведений у стан очікування до моменту завершення поточної транзакції.

Завдяки описаному підходу при одночасному зверненні кількох клієнтів до останнього флакона перша транзакція захоплює блокування, виконує перевірку, списує залишок та зберігає зміни. Друга транзакція весь цей час перебуває в очікуванні, а після розблокування зчитує вже оновлений рядок із нульовим залишком і безпомилково переривається помилкою валідації.

Усередині заблокованої секції виконується перевірка наявності достатньої кількості товару. У разі виявлення дефіциту генерується системне виключення, яке автоматично призводить до відкату всієї транзакції. Це означає, що попередньо створений базовий запис замовлення також безповоротно видаляється з бази.

Безпосереднє зменшення складського залишку реалізовано за допомогою спеціальних виразів оновлення на стороні бази даних. Замість вразливої схеми зчитування поточного значення в оперативну пам'ять сервера, виконання віднімання та зворотного запису, сервер формує декларативну команду арифметичної операції. За цією командою віднімання замовленої кількості виконується самим рушієм бази даних. Це створює додатковий рівень захисту від конкурентної модифікації та зменшує кількість мережових звернень до сховища.

На фінальних кроках алгоритм створює прив'язані записи позицій кошика із жорсткою фіксацією ціни продажу, підраховує загальну вартість сформованого замовлення та оновлює відповідний атрибут базового запису. Після цього транзакція успішно фіксується, мережеві блокування знімаються, а клієнтський додаток отримує сформовану відповідь.

Аналогічний архітектурний підхід застосовується для функціоналу додавання нових позицій до існуючого замовлення через адміністративний інтерфейс, що гарантує одноманітність та надійність обробки складських операцій. Лістинг сервісного методу створення замовлення наведено у Додатку Б.

		Кравченко К.М.			ІПЗ.КР.Б – 121 – 26– ПЗ	Арк.
		Єфремов Ю.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		45

Характерною особливістю функціонування будь-якого майданчика електронної комерції є динамічне ціноутворення. Вартість товарних позицій постійно змінюється під впливом інфляційних процесів, маркетингових акцій або регулярної переоцінки складських запасів. Незважаючи на це, сформоване клієнтом замовлення виступає як первинний фінансовий документ, який повинен гарантовано зберігати абсолютну незмінність. Сума укладеної транзакції не може підлягати автоматичному перерахунку при перегляді історії покупок через певний проміжок часу, навіть за умови суттєвої зміни актуальної вартості товару в загальному каталозі.

Для забезпечення цього суворого інваріанту в архітектуру бази даних та сервісного шару платформи інтегровано патерн фіксації знімка ціни. Програмна сутність позиції замовлення містить власне поле ціни, реалізоване з використанням спеціалізованого типу даних із фіксованою точністю. Використання саме такого типу повністю усуває типові апаратні похибки, які притаманні числам із рухомою комою під час виконання точних фінансових розрахунків. Зазначене поле принципово ізольоване від поля вартості у базовій моделі товарної варіації, оскільки воно призначене для статичного збереження ціни виключно на момент створення позиції в чеку.

Механізм наповнення цього поля інкапсульовано безпосередньо у метод збереження моделі позиції замовлення. Програмний алгоритм працює таким чином, що під час ініціалізації збереження об'єкта в базу даних спочатку перевіряється наявність уже заданої вартості. Якщо значення відсутнє, система автоматично копіює актуальну ціну з відповідного варіанту парфуму у власне поле позиції. У випадку, коли ціна вже була явно визначена під час імпорту історичних даних або при ручному коригуванні через панель адміністратора, алгоритм пропускає етап копіювання і безпечно зберігає існуюче значення.

Такий підхід надає багато переваг - по-перше, процеси генерації електронних чеків та побудови комплексних аналітичних звітів стають максимально оптимізованими. Для обчислення загальної суми продажів достатньо виконати швидке агрегаційне додавання полів у межах однієї таблиці,

		Кравченко К.М.			ІПЗ.КР.Б – 121 – 26– ПЗ	Арк.
		Єфремов Ю.М.				46
Змн.	Арк.	№ докум.	Підпис	Дата		

уникаючи необхідності виконання важких об'єднань із таблицями історії зміни цін. По-друге, така архітектура [13] автоматично задовольняє жорсткі бухгалтерські та податкові вимоги до первинної документації, перетворюючи кожне окреме замовлення на повністю самодостатній інформаційний об'єкт. По-третє, суттєво спрощується логіка проведення операцій повернення коштів, адже при скасуванні покупки система завжди повертає клієнту саме ту суму, яка була надійно зафіксована у конкретній позиції замовлення, повністю ігноруючи поточні ринкові коливання.

Лістинг реалізації механізму фіксації знімка ціни наведено у Додатку В.

Специфіка платформи полягає в тому, що одна парфумерна композиція може реалізовуватися у різних об'ємах, кожен з яких характеризується власною ціною та індивідуальним складським обліком. Для точного відображення цієї бізнес-моделі на рівні схеми бази даних застосовано відношення один-до-багатьох, де єдиній сутності парфуму підпорядковується множина товарних варіацій. Саме конкретна варіація, а не базовий аромат, виступає самостійною одиницею складського та фінансового обліку. Для зберігання інформації про наявний залишок використовується цілочисельний тип даних із жорстким вбудованим обмеженням виключно додатних значень.

Процес зменшення складських запасів суворо інкапсульовано в єдиній точці доступу — сервісному об'єкті створення або редагування замовлення. Операція декременту виконується виключно в межах транзакційної секції з накладанням ексклюзивного блокування на рядок бази даних та застосуванням атомарних виразів оновлення. Такий комплексний підхід гарантує абсолютну неможливість продажу товару понад його фактичну наявність на складі. Перед безпосереднім списанням алгоритм виконує явну перевірку достатності зарезервованого об'єму. У разі виявлення дефіциту система генерує помилку валідації рівня бізнес-логіки. Формування зрозумілого семантичного повідомлення про нестачу конкретної позиції покращує користувацький досвід, оскільки клієнтський інтерфейс може коректно відобразити суть проблеми та

		Кравченко К.М.			ІПЗ.КР.Б – 121 – 26– ПЗ	Арк.
		Єфремов Ю.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		47

запропонувати альтернативні варіанти замість демонстрації незрозумілої технічної помилки бази даних.

Збільшення залишків, або повернення продукції на віртуальний склад, ініціюється в окремих логістичних сценаріях, таких як скасування транзакції адміністратором або повернення товару клієнтом. Ці операції також обгортаються у транзакційний контур і використовують механізм атомарного оновлення на стороні бази даних. Однак, на відміну від процесу списання, інкремент виконується без застосування ексклюзивного блокування рядків. Це рішення обґрунтовується тим, що математична операція додавання принципово не створює ризику від'ємного переповнення, тому блокування доступу для інших паралельних запитів є надлишковим і лише безпідставно знижуватиме загальну пропускну здатність системи.

Окремо варто відзначити концептуальне спрощення спроектованої доменної моделі. У системі відсутні виокремлені сутності складського приміщення чи окремої товарної партії. Товарна варіація самостійно виконує роль як номенклатурної позиції, так і облікової одиниці. Це свідомий інженерний компроміс, який повністю виправдовується характером обраної бізнес-стратегії. Розроблена платформа орієнтована на торгівлю через єдиний центральний розподільчий центр, а не на управління багатоскладовою логістикою з кількома фізичними локаціями. У разі виникнення майбутньої потреби масштабування до складної мультискладської моделі поточна архітектура дозволить безболісно розширити структуру шляхом простого введення проміжної сутності рівня запасів без необхідності руйнування та переписування базових абстракцій каталогу продукції. Лістинг програмної реалізації скінченного автомата статусів замовлення наведено у Додатку Г.

Управління життєвим циклом транзакцій вимагає суворого контролю над етапами їх виконання. Замість використання вільного текстового поля для фіксації поточного стану, розроблена система моделює поведінку замовлення як скінченний автомат із п'ятьма етапами. Для реалізації цієї архітектури застосовано механізм Enum із типізованими значеннями.

		Кравченко К.М.			ІПЗ.КР.Б – 121 – 26– ПЗ	Арк.
		Єфремов Ю.М.				48
Змн.	Арк.	№ докум.	Підпис	Дата		

Життєвий цикл об'єкта розпочинається з переходу до початкового стану нового замовлення. Цей перехід ініціюється системою автоматично, але виключно за умови успішного завершення транзакції створення. Це гарантує, що об'єкт потрапляє до загального пулу обробки з уже обчисленою підсумковою вартістю та надійно зарезервованими складськими залишками. Основна гілка алгоритму представляє собою лінійну прогресію, яка охоплює етапи підтвердження, логістичного відправлення та фінального виконання. Зміна цих етапів здійснюється операційним персоналом.

Для забезпечення високого рівня безпеки комерційної діяльності кожен перехід між станами захищений строгим авторизаційним бар'єром. Він дозволяє модифікувати статуси лише користувачам із відповідними привілеями адміністратора або менеджера. Окрему увагу при проєктуванні було приділено підтримці бізнес-інваріантів. Зокрема, перехід від нового замовлення до підтвердженого жорстко контролюється на рівні ядра системи: якщо в процесі обробки чек з певних причин залишився порожнім, подальша зміна статусу блокується з генерацією відповідної помилки валідації.

Паралельно з основним логістичним сценарієм функціонує альтернативна гілка скасування транзакції. Переведення замовлення в анульований стан допускається з будь-якого попереднього активного етапу обробки. Критично важливим наслідком такої дії є запуск вбудованого компенсаційного механізму. Цей алгоритм ініціює зворотну транзакцію, автоматично повертаючи зарезервовані об'єми парфумерної продукції на віртуальний складський баланс, чим забезпечує їх доступність для інших покупців та запобігає безпідставному заморожуванню товарних активів.

Спроектований скінченний автомат містить два кінцеві стани, які свідчать про успішне виконання або остаточне скасування угоди. Архітектура програми категорично блокує будь-які подальші переходи з цих етапів. Зазначене обмеження впроваджено цілеспрямовано для захисту комерційної аналітики від несанкціонованої чи випадкової модифікації історичних даних минулих періодів. Строга формалізація станів відіграє вирішальну роль для модуля

		Кравченко К.М.			ІПЗ.КР.Б – 121 – 26– ПЗ	Арк.
		Єфремов Ю.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		49

статистичних розрахунків. Усі ключові показники ефективності торгового майданчика агрегуються виключно на основі успішно завершених угод. Саме в момент переходу до цього термінального статусу замовлення починає впливати на показники загальної динаміки продажів, формування рейтингів топових товарів та визначення ольфакторних вподобань аудиторії.

Для оптимізації продуктивності та усунення затримок під час виконання ресурсомістких аналітичних обчислень відповідне поле статусу було проіндексовано на рівні схеми бази даних. Лістинг програмної реалізації статусів замовлення наведено у Додатку Д.

Висновки до другого розділу

У другому розділі було здійснено комплексне проєктування платформи для управління продажами нішевої парфумерії. На основі аналізу вимог користувачів визначено ключових акторів та змодельовано основні варіанти використання системи. Це дозволило сформувати об'єктно-орієнтовану структуру майданчика. Для надійного збереження комерційної інформації розроблено реляційну базу даних. Важливим архітектурним рішенням стала побудова схеми даних із розділенням базових парфумерних композицій та конкретних товарних варіацій, а також впровадження механізму логічного видалення записів для збереження історичної цілісності фінансових звітів.

Значну увагу було приділено проєктуванню складних алгоритмів роботи торгового майданчика. За допомогою інструментів візуального моделювання формалізовано критично важливі процеси електронної комерції. Зокрема, розроблено алгоритм управління статусами покупок та створено безпечний механізм прозорості ротації токенів авторизації клієнтів.

Під час опису реалізації серверної та клієнтської логіки обґрунтовано вибір застосованих інженерних концепцій із фокусом на семантику процесів. Застосування модульної архітектури, ізоляція бізнес-логіки та впровадження класичних патернів проєктування забезпечили високу надійність системи. Інтеграція механізму створення статичного знімка ціни для кожної позиції в чеку

		Кравченко К.М.			ІПЗ.КР.Б – 121 – 26– ПЗ	Арк.
		Єфремов Ю.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		50

гарантувала захист фінансової історії клієнтів від майбутніх коливань вартості товарів у каталозі. У сукупності всі ухвалені архітектурні рішення утворюють міцний концептуальний фундамент для стабільного функціонування та подальшого масштабування розробленого програмного комплексу.

		Кравченко К.М.			ІПЗ.КР.Б – 121 – 26– ІЗ	Арк.
		Єфремов Ю.М.				51
Змн.	Арк.	№ докум.	Підпис	Дата		

РОЗДІЛ 3. ІНТЕРФЕЙС ТА ПОРЯДОК РОБОТИ З ПЛАТФОРМОЮ ДЛЯ УПРАВЛІННЯ ПРОДАЖАМИ НІШЕВОЇ ПАРФУМЕРІЇ

3.1. Порядок встановлення та налаштування параметрів додатку

Для забезпечення стабільного функціонування розробленого програмного комплексу необхідно дотримуватися чітко визначених апаратних, системних та конфігураційних вимог.

Для повноцінної роботи програмного додатку було визначено технічні характеристики апаратного та системного забезпечення сервера. Насамперед для розгортання платформи необхідний процесор, що має щонайменше чотири обчислювальні ядра. Також для швидкої обробки запитів сервер повинен мати від 8 ГБ оперативної пам'яті. Для збереження реляційної бази даних MySQL, медіафайлів у сховищі MinIO та образів контейнерів на жорсткому диску потрібно виділити мінімум 20 ГБ вільного простору. Операційною системою на сервері має бути Linux, наприклад Ubuntu 22.04 LTS, або інше середовище, яке підтримує роботу з технологією Docker. Окрім цього, для сервера потрібне стабільне підключення до мережі Інтернет. Це обов'язкова умова для взаємодії веб-сайту із сервісом авторизації Google, платіжною системою LiqPay та API служби доставки «Нова Пошта». Для того, щоб проксі-сервер Nginx міг коректно приймати запити від користувачів, на сервері мають бути відкриті мережеві порти 80 та 443.

Для кращого схематичного розуміння усіх важливих сервісів та модулів системи було розроблено діаграми розгортання платформи, що зображено на рис. 3.1.

Процес розгортання платформи [14] суворо регламентований та складається з певної послідовності дій. Насамперед на цільовому сервері потрібно встановити технологію Docker та інструмент Docker Compose. Після цього необхідно завантажити актуальний програмний код проєкту з віддаленого репозиторію.

		Кравченко К.М.			ІПЗ.КР.Б – 121 – 26– ПЗ	Арк.
		Єфремов Ю.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		52

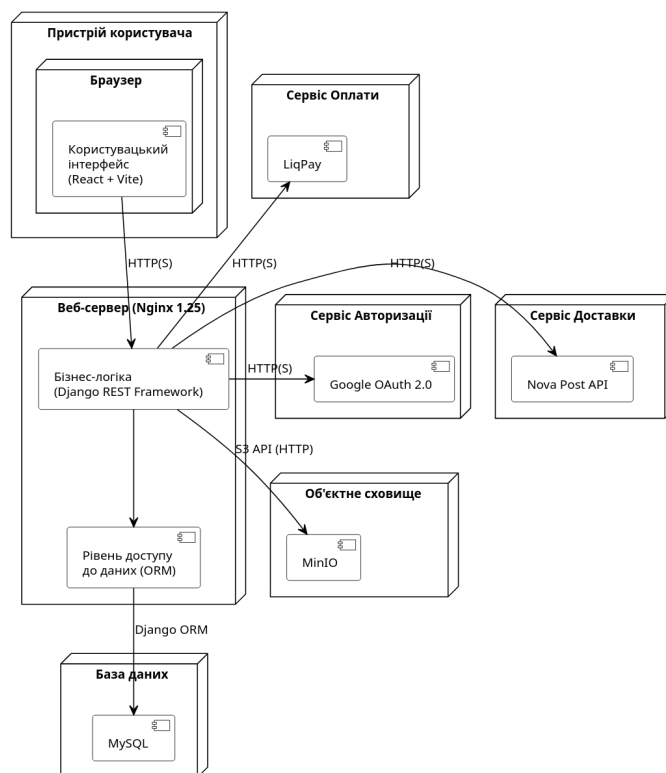


Рисунок 3.1 – Діаграма розгортання системи

Процес розгортання платформи суворо регламентований та складається з певної послідовності дій. Задля управління багатоконтейнерною інфраструктурою застосовується інструмент оркестрації Docker Compose, який дозволяє описати всі необхідні сервіси, їхні параметри та мережеві взаємозв'язки в єдиному файлі. Насамперед на цільовому сервері потрібно встановити актуальні версії технології Docker та інструменту Docker Compose. Після цього необхідно клонувати програмний код проєкту з віддаленого репозиторію.

Окрему увагу під час конфігурації необхідно приділити налаштуванню змінних середовища для серверної частини, яка побудована на базі фреймворку Django. У відповідному локальному конфігураційному файлі задаються секретні ключі безпеки, параметри режиму налагодження та перелік дозволених мережевих хостів. Для забезпечення коректної взаємодії бекенду з реляційною базою даних вказуються реквізити доступу до MySQL, зокрема ім'я бази, ім'я користувача, пароль та мережевий порт підключення. Також у цьому файлі прописуються параметри для авторизації в об'єктному сховищі MinIO, де

зберігатимуться фотографії товарів, логотипи брендів та інший статичний медіаконтент.

Після підготовки параметрів серверної частини здійснюється конфігурація клієнтського додатка та зовнішніх інтеграцій. Для фронтенду, розробленого за допомогою бібліотеки React, необхідно вказати базову адресу для мережових звернень до прикладного програмного інтерфейсу сервера. Окрім цього, для коректної роботи системи ідентифікації користувачів потрібно зареєструвати додаток у сервісі Google Cloud Console та перенести отримані ключі авторизації до налаштувань проєкту. Щоб інтернет-магазин мав змогу приймати онлайн-оплати від клієнтів, створюється обліковий запис мерчанта у платіжній системі LiqPay, звідки отримуються та додаються відповідні публічний і приватний криптографічні ключі. Аналогічно отримується API-ключ від логістичної компанії «Нова Пошта» для активації функціоналу пошуку відділень та розрахунку вартості доставки під час оформлення замовлення.

Важливим аспектом етапу розгортання є налаштування збереження даних при можливих перезапусках системи. Оскільки програмні контейнери є тимчасовими за своєю природою, для реляційної бази даних та об'єктного сховища налаштовуються постійні томи (volumes). Це гарантує, що вся накопичена інформація про комерційні транзакції та завантажені графічні матеріали не буде втрачена під час оновлення програмного забезпечення.

Після заповнення всіх необхідних параметрів безпосередній запуск системи здійснюється за допомогою однієї консольної команди оркестратора Docker Compose. Цей інструмент автоматично створює та запускає в ізольованих контейнерах усі необхідні компоненти. Розгорнута інфраструктура платформи складається з п'яти основних сервісів, кожен з яких виконує чітко визначену роль. Першим у системі ініціалізується контейнер бази даних MySQL, який відповідає за постійне збереження всієї текстової та транзакційної інформації платформи. Наступним запускається серверна частина на базі Django, яка працює під управлінням веб-сервера Gunicorn і виконує обробку бізнес-логіки інтернет-магазину. Клієнтська частина додатку розгортається у власному контейнері за

		Кравченко К.М.			ІПЗ.КР.Б – 121 – 26– ІЗ	Арк.
		Єфремов Ю.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		54

допомогою інструменту збірки Vite, забезпечуючи швидку взаємодію з користувачами через графічний інтерфейс. Для зберігання неструктурованих даних активується окремий контейнер сховища MinIO. Всі зовнішні HTTP-запити до системи приймає та маршрутизує проксі-сервер Nginx, який розподіляє трафік між клієнтським додатком і сервером прикладного програмного інтерфейсу.

Після успішної побудови та фонового запуску всіх контейнерів необхідно виконати міграції бази даних, що дозволяє автоматично створити необхідні реляційні таблиці, індекси та зв'язки згідно з розробленими моделями предметної області. Наступним кроком є створення облікового запису головного адміністратора системи, який матиме розширені права для управління каталогом ароматів, доступу до аналітики продажів та зміни статусів замовлень.

Весь процес розгортання завершується комплексною перевіркою працездатності створеного програмного комплексу. Адміністратор перевіряє доступність клієнтського веб-додатка через браузер, а також коректність функціонування серверного інтерфейсу за допомогою автоматично згенерованої сторінки документації Swagger, яка дозволяє виконати тестові запити до бекенду. Обов'язковою є перевірка графічного веб-інтерфейсу управління MinIO для підтвердження успішного доступу до віртуального контейнера медіафайлів. Для фінальної верифікації здійснюється тестова реєстрація користувача на платформі, додавання товару до кошика та оформлення пробного замовлення з перевіркою роботи логістичних і платіжних модулів. Успішне проходження всіх перелічених кроків свідчить про повну готовність спроектованої веб-платформи до стабільної експлуатації.

3.2. Структура інтерфейсу додатку

При вході на сайт користувач бачить каталог товарів платформи нішевої парфумерії, де має можливість переглядати доступні до замовлення товари (рис. 3.2).

		Кравченко К.М.			ІПЗ.КР.Б – 121 – 26– ІЗ	Арк.
		Єфремов Ю.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		55

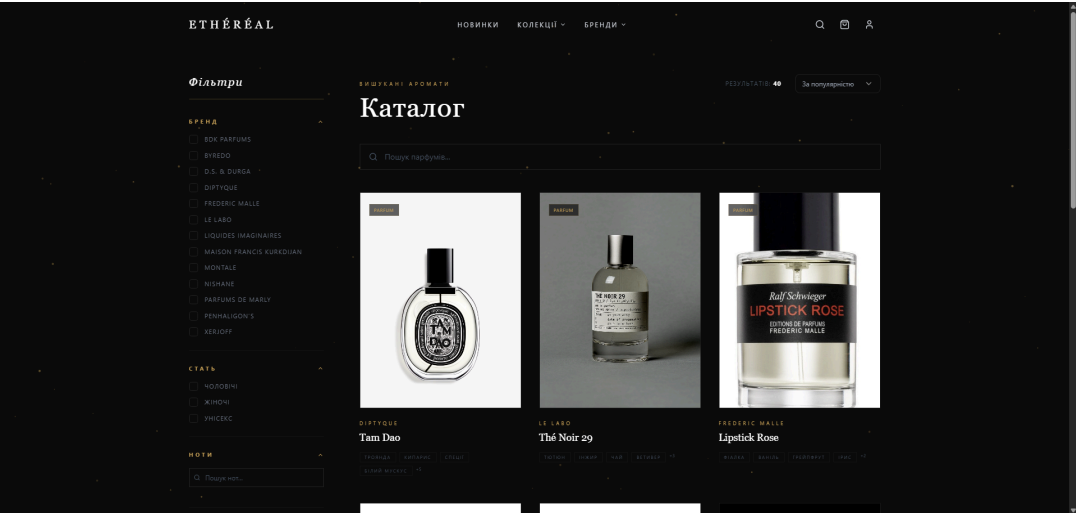


Рисунок 3.2 – Каталог товарів

Звідси користувач вже має можливість користуватися платформою для здійснення пошуку товару та оформлення замовлення. Велика кількість фільтрів за брендами, нотами, статтю, популярністю тощо дозволяють ефективно шукати бажаний перелік парфумів(рис. 3.3):

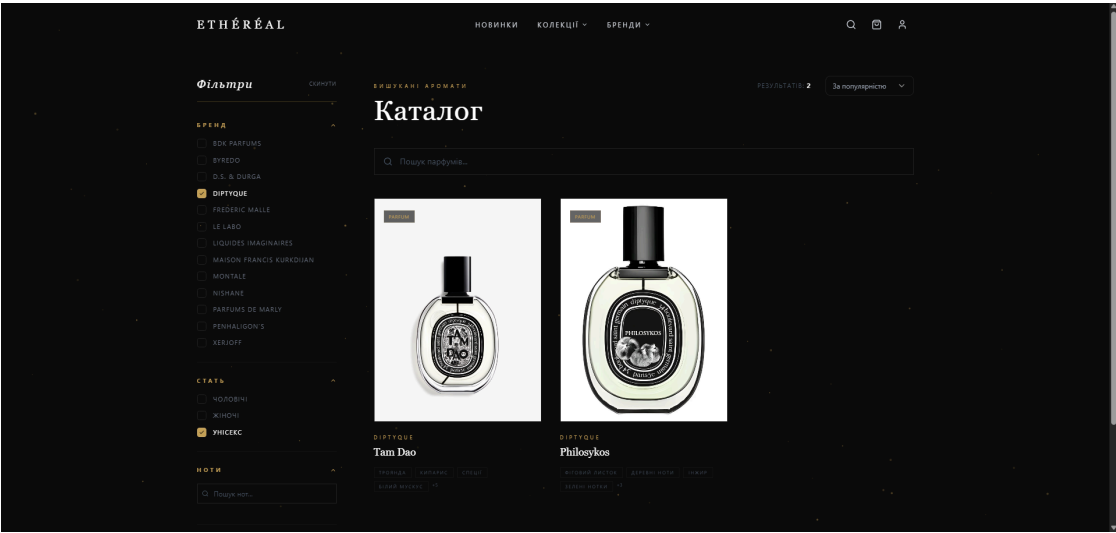


Рисунок 3.3 – Застосування фільтрів

При натисканні на товар покупця спрямує на сторінку перегляду обраного видк продукції. Приклад детальної інформації про товар зображено на рис. 3.4.

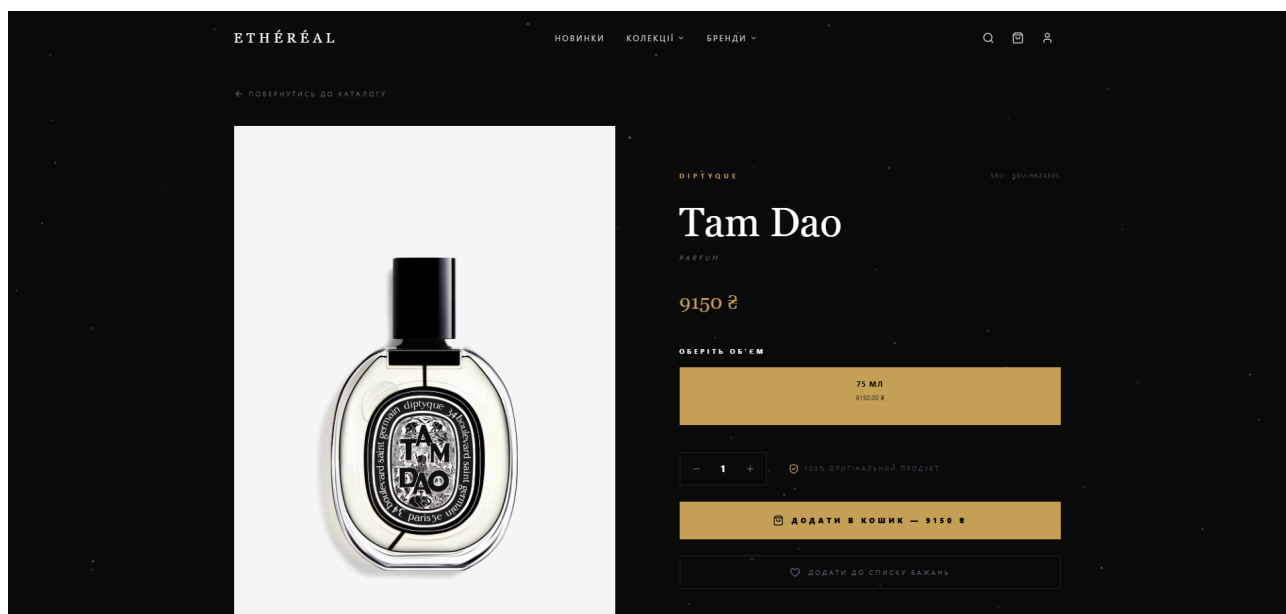


Рисунок 3.4 – Перегляд товару

Тут користувач може переглянути фото, обрати об'єм, а також кількість товару, додати його в список бажаного на майбутнє або ж покласти його до кошика (рис. 3.5). Операції з додавання-віднімання кількості продукції або ж її видалення з кошика можна також виконати і у кошику.

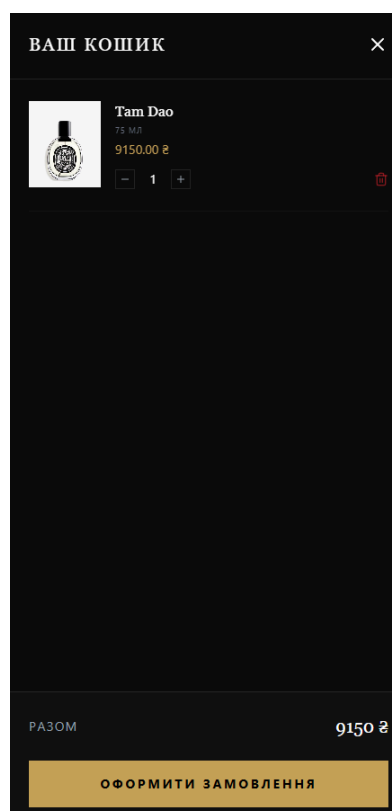


Рисунок 3.5 – Перегляд кошика

		Кравченко К.М.			ІПЗ.КР.Б – 121 – 26– ІЗ	Арк.
		Єфремов Ю.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		57

Після ознайомлення та погодження з вмістом корзини користувач має змогу оформити замовлення. Процес поділено на дві частини – спершу потрібно ввести контактну інформацію (рис. 3.6):

Рисунок 3.6 – Заповнення контактної інформації

У наступному етапі сайт просить обрати тип та адресу для успішної доставки (рис. 3.7):

Рисунок 3.7 – Заповнення інформації про доставку

Крім здійснення покупок, клієнт також має можливість створити обліковий запис для оформлення нових замовлень, вигляд панелі авторизації зображено на рис. 3.8.

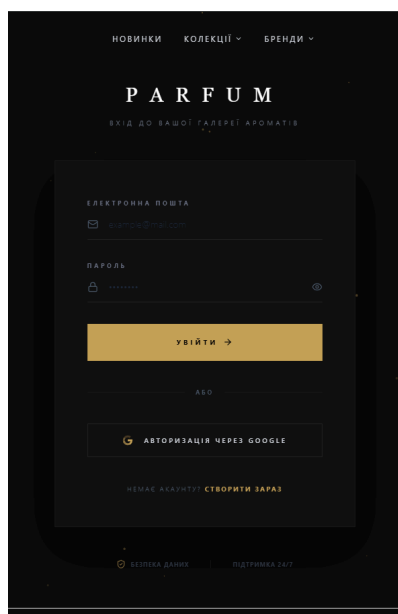


Рисунок 3.8 – Вікно авторизації

Зареєстрований користувач має додаткові переваги у вигляді можливості перегляду власної історії покупок, а також здійснення налаштувань акаунту – зміни імені, адреси за замовчуванням тощо (рис. 3.9).

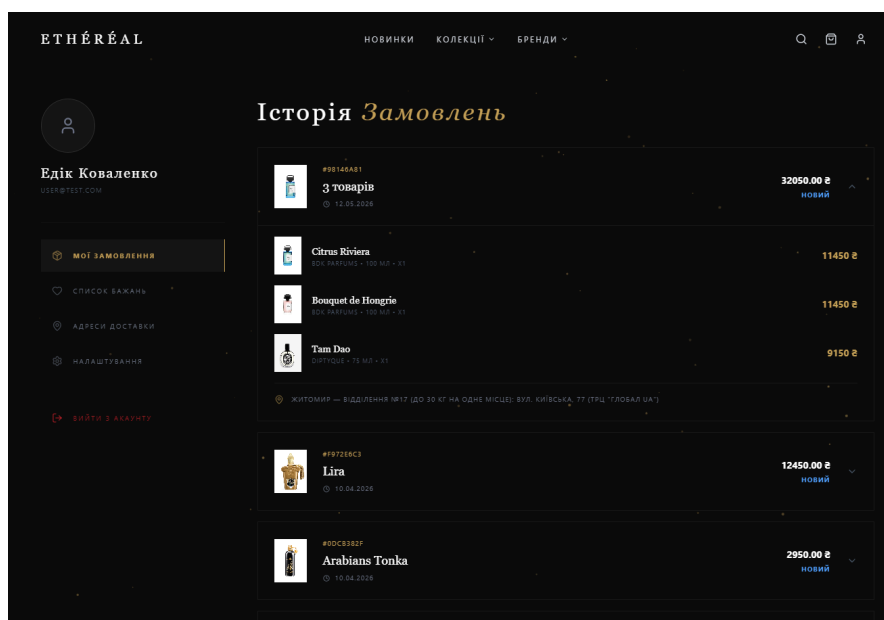


Рисунок 3.9 – Історія замовлень

Будь-яке оформлене замовлення далі потрапляє на обробку до менеджера веб-сайту. На рис. 3.10 зображено замовлення, що було здійснено у попередніх кроках.

		Кравченко К.М.			ПЗ.КР.Б – 121 – 26– ПЗ	Арк.
		Єфремов Ю.М.				59
Змн.	Арк.	№ докум.	Підпис	Дата		

Замовлення #98146a81

ДАНІ КЛІЄНТА

Андрій Коваленко

+380657847542

ДОСТАВКА

Житомир

Відділення №17 (до 30 кг на одне місце): вул. Київська, 77 (ТРЦ "Глобал UA")

СТАТУС

Підтверджено

ТОВАРИ

Citrus Riviera — BDK Parfums 100ml	1	11450.00 грн	×
Bouquet de Hongrie — BDK Parfums 100ml	1	11450.00 грн	×
Tam Dao — Diptyque 75ml	1	9150.00 грн	×
Разом		32050.00 грн	

Пошук парфуму...

1

+ Додати

Скасувати

Зберегти

Рисунок 3.10 – Обробка замовлення

Менеджер має можливість змінювати статус замовлення, чим дозволяє просувати його на подальшу обробку аж до успішного завершення циклу. Додатковим функціоналом є властивість додавати до переліку інші товари на прохання користувача, якщо він бажає отримати його в межах даного замовлення.

Крім обробки замовлень, в перелік зони відповідальності менеджера входить процес додавання нових брендів парфумів (рис. 3.11), що містить логотип, назву, країну походження та короткий опис.

		Кравченко К.М.			ІПЗ.КР.Б – 121 – 26– ПЗ	Арк.
		Єфремов Ю.М.				60
Змн.	Арк.	№ докум.	Підпис	Дата		

Додати бренд ×

Логотип

Назва

Країна

Опис

Скасувати **Додати**

Рисунок 3.11 – Додавання бренду

Також є доступ до додавання нових нот, що допомагають користувачам краще зрозуміти ольфакторність парфумів та їх аромат (рис. 3.12):

Додати ноту ×

Назва

Тип

Базова

Скасувати **Додати**

Рисунок 3.12 – Додавання нот

Внесення та редагування даних про товар також є в переліку можливостей менеджера системи (рис. 3.13), що грає ключову роль у пошуковій оптимізації та формуванні початкових вражень у потенційного покупця.

		Кравченко К.М.			ІПЗ.КР.Б – 121 – 26– ІЗ	Арк.
		Єфремов Ю.М.				61
Змн.	Арк.	№ докум.	Підпис	Дата		

Редагувати товар

Назва

Intense Café

Опис

Кавово-ванільний аромат із трояндою: один із найпопулярніших гурманських ароматів.

Концентрація

Parfum

Стать

Жіночий

Бренд

Montale

Ноти

Білий мускус

Ваніль

Кава

Квіткові ноти

Троянда

Зображення

Додати зображення



Скасувати

Зберегти

Рисунок 3.13 – Додавання товару

Так як кожен товар може містити різні об'єми, ціни, а також має властивість закінчуватись і потребує поповнення – усі ці потреби на платформі закриває розділ “Склад”. Приклад вікна зі складською інформацією про парфум зображений на рис. 3.14.

Варіанти — Intense Café

ОБ'ЄМ	ЦІНА	КІЛЬКІСТЬ	SKU		
20 ml	2850.00 грн	9	SKU-D5R2T7Y		
100 ml	6900.00 грн	5	SKU-E8U4I1O		

Додати варіант

Об'єм (ml)

Ціна

Кількість

SKU

Додати

Рисунок 3.14 – Складська інформація про товар

		Кравченко К.М.			ПІЗ.КР.Б – 121 – 26– ПЗ	Арк.
		Єфремов Ю.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		62

Найважливішим актором системи є адміністратор платформи. Він має доступ до усіх розділів системи для повного контролю над веб-сайтом. Адміністратор має можливість переглядати аналітику, щоб здійснювати аналіз найбільш продаваних товарів та краще розуміти потреби кінцевого користувача (рис. 3.15).

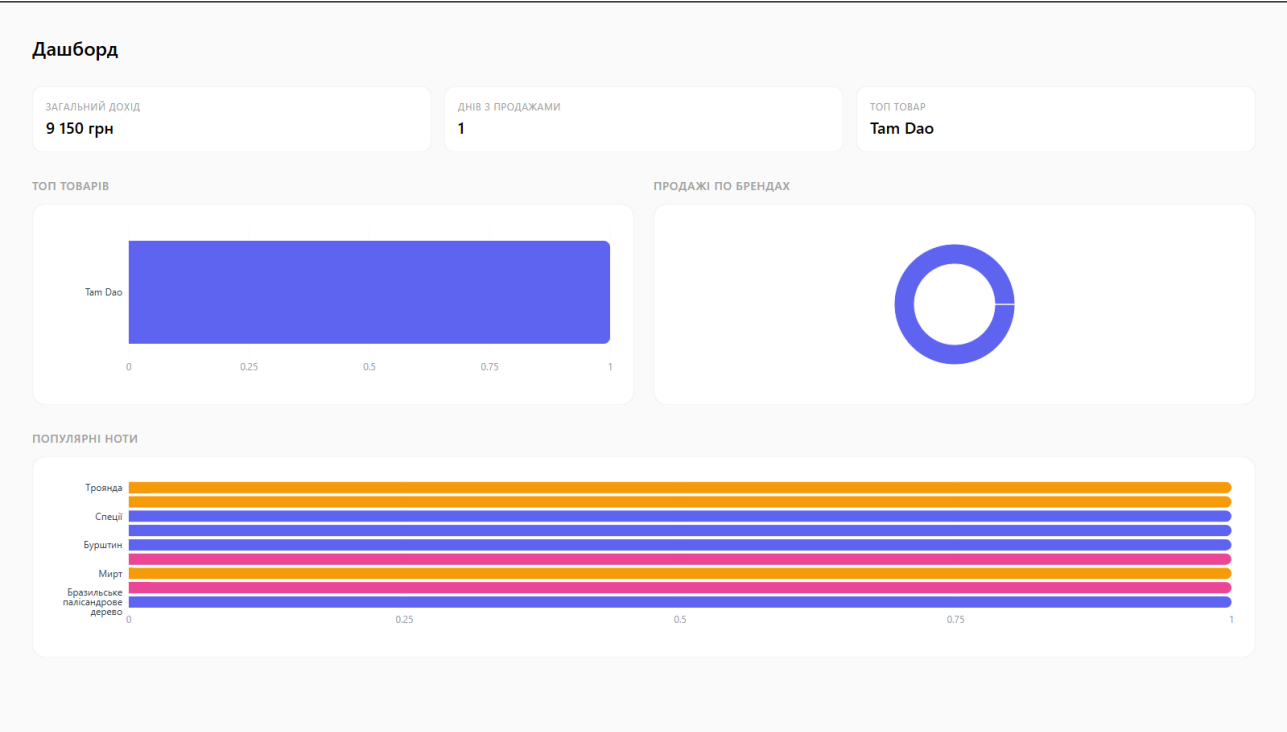


Рисунок 3.15 – Складська інформація про товар

Серед додаткового функціоналу є робота над користувачами системи – а саме перегляд інформації про них та надання прав (рис. 3.16).

Користувачі

Ім'я

Едік Коваленко

manager test

test

Всього: 3

Едік Коваленко

Активний 04.04.2026

АКАУНТ

Ім'я

Едік Коваленко

Email

user@test.com

Роль

Клієнт

КОНТАКТНІ ДАНІ

Ім'я

Прізвище

Телефон

Зберегти

Зблокувати

ЗАМОВЛЕННЯ (5)

Ім'я	Ціна	Разом
#98146a11 Citrus Riviera 100ml x 1	11450.00 грн	11450.00 грн
Bouquet de Hongrie 100ml x 1	11450.00 грн	11450.00 грн
Tam Dao 75ml x 1	9150.00 грн	9150.00 грн
Разом		32050.00 грн
12.05.2026		
#972efc3 Lira 100ml x 1	12450.00 грн	12450.00 грн
Разом		12450.00 грн
10.04.2026		
#04c33829 Arabians Tonka 20ml x 1	2950.00 грн	2950.00 грн
Разом		2950.00 грн
10.04.2026		
#a6948ica0 Arabians Tonka 20ml x 1	2950.00 грн	2950.00 грн
Intense Cafe 20ml x 1	2950.00 грн	2950.00 грн

Рисунок 3.16 – Інформація про користувача

У разі виявлення певних порушень адміністратор має право заблокувати користувача, після чого він втратить можливість заходити на сайт і виконувати будь-які дії (рис. 3.17).

Користувачі					Пошук...	Всі ролі ▾
ІМ'Я	EMAIL	РОЛЬ	СТАТУС	ДАТА РЕЄСТРАЦІЇ		
Едік Коваленко	user@test.com	Клієнт	Заблокований	04.04.2026		

Рисунок 3.17 – Користувач заблокований

Розглянутий функціонал платформи для продажу нішевої парфумерії повністю покриває потреби як покупців, так і працівників веб-сайту для забезпечення його злагодженої роботи.

3.3. Тестування роботи розробленого додатку

Перевірка коректності функціонування програмного забезпечення є важливим етапом розробки, який гарантує відповідність реалізованого інтернет-магазину початковим технічним вимогам. Ця процедура спрямована на виявлення прихованих дефектів у коді, усунення логічних помилок та забезпечення загальної стабільності веб-додатка. Головною метою проведеного тестування стало підтвердження безвідмовної роботи системи, а також контроль правильного збереження даних в умовах одночасного використання платформи багатьма клієнтами.

Насамперед для перевірки базової працездатності знадобилася перевірка роботи механізму логічного видалення записів (Soft-delete). Для цього було створено тестовий запис парфуму, після чого викликано метод його видалення. Перевірка підтвердила, що запис не зник з бази даних фізично, а лише отримав відповідний статус видаленого. Наступний запит до загального каталогу товарів показав, що цей парфум більше не відображається для звичайних користувачів, проте залишається доступним для адміністратора.

Після перевірки серверних методів було застосовано стратегію ручного тестування клієнтського інтерфейсу. У процесі перевірки було пройдено повний шлях клієнта. Спочатку було протестовано процедуру звичайної реєстрації за допомогою електронної пошти та пароля. Далі було перевірено альтернативний

		Кравченко К.М.			ІПЗ.КР.Б – 121 – 26– ПЗ	Арк.
		Єфремов Ю.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		64

метод входу через сервіс Google OAuth. Тестування показало, що система успішно отримує дані з облікового запису Google та автоматично створює профіль користувача на платформі без необхідності вводити додаткові дані. Після авторизації було перевірено роботу каталогу парфумерії. Фільтрація товарів за брендами, ціною та парфумерними нотами, а також текстовий пошук відпрацювали коректно і без затримок.

Наступним кроком стало тестування процесу оформлення покупки. Тестовий товар у вигляді відливанту певного об'єму був доданий до кошика. Під час перевірки сторінку браузера було навмисно перезавантажено, щоб переконатися у збереженні стану кошика. Оскільки дані кошика зберігаються у локальному сховищі браузера за допомогою бібліотеки Zustand, після оновлення сторінки товар залишився на місці. Під час оформлення замовлення було перевірено інтеграцію з логістичним сервісом служби доставки «Нова Пошта». Динамічний пошук міст та відділень спрацював успішно, підтягуючи актуальні дані через програмний інтерфейс (API) поштового оператора. Замовлення було успішно створено, а у відповідному розділі профілю з'явилася інформація про його статус.

Окремим важливим етапом стала перевірка панелі адміністратора, яка призначена для управління контентом інтернет-магазину. Під час входу в систему з правами адміністратора було підтверджено, що інтерфейс автоматично надає доступ до прихованих розділів. Було протестовано процес створення нової картки товару, зокрема завантаження фотографій парфумів до об'єктного сховища MinIO. Завантажені зображення успішно збереглися на окремому сервері, а платформа отримала прямі посилання на них. Також було перевірено механізм реляційних обмежень бази даних. Адміністратор спробував видалити бренд, до якого вже були прив'язані існуючі парфуми. Система коректно заблокувала цю дію та вивела повідомлення про помилку, що гарантує захист від випадкового руйнування зв'язків у базі даних.

Для аналізу стабільності підсистеми безпеки було протестовано механізм оновлення токенів доступу (JWT). Оскільки основний токен доступу має

		Кравченко К.М.			ІПЗ.КР.Б – 121 – 26– ПЗ	Арк.
		Єфремов Ю.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		65

короткий час життя для забезпечення безпеки, система повинна непомітно для користувача оновлювати його. Під час тестування користувач залишався на сторінці без активних дій більше п'ятнадцяти хвилин. Після завершення терміну дії токена була здійснена спроба переходу до захищеного розділу профілю. Перехоплювач запитів на стороні клієнта успішно зафіксував помилку доступу, автоматично відправив запит на отримання нового токена за допомогою токена оновлення і повторив початкову дію. Користувач не помітив процесу переавторизації, що свідчить про правильну реалізацію даного механізму.

З метою перевірки надійності механізмів обробки даних було застосовано техніку аналізу граничних значень. Зокрема, було перевірено поля введення пароля (не менше восьми символів), формату електронної пошти та дотримання діапазонів числових значень. Спроби залишити відгук на товар з оцінкою, що виходить за межі від одного до п'яти балів, були успішно відхилені валідатором. Аналогічно система не дозволила додати до кошика кількість товару, яка дорівнює нулю або має від'ємне значення. Валідація спрацювала як на стороні клієнтського інтерфейсу, так і на рівні серверної обробки.

Найбільш критичним сценарієм, який вимагав ретельного тестування, стала перевірка транзакційної цілісності складу при одночасних покупках. Для цього було змодельовано ситуацію стану гонки (Race condition), коли два різні користувачі одночасно намагаються оформити замовлення на останній доступний відливант парфуму. Завдяки реалізованому механізму блокування рядків у базі даних, сервер успішно опрацював перший запит, атомарно віднявши необхідну кількість мілілітрів від загального залишку. Другий запит, який надійшов на мілісекунди пізніше, був відхилений системою з повідомленням про недостатню кількість товару на складі.

На фінальному етапі тестування було перевірено адаптивність розробленого веб-додатка. За допомогою вбудованих інструментів розробника у браузері було змодельовано відображення сайту на екранах мобільних телефонів та планшетів. Усі елементи інтерфейсу, включаючи навігаційне меню, сітку товарів та форми оформлення замовлення, коректно підлаштувалися під різні

		Кравченко К.М.			ІПЗ.КР.Б – 121 – 26– ІЗ	Арк.
		Єфремов Ю.М.				66
Змн.	Арк.	№ докум.	Підпис	Дата		

розміри екранів. В результаті проведеного комплексного тестування було зроблено висновок, що створений програмний продукт функціонує стабільно, не містить критичних помилок та повністю готовий до розгортання і використання у реальному комерційному середовищі.

Висновки до третього розділу

У третьому розділі було детально розглянуто процеси підготовки, розгортання та перевірки якості програмного комплексу. Для візуалізації фізичної архітектури побудовано діаграму розгортання, яка наочно демонструє алгоритм взаємодії між клієнтським додатком, сервером, реляційною базою даних, ізольованим об'єктним сховищем медіафайлів та підключеними зовнішніми сервісами. Використання технології контейнеризації дозволило стандартизувати процес інсталяції платформи, забезпечити незалежність мікросервісів та усунути ризики конфліктів системних залежностей.

Значну увагу приділено реалізації користувацького інтерфейсу, який спроектовано за принципами односторінкового застосунку (SPA). Це архітектурне рішення забезпечило високу чуйність платформи та мінімізацію мережових затримок під час взаємодії користувачів із каталогом продукції, кошиком і панеллю управління. Впровадження адаптивного дизайну дозволило досягти коректного відображення контенту на різних пристроях.

Завершальним етапом стала перевірка надійності розробленого рішення з метою виявлення та попередження програмних дефектів. За допомогою методу димового тестування було верифіковано безперебійність основних сценаріїв — від реєстрації облікового запису до успішного атомарного оформлення замовлення з розрахунком доставки. Застосування техніки перевірки причинно-наслідкових зв'язків дозволило глибоко проаналізувати логіку управління медіафайлами, роботу механізму м'якого видалення та алгоритми розмежування прав доступу на базі рольової моделі. Використання аналізу граничних значень відіграло критичну роль у забезпеченні

		Кравченко К.М.			ІПЗ.КР.Б – 121 – 26– ПЗ	Арк.
		Єфремов Ю.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		67

транзакційної цілісності складу: було успішно підтверджено, що система ефективно попереджує помилки, миттєво блокуючи створення замовлень у разі недостатнього залишку або введення некоректних даних.

Результати проведеного тестування засвідчили високий рівень стабільності та відмовостійкості платформи, підтвердивши її цілковиту готовність до впровадження у реальне комерційне середовище електронної комерції.

		Кравченко К.М.			ІПЗ.КР.Б – 121 – 26– ІЗ	Арк.
		Єфремов Ю.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		68

ВИСНОВКИ

У рамках виконання випускної кваліфікаційної роботи було успішно спроектовано та програмно реалізовано повнофункціональну вебплатформу для управління продажами нішевої парфумерії. Головна мета проєкту, що полягала у створенні надійного цифрового інструменту для автоматизації процесів електронної комерції з урахуванням специфіки парфумерного сегмента, повністю досягнута.

На етапі системного аналізу та концептуального моделювання було сформовано технічне завдання, визначено ключових акторів та формалізовано їхню взаємодію з платформою. Обґрунтовано вибір клієнт-серверної архітектури із застосуванням концепції односторінкового застосунку для клієнтської частини. Проєктування реляційної бази даних здійснювалося з фокусом на гарантування цілісності історичної інформації, що стало можливим завдяки наскрізному впровадженню архітектурного інваріанта логічного видалення записів. Для забезпечення високоточного складського та фінансового обліку доменну модель було оптимізовано шляхом розділення абстрактних базових ароматів та їхніх конкретних товарних варіацій за об'ємом.

Вагомим інженерним здобутком є розробка відмовостійких алгоритмів для обслуговування критичних бізнес-процесів платформи. Інтеграція класичних патернів проєктування дозволила надійно інкапсулювати складну бізнес-логіку та досягти низької зв'язності коду. З метою запобігання виникненню стану гонки ресурсів під час паралельних покупок товарів, реалізовано транзакційний алгоритм атомарного створення замовлення із застосуванням механізму блокування рядків бази даних. Додатковий рівень фінансової безпеки забезпечено впровадженням механізму генерації статичного знімка ціни в момент оформлення кошика, що захищає чеки від майбутніх коливань вартості. Керування логістичними етапами формалізовано через систему статусів замовлення, а процеси ідентифікації посилено алгоритмом ротації токенів.

		Кравченко К.М.			ІПЗ.КР.Б – 121 – 26– ПЗ	Арк.
		Єфремов Ю.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		69

Фінальний етап розробки охоплював фізичне розгортання програмного комплексу та його комплексну верифікацію. Використання технології контейнеризації дозволило ізолювати середовища виконання, усунути конфлікти системних залежностей та стандартизувати процес запуску платформи на будь-якому серверному обладнанні. Програмний комплекс було успішно інтегровано із зовнішніми хмарними сервісами авторизації, платіжними шлюзами та логістичними інтерфейсами вітчизняних поштових операторів. Проведене багаторівневе тестування підтвердило абсолютну стабільність платформи, її стійкість до некоректних вхідних даних та безпомилкову алгоритмічну обробку складських залишків.

У підсумку, створений програмний продукт є довершеним інженерним рішенням, повністю готовим до впровадження у реальне комерційне середовище. Ухвалені архітектурні рішення гарантують високий рівень продуктивності і безпеки, а також утворюють міцний програмний фундамент для безперешкодного масштабування платформи відповідно до майбутніх потреб бізнесу.

		Кравченко К.М.			ІПЗ.КР.Б – 121 – 26– ПЗ	Арк.
		Єфремов Ю.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		70

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Percival H., Gregory B. Architecture Patterns with Python. O'Reilly Media, 2020.
2. Vincent W. S. Django for APIs : Build web APIs with Python and Django. Leanpub, 2019.
3. Vincent W. S. Django for Beginners : Build websites with Python & Django. Leanpub, 2018.
4. Django Documentation. URL: <https://docs.djangoproject.com/>
5. Документація React. URL: <https://react.dev/>
6. Slatkin B. Effective Python : 90 Specific Ways to Write Better Python. Pearson Education, 2020.
7. Docker Documentation. URL: <https://docs.docker.com/>
8. Banks A., Porcello E. Learning React : Functional Web Development with React and Redux. O'Reilly Media, 2017.
9. Massé M. REST API Design Rulebook. O'Reilly Media, 2012.
10. Downey A. Think Python : How to Think Like a Computer Scientist. Green Tea Press, 2015.
11. MySQL Documentation. URL: <https://dev.mysql.com/doc/>
12. MySQL 8.0 Reference Manual. URL: <https://dev.mysql.com/doc/refman/8.0/en/>
13. Martin R. C. Clean Architecture : A Craftsman's Guide to Software Structure and Design. Prentice Hall, 2017.
14. Matthias K., Kane S. P. Docker : Up & Running. O'Reilly Media, 2018.
15. Evans E. Domain-Driven Design : Tackling Complexity in the Heart of Software. Addison-Wesley, 2003.
16. Schwartz B., Zaitsev P., Tkachenko V. High Performance MySQL. O'Reilly Media, 2012.
17. MinIO Documentation. URL: <https://min.io/docs/>
18. Патерн Стан. URL: <https://refactoring.guru/uk/design-patterns/state>

		Кравченко К.М.			ІПЗ.КР.Б – 121 – 26– ІЗ	Арк.
		Єфремов Ю.М.				71
Змн.	Арк.	№ докум.	Підпис	Дата		

19. UI/UX дизайн: принципи та методи створення зручного інтерфейсу користувача. URL: <https://whileweb.com/uk/blog/uiux-dizajn-principi-ta-metodi-stvorennya-zruchnogo-interfejsu-koristuvacha/>
20. Olfactory notes: the ultimate guide to understanding your perfume. URL: <https://ulric-de-varens.com/en/blogs/post/guide-olfactory-notes-perfume>
21. Newman S. Building Microservices : Designing Fine-Grained Systems. 2nd ed. O'Reilly Media, 2021.
22. Richardson C. Microservices Patterns : With Examples in Java. Manning Publications, 2018.
23. Kleppmann M. Designing Data-Intensive Applications. O'Reilly Media, 2017.
24. Gamma E., Helm R., Johnson R., Vlissides J. Design Patterns : Elements of Reusable Object-Oriented Software. Addison-Wesley, 1994.
25. Fowler M. Patterns of Enterprise Application Architecture. Addison-Wesley, 2002.
26. Django REST Framework Documentation. URL: <https://www.django-rest-framework.org/>
27. Fielding R. T. Architectural Styles and the Design of Network-based Software Architectures : PhD dissertation. University of California, Irvine, 2000.
28. NGINX Documentation. URL: <https://nginx.org/en/docs/>
29. OWASP Top Ten : The Ten Most Critical Web Application Security Risks. URL: <https://owasp.org/www-project-top-ten/>
30. JSON Web Token (JWT). RFC 7519. URL: <https://datatracker.ietf.org/doc/html/rfc7519>

ДОДАТКИ

		Кравченко К.М.			ІПЗ.КР.Б – 121 – 26– ІЗ	Арк.
		Єфремов Ю.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		73

Додаток А

```
class SoftDeleteManager(models.Manager):
    def get_queryset(self):
        return super().get_queryset().filter(is_deleted=False)

class SoftDeleteModel(models.Model):
    id = models.UUIDField(primary_key=True, default=uuid.uuid4, editable=False)
    is_deleted = models.BooleanField(default=False)
    created_at = models.DateTimeField(auto_now_add=True, verbose_name="Created At")
    updated_at = models.DateTimeField(auto_now=True, verbose_name="Updated At")
    deleted_at = models.DateTimeField(null=True, blank=True, verbose_name="Deleted At")

    objects = SoftDeleteManager()

    all_objects = models.Manager()

    class Meta:
        abstract = True

    def delete(self, using=None, keep_parents=False):
        self.is_deleted = True
        self.deleted_at = timezone.now()
        self.save(using=using)

    def restore(self):
        self.is_deleted = False
        self.deleted_at = None
        self.save()
```

Додаток Б

```
class OrderService:
    @staticmethod
    @transaction.atomic
    def create_order(user, customer_data, items_data, nova_poshta_branch_id=None, user_id=None,
        delivery_method='np',
        courier_address=""):
        customer = None

        if user_id:
            try:
                linked_user = User.objects.get(id=user_id)
                customer, _ = Customer.objects.update_or_create(
```

		Кравченко К.М.			ІІЗ.КР.Б – 121 – 26– ІІЗ	Арк.
		Єфремов Ю.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		74

```

        user=linked_user,
        defaults={
            'first_name': customer_data.get('first_name', ""),
            'last_name': customer_data.get('last_name', ""),
            'phone': customer_data.get('phone', ""),
        }
    )
except User.DoesNotExist:
    pass
elif user and user.is_authenticated:
    customer, _ = Customer.objects.update_or_create(
        user=user,
        defaults={
            'first_name': customer_data.get('first_name', ""),
            'last_name': customer_data.get('last_name', ""),
            'phone': customer_data.get('phone', ""),
        }
    )
else:
    customer = Customer.objects.create(**customer_data)

order = Order.objects.create(
    customer=customer,
    total_price=0,
    status=Order.Status.NEW,
    delivery_method=delivery_method,
    nova_poshta_branch_id=nova_poshta_branch_id if delivery_method == 'np' else None,
    courier_address=courier_address if delivery_method == 'courier' else "",
)

if delivery_method == 'np' and order.nova_poshta_branch and order.customer.user:
    SavedAddress.objects.get_or_create(
        user=order.customer.user,
        city=order.nova_poshta_branch.city,
        nova_poshta_branch=order.nova_poshta_branch,
        defaults={'label': f'{order.nova_poshta_branch.city.name}'}
    )

total_price = 0

for item in items_data:
    variant_id = item['product_variant_id']
    quantity = item['quantity']

    try:

```

```

        variant = ProductVariant.objects.select_for_update().get(id=variant_id)
except ProductVariant.DoesNotExist:
    raise ValidationError(f"Product variant {variant_id} not found.")

if variant.quantity < quantity:
    raise ValidationError(
        f'Insufficient stock for {variant.perfume.name} ({variant.volume_ml}ml). "
        f"Available: {variant.quantity}, Requested: {quantity}"
    )

variant.quantity = F('quantity') - quantity
variant.save()

OrderItem.objects.create(
    order=order,
    product_variant=variant,
    quantity=quantity,
    price=variant.price
)
total_price += variant.price * quantity

order.total_price = total_price
order.save()
return order

```

Додаток В

```

class OrderItem(SoftDeleteModel):
    order = models.ForeignKey(Order, on_delete=models.CASCADE, related_name='items')
    product_variant = models.ForeignKey(ProductVariant, on_delete=models.PROTECT,
related_name='order_items')
    quantity = models.PositiveIntegerField(_("Quantity"), default=1)
    price = models.DecimalField(_("Price at moment"), max_digits=10, decimal_places=2)
    class Meta:
        verbose_name = _("Order Item")
        verbose_name_plural = _("Order Items")
        db_table = 'order_items'
    def save(self, *args, **kwargs):
        if not self.price:
            self.price = self.product_variant.price
        super().save(*args, **kwargs)

```

		Кравченко К.М.			ІІЗ.КР.Б – 121 – 26– ІІЗ	Арк.
		Єфремов Ю.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		76

```

OrderItem.objects.create(
    order=order,
    product_variant=variant,
    quantity=quantity,
    price=variant.price,
)
total_price += variant.price * quantity
from django.db.models import F, Sum
from .models import OrderItem
order.total_price = OrderItem.objects.filter(order=order).aggregate(
    total=Sum(F('price') * F('quantity'))
)['total'] or 0
order.save()
from rest_framework import serializers
from .models import OrderItem

class OrderItemReadSerializer(serializers.ModelSerializer):
    perfume_id = serializers.CharField(source='product_variant.perfume.id', read_only=True)
    perfume_name = serializers.CharField(source='product_variant.perfume.name', read_only=True)
    brand_name = serializers.CharField(source='product_variant.perfume.brand.name',
read_only=True)
    volume = serializers.IntegerField(source='product_variant.volume_ml', read_only=True)
    perfume_image = serializers.SerializerMethodField()
    class Meta:
        model = OrderItem
        fields = ['id', 'perfume_id', 'perfume_name', 'brand_name', 'volume', 'quantity', 'price',
'perfume_image']

    def get_perfume_image(self, obj):
        image = obj.product_variant.perfume.images.order_by('position').first()
        if image:
            return image.image.url
        return None

```

```

class ProductVariant(SoftDeleteModel):
    perfume = models.ForeignKey(Perfume, on_delete=models.CASCADE, related_name='variants')
    volume_ml = models.PositiveIntegerField(_("Volume (ml)"))
    price = models.DecimalField(_("Price"), max_digits=10, decimal_places=2)
    quantity = models.PositiveIntegerField(_("Stock Quantity"), default=0)
    sku = models.CharField(_("SKU"), max_length=50, unique=True, db_index=True)

    class Meta:
        verbose_name = _("Product Variant")
        verbose_name_plural = _("Product Variants")
        db_table = 'product_variants'
        ordering = ['volume_ml']

    def __str__(self):
        return f'{self.perfume.name} {self.volume_ml}ml'

for item in items_data:
    variant_id = item['product_variant_id']
    quantity = item['quantity']

    try:
        variant = ProductVariant.objects.select_for_update().get(id=variant_id)
    except ProductVariant.DoesNotExist:
        raise ValidationError(f"Product variant {variant_id} not found.")

    if variant.quantity < quantity:
        raise ValidationError(
            f"Insufficient stock for {variant.perfume.name} ({variant.volume_ml}ml). "
            f"Available: {variant.quantity}, Requested: {quantity}"
        )

    variant.quantity = F('quantity') - quantity
    variant.save()

    OrderItem.objects.create(
        order=order,
        product_variant=variant,
        quantity=quantity,
        price=variant.price
    )
    total_price += variant.price * quantity

class OrderItemViewSet(viewsets.GenericViewSet, mixins.CreateModelMixin,
                        mixins.UpdateModelMixin, mixins.DestroyModelMixin):

```

		Кравченко К.М.			ІІЗ.КР.Б – 121 – 26– ІІЗ	Арк.
		Єфремов Ю.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		78

```

permission_classes = [IsManagerOrAdmin]
http_method_names = ['post', 'patch', 'delete', 'head', 'options']

def partial_update(self, request, *args, **kwargs):
    item = self.get_object()
    new_quantity = request.data.get('quantity')

    if new_quantity:
        diff = new_quantity - item.quantity
        variant = item.product_variant
        if diff > 0 and variant.quantity < diff:
            return Response(
                {'detail': f'Insufficient stock. Available: {variant.quantity}'},
                status=status.HTTP_400_BAD_REQUEST
            )
        variant.quantity = F('quantity') - diff
        variant.save()
        item.quantity = new_quantity
        item.save()

    order = item.order
    order.total_price = OrderItem.objects.filter(order=order).aggregate(
        total=Sum(F('price') * F('quantity'))
    )['total'] or 0
    order.save()
    return Response(OrderItemReadSerializer(item).data)

def destroy(self, request, *args, **kwargs):
    item = self.get_object()
    order = item.order
    if order.items.filter(is_deleted=False).count() <= 1:
        return Response(
            {'detail': 'An order cannot be placed without items.'},
            status=status.HTTP_400_BAD_REQUEST
        )
    variant = item.product_variant
    variant.quantity = F('quantity') + item.quantity
    variant.save()
    item.delete()
    order.total_price = OrderItem.objects.filter(order=order).aggregate(
        total=Sum(F('price') * F('quantity'))
    )['total'] or 0
    order.save()
    return Response(status=status.HTTP_204_NO_CONTENT)

```

```

class Order(SoftDeleteModel):
    delivery_method = models.CharField(
        _("Delivery Method"),
        max_length=20,
        choices=[('np', 'Nova Poshta'), ('courier', 'Courier')],
        default='np'
    )
    nova_poshta_branch = models.ForeignKey(
        NovaPoshtaBranch,
        on_delete=models.PROTECT,
        related_name='orders',
        verbose_name=_("Nova Poshta Branch"),
        null=True,
        blank=True
    )
    courier_address = models.TextField(_("Courier Address"), blank=True, default="")
    customer = models.ForeignKey(Customer, on_delete=models.PROTECT, related_name='orders')
    total_price = models.DecimalField(_("Total Price"), max_digits=12, decimal_places=2,
    default=0)

class Status(models.TextChoices):
    NEW = 'new', _('New')
    CONFIRMED = 'confirmed', _('Confirmed')
    SHIPPED = 'shipped', _('Shipped')
    COMPLETED = 'completed', _('Completed')
    CANCELED = 'canceled', _('Canceled')

status = models.CharField(
    _("Status"),
    max_length=20,
    choices=Status.choices,
    default=Status.NEW,
    db_index=True
)

class Meta:
    verbose_name = _("Order")
    verbose_name_plural = _("Orders")
    db_table = 'orders'
    ordering = ['-created_at']

def __str__(self):
    return f"Order #{str(self.id)[8]} - {self.status}"

```

		Кравченко К.М.			ІПЗ.КР.Б – 121 – 26– ІЗ	Арк.
		Єфремов Ю.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		80

```

class OrderViewSet(viewsets.GenericViewSet,
                    mixins.CreateModelMixin,
                    mixins.ListModelMixin,
                    mixins.RetrieveModelMixin,
                    mixins.UpdateModelMixin,
                    mixins.DestroyModelMixin):
    http_method_names = ['get', 'post', 'patch', 'head', 'options']

    def get_permissions(self):
        if self.action == 'create':
            return [AllowAny()]
        if self.action == 'partial_update':
            return [IsManagerOrAdmin()]
        return [IsAuthenticated()]
    def partial_update(self, request, *args, **kwargs):
        order = self.get_object()
        if 'nova_poshta_branch' in request.data:
            order.nova_poshta_branch_id = request.data['nova_poshta_branch']
        if 'status' in request.data:
            order.status = request.data['status']
        if not order.items.filter(is_deleted=False).exists():
            return Response(
                {'detail': 'An order cannot be placed without items.'},
                status=status.HTTP_400_BAD_REQUEST
            )
        order.save()
        if 'customer' in request.data:
            customer_data = request.data['customer']
            serializer = CustomerSerializer(order.customer, data=customer_data, partial=True)
            serializer.is_valid(raise_exception=True)
            serializer.save()

        return Response(OrderReadSerializer(order).data)

```

		Кравченко К.М.			ІІЗ.КР.Б – 121 – 26– ІІЗ	Арк.
		Єфремов Ю.М.				
Змн.	Арк.	№ докум.	Підпис	Дата		81