

**ДЕРЖАВНИЙ УНІВЕРСИТЕТ
ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНИХ ТЕХНОЛОГІЙ
НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ
ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ
КАФЕДРА ІНЖЕНЕРІЇ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ**

КВАЛІФІКАЦІЙНА РОБОТА

на тему: «Розробка онлайн-магазину парфумерії "AromaShop"»

на здобуття освітнього ступеня бакалавра
зі спеціальності 121 Інженерія програмного забезпечення
освітньо-професійної програми «Інженерія програмного забезпечення»

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

_____ Владислав МЕЛЬНИК
(підпис)

Виконав: здобувач вищої освіти групи ПД-44

_____ Владислав МЕЛЬНИК

Керівник: _____ Вікторія КОРЕЦЬКА
канд.пед.наук, доц.

Рецензент: _____

Київ 2026

**ДЕРЖАВНИЙ УНІВЕРСИТЕТ
ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНИХ ТЕХНОЛОГІЙ**

Навчально-науковий інститут інформаційних технологій

Кафедра Інженерії програмного забезпечення

Ступінь вищої освіти Бакалавр

Спеціальність 121 Інженерія програмного забезпечення

Освітньо-професійна програма «Інженерія програмного забезпечення»

ЗАТВЕРДЖУЮ

Завідувач кафедри

Інженерії програмного забезпечення

_____ Ірина ЗАМРІЙ

« ____ » _____ 2026 р.

**ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ**

Мельнику Владиславу Олександровичу

1. Тема кваліфікаційної роботи: «Розробка онлайн-магазину парфумерії "AromaShop"»

керівник кваліфікаційної роботи канд.пед.наук, доц. Вікторія КОРЕЦЬКА,

затверджені наказом Державного університету інформаційно-комунікаційних технологій від «20» лютого 2026 року № 51.

2. Строк подання кваліфікаційної роботи «29» травня 2026 року.

3. Вихідні дані до кваліфікаційної роботи: теоретичні відомості про електронну комерцію, принципи організації онлайн-продажу парфумерної продукції, підходи до побудови каталогів товарів, фільтрації ароматів за характеристиками, оформлення замовлень, роботи з кошиком, відгуками користувачів та адміністративним керуванням асортиментом, а також технічна документація щодо використання Java 21, Spring Boot, Spring Security, JWT, Spring Data JPA, PostgreSQL, Liquibase, React, TypeScript, Vite, Docker та Docker Compose.

4. Зміст розрахунково-пояснювальної записки:

1. Огляд та аналіз предметної галузі онлайн-продажу парфумерної продукції, особливостей вибору ароматів, структурування каталогу, пошуку, фільтрації та оформлення замовлень.
2. Аналіз існуючих програмних засобів для онлайн-продажу парфумерії, зокрема спеціалізованих інтернет-магазинів, маркетплейсів та beauty-платформ.
3. Огляд засобів реалізації web-застосунку AromaShop для онлайн-продажу парфумерії.
4. Проектування архітектури web-застосунку, структури бази даних, доменної моделі, основних модулів системи та сценаріїв взаємодії користувачів із платформою.
5. Програмна реалізація та опис функціонування web-застосунку AromaShop, зокрема каталогу товарів, кошика, замовлень, відгуків, адміністративної панелі та функції «Підбір за подією».
6. Тестування основних функцій web-застосунку та оцінка відповідності програмного продукту поставленим функціональним і нефункціональним вимогам.

5. Перелік графічного матеріалу: *презентація*

1. Аналіз аналогів програмного забезпечення для онлайн-продажу парфумерної продукції.
2. Вимоги до web-застосунку AromaShop.
3. Архітектура програмної реалізації web-застосунку AromaShop.
4. Схема взаємодії React, Java Spring Boot, PostgreSQL, REST API та Docker Compose.
5. Діаграма варіантів використання web-застосунку AromaShop.
6. Діаграма класів доменної моделі системи.
7. ER-діаграма бази даних web-застосунку AromaShop.
8. Алгоритм роботи функції «Підбір за подією».
9. Екранні форми web-застосунку
10. Апробація результатів дослідження.

6. Дата видачі завдання «23» лютого 2026 року.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів кваліфікаційної роботи	Строк виконання етапів роботи	Примітка
1	Підбір та аналіз науково-технічної літератури за темою онлайн-продажу парфумерної продукції	23.02–28.02.2026	
2	Аналіз та дослідження існуючих аналогів програмного забезпечення для онлайн-продажу парфумерії	01.03–07.03.2026	
3	Аналіз предметної галузі онлайн-продажу парфумерії та визначення вимог до web-застосунку	08.03–15.03.2026	
4	Огляд засобів реалізації web-застосунку AromaShop	16.03–22.03.2026	
5	Проектування web-застосунку AromaShop, логічної структури системи, бази даних та UML-діаграм	23.03–05.04.2026	
6	Програмна реалізація серверної та клієнтської частин web-застосунку	06.04–26.04.2026	
7	Тестування web-застосунку AromaShop та оформлення результатів тестування	27.04–03.05.2026	
8	Оформлення роботи: вступ, висновки, реферат, список джерел	04.05–10.05.2026	
9	Розробка демонстраційних матеріалів та підготовка до попереднього захисту роботи	11.05–17.05.2026	
10	Попередній захист роботи	18.05–22.05.2026	

Здобувач вищої освіти

_____ (підпис)

Владислав МЕЛЬНИК

Керівник
кваліфікаційної роботи

_____ (підпис)

Вікторія КОРЕЦЬКА

РЕФЕРАТ

Текстова частина кваліфікаційної роботи на здобуття освітнього ступеня бакалавра: 60 стор., 2 табл., 27 рис., 27 джерел.

Мета роботи – спрощення процесу вибору, пошуку та придбання парфумерної продукції за рахунок розробки web-застосунку AromaShop.

Об'єкт дослідження – процеси онлайн-продажу парфумерної продукції.

Предмет дослідження – засоби, методи та технології реалізації web-застосунку AromaShop для онлайн-продажу парфумерії.

Короткий зміст роботи: у кваліфікаційній роботі проведено аналіз предметної галузі онлайн-продажу парфумерної продукції та визначено основні проблеми, пов'язані з вибором ароматів в інтернет-магазинах.

Проаналізовано існуючі програмні засоби для онлайн-продажу парфумерії, зокрема BROCARD, MAKEUP, ROZETKA та EVA.UA. Визначено їхні переваги та недоліки, серед яких перевантаженість інтерфейсу, надмірно широкий асортимент, недостатня спеціалізація саме на парфумерній продукції, обмежені можливості персоналізованого вибору ароматів та відсутність повноцінного підбору товарів за життєвою ситуацією.

У роботі сформовано функціональні та нефункціональні вимоги до web-застосунку AromaShop, обґрунтовано вибір засобів реалізації клієнтської частини, серверної частини, бази даних, системи аутентифікації, міграцій, тестування та контейнеризації. Для реалізації web-застосунку використано Java 21, Spring Boot, Spring Web, Spring Security, JWT, Spring Data JPA, Hibernate, PostgreSQL, Liquibase, React, TypeScript, Vite, React Router, TanStack React Query, Swagger/OpenAPI, Docker, Docker Compose, Maven та Git [20].

Спроектовано загальну архітектуру web-застосунку, логічну структуру системи, модель предметної області, структуру бази даних, діаграму варіантів

використання, діаграму класів доменної моделі, ER-діаграму, діаграму послідовності оформлення замовлення та алгоритм роботи функції «Підбір за подією».

Особливістю розробленої системи є функція «Підбір за подією», яка дозволяє користувачу обирати аромат не лише за назвою або брендом, а за конкретною ситуацією використання. Система підтримує сценарії підбору для побачення, офісу, подарунка, щоденного використання та вечірнього виходу.

Проведено тестування web-застосунку AromaShop, перевірено коректність роботи авторизації, реєстрації, каталогу товарів, пошуку, фільтрації, сторінки товару, кошика, оформлення замовлення, історії замовлень, відгуків, профілю користувача, адміністративної панелі та функції «Підбір за подією».

КЛЮЧОВІ СЛОВА: AROMASHOP, ОНЛАЙН-МАГАЗИН, ПАРФУМЕРІЯ, WEB-ЗАСТОСУНОК, ЕЛЕКТРОННА КОМЕРЦІЯ, КАТАЛОГ ТОВАРІВ, ПІДБІР ЗА ПОДІЄЮ, КОШИК, ЗАМОВЛЕННЯ, ВІДГУКИ, JAVA, SPRING BOOT, REACT, TYPESCRIPT, POSTGRESQL, LIQUIBASE, JWT, DOCKER COMPOSE.

ЗМІСТ

ВСТУП.....	13
1 АНАЛІЗ ЗАДАЧІ ОНЛАЙН-ПРОДАЖУ ПАРФУМЕРНОЇ ПРОДУКЦІЇ	16
1.1 Типовий процес вибору та придбання парфумерії	16
1.2 Особливості предметної галузі продажу парфумерії	18
1.3 Проблеми, що виникають під час онлайн-вибору ароматів.....	19
1.4 Існуючі технологічні рішення для онлайн-продажу парфумерії.....	21
1.5 Аналіз аналогів	22
1.6 Роль онлайн-магазину парфумерії як цифрового сервісу	27
1.7 Представлення парфумерної продукції в цифровій формі	28
2 ВИБІР ТЕХНІЧНИХ ЗАСОБІВ ДЛЯ РЕАЛІЗАЦІЇ WEB-ЗАСТОСУНКУ	
AROMASHOP	31
2.1 Визначення функціональних та нефункціональних вимог	32
2.2 Вибір мови програмування та фреймворку	34
2.3 Вибір засобів реалізації клієнтського web-інтерфейсу	35
2.4 Вибір бази даних та засобів керування міграціями	37
2.5 Вибір засобів аутентифікації, авторизації та взаємодії через API	38
2.6 Вибір інструментів контейнеризації, тестування та контролю версій.....	41
3 ПРОЄКТУВАННЯ WEB-ЗАСТОСУНКУ ДЛЯ ОНЛАЙН-ПРОДАЖУ	
ПАРФУМЕРІЇ	42
3.1 Проєктування загальної архітектури web-застосунку	42
3.2 Проєктування логічної структури системи.....	45
3.3 Проєктування структури бази даних.....	47
3.4 Проєктування серверної частини Spring Boot Backend	50
3.5 Проєктування функції «Підбір за подією»	51
3.6 Проєктування клієнтського web-інтерфейсу.....	53
4 РЕАЛІЗАЦІЯ ТА ТЕСТУВАННЯ WEB-ЗАСТОСУНКУ AROMASHOP.....	56
4.1 Реалізація авторизації користувача та профілю.....	56
4.2 Реалізація клієнтського web-інтерфейсу.....	59
4.3 Реалізація серверної частини для роботи з товарами, брендами та	
користувачами	63
4.4 Реалізація кошика, замовлень, відгуків та історії покупок	65
4.5 Реалізація функції «Підбір за подією» та адміністративної панелі	68
4.6 Тестування основних функцій web-застосунку	71
ВИСНОВКИ	75
ПЕРЕЛІК ПОСИЛАНЬ	78
ДОДАТОК А. ДЕМОНСТРАЦІЙНІ МАТЕРІАЛИ.....	81
ДОДАТОК Б. ЛІСТИНГИ ПРОГРАМНИХ МОДУЛІВ	89

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

API – інтерфейс прикладного програмування, що використовується для взаємодії між програмними компонентами або зовнішніми сервісами.

REST – архітектурний стиль організації обміну даними між клієнтською та серверною частинами web-застосунку.

HTTP – протокол передавання даних у мережі Інтернет.

HTTPS – захищена версія протоколу HTTP, що забезпечує безпечний обмін даними між клієнтом і сервером.

JSON – текстовий формат обміну даними, який використовується для передавання структурованої інформації між frontend- і backend-частинами системи [23].

Frontend – клієнтська частина web-застосунку, з якою безпосередньо взаємодіє користувач [23].

Backend – серверна частина web-застосунку, яка виконує бізнес-логіку, обробляє запити та працює з базою даних.

SPA – односторінковий web-застосунок, у якому взаємодія з користувачем відбувається без повного перезавантаження сторінки.

Java – об'єктно-орієнтована мова програмування, що використовується для реалізації серверної частини web-застосунку.

Spring – фреймворк для створення серверних Java-застосунків.

Spring Boot – інструмент екосистеми Spring, який спрощує створення, налаштування та запуск backend-сервісів [19].

Spring Web – модуль Spring, який використовується для створення web-застосунків і REST API [19; 26].

Spring Security – модуль Spring для реалізації механізмів аутентифікації, авторизації та захисту доступу [20].

JWT – JSON Web Token, токен, який використовується для безпечної передачі даних про авторизованого користувача [20].

Spring Data JPA – модуль Spring для роботи з реляційними базами даних через об'єктну модель [19].

Hibernate – ORM-фреймворк, який забезпечує відображення Java-класів на таблиці реляційної бази даних [19; 21].

PostgreSQL – об'єктно-реляційна система керування базами даних [21].

Liquibase – інструмент для керування змінами структури бази даних за допомогою міграцій [21; 22].

React – бібліотека JavaScript для створення компонентного користувацького інтерфейсу [10; 23].

TypeScript – мова програмування, що розширює JavaScript за рахунок статичної типізації [24].

Vite – інструмент для швидкого створення, запуску та збирання frontend-застосунків [23; 25].

React Router – бібліотека для організації маршрутизації між сторінками клієнтського web-застосунку [23].

TanStack React Query – бібліотека для роботи із серверними даними, кешування запитів і синхронізації стану frontend-частини з backend [23].

Maven – інструмент для збирання Java-проектів і керування залежностями.

Swagger/OpenAPI – засоби опису, документування та тестування REST API [26].

Docker – платформа контейнеризації, що використовується для запуску компонентів системи в ізольованому середовищі [27].

Docker Compose – інструмент для опису та запуску багатоконтейнерних застосунків [27].

Git – система контролю версій, що використовується для відстеження змін у програмному коді.

GitHub – web-сервіс для зберігання репозиторіїв, спільної роботи з кодом та контролю версій.

AromaShop – web-застосунок для онлайн-продажу парфумерної продукції [1; 2; 3; 4; 6].

E-commerce – електронна комерція, тобто купівля та продаж товарів або послуг через інтернет [1; 2; 3; 4; 6].

Каталог товарів – структурований перелік товарних позицій, доступних для перегляду користувачем.

Кошик – функціональний модуль web-застосунку, у якому користувач зберігає обрані товари перед оформленням замовлення.

Замовлення – запис у системі, що містить інформацію про придбані товари, контактні дані користувача, адресу доставки, статус і суму покупки.

Адміністративна панель – частина web-застосунку, призначена для керування товарами, брендами, користувачами, замовленнями та відгуками.

IFRA – міжнародна асоціація ароматичних речовин, стандарти якої використовуються для оцінки безпечності ароматичних компонентів [16].

GS1 – міжнародна система стандартів для ідентифікації, маркування та обміну даними про товари [17].

ER-діаграма – діаграма сутність-зв'язок, що використовується для моделювання структури бази даних [21].

UML – уніфікована мова моделювання, яка використовується для опису структури та поведінки програмної системи.

DTO – об'єкт передавання даних між шарами або компонентами програмної системи.

IDE – інтегроване середовище розробки програмного забезпечення.

ВСТУП

У сучасних умовах розвитку електронної комерції онлайн-продаж товарів є одним із найбільш поширених способів взаємодії між продавцем і покупцем. Інтернет-магазини дозволяють користувачам переглядати асортимент, порівнювати товари, оформлювати замовлення дистанційно та отримувати актуальну інформацію про наявність продукції. Особливо важливою така форма продажу є для парфумерної продукції, оскільки асортимент ароматів є широким, а вибір товару залежить не лише від ціни або бренду, а й від індивідуальних вподобань покупця [1; 2; 3; 4; 6].

Актуальність теми кваліфікаційної роботи зумовлена необхідністю створення зручного спеціалізованого web-застосунку для онлайн-продажу парфумерії. На відміну від універсальних інтернет-магазинів, система для продажу ароматів повинна враховувати специфіку предметної галузі: ароматичне сімейство, концентрацію, об'єм, парфумерні ноти, рейтинг, відгуки, сезонність, стиль аромату, наявність товару та відповідність стандартам якості. Якщо такі характеристики подані неповно або незручно, користувачу складніше прийняти рішення про покупку [1; 2; 3; 4; 6].

У реальних умовах користувач часто не знає точної назви аромату або бренду, але розуміє ситуацію, для якої потрібен парфум. Наприклад, покупець може шукати аромат для офісу, побачення, подарунка, щоденного використання або вечірнього виходу. Звичайна фільтрація за ціною, брендом і об'ємом не завжди допомагає вирішити таку задачу. У зв'язку з цим актуальною є розробка web-застосунку AromaShop, який поєднує класичні функції інтернет-магазину з функцією предметно орієнтованого підбору ароматів за подією [1; 2; 3; 4; 6].

Розроблюваний web-застосунок AromaShop призначений для перегляду каталогу парфумерної продукції, пошуку й фільтрації товарів, перегляду детальної інформації про аромат, роботи з кошиком, оформлення замовлень, перегляду історії покупок, залишення відгуків і керування профілем користувача. Для

адміністратора система повинна забезпечувати створення, редагування та видалення товарів, керування брендами, користувачами, замовленнями, залишками та відгуками [1; 3; 4; 10].

Особливо важливим елементом системи є функція «Підбір за подією». Вона дозволяє користувачу обрати ситуацію використання аромату, наприклад побачення, офіс, подарунок, кожен день або вечір. Після цього система задає уточнювальні питання, формує критерії підбору та пропонує товари з внутрішньої бази магазину. Такий підхід спрощує вибір для користувачів, які не мають глибоких знань про парфумерні бренди або ароматичні сімейства [1; 3; 4; 10].

Мета роботи – спрощення процесу вибору, пошуку та придбання парфумерної продукції за рахунок розробки web-застосунку [1; 3; 4; 10].

Об'єкт дослідження – процеси онлайн-продажу парфумерної продукції [1; 2; 3; 4; 6].

Предмет дослідження – засоби, методи та технології реалізації web-застосунку AromaShop для онлайн-продажу парфумерії [1; 2; 3; 4; 6].

Для досягнення поставленої мети необхідно вирішити такі завдання:

1. Проаналізувати предметну галузь онлайн-продажу парфумерної продукції та визначити основні проблеми вибору ароматів в інтернет-магазинах [1; 2; 3; 4; 6].
2. Дослідити існуючі програмні засоби для онлайн-продажу парфумерії, порівняти їх функціональні можливості та визначити основні недоліки [1; 2; 3; 4; 6].
3. Сформулювати функціональні та нефункціональні вимоги до web-застосунку AromaShop.
4. Обґрунтувати вибір технологій для реалізації web-застосунку, зокрема Java 21, Spring Boot, React, TypeScript, PostgreSQL, Liquibase, JWT та Docker Compose [18; 19; 20; 21; 22; 23].
5. Спроекувати архітектуру web-застосунку, структуру бази даних, доменну модель і основні UML-діаграми системи [21].

6. Реалізувати серверну частину для роботи з товарами, брендами, користувачами, кошиком, замовленнями, відгуками та рекомендаціями.
7. Реалізувати клієнтський web-інтерфейс для перегляду каталогу, сторінки товару, кошика, оформлення замовлення, історії покупок, профілю користувача та адміністративної панелі.
8. Розробити функцію «Підбір за подією» для формування рекомендацій ароматів на основі ситуації використання та уточнювальних параметрів.
9. Провести тестування основних функцій web-застосунку та оцінити відповідність розробленого програмного забезпечення поставленим вимогам.

Методи дослідження. У бакалаврській роботі використовуються такі методи: аналіз предметної галузі, порівняльний аналіз існуючих програмних засобів онлайн-продажу парфумерії, об'єктно-орієнтоване проектування, UML-модельовання, проектування структури реляційної бази даних, розробка клієнт-серверного web-застосунку, реалізація REST API, тестування програмного забезпечення та аналіз результатів роботи системи [1; 2; 3; 4; 6; 21].

Практична значущість одержаних результатів полягає в тому, що розроблений web-застосунок може бути використаний як основа для створення спеціалізованого онлайн-магазину парфумерії. AromaShop дозволяє об'єднати в одному інтерфейсі перегляд товарів, пошук, фільтрацію, кошик, замовлення, відгуки, профіль користувача, адміністративне керування та підбір ароматів за подією.

У майбутньому систему можна розширити додаванням онлайн-оплати, інтеграції зі службами доставки, персональних рекомендацій, системи промокодів, обраних товарів, push-сповіщень і розширеної аналітики продажів [1; 3; 4; 10].

1 АНАЛІЗ ЗАДАЧІ ОНЛАЙН-ПРОДАЖУ ПАРФУМЕРНОЇ ПРОДУКЦІЇ

Онлайн-продаж парфумерної продукції є частиною сфери електронної комерції, яка передбачає організацію дистанційної взаємодії між продавцем і покупцем за допомогою web-застосунку. На відміну від звичайного фізичного магазину, де покупець може безпосередньо протестувати аромат, оцінити флакон, стійкість і звучання композиції, онлайн-магазин повинен компенсувати цю відсутність за рахунок повної, структурованої та зрозумілої інформації про товар [1; 2; 3; 4; 6].

У межах даної кваліфікаційної роботи розглядається задача розробки web-застосунку «AromaShop», призначеного для онлайн-продажу парфумерії. Система повинна забезпечувати перегляд каталогу товарів, пошук і фільтрацію ароматів, перегляд детальної інформації про товар, роботу з кошиком, оформлення замовлень, перегляд історії замовлень, залишення відгуків, а також адміністративне керування товарами, брендами, користувачами та замовленнями [1; 2; 3; 4; 6].

Особливістю проекту є його орієнтація саме на парфумерну продукцію, а не на універсальний продаж будь-яких товарів. Тому в системі необхідно врахувати такі специфічні характеристики, як сімейство аромату, концентрація, об'єм, парфумерні ноти, рейтинг, відгуки, наявність на складі, а також відповідність стандартам IFRA та GS1. Додатковою функцією системи є «Підбір за подією», яка дозволяє користувачу обрати аромат не лише за назвою або брендом, а за життєвою ситуацією: для побачення, офісу, подарунка, щоденного використання або вечірнього виходу [16; 17].

1.1 Типовий процес вибору та придбання парфумерії

Типовий процес придбання парфумерії в інтернет-магазині складається з кількох послідовних етапів. Спочатку користувач відкриває головну сторінку web-

застосунку, після чого переходить до каталогу товарів. У каталозі він може переглядати доступні аромати, застосовувати пошук, фільтри та сортування. Після вибору певного товару користувач відкриває сторінку детальної інформації, де переглядає опис, бренд, ціну, об'єм, концентрацію, ноти аромату, наявність на складі та відгуки інших покупців [1; 2; 3; 4; 6].

Якщо товар підходить користувачу, він додає його до кошика. У кошику можна змінити кількість товарів, видалити зайві позиції та перевірити загальну суму замовлення. Після цього користувач переходить до оформлення замовлення, де вводить контактні дані, адресу доставки та підтверджує покупку. Після створення замовлення система зберігає його у базі даних і надає користувачу можливість переглядати статус у власному кабінеті [1; 3; 4; 10].

Для авторизованого користувача процес є більш повним, оскільки він може не лише оформити замовлення, а й переглядати історію покупок, відкривати деталі замовлень, редагувати профіль, залишати відгуки та оцінки товарам. Гість має обмежені можливості: він може переглядати каталог, користуватися пошуком, фільтрами, відкривати сторінки товарів і використовувати функцію підбору за подією, але для додавання товарів до кошика та оформлення замовлення йому потрібно авторизуватися або зареєструватися [1; 3; 4; 10].

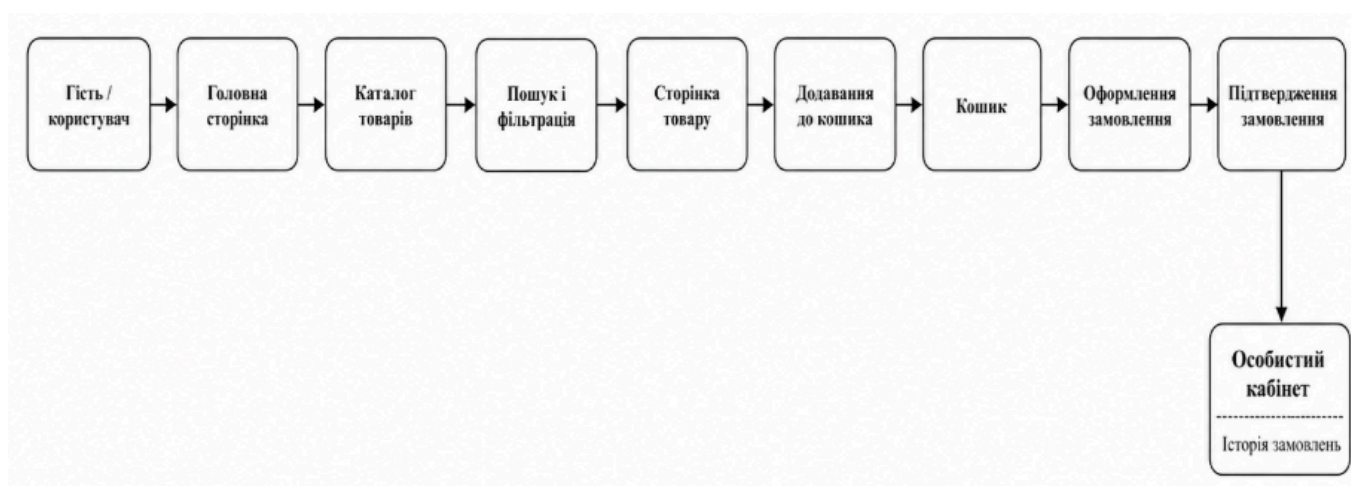


Рис. 1.1 Загальна схема процесу вибору та придбання парфумерії в онлайн-магазині

З точки зору бізнес-процесу, онлайн-магазин парфумерії повинен забезпечувати безперервний шлях користувача від первинного перегляду товарів до завершення покупки. Важливо, щоб цей шлях був простим і логічним, оскільки надмірна кількість дій або складна структура інтерфейсу може призвести до того, що покупець не завершить оформлення замовлення [10].

Для системи «AromaShop» типовий процес взаємодії користувача з магазином можна подати такою послідовністю: відкриття сайту, перегляд каталогу, пошук або фільтрація товарів, відкриття сторінки товару, додавання до кошика, редагування кошика, оформлення замовлення, перегляд статусу замовлення. Така логіка відповідає загальним принципам побудови e-commerce систем, але доповнюється специфічними для парфумерії характеристиками [1; 2; 3; 4; 6].

1.2 Особливості предметної галузі продажу парфумерії

Продаж парфумерної продукції має низку особливостей, які відрізняють цю предметну галузь від продажу більш стандартних товарів. Парфумерія не оцінюється лише за ціною, брендом або зовнішнім виглядом. Для покупця важливими є ароматичне сімейство, концентрація, сезонність, інтенсивність, стійкість, ноти композиції, призначення аромату та його доречність у певній ситуації [1; 3; 4; 10].

До основних характеристик парфумерного товару можна віднести: [16; 17]

- назву товару;
- бренд;
- опис аромату;
- короткий опис для картки товару;
- ціну та акційну ціну;
- об'єм флакона;
- концентрацію;
- ароматичне сімейство;
- верхні, середні та базові ноти;

- зображення товару;
- рейтинг і відгуки;
- кількість на складі;
- відповідність стандартам IFRA та GS1.

Сімейство аромату дозволяє класифікувати товар за загальним характером композиції. Наприклад, аромат може бути квітковим, деревним, цитрусовим, східним, фужерним, свіжим або солодким. Концентрація визначає тип парфумерної продукції: парфуми, парфумована вода, туалетна вода, одеколон тощо. Об'єм флакона впливає на ціну, зручність використання і вибір покупця [1; 3; 4; 10].

Важливим елементом опису парфумерії є парфумерна піраміда. Вона включає верхні, середні та базові ноти. Верхні ноти відчуються першими після нанесення аромату, середні формують основний характер композиції, а базові відповідають за глибину та стійкість. У web-застосунку така інформація повинна бути структурована так, щоб користувач міг швидко зрозуміти, який характер має аромат [1; 3; 4; 10].

Також для онлайн-магазину парфумерії важливими є відгуки й рейтинги. Оскільки покупець не може протестувати аромат фізично, думки інших користувачів допомагають оцінити товар. Відгуки можуть містити інформацію про стійкість, шлейф, відповідність опису, доречність для певної ситуації або сезону [1; 3; 4; 10].

Окремою особливістю системи «AromaShop» є врахування стандартів IFRA та GS1. IFRA пов'язана з безпечністю використання ароматичних компонентів, а GS1 — з ідентифікацією товарів через глобальні стандарти маркування. Для користувача така інформація може бути додатковим підтвердженням якості та структурованості товарних даних [16; 17].

1.3 Проблеми, що виникають під час онлайн-вибору ароматів

Під час онлайн-вибору парфумерної продукції користувач стикається з низкою проблем, які ускладнюють прийняття рішення про покупку. Найбільш очевидною проблемою є неможливість фізично протестувати аромат. У звичайному магазині покупець може нанести аромат на блотер або шкіру, оцінити його розкриття та зрозуміти, чи підходить він особисто йому. В онлайн-магазині такої можливості немає, тому користувач залежить від опису, нот, відгуків і візуального представлення товару [1; 3; 4; 10].

Ще однією проблемою є складність вибору без знання конкретної назви або бренду. Часто покупець не має чіткого запиту на певний товар, а лише розуміє ситуацію, для якої йому потрібен аромат. Наприклад, користувач може шукати парфум для офісу, подарунка, побачення або вечірнього заходу, але не знати, який саме бренд чи аромат йому підійде. У більшості онлайн-магазинів така задача вирішується недостатньо зручно, оскільки користувач змушений самостійно переглядати десятки товарів [1; 3; 4; 10].

Також поширеною проблемою є недостатньо гнучка фільтрація. У багатьох інтернет-магазинах фільтри обмежуються брендом, ціною, статтю або об'ємом. Однак для парфумерії цього недостатньо. Користувачу можуть бути потрібні фільтри за ароматичним сімейством, концентрацією, нотами, стилем, призначенням, сезонністю або інтенсивністю. Якщо таких параметрів немає, пошук стає менш точним [1; 2; 3; 4; 6].

Окрему проблему становлять неінформативні товарні картки. Якщо сторінка товару містить лише назву, бренд, ціну та короткий опис, користувач не отримує достатньо даних для вибору. Для парфумерної продукції важливо показувати не тільки загальні характеристики, а й ноти, концентрацію, сімейство, відгуки, рейтинг, наявність і додаткові пояснення [1; 3; 4; 10].

До основних проблем онлайн-вибору парфумерії можна віднести:

- неможливість фізично протестувати аромат;
- складність вибору без знання конкретного бренду або назви;
- недостатньо гнучку фільтрацію товарів;
- неінформативні товарні картки;

- неактуальну інформацію про наявність;
- перевантажений або незручний інтерфейс;
- відсутність підбору товарів за ситуацією використання.

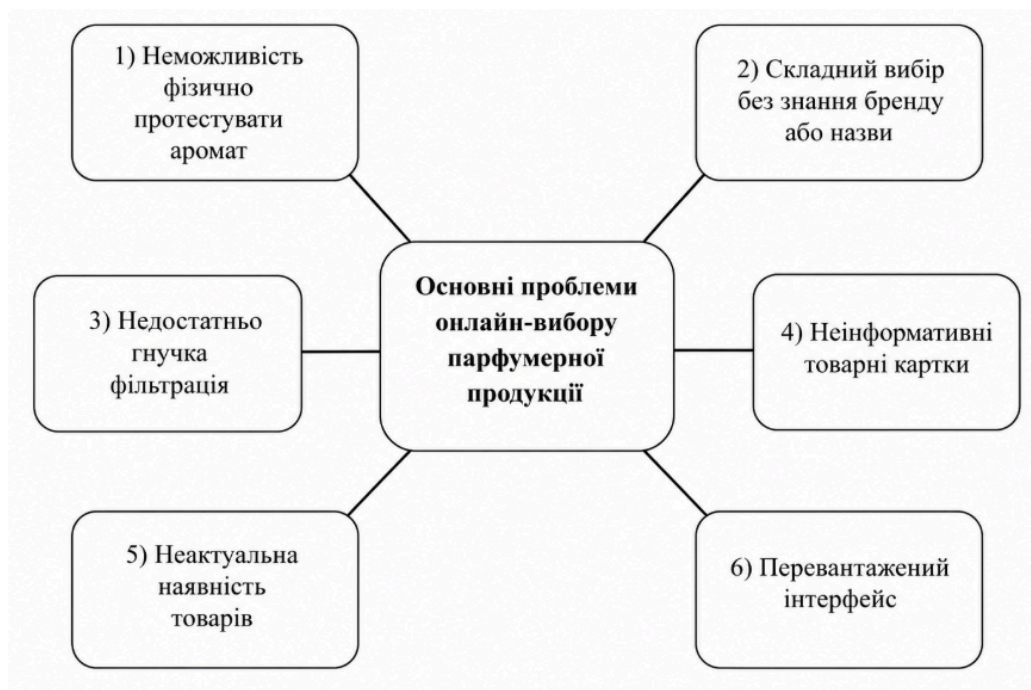


Рис. 1.2 Основні проблеми онлайн-вибору парфумерної продукції

У системі «AromaShop» частину цих проблем передбачається вирішити за рахунок структурованого каталогу, розширених характеристик товарів, відгуків користувачів, фільтрації за спеціалізованими параметрами та функції «Підбір за подією». Такий підхід дозволяє зробити вибір парфумерії більш зрозумілим навіть для користувачів, які не мають глибоких знань про бренди або ароматичні групи [1; 3; 4; 10].

1.4 Існуючі технологічні рішення для онлайн-продажу парфумерії

Для онлайн-продажу парфумерної продукції можуть використовуватися різні типи програмних рішень. Найпоширенішими є спеціалізовані інтернет-магазини, універсальні маркетплейси, beauty-платформи та web-застосунки окремих

торговельних мереж. Кожен із цих типів має власні переваги й обмеження [1; 2; 3; 4; 6].

Спеціалізовані інтернет-магазини орієнтовані саме на продаж парфумерії та косметики. Їхньою перевагою є глибша предметна структура каталогу, наявність тематичних категорій, поділ за брендами, ароматичними групами та типами продукції. Такі системи краще відповідають потребам користувачів, які цілеспрямовано шукають парфумерію [1; 2; 3; 4; 6].

Універсальні маркетплейси, навпаки, продають товари з багатьох категорій. Вони мають великий асортимент, розвинену систему пошуку, відгуків, рейтингів і доставки. Проте парфумерія в таких системах є лише однією з багатьох категорій, тому інтерфейс і фільтри не завжди враховують специфіку вибору ароматів [1; 3; 4; 10].

Beauty-платформи поєднують продаж парфумерії, косметики, доглядових засобів та інших товарів для краси. Їх перевагою є широка аудиторія та можливість комбінувати різні групи товарів. Водночас такий підхід може ускладнювати пошук саме парфумерної продукції через велику кількість супутніх категорій [1; 3; 4; 10].

Окремі торговельні мережі також можуть мати власні web-застосунки або сайти, де користувачі переглядають асортимент, акції, бонусні пропозиції та оформлюють замовлення. Такі рішення часто поєднують онлайн-продаж із фізичними магазинами, що дозволяє використовувати самовивіз, бонусні програми або перевірку наявності товарів у конкретній точці продажу [1; 2; 3; 4; 6].

Для розробки «AromaShop» доцільно врахувати переваги кожного з цих підходів, але не копіювати їх повністю. Система повинна бути достатньо спеціалізованою для парфумерії, але при цьому не перевантаженою зайвими функціями. Основний акцент має бути зроблений на зручний каталог, зрозумілу сторінку товару, фільтрацію, кошик, оформлення замовлення, відгуки та підбір за подією [1; 3; 4; 10].

1.5 Аналіз аналогів

Для визначення функціональних особливостей майбутнього web-застосунку доцільно проаналізувати існуючі програмні рішення, які використовуються для онлайн-продажу парфумерії. У межах роботи розглянуто такі аналоги: BROCARD, MAKEUP, ROZETKA та EVA.UA [1; 2; 3; 4; 6; 11].

BROCARD є спеціалізованим інтернет-магазином парфумерії та косметики. Його перевагою є орієнтація саме на продукцію beauty-сегменту. У каталозі представлені різні категорії парфумерії, бренди, акційні пропозиції та товарні картки з описом. Такий підхід є корисним для аналізу, оскільки демонструє, як може бути організований спеціалізований магазин для продажу ароматів [1; 2; 3; 4; 6; 11].

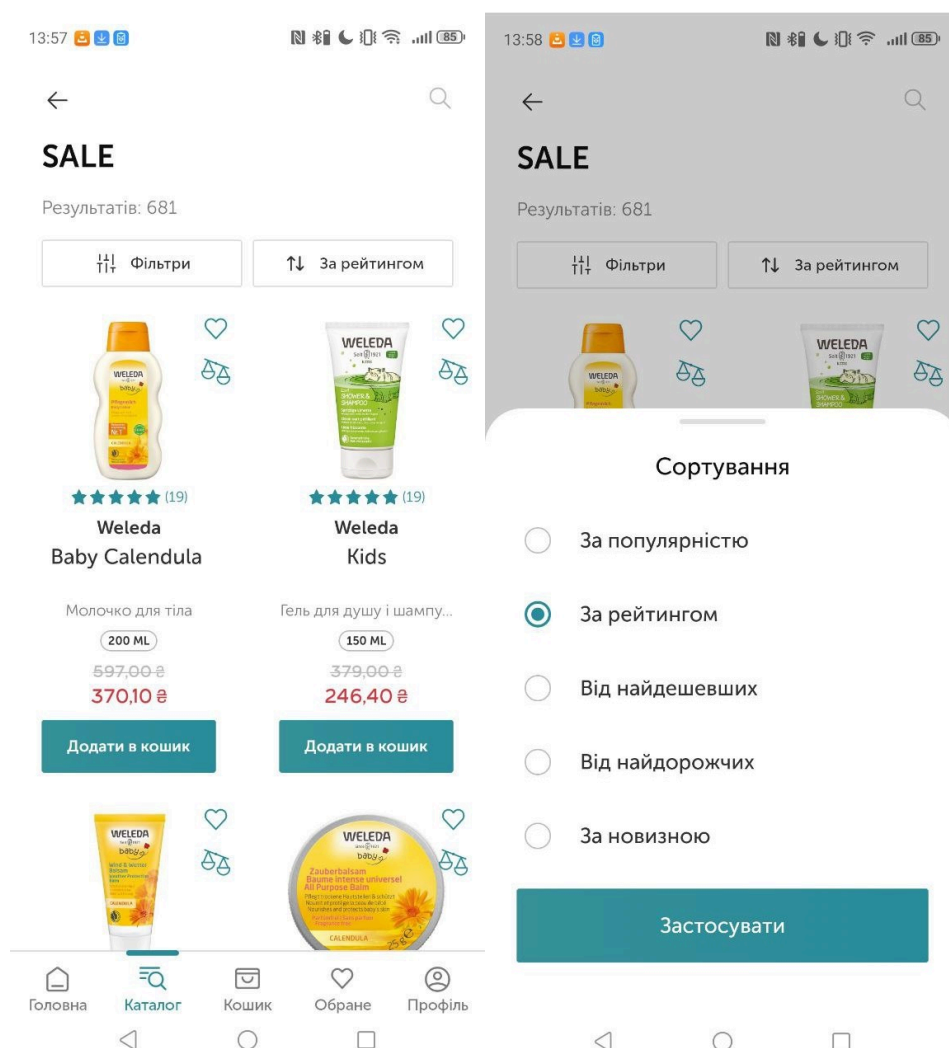


Рис. 1.3 Екранні форми інтернет-магазину BROCARD

Перевагами BROCARD є вузька спеціалізація, представлення брендової продукції, наявність тематичних категорій і можливість оформлення замовлення через web-інтерфейс. До недоліків можна віднести певну насиченість інтерфейсу та велику кількість категорій, що може ускладнювати швидкий вибір для користувача, який не знає точної назви товару [10; 11; 12].

MAKEUP є популярним інтернет-магазином косметики та парфумерії. Він має великий каталог товарів, пошук, фільтрацію, сторінки товарів, відгуки покупців і функції оформлення замовлень. MAKEUP є прикладом великої e-commerce системи, де парфумерія поєднується з косметикою та іншими товарами для догляду [1; 2; 3; 4; 6; 13].

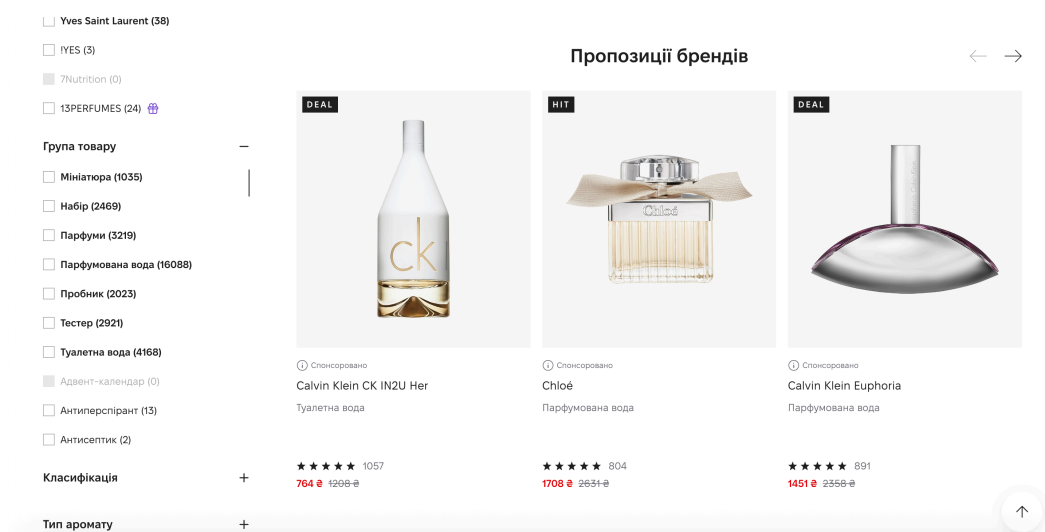


Рис. 1.4 Екранні форми інтернет-магазину MAKEUP

Перевагами MAKEUP є широкий асортимент, велика кількість відгуків, зручна система пошуку та доступність великої кількості товарів. Недоліком є те, що через масштабність каталогу користувачу може бути складніше зосередитися саме на виборі аромату. Також частина фільтрів є загальною для великого магазину, а не спеціально адаптованою під логіку вибору парфумерії [13].

ROZETKA є великим українським маркетплейсом, у якому парфумерія представлена як одна з категорій товарів. Платформа має пошук, фільтрацію, відгуки, рейтинги, акції, різних продавців і варіанти доставки. Така система є

потужною з точки зору електронної комерції, але її структура значно складніша за структуру вузькоспеціалізованого магазину [1; 2; 3; 4; 6; 14].

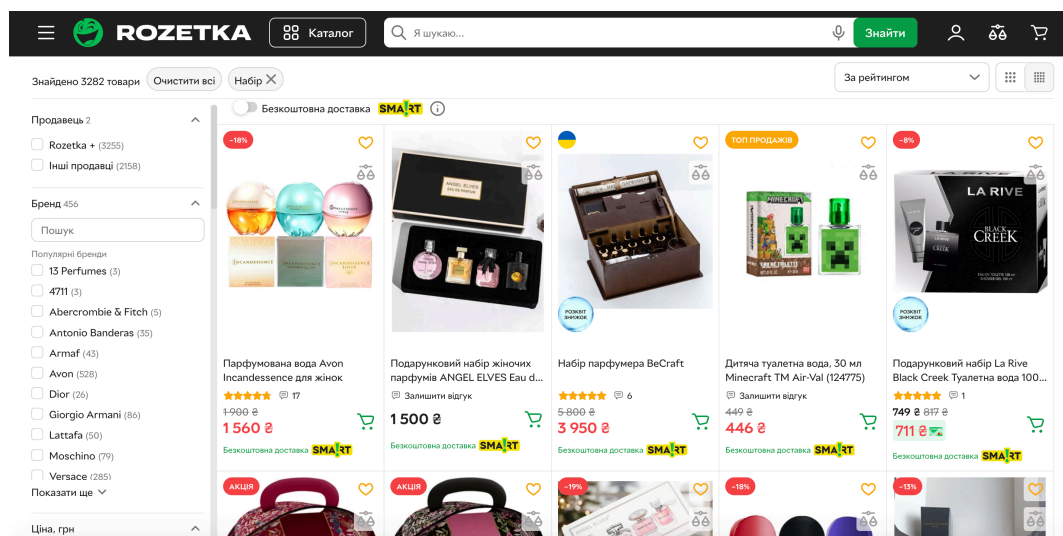


Рис. 1.5 Екранні форми розділу парфумерії ROZETKA

Перевагами ROZETKA є велика кількість товарів, довіра користувачів, розвинена система відгуків і зручні механізми оформлення замовлення. Недоліком є перевантаженість інтерфейсу, наявність багатьох продавців і велика кількість додаткових блоків, які можуть відволікати користувача від вибору парфумерії [10; 14].

EVA.UA є інтернет-магазином мережі EVA, який поєднує продаж парфумерії, косметики, засобів догляду та побутових товарів. У розділі парфумерії реалізовано товарні картки, фільтри, акції, бонусні пропозиції та відгуки користувачів [1; 2; 3; 4; 6; 15].

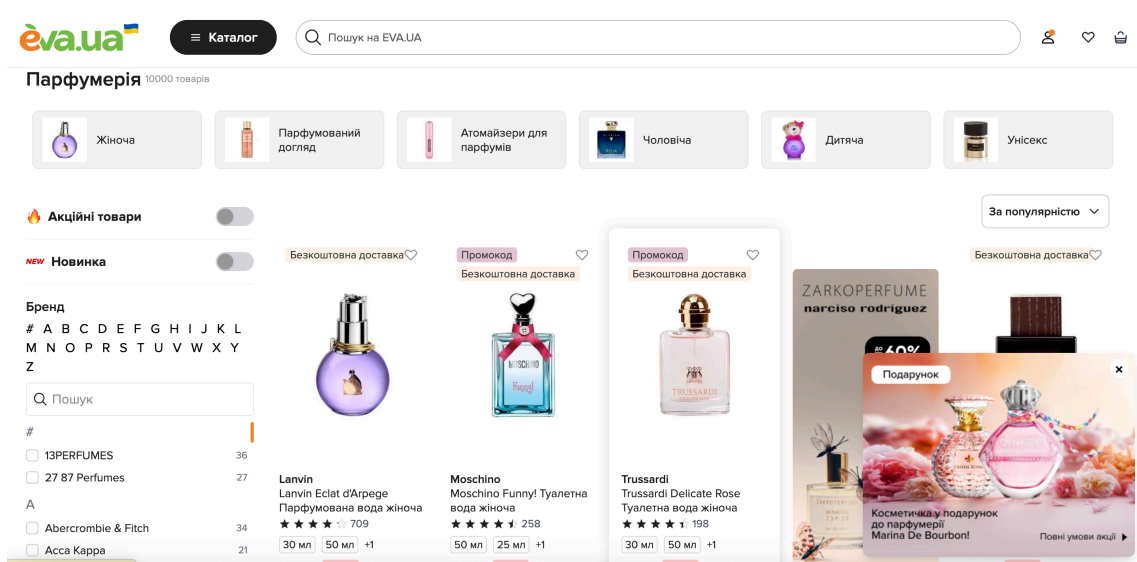


Рис. 1.6 Екранні форми інтернет-магазину EVA.UA

Перевагами EVA.UA є зрозумілі товарні картки, наявність акцій, бонусної системи та можливість швидкого додавання товарів до кошика. Недоліком є те, що магазин не є вузькоспеціалізованим лише на парфумерії, тому його структура містить багато суміжних категорій [15].

Зведений аналіз аналогів наведено в таблиці 1.1.

Таблиця 1.1

Порівняльна характеристика програмних засобів для онлайн-продажу парфумерії

Показник	BROCARD	MAKEUP	ROZETKA	EVA.UA
Каталог парфумерії	+	+	+	+
Фільтрація за ціною/брендом/об'ємом	+	+	+	+
Поділ за ароматичним сімейством	Є окремі групи ароматів	Групи ароматів подані не всюди	Залежить від продавця або категорії	Фільтр за сімейством: квіткові, деревні, цитрусові, східні тощо
Відображення парфумерних нот	Ноти подані не для всіх товарів	Ноти часто є в описі	-	Ноти аромату винесені в окремий блок кожного товару

Детальна сторінка товару	+	+	+	+
Відгуки та рейтинг	+	+	+	+
Оформлення замовлення	+	+	+	+
Підбір аромату за подією	-	-	-	-
Пояснення причини рекомендації	-	-	-	-

Аналіз аналогів показав, що більшість сучасних інтернет-магазинів має базовий набір e-commerce функцій: каталог, пошук, фільтрацію, кошик, замовлення та відгуки. Водночас у розглянутих системах майже відсутній спеціалізований підбір аромату за життєвою ситуацією. Саме тому у проєкті «AromaShop» доцільно реалізувати функцію «Підбір за подією», яка дозволить користувачу обирати аромат не лише через фільтри, а через зрозумілий сценарій використання [1; 2; 3; 4; 6].

1.6 Роль онлайн-магазину парфумерії як цифрового сервісу

Сучасний онлайн-магазин парфумерії є не просто електронним каталогом товарів. Він виконує роль цифрового сервісу, який поєднує функції представлення продукції, взаємодії з клієнтами, обробки замовлень, збирання відгуків і адміністрування асортименту. Такий сервіс повинен бути зручним як для покупця, так і для адміністратора [1; 3; 4; 10].

Для покупця онлайн-магазин має забезпечувати швидкий доступ до товарів, зрозумілу структуру каталогу, можливість пошуку, фільтрації, перегляду опису аромату, додавання товару до кошика та оформлення замовлення. Для адміністратора система повинна надавати інструменти для створення нових товарів, редагування характеристик, керування брендами, контролю залишків, перегляду замовлень і зміни їх статусів [1; 3; 4; 10].

У системі «AromaShop» важливим елементом цифрового сервісу є поділ користувачів за ролями. Гість може переглядати каталог, шукати товари,

відкривати сторінки товарів, читати відгуки та користуватися функцією підбору. Авторизований користувач додатково може працювати з кошиком, оформлювати замовлення, переглядати історію покупок, залишати відгуки та редагувати профіль. Адміністратор має доступ до керування товарами, брендами, замовленнями та користувачами [1; 3; 4; 10].

Цифровий сервіс також повинен забезпечувати актуальність даних. Якщо товар відсутній на складі, користувач повинен бачити це до оформлення замовлення. Якщо адміністратор змінює ціну або кількість товару, інформація повинна оновлюватися в системі. Такий підхід знижує ризик помилок і підвищує довіру користувачів до магазину [1; 3; 4; 10].

Окреме значення має робота з відгуками. Відгуки дозволяють користувачам отримати суб'єктивну оцінку аромату від інших покупців. Це особливо важливо для парфумерії, оскільки сприйняття запаху є індивідуальним. У системі відгуки можуть містити оцінку, коментар і дату створення. На основі цих даних формується загальне уявлення про товар [1; 3; 4; 10].

Функція «Підбір за подією» робить онлайн-магазин більш персоналізованим. Вона дозволяє користувачу не починати пошук із конкретної назви товару, а обрати ситуацію, для якої потрібен аромат. Наприклад, якщо користувач шукає парфум для подарунка, система може уточнити, кому призначений подарунок, який бюджет і який стиль аромату потрібен. Після цього користувач отримує список відповідних товарів із поясненням причини рекомендації [1; 3; 4; 10].

Таким чином, «AromaShop» розглядається як спеціалізований цифровий сервіс для онлайн-продажу парфумерії, який поєднує класичні функції інтернет-магазину з предметно орієнтованими можливостями вибору ароматів [1; 2; 3; 4; 6].

1.7 Представлення парфумерної продукції в цифровій формі

Для коректної роботи онлайн-магазину парфумерна продукція повинна бути представлена у цифровій формі як структурована сутність. У системі «AromaShop»

такою сутністю є товар, який містить не лише базові поля, а й спеціалізовані характеристики, необхідні саме для парфумерії [1; 3; 4; 10].

Базова інформація про товар включає назву, унікальний ідентифікатор, slug для формування URL-адреси, короткий опис, повний опис, ціну, акційну ціну та кількість на складі. Ці дані використовуються для відображення товару в каталозі, пошуку, фільтрації, сторінці товару, кошику та замовленнях [1; 3; 4; 10].

До спеціалізованих характеристик належать бренд, сімейство аромату, концентрація, об'єм, парфумерні ноти, зображення, рейтинг, відгуки, коди ідентифікації та ознаки відповідності стандартам. Саме ці параметри дозволяють відрізнити парфумерний товар від звичайної товарної позиції в універсальному інтернет-магазині [1; 2; 3; 4; 6].



Рис. 1.7 Цифрове представлення товару парфумерії в системі AromaShop

У базі даних товар пов'язується з брендом, зображеннями, нотами, відгуками, позиціями кошика, позиціями замовлення та рекомендаціями для подій. Така структура дозволяє використовувати один і той самий товар у різних сценаріях

роботи системи. Наприклад, товар може відображатися в каталозі, бути доданим до кошика, потрапляти до замовлення, мати відгуки та одночасно використовуватися у функції «Підбір за подією» [1; 3; 4; 10].

Парфумерні ноти доцільно зберігати окремими сутностями, оскільки одна й та сама нота може використовуватися в багатьох товарах. Наприклад, бергамот, ваніль, жасмин, сандал або мускус можуть бути частиною різних ароматів. Зв'язок між товаром і нотою містить тип ноти: верхня, середня або базова. Це дозволяє відображати парфумерну піраміду в логічному вигляді [1; 3; 4; 10].

Зображення товару також доцільно зберігати окремо, оскільки один товар може мати головне фото, додаткові фото та зображення масштабу. Для кожного зображення можна зберігати URL, альтернативний текст, ознаку головного зображення та порядок відображення [1; 3; 4; 10].

Відгуки пов'язуються з користувачем і товаром. Це дозволяє контролювати, хто залишив відгук, до якого товару він належить, яку оцінку поставив користувач і який текст коментаря було додано. На основі відгуків може формуватися середній рейтинг товару [1; 3; 4; 10].

Окрему роль відіграє цифрове представлення рекомендацій за подією. Для цього система може використовувати тип події, тип отримувача, сезон, стиль, товар і пояснення причини рекомендації. Наприклад, якщо користувач обирає аромат для офісу, система може показати легкі, ненав'язливі та нейтральні аромати, які є в наявності. Якщо користувач шукає подарунок, система може врахувати отримувача, бюджет і бажаний стиль [1; 3; 4; 10].

Таким чином, цифрове представлення парфумерного товару в системі «AromaShop» повинно бути достатньо деталізованим, щоб підтримувати не лише стандартні функції онлайн-магазину, а й спеціалізовані сценарії вибору ароматів. Це забезпечує більш точний пошук, зручнішу фільтрацію, інформативні сторінки товарів і можливість реалізації персоналізованого підбору за подією [1; 3; 4; 10].

2 ВИБІР ТЕХНІЧНИХ ЗАСОБІВ ДЛЯ РЕАЛІЗАЦІЇ WEB-ЗАСТОСУНКУ AROMASHOP

Розробка web-застосунку для онлайн-продажу парфумерної продукції потребує вибору таких технічних засобів, які забезпечують стабільну роботу клієнтської та серверної частин, надійне зберігання даних, захист користувацької інформації, зручну взаємодію через API та можливість подальшого масштабування системи.

Для проєкту AromaShop важливо обрати технології, які дозволяють реалізувати каталог товарів, пошук, фільтрацію, сторінку товару, кошик, оформлення замовлень, відгуки, профіль користувача, адміністративну панель і функцію «Підбір за подією» [1; 2; 3; 4; 6].

Оскільки система є web-застосунком, доцільним є використання клієнт-серверної архітектури. У такій архітектурі клієнтська частина відповідає за відображення інтерфейсу та взаємодію з користувачем, серверна частина виконує бізнес-логіку та обробляє запити, а база даних зберігає інформацію про користувачів, товари, бренди, замовлення, кошики, відгуки й рекомендації [10; 21].

Для реалізації AromaShop обрано технологічний стек, який включає Java 21, Spring Boot, Spring Web, Spring Security, JWT, Spring Data JPA, Hibernate, PostgreSQL, Liquibase, React, TypeScript, Vite, React Router, TanStack React Query, Swagger/OpenAPI, Docker, Docker Compose, Maven та Git. Такий набір засобів дозволяє реалізувати повноцінний web-застосунок із розділенням відповідальності між frontend, backend і базою даних [18; 19; 20; 21; 22; 23].

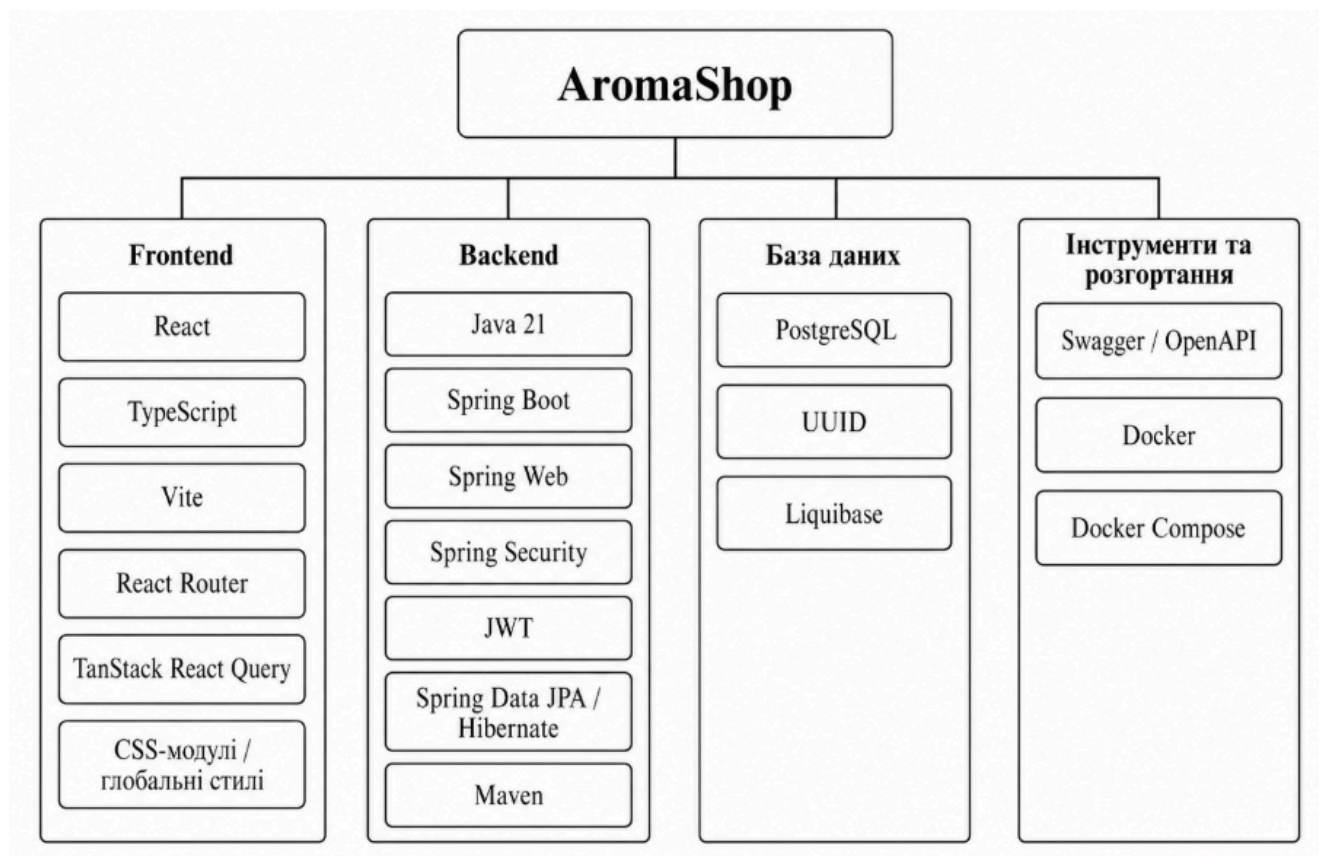


Рис. 2.1 Технологічний стек web-застосунку AromaShop

2.1 Визначення функціональних та нефункціональних вимог

Перед вибором технічних засобів необхідно визначити вимоги до майбутньої системи. Функціональні вимоги описують, які можливості повинен мати web-застосунок, а нефункціональні — яким чином система повинна працювати з точки зору швидкодії, безпеки, зручності, масштабованості та надійності [18; 19; 21; 23; 24].

До основних функціональних вимог web-застосунку AromaShop належать:

1. Реєстрація та авторизація користувачів [20].
2. Перегляд каталогу парфумерної продукції.
3. Пошук товарів за назвою та брендом.
4. Фільтрація товарів за брендом, ароматичним сімейством, концентрацією, об'ємом і ціною.
5. Перегляд детальної інформації про товар.
6. Додавання товарів до кошика.

7. Зміна кількості товарів у кошику та видалення позицій.
8. Оформлення замовлення.
9. Перегляд історії замовлень і деталей конкретного замовлення.
10. Залишення відгуків та оцінювання товарів.
11. Редагування профілю користувача.
12. Адміністративне керування товарами, брендами, користувачами та замовленнями.
13. Використання функції «Підбір за подією» для вибору аромату під конкретну ситуацію.

Нефункціональні вимоги визначають якісні характеристики системи. Для AromaShop доцільно встановити такі вимоги:

1. Швидкодія: основні сторінки системи повинні завантажуватися не довше ніж за 3 секунди.
2. Час пошуку: пошук і фільтрація товарів повинні виконуватися в межах 1–2 секунд.
3. Зручність використання: оформлення замовлення має виконуватися не більше ніж за 4–5 основних кроків.
4. Адаптивність: інтерфейс повинен коректно відображатися на екранах від 360 px до 1920 px.
5. Надійність: система повинна підтримувати не менше 100 одночасних користувачів у базовій конфігурації.
6. Безпека: паролі користувачів повинні зберігатися у захешованому вигляді.
7. Доступність: web-застосунок повинен бути доступним не менше 99% часу в умовах штатної роботи.
8. Масштабованість: система повинна дозволяти додавання нових товарів, брендів, категорій і сценаріїв підбору без зміни основної архітектури.
9. Супроводжуваність: програмний код повинен бути розділений на логічні модулі та шари.
10. Актуальність даних: інформація про залишки товарів повинна оновлюватися після оформлення замовлення.

Такі вимоги визначають подальший вибір технологій. Наприклад, потреба в захищеній авторизації обґрунтовує використання Spring Security та JWT, необхідність роботи зі структурованими даними — PostgreSQL, а вимога до зручного інтерфейсу — React і TypeScript [10; 20; 21; 23; 24].

2.2 Вибір мови програмування та фреймворку

Для реалізації серверної частини web-застосунку AromaShop обрано мову програмування Java 21. Java є однією з найбільш поширених мов для створення серверних застосунків, оскільки підтримує об'єктно-орієнтований підхід, має розвинену екосистему бібліотек, високу стабільність і добре підходить для побудови багаторівневих систем [18].

У межах проєкту AromaShop Java використовується для реалізації основної бізнес-логіки: роботи з користувачами, товарами, брендами, кошиком, замовленнями, відгуками та рекомендаціями. Об'єктно-орієнтований підхід дозволяє представити предметну область у вигляді доменних сутностей, наприклад `UserEntity`, `ProductEntity`, `BrandEntity`, `CartEntity`, `OrderEntity`, `ReviewEntity` та `EventRecommendationEntity` [18].

Для спрощення розробки серверної частини обрано Spring Boot. Цей фреймворк дозволяє швидко створювати backend-застосунки, налаштовувати REST API, підключати базу даних, реалізовувати безпеку та структурувати проєкт за шарами. У системі AromaShop Spring Boot є основою серверної частини, яка приймає HTTP-запити від frontend, обробляє їх, звертається до бази даних і повертає відповідь у форматі JSON [19; 21; 23; 26].

Серверна частина web-застосунку будується за класичною багаторівневою структурою:

- `controller` — шар контролерів, який приймає HTTP-запити;
- `service` — шар бізнес-логіки;
- `repository` — шар доступу до бази даних;
- `entity` — доменні сутності;

- dto — об'єкти передавання даних;
- security — компоненти авторизації та захисту;
- config — конфігураційні класи.

Spring Web використовується для створення REST-контролерів. За його допомогою реалізуються endpoint-и для роботи з товарами, брендами, кошиком, замовленнями, користувачами, відгуками та рекомендаціями. Наприклад, клієнтська частина може надіслати запит на отримання списку товарів, а backend повертає масив товарних позицій із необхідними характеристиками [19].

Spring Data JPA та Hibernate використовуються для роботи з базою даних через об'єктну модель. Це дозволяє зменшити кількість ручного SQL-коду і працювати з таблицями бази даних через Java-класи та репозиторії. Наприклад, ProductRepository може використовуватися для отримання товарів за брендом, ціною, сімейством аромату або наявністю на складі [19; 21].

Для збирання backend-проєкту використовується Maven. Він дозволяє керувати залежностями, запускати проєкт, виконувати тести та формувати готову збірку застосунку [18; 19; 21; 23; 24].

2.3 Вибір засобів реалізації клієнтського web-інтерфейсу

Для реалізації клієнтської частини web-застосунку AromaShop обрано React. React є бібліотекою для створення компонентного користувацького інтерфейсу. Компонентний підхід є зручним для інтернет-магазину, оскільки окремі частини інтерфейсу можна реалізувати як незалежні компоненти: картка товару, список товарів, фільтри, кошик, форма оформлення замовлення, блок відгуків, сторінка профілю та адміністративна панель [1; 2; 3; 4; 6; 10].

У системі AromaShop клієнтська частина повинна забезпечувати роботу таких сторінок:

- головна сторінка;
- каталог товарів;
- сторінка детального перегляду товару;

- сторінка підбору за подією;
- сторінка авторизації;
- сторінка реєстрації;
- кошик;
- оформлення замовлення;
- історія замовлень;
- деталі замовлення;
- профіль користувача;
- адміністративна панель товарів;
- адміністративна панель брендів;
- адміністративна панель замовлень;
- адміністративна панель користувачів.

Для підвищення надійності frontend-коду використовується TypeScript. Він додає статичну типізацію до JavaScript, що дозволяє зменшити кількість помилок під час розробки та зробити код більш зрозумілим. У проєкті TypeScript доцільно використовувати для опису типів товарів, користувачів, замовлень, кошика, відгуків і відповідей API [23; 24].

React Router використовується для організації маршрутизації між сторінками. Наприклад, маршрути можуть відповідати сторінкам каталогу, товару, кошика, оформлення замовлення, профілю користувача та адміністративних розділів [23].

TanStack React Query використовується для роботи із серверними даними. Ця бібліотека дозволяє зручно виконувати HTTP-запити, кешувати отримані дані, оновлювати інформацію після змін і синхронізувати стан клієнтської частини з backend. Для AromaShop це важливо, оскільки каталог товарів, кошик, замовлення й відгуки повинні регулярно отримуватися з серверної частини [23].

Клієнтська частина взаємодіє з backend через REST API. Дані передаються у форматі JSON. Наприклад, frontend надсилає запит на отримання товарів, backend повертає список товарів, після чого React-компоненти відображають їх у каталозі [23; 26].

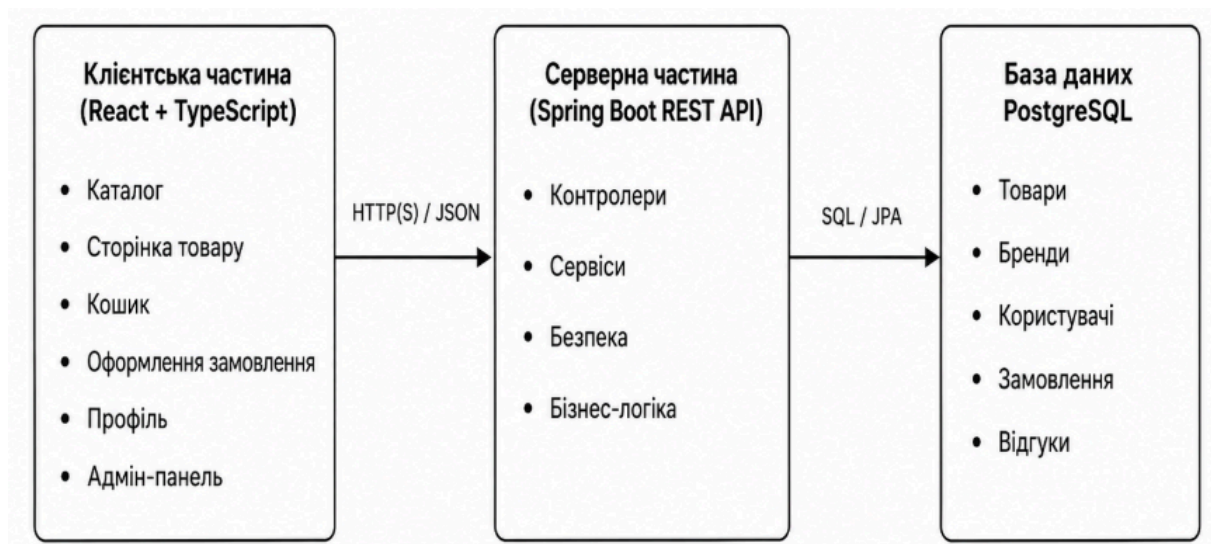


Рис. 2.2 Схема взаємодії клієнтської та серверної частин через REST API

2.4 Вибір бази даних та засобів керування міграціями

Для зберігання даних web-застосунку AromaShop обрано PostgreSQL. Це реляційна система керування базами даних, яка добре підходить для зберігання структурованої інформації та підтримує складні зв'язки між таблицями. Для онлайн-магазину така база даних є доцільною, оскільки система повинна працювати з великою кількістю взаємопов'язаних сутностей [21].

У базі даних AromaShop необхідно зберігати такі основні групи даних:

- користувачі;
- бренди;
- товари;
- зображення товарів;
- парфумерні ноти;
- зв'язки товарів із нотами;
- кошики;
- позиції кошика;
- замовлення;
- позиції замовлень;
- відгуки;

- типи подій для підбору;
- рекомендації товарів за подіями.

PostgreSQL дозволяє організувати ці дані у вигляді таблиць із первинними та зовнішніми ключами. Наприклад, один бренд може мати багато товарів, один користувач може мати багато замовлень, один товар може мати багато відгуків, а один аромат може бути пов'язаний із кількома нотами через проміжну таблицю `product_notes` [21].

Для ідентифікаторів сутностей у проєкті доцільно використовувати UUID. Такий підхід дозволяє створювати унікальні ідентифікатори для записів і зменшує ризик конфліктів при масштабуванні або перенесенні даних між середовищами [18; 19; 21; 23; 24].

Для керування структурою бази даних використовується Liquibase. Цей інструмент дозволяє описувати зміни схеми бази даних у вигляді міграцій. Наприклад, створення таблиць `users`, `products`, `orders` або `reviews` може бути оформлене у `changelog`-файлах. Це дозволяє контролювати історію змін бази даних і синхронізувати структуру між різними середовищами розробки [21; 22].

Перевагою Liquibase є те, що всі зміни бази даних зберігаються у проєкті разом із програмним кодом. Це спрощує командну розробку, тестування та розгортання застосунку. Якщо потрібно додати нову таблицю або поле, створюється окрема міграція, яка може бути виконана автоматично під час запуску системи [21; 22].

2.5 Вибір засобів аутентифікації, авторизації та взаємодії через API

Оскільки web-застосунок AromaShop передбачає роботу з особистими кабінетами, замовленнями, профілями користувачів та адміністративними функціями, важливо реалізувати механізми аутентифікації та авторизації. Для цього використовується Spring Security у поєднанні з JWT [20].

Аутентифікація дозволяє перевірити, ким є користувач. Для цього користувач вводить email і пароль, після чого сервер перевіряє ці дані. Якщо дані

коректні, система генерує JWT-токен. Цей токен надалі використовується для доступу до захищених ресурсів [20].

Авторизація визначає, які дії дозволені конкретному користувачу. У системі AromaShop передбачено три основні ролі: [20]

- гість;
- авторизований користувач;
- адміністратор.

Гість може переглядати каталог, шукати товари, застосовувати фільтри, відкривати сторінки товарів, читати відгуки та користуватися функцією «Підбір за подією». Авторизований користувач додатково може працювати з кошиком, оформлювати замовлення, переглядати історію покупок, залишати відгуки та редагувати профіль. Адміністратор має доступ до керування товарами, брендами, користувачами, замовленнями, відгуками та залишками товарів [18; 19; 21; 23; 24].

JWT підходить для реалізації авторизації в клієнт-серверному web-застосунку, оскільки дозволяє передавати інформацію про користувача у вигляді токена. Після входу користувача frontend зберігає токен і додає його до наступних запитів у заголовок Authorization. Backend перевіряє токен і визначає, чи має користувач право виконати певну дію [20; 23].

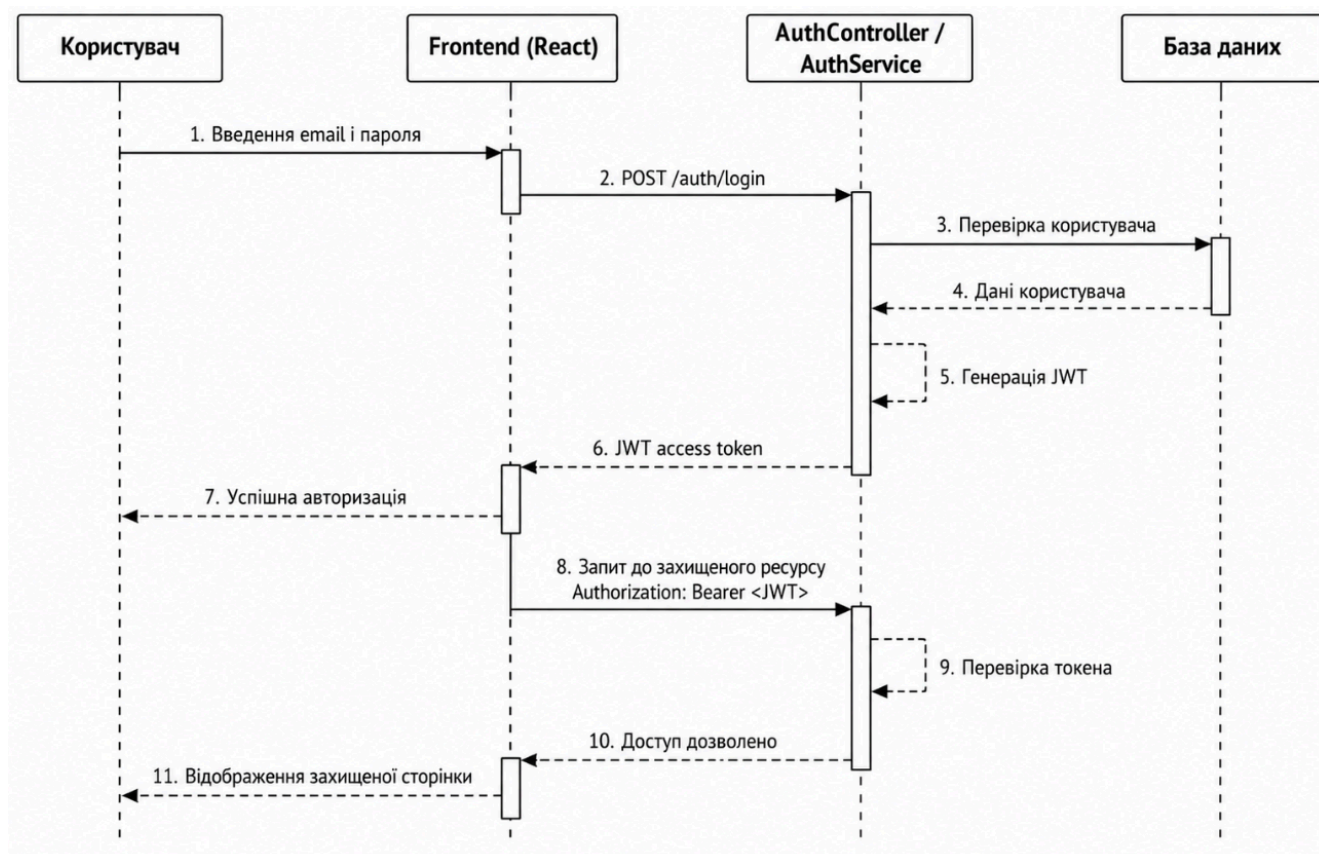


Рис. 2.3 Схема авторизації користувача за допомогою JWT

Взаємодія між frontend і backend реалізується через REST API. Кожна група функцій системи може мати окремі endpoint-и. Наприклад: [23; 26]

- /api/auth — реєстрація та авторизація;
- /api/products — робота з товарами;
- /api/brands — робота з брендами;
- /api/cart — робота з кошиком;
- /api/orders — робота із замовленнями;
- /api/reviews — робота з відгуками;
- /api/users — робота з користувачами;
- /api/recommendations — підбір ароматів за подією.

Для документування REST API використовується Swagger/OpenAPI. Це дозволяє переглядати доступні endpoint-и, параметри запитів, формати відповідей і тестувати API безпосередньо через web-інтерфейс документації. Такий інструмент є корисним як під час розробки, так і під час тестування системи [26].

2.6 Вибір інструментів контейнеризації, тестування та контролю версій

Для запуску компонентів системи в ізолюваному середовищі доцільно використовувати Docker. Docker дозволяє створити окремі контейнери для backend, frontend і бази даних. Це спрощує запуск системи на різних комп'ютерах і зменшує ризик проблем, пов'язаних із відмінностями середовища [21; 23; 27].

Docker Compose використовується для запуску кількох контейнерів одночасно. У випадку AromaShop можна описати окремі сервіси: [27]

- frontend — клієнтська частина React + Vite;
- backend — серверна частина Java + Spring Boot;
- db — база даних PostgreSQL.

Також у Docker Compose можна налаштувати мережу між контейнерами, змінні середовища, порти та volume для збереження даних PostgreSQL. Це дозволяє запускати весь web-застосунок однією командою та швидко відтворювати середовище розробки або тестування [21; 27].

Для тестування API може використовуватися Swagger/OpenAPI або Postman. За допомогою цих інструментів можна перевірити коректність endpoint-ів, передавання параметрів, створення товарів, авторизацію, роботу кошика, оформлення замовлення та інші сценарії. [20; 26].

Для контролю версій використовується Git. Він дозволяє відстежувати зміни у програмному коді, повертатися до попередніх версій, створювати гілки для окремих функцій і підтримувати порядок у процесі розробки. GitHub може використовуватися як віддалений репозиторій для збереження коду, спільної роботи та перегляду історії змін [18; 19; 21; 23; 24].

3 ПРОЄКТУВАННЯ WEB-ЗАСТОСУНКУ ДЛЯ ОНЛАЙН-ПРОДАЖУ ПАРФУМЕРІЇ

Проектування web-застосунку AromaShop є важливим етапом розробки, оскільки саме на цьому етапі визначається структура майбутньої системи, взаємодія її компонентів, модель даних, ролі користувачів, основні сценарії роботи та логіка реалізації ключових функцій. Для онлайн-магазину парфумерії важливо не лише забезпечити стандартні можливості електронної комерції, а й врахувати особливості предметної галузі: парфумерні ноти, ароматичні сімейства, концентрацію, об'єм, відгуки, рейтинги, наявність товарів і підбір ароматів за подією [1; 2; 3; 4; 6].

У межах проектування web-застосунку AromaShop було визначено загальну архітектуру системи, логічну структуру, доменну модель, структуру бази даних, сценарії взаємодії користувачів із системою та алгоритм роботи функції «Підбір за подією». Проектування виконувалося з урахуванням поділу системи на клієнтську частину, серверну частину та базу даних [21].

3.1 Проектування загальної архітектури web-застосунку

Web-застосунок AromaShop проектується за клієнт-серверною архітектурою. Такий підхід дозволяє розділити відповідальність між окремими частинами системи: клієнтська частина відповідає за інтерфейс користувача, серверна частина — за бізнес-логіку, а база даних — за зберігання інформації [21].

Клієнтська частина реалізується за допомогою React і TypeScript. Вона забезпечує відображення сторінок web-застосунку, обробку дій користувача, навігацію між сторінками та передавання запитів до серверної частини. Через клієнтський інтерфейс користувач може переглядати каталог, відкривати сторінки товарів, користуватися фільтрами, додавати товари до кошика, оформлювати замовлення, переглядати історію покупок і працювати з профілем [23; 24].

Серверна частина реалізується за допомогою Java 21 та Spring Boot. Вона приймає HTTP-запити від frontend-частини, перевіряє права доступу, виконує бізнес-логіку, працює з базою даних і повертає результати у форматі JSON. На серверному рівні реалізуються модулі авторизації, роботи з товарами, брендами, кошиком, замовленнями, відгуками, користувачами та рекомендаціями за подією [18; 19; 20; 23].

База даних PostgreSQL використовується для зберігання структурованої інформації. У ній зберігаються дані про користувачів, товари, бренди, парфумерні ноти, зображення товарів, кошики, замовлення, позиції замовлень, відгуки та рекомендації. Така структура дозволяє забезпечити цілісність даних і реалізувати зв'язки між основними сутностями системи [21].

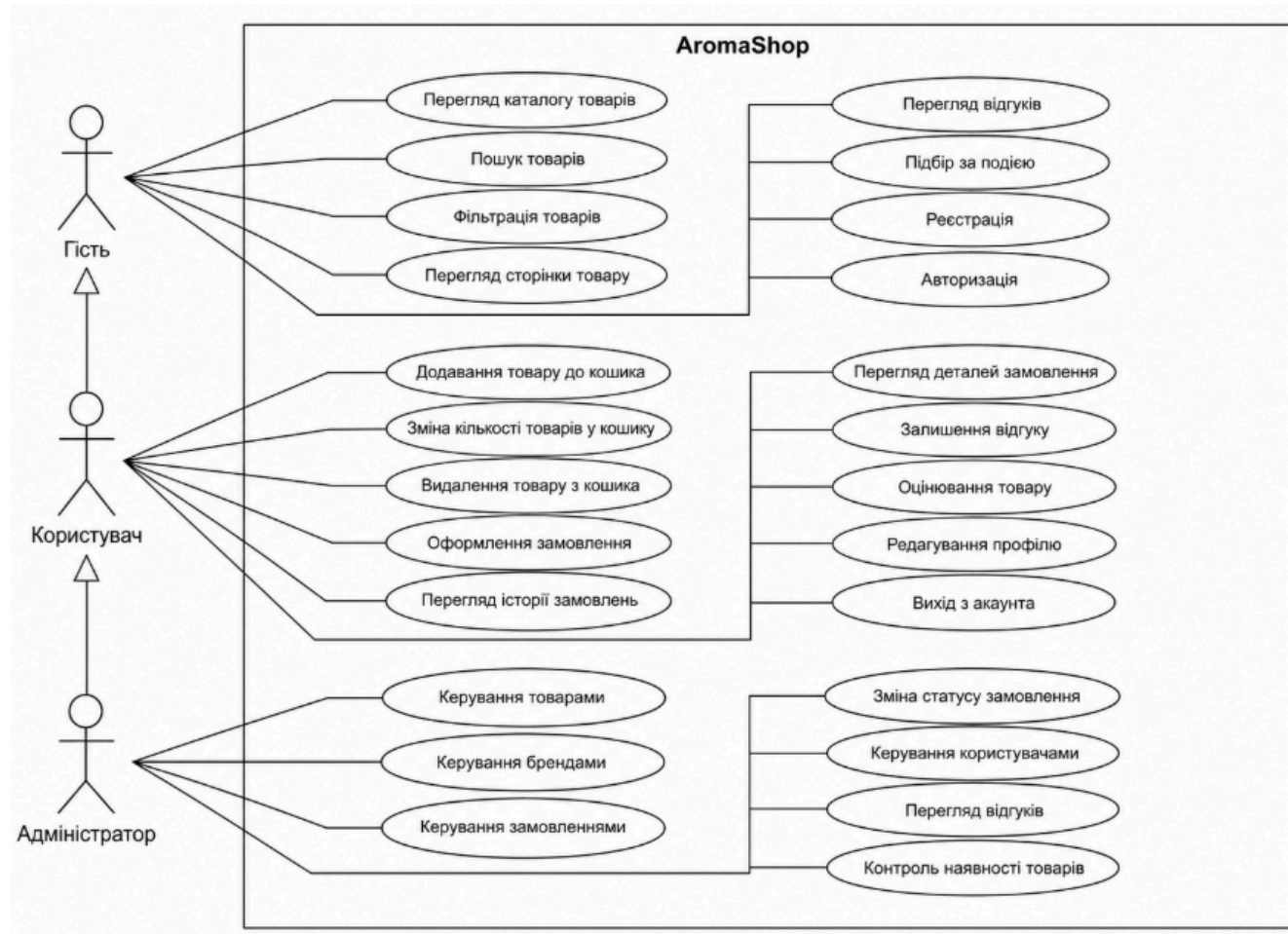


Рис. 3.1 Діаграма варіантів використання web-застосунку AromaShop

Діаграма варіантів використання відображає основні ролі системи та доступні їм функції. У системі передбачено три ролі: гість, авторизований користувач і адміністратор. Гість може переглядати каталог, шукати та фільтрувати товари, відкривати сторінку товару, переглядати відгуки, користуватися підбором за подією, а також перейти до реєстрації або авторизації. Авторизований користувач наслідує можливості гостя та додатково може працювати з кошиком, оформлювати замовлення, переглядати історію покупок, залишати відгуки, оцінювати товари, редагувати профіль і виходити з акаунта. Адміністратор наслідує можливості авторизованого користувача та має доступ до керування товарами, брендами, замовленнями, користувачами, відгуками й контролю наявності товарів [20].

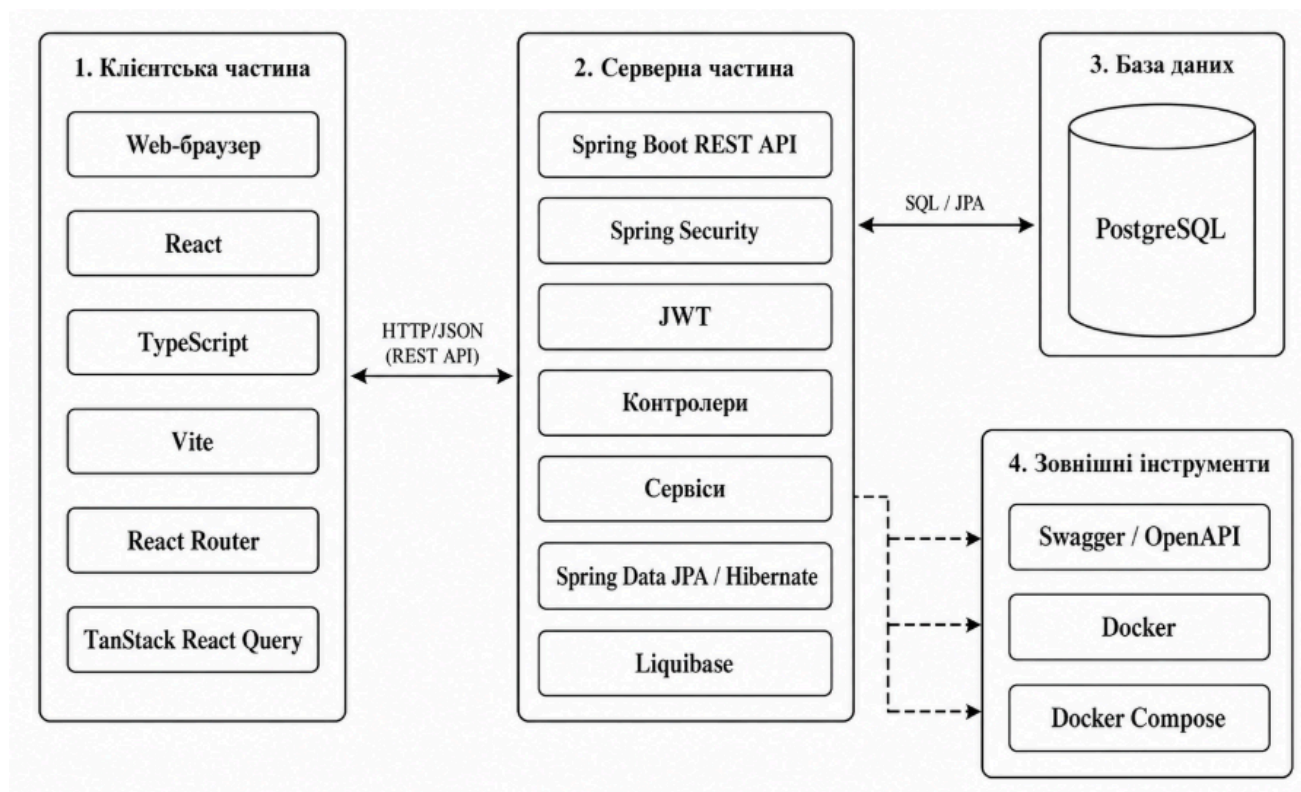


Рис. 3.2 Архітектура web-застосунку AromaShop

Загальна архітектура AromaShop включає клієнтську частину, серверну частину, базу даних і зовнішні інструменти. Клієнтська частина працює у web-браузері та взаємодіє із серверною частиною через REST API. Серверна частина містить контролери, сервіси, репозиторії, компоненти безпеки та механізм міграцій

бази даних. База даних PostgreSQL зберігає всі основні дані системи. Для документування API використовується Swagger/OpenAPI, а для контейнеризації — Docker і Docker Compose [21; 22; 26; 27].

3.2 Проєктування логічної структури системи

Логічна структура web-застосунку AromaShop будується за модульним принципом. Кожен модуль відповідає за окрему групу функцій, що спрощує розробку, тестування та подальше розширення системи. Такий підхід дозволяє уникнути змішування логіки різних частин застосунку та забезпечує зручність супроводу коду [19; 21; 23; 26].

До основних логічних модулів системи належать:

1. Модуль аутентифікації та авторизації [20].
2. Модуль користувачів.
3. Модуль товарів.
4. Модуль брендів.
5. Модуль парфумерних нот.
6. Модуль зображень товарів.
7. Модуль кошика.
8. Модуль замовлень.
9. Модуль відгуків.
10. Модуль підбору за подією.
11. Адміністративний модуль.

Модуль аутентифікації та авторизації відповідає за реєстрацію користувачів, вхід у систему, генерацію JWT-токена та перевірку прав доступу. Він забезпечує розмежування можливостей між гостем, авторизованим користувачем і адміністратором [20].

Модуль користувачів відповідає за зберігання та оновлення інформації про користувачів. До таких даних належать email, username, пароль, роль, повне ім'я, телефон, адреса та дата створення акаунта. Через цей модуль користувач може

редагувати профіль, а адміністратор — переглядати та керувати користувачами [19; 21; 23; 26].

Модуль товарів є одним із центральних у системі. Він відповідає за створення, редагування, видалення та перегляд парфумерної продукції. Товар має назву, опис, бренд, ціну, акційну ціну, кількість на складі, ароматичне сімейство, концентрацію, об'єм, код EAN/UPC, ознаки відповідності GS1 та IFRA, зображення й ноти [16; 17].

Модуль брендів використовується для керування інформацією про виробників парфумерії. Один бренд може мати багато товарів, тому бренд є окремою сутністю, пов'язаною з товарами через відношення «один до багатьох» [19; 21; 23; 26].

Модуль парфумерних нот відповідає за зберігання нот ароматів. Оскільки одна й та сама нота може використовуватися в багатьох товарах, між товарами й нотами створюється проміжна сутність `ProductNoteEntity`. Вона визначає, до якого товару належить нота, який тип має нота — верхня, середня або базова — і в якому порядку вона відображається [19; 21; 23; 26].

Модуль кошика дозволяє авторизованому користувачу додавати товари, змінювати їх кількість і видаляти позиції. Кошик пов'язаний з конкретним користувачем і містить набір позицій `CartItemEntity` [19; 21; 23; 26].

Модуль замовлень відповідає за створення замовлення на основі кошика, збереження контактних даних користувача, адреси доставки, статусу замовлення, загальної суми та складу покупки. Для збереження товарів у межах замовлення використовується сутність `OrderItemEntity` [19; 21; 23; 26].

Модуль відгуків дозволяє користувачам залишати оцінки та коментарі до товарів. Кожен відгук пов'язаний із конкретним користувачем і конкретним товаром.

Модуль підбору за подією є спеціалізованою частиною системи. Він дозволяє користувачу обрати ситуацію, для якої потрібен аромат, відповісти на уточнювальні питання та отримати список відповідних товарів з поясненням причини рекомендації [19; 21; 23; 26].

Адміністративний модуль забезпечує керування товарами, брендами, користувачами, замовленнями, відгуками та залишками. Він доступний лише користувачам із роллю адміністратора [19; 21; 23; 26].

3.3 Проєктування структури бази даних

База даних web-застосунку AromaShop проєктується як реляційна модель, у якій основні сутності системи представлені окремими таблицями. Такий підхід дозволяє забезпечити цілісність даних, уникнути дублювання інформації та реалізувати зв'язки між товарами, користувачами, кошиками, замовленнями, відгуками й рекомендаціями [21].

Основними таблицями бази даних є: [21]

- users;
- brands;
- products;
- product_images;
- notes;
- product_notes;
- carts;
- cart_items;
- orders;
- order_items;
- reviews;
- recommendation_events;
- event_recommendations.

Таблиця users зберігає інформацію про користувачів системи. Вона містить ідентифікатор користувача, email, username, пароль, роль, повне ім'я, телефон, адресу та дату створення. Користувач може мати один кошик, багато замовлень і багато відгуків.

Таблиця `brands` містить інформацію про бренди парфумерії. До її полів належать ідентифікатор, назва, опис і URL логотипу. Один бренд може бути пов'язаний із багатьма товарами.

Таблиця `products` зберігає основну інформацію про товари. Вона містить назву, `slug`, опис, короткий опис, посилання на бренд, ціну, акційну ціну, кількість на складі, ароматичне сімейство, концентрацію, об'єм, код EAN/UPC, ознаки відповідності GS1 та IFRA, а також примітку IFRA.

Таблиця `product_images` використовується для зберігання зображень товару. Один товар може мати кілька зображень: головне фото, додаткові фото та зображення масштабу. Для кожного зображення зберігаються URL, альтернативний текст, ознака головного зображення, ознака зображення масштабу та порядок відображення.

Таблиця `notes` зберігає назви парфумерних нот. Таблиця `product_notes` є проміжною і визначає зв'язок між товарами та нотами. У ній також зберігається тип ноти: верхня, середня або базова. Це дозволяє формувати парфумерну піраміду для кожного товару.

Таблиця `carts` зберігає кошики користувачів, а `cart_items` — окремі позиції кошика. Один кошик може містити багато товарів, а кожна позиція містить посилання на товар і кількість.

Таблиця `orders` зберігає замовлення користувачів. Вона містить загальну суму, статус, адресу доставки, ім'я, телефон, email і дату створення. Таблиця `order_items` зберігає товари, які входять до конкретного замовлення, їх кількість і ціну на момент покупки.

Таблиця `reviews` використовується для зберігання відгуків. Вона містить посилання на користувача, товар, оцінку, коментар і дату створення.

Таблиця `recommendation_events` містить типи подій для підбору ароматів, наприклад побачення, офіс, подарунок, кожен день або вечір. Таблиця `event_recommendations` пов'язує події з товарами та містить додаткові параметри: тип отримувача, сезон, стиль і пояснення, чому аромат підходить до обраної ситуації.

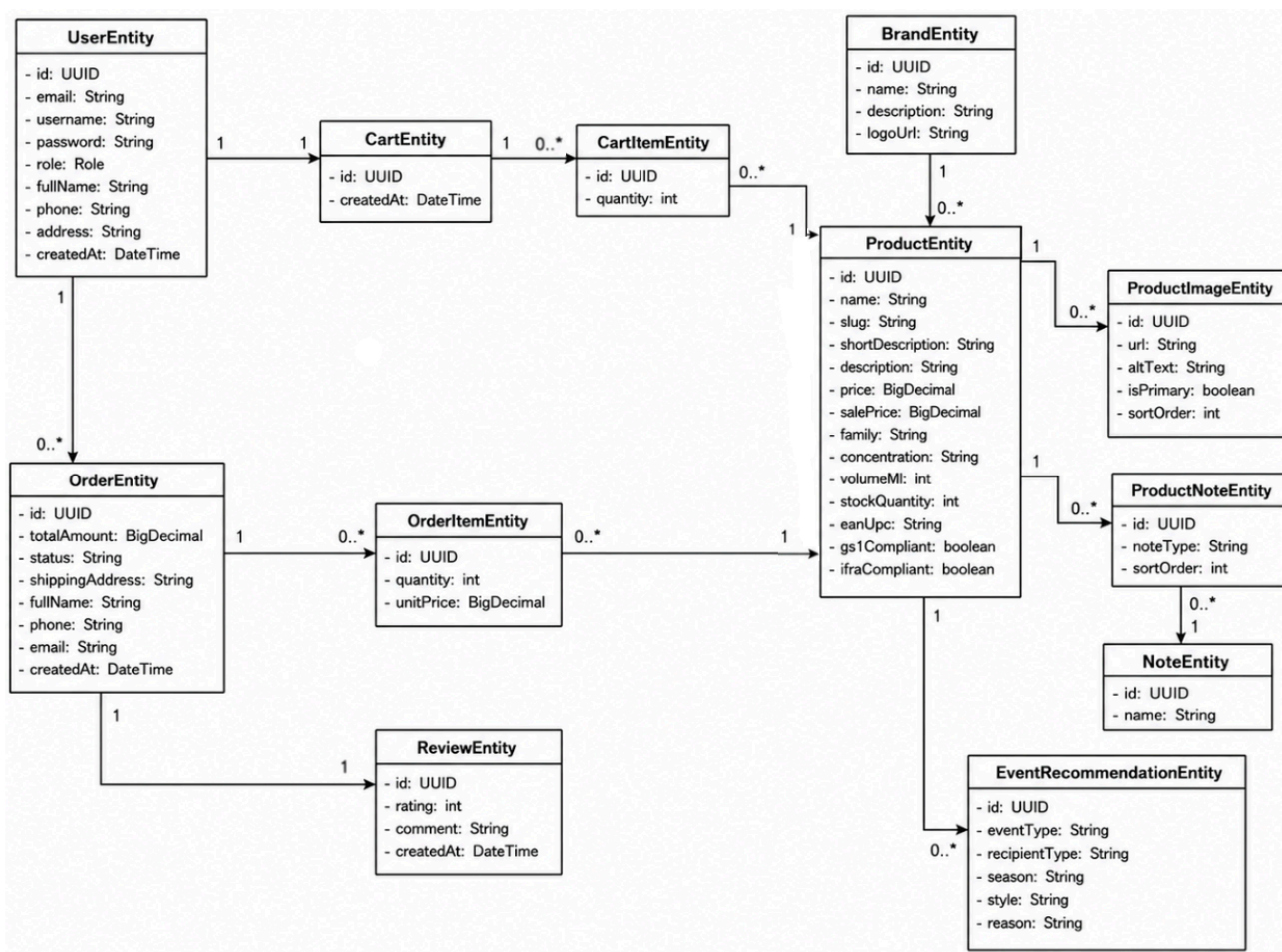


Рис. 3.3 Діаграма класів доменної моделі AromaShop

Діаграма класів доменної моделі відображає основні сутності системи та зв'язки між ними. Вона дозволяє побачити, як користувач пов'язаний із кошиком, замовленнями та відгуками, як товар пов'язаний із брендом, зображеннями, нотами, відгуками, позиціями кошика, позиціями замовлення та рекомендаціями [19; 21; 23; 26].

ER-діаграма бази даних показує таблиці, первинні та зовнішні ключі, типи даних і зв'язки між сутностями. Вона використовується як основа для реалізації структури бази даних у PostgreSQL і подальшого створення міграцій Liquibase [21; 22].

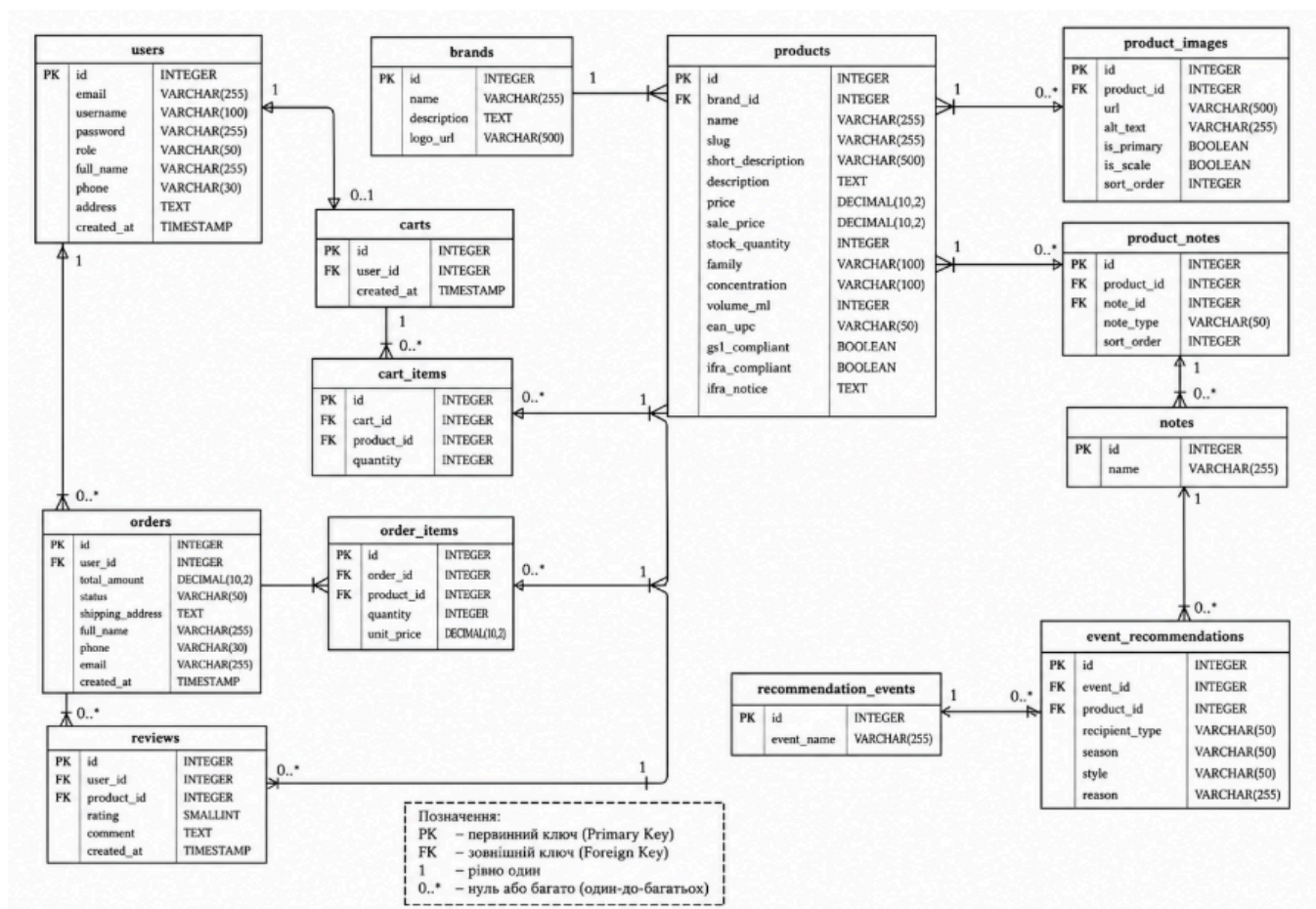


Рис. 3.6 ER-діаграма бази даних web-застосунку AromaShop

3.4 Проєктування серверної частини Spring Boot Backend

Серверна частина AromaShop проєктується за багаторівневою архітектурою. Основними рівнями є контролери, сервіси, репозиторії, доменні сутності, DTO та компоненти безпеки. Такий підхід дозволяє розділити логіку обробки HTTP-запитів, бізнес-операції та роботу з базою даних [19; 21; 23; 26].

Контролери відповідають за приймання запитів від клієнтської частини. Наприклад, `ProductController` обробляє запити, пов'язані з товарами, `OrderController` — із замовленнями, `CartController` — з кошиком, `AuthController` — з авторизацією та реєстрацією, `ReviewController` — з відгуками [20].

Сервісний шар містить основну бізнес-логіку системи. Саме в сервісах виконується перевірка даних, формування відповідей, створення замовлень, оновлення кількості товарів, перевірка наявності, створення відгуків і формування

рекомендацій. До основних сервісів належать AuthServiceImpl, ProductServiceImpl, CartServiceImpl, OrderServiceImpl, ReviewServiceImpl, UserServiceImpl, BrandServiceImpl та EventRecommendationServiceImpl [19; 21; 23; 26].

Репозиторії відповідають за доступ до бази даних. Вони реалізуються за допомогою Spring Data JPA та дозволяють виконувати операції пошуку, створення, оновлення й видалення записів. Наприклад, ProductRepository може забезпечувати пошук товарів за назвою, брендом, сімейством аромату, концентрацією або ціновим діапазоном [19; 21].

DTO використовуються для передавання даних між клієнтською та серверною частинами. Це дозволяє не передавати напряду доменні сутності та контролювати структуру відповіді API. Наприклад, ProductDto може містити лише ті поля, які потрібні для відображення товару в каталозі, а ProductDetailsDto — розширену інформацію для сторінки товару [19; 21; 23; 26].

Компоненти безпеки відповідають за перевірку JWT-токена, захист endpoint-ів і визначення прав доступу. Публічні endpoint-и можуть бути доступні гостям, а захищені — лише авторизованим користувачам або адміністраторам [20].

3.5 Проєктування функції «Підбір за подією»

Функція «Підбір за подією» є однією з ключових особливостей web-застосунку AromaShop. Вона призначена для користувачів, які не знають конкретної назви аромату або бренду, але розуміють ситуацію, для якої їм потрібен парфум [19; 21; 23; 26].

У системі передбачено кілька основних сценаріїв підбору:

- на побачення;
- в офіс;
- на подарунок;
- на кожен день;
- на вечір.

Після вибору події система задає уточнювальні питання. Наприклад, для подарунка користувач може вказати, кому призначений аромат: дівчині, хлопцю, мамі, колезі або як універсальний варіант. Також може бути враховано бюджет і бажаний стиль: свіжий, солодкий, елегантний або класичний [19; 21; 23; 26].

Для побачення система може уточнювати, для кого обирається аромат — для жінки, чоловіка або унісекс, а також яке враження потрібно створити: романтичне, тепле, виразне або легке [19; 21; 23; 26].

Для офісу важливими є легкість, нейтральність і ненав'язливість аромату. Для вечірнього виходу можуть враховуватися сезон, насиченість і стиль композиції: деревний, східний, квітковий або солодкий [19; 21; 23; 26].

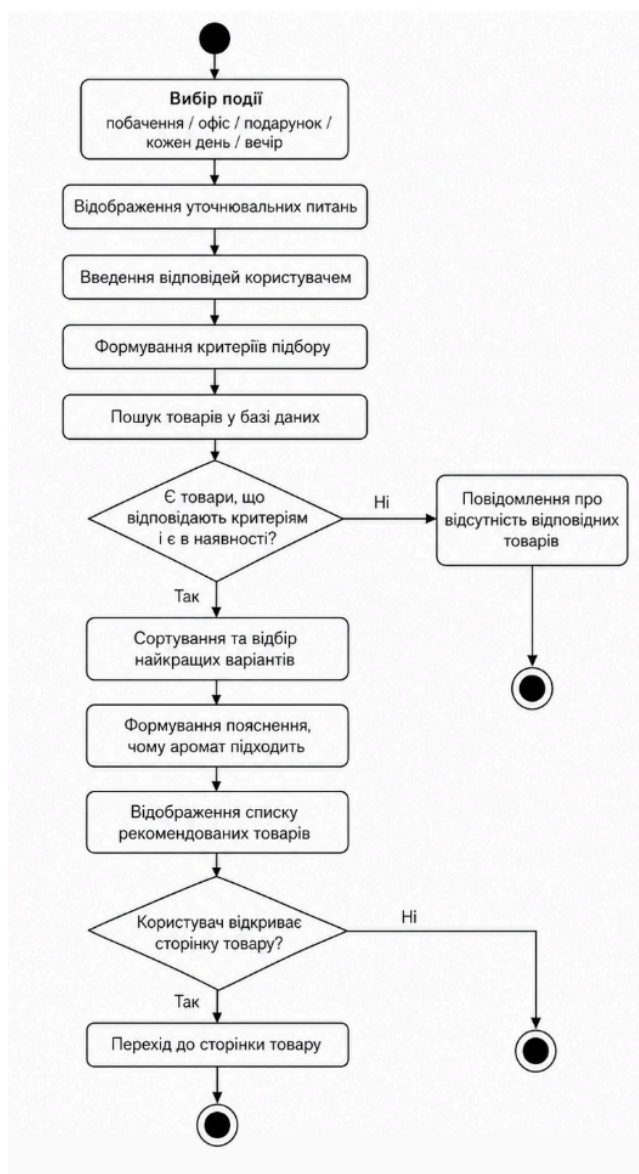


Рис. 3.4 Діаграма роботи функції «Підбір за подією»

У базі даних функція підбору реалізується через таблиці `recommendation_events` та `event_recommendations`. Перша таблиця містить типи подій, а друга — зв'язок між подією, товаром і додатковими параметрами. Поле `reason` використовується для пояснення, чому конкретний аромат підходить до обраної ситуації [19; 21; 23; 26].

Наприклад, якщо користувач обирає сценарій «офіс», система може запропонувати легкі та ненав'язливі аромати з помірною інтенсивністю. Якщо користувач обирає «вечір», система може підібрати більш насичені композиції з деревними, східними або солодкими нотами [19; 21; 23; 26].

Такий механізм дозволяє зробити вибір парфумерії більш інтуїтивним. Користувач не повинен знати конкретні бренди або ноти, оскільки система допомагає перейти від життєвої ситуації до конкретних товарів [19; 21; 23; 26].

3.6 Проєктування клієнтського web-інтерфейсу

Клієнтський web-інтерфейс AromaShop повинен бути простим, зрозумілим і зручним для користувача. Основна мета інтерфейсу — забезпечити швидкий доступ до каталогу, фільтрів, сторінки товару, кошика, оформлення замовлення, профілю та функції підбору за подією [10].

Компонентна структура frontend-частини повинна забезпечувати повторне використання елементів. Наприклад, компонент `ProductCard` може використовуватися як у каталозі, так і в результатах підбору за подією. Компонент `FilterPanel` відповідає за фільтрацію товарів. Компонент `CartItem` використовується для відображення позицій кошика. Компонент `ReviewList` показує відгуки на сторінці товару [23].

Для навігації між сторінками використовується `React Router`. Наприклад, каталог може бути доступний за маршрутом `/catalog`, сторінка товару — за `/products/:slug`, кошик — за `/cart`, оформлення замовлення — за `/checkout`, а адміністративна панель — за `/admin` [23].

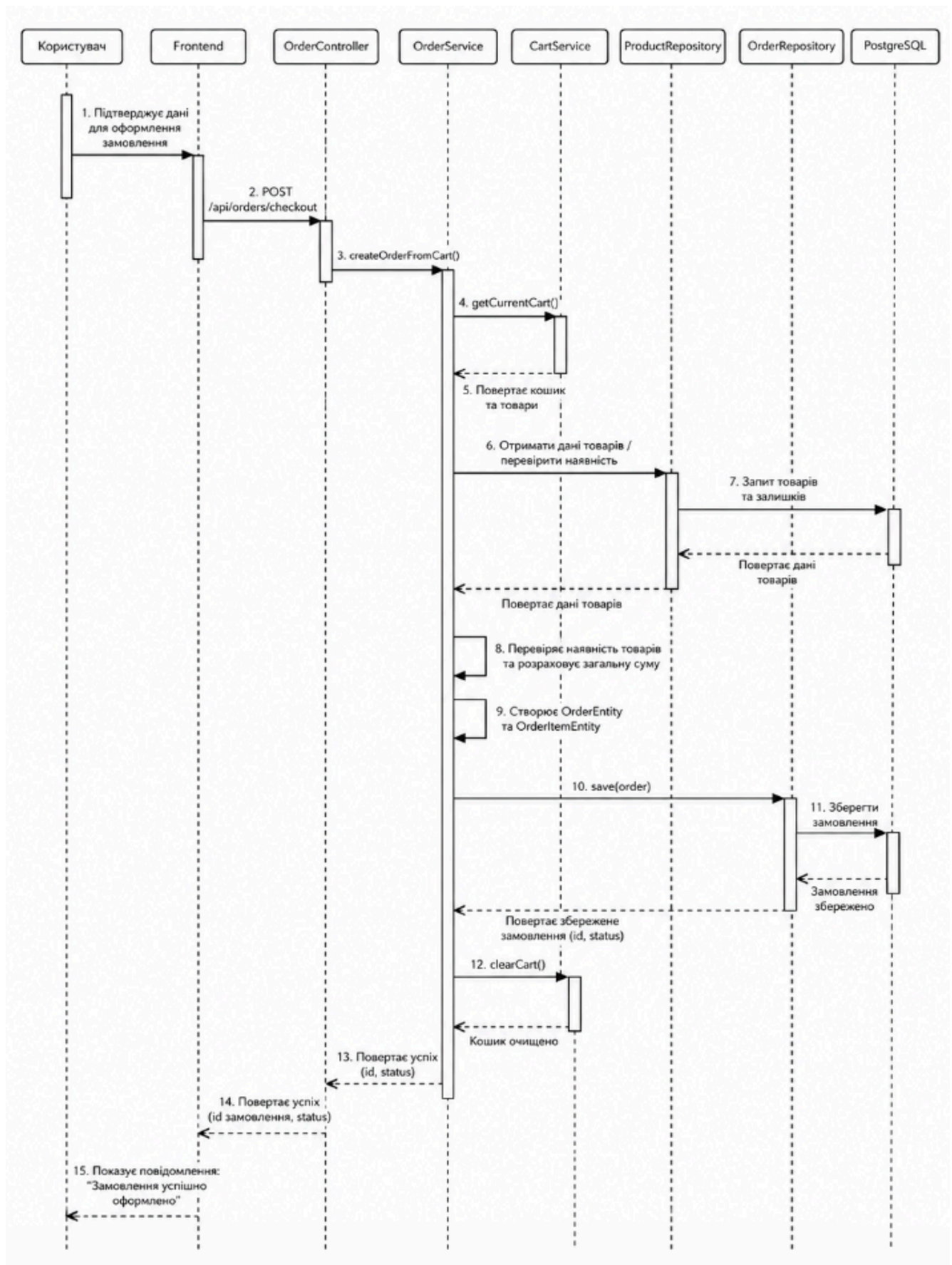


Рис. 3.5 Діаграма послідовності оформлення замовлення

Для отримання даних із backend використовується TanStack React Query. Він дозволяє завантажувати товари, бренди, кошик, замовлення, відгуки й рекомендації, а також оновлювати дані після виконання дій користувача. Наприклад, після додавання товару до кошика система може автоматично оновити кількість позицій у кошику [23].

Діаграма послідовності оформлення замовлення відображає взаємодію між користувачем, frontend, OrderController, OrderService, CartService, ProductRepository, OrderRepository і PostgreSQL. Спочатку користувач підтверджує дані для оформлення замовлення, після чого frontend надсилає POST-запит до серверної частини. Сервер перевіряє кошик, отримує дані товарів, перевіряє наявність, створює замовлення, зберігає його в базі даних, очищує кошик і повертає користувачу повідомлення про успішне оформлення [21; 23].

Адміністративний інтерфейс повинен бути відокремлений від звичайного користувацького. Адміністратор має доступ до списку товарів, створення нового товару, редагування характеристик, видалення товарів, керування брендами, перегляду замовлень, зміни статусів і перегляду користувачів. Доступ до цих сторінок повинен перевірятися через роль користувача [19; 21; 23; 26].

Отже, проектування web-застосунку AromaShop передбачає створення цілісної клієнт-серверної системи з чіткою архітектурою, структурованою базою даних, розмежуванням ролей, модульною серверною частиною та зручним frontend-інтерфейсом [1; 2; 3].

Особливу роль у системі відіграє функція «Підбір за подією», яка розширює стандартну логіку інтернет-магазину та робить процес вибору парфумерії більш персоналізованим [4; 6; 23].

4 РЕАЛІЗАЦІЯ ТА ТЕСТУВАННЯ WEB-ЗАСТОСУНКУ AROMASHOP

Реалізація web-застосунку AromaShop виконувалася відповідно до спроектованої клієнт-серверної архітектури. Основна увага під час розробки була зосереджена на створенні функціонального онлайн-магазину парфумерної продукції, який дозволяє користувачу переглядати каталог ароматів, застосовувати пошук і фільтрацію, відкривати сторінку товару, працювати з кошиком, оформлювати замовлення, залишати відгуки та використовувати функцію «Підбір за подією». Для адміністратора було реалізовано окремі можливості керування товарами, брендами, користувачами та замовленнями [18; 19; 21; 23; 26].

Серверна частина web-застосунку була реалізована на основі Java 21 та Spring Boot. Вона відповідає за обробку запитів, виконання бізнес-логіки, роботу з базою даних PostgreSQL, перевірку прав доступу користувачів і формування відповідей для клієнтської частини. Клієнтська частина була реалізована з використанням React, TypeScript і Vite. Вона забезпечує відображення сторінок системи, взаємодію користувача з елементами інтерфейсу та передавання запитів до backend через REST API [10; 18; 19; 21; 23; 24].

Реалізація системи виконувалася поетапно. Спочатку було створено базову структуру backend- і frontend-проектів, налаштовано підключення до бази даних, реалізовано сутності та репозиторії. Далі було розроблено модулі авторизації, каталогу товарів, кошика, замовлень, відгуків і адміністративного керування. Після цього було реалізовано клієнтські сторінки та сценарії взаємодії користувача із системою. Завершальним етапом стало тестування основних функцій web-застосунку [20; 21; 23].

4.1 Реалізація авторизації користувача та профілю

Реалізація авторизації користувача є одним із базових етапів створення web-застосунку AromaShop, оскільки частина функцій системи доступна лише

авторизованим користувачам. До таких функцій належать додавання товарів до кошика, оформлення замовлень, перегляд історії покупок, залишення відгуків, редагування профілю та доступ до адміністративної панелі [20].

У системі передбачено три основні ролі користувачів: гість, авторизований користувач і адміністратор. Гість може переглядати каталог товарів, шукати та фільтрувати продукцію, відкривати сторінки товарів, переглядати відгуки та користуватися функцією «Підбір за подією». Авторизований користувач має всі можливості гостя, а також може додавати товари до кошика, оформлювати замовлення, переглядати історію покупок, залишати оцінки та відгуки. Адміністратор має розширені права доступу та може керувати товарами, брендами, користувачами й замовленнями [18; 19; 21; 23; 26].

Для реалізації механізму безпеки було використано Spring Security та JWT. Під час реєстрації користувач вводить email, username, пароль та інші необхідні дані. Після відправлення форми backend перевіряє коректність введених даних, зокрема унікальність email, довжину пароля та наявність обов'язкових полів. Якщо дані є коректними, створюється новий запис у таблиці users, а пароль зберігається у захешованому вигляді [20].

Під час входу в систему користувач вводить email і пароль. Backend перевіряє, чи існує користувач із таким email, порівнює введений пароль із захешованим значенням у базі даних і, у разі успішної перевірки, генерує JWT-токен. Цей токен передається клієнтській частині та використовується у подальших запитах до захищених endpoint-ів [20].

Основна логіка авторизації реалізована в сервісі AuthServiceImpl. У ньому передбачено методи registerUser(), loginUser() та generateToken(). Метод registerUser() відповідає за створення нового користувача, loginUser() — за перевірку даних входу, а generateToken() — за формування токена доступу. На рівні контролера AuthController реалізовано endpoint-и для реєстрації та авторизації користувача [20].

AROMASHOP Search fragrances Search CATALOG OCCASIONS CART ORDERS Log in Sign up

WELCOME BACK Access your cart, orders, and wishlist.

Log in

Email
anna@aromashop.local

Password

Log in

Don't have an account? Create one

AROMASHOP Колекція ароматів і турбота про ваш досвід покупки.

ПРО МАГАЗИН Про нас Контакти

КАТАЛОГ Жіночі аромати Чоловічі аромати

СЕРВІС Поширені запитання Відгуки

Рис. 4.1 Екранна форма авторизації користувача в системі AromaShop

Після входу в систему користувач отримує доступ до особистого профілю. У профілі відображаються основні персональні дані: email, username, повне ім'я, телефон, адреса доставки та роль користувача. Також передбачено можливість редагування окремих полів профілю. Це дозволяє користувачу оновлювати контактні дані без створення нового акаунта [18; 19; 21; 23; 26].

Редагування профілю виконується через відповідний endpoint у UserController. Клієнтська частина надсилає оновлені дані, після чого backend перевіряє права доступу й оновлює запис у базі даних. Для звичайного користувача доступне редагування лише власного профілю, тоді як адміністратор може переглядати список користувачів і керувати ними через адміністративну панель [18; 19; 21; 23; 26].

AROMASHOP Search fragrances Search CATALOG OCCASIONS CART ORDERS PROFILE Anna Log out

YOUR ACCOUNT Manage your preferences and view past orders.

Profile

Account details

Email anna@aromashop.local

Username Anna

Profile details

Full name
Anna

Gender
Female

Phone
+380958738849

Address

Save changes

Рис. 4.2 Екранна форма профілю користувача web-застосунку AromaShop

Окрему увагу було приділено розмежуванню доступу. Наприклад, endpoint-и для перегляду каталогу є відкритими, оскільки ними може користуватися гість. Натомість endpoint-и для оформлення замовлення, перегляду історії покупок або створення відгуку потребують авторизації. Endpoint-и для створення, редагування та видалення товарів доступні лише користувачам із роллю адміністратора [20].

Таким чином, реалізований механізм авторизації забезпечує базовий рівень безпеки web-застосунку, дозволяє розмежовувати права доступу та створює основу для персоналізованої взаємодії користувача із системою [20].

4.2 Реалізація клієнтського web-інтерфейсу

Клієнтський web-інтерфейс AromaShop було реалізовано з використанням React, TypeScript і Vite. Основною метою frontend-частини є забезпечення зручної взаємодії користувача з функціями онлайн-магазину. Інтерфейс повинен бути зрозумілим, логічним і достатньо простим, щоб користувач міг швидко перейти від перегляду каталогу до оформлення замовлення [23; 24; 25].

Структура клієнтської частини побудована за компонентним підходом. Окремі сторінки та елементи інтерфейсу реалізовано у вигляді незалежних компонентів. Це дозволяє повторно використовувати окремі блоки, спрощує супровід коду та робить frontend більш гнучким [10; 23].

До основних сторінок клієнтської частини належать:

- головна сторінка;
- каталог товарів;
- сторінка детального перегляду товару;
- сторінка підбору аромату за подією;
- сторінка авторизації;
- сторінка реєстрації;
- кошик;
- оформлення замовлення;

- історія замовлень;
- деталі замовлення;
- профіль користувача;
- адміністративна панель товарів;
- адміністративна панель брендів;
- адміністративна панель замовлень;
- адміністративна панель користувачів.

Для маршрутизації між сторінками було використано React Router. Кожна сторінка має окремий маршрут. Наприклад, каталог доступний за маршрутом /catalog, сторінка товару — за /products/:slug, кошик — за /cart, оформлення замовлення — за /checkout, профіль — за /profile, а адміністративні сторінки — за маршрутами, які починаються з /admin [23].

Головна сторінка виконує роль початкової точки взаємодії користувача із системою. На ній можуть бути розміщені коротка інформація про магазин, посилання на каталог, блоки популярних товарів, акційні пропозиції та перехід до функції «Підбір за подією». Така структура дозволяє користувачу швидко зорієнтуватися та перейти до потрібного розділу [18; 19; 21; 23; 26].

Каталог товарів є однією з основних сторінок системи. Він відображає список доступної парфумерної продукції у вигляді товарних карток. Кожна картка містить зображення товару, назву, бренд, ціну, об'єм, короткий опис, рейтинг і кнопку переходу до сторінки товару. За потреби також може відображатися акційна ціна або статус наявності [18; 19; 21; 23; 26].

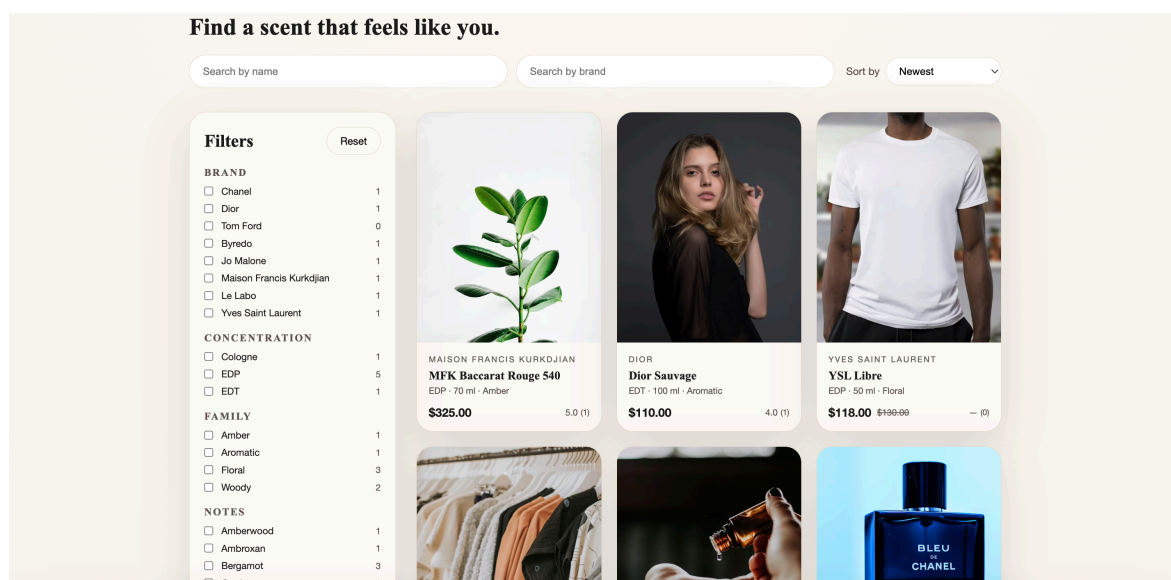


Рис. 4.3 Екранна форма каталогу товарів web-застосунку AromaShop

Для зручності користувача в каталозі реалізовано пошук і фільтрацію. Пошук дозволяє знайти товар за назвою або брендом. Фільтрація дає змогу обмежити список товарів за брендом, ароматичним сімейством, концентрацією, об'ємом, ціною або наявністю. Такий підхід є важливим для парфумерного магазину, оскільки асортимент може містити велику кількість товарів із різними характеристиками [18; 19; 21; 23; 26].

Окрім базових фільтрів, у системі важливо забезпечити логічне поєднання параметрів товару між собою. Наприклад, користувач може одночасно обрати певний бренд, ароматичне сімейство, концентрацію та ціновий діапазон, після чого система повинна відобразити лише ті позиції, які відповідають усім заданим умовам. Це зменшує кількість зайвих результатів у каталозі та скорочує час вибору потрібного аромату. Такий підхід також підвищує зручність роботи з великим асортиментом, оскільки користувач отримує більш точну добірку товарів без необхідності вручну переглядати весь каталог.

Сторінка товару містить розширену інформацію про аромат. На ній відображаються назва, бренд, повний опис, ціна, акційна ціна, об'єм, концентрація, ароматичне сімейство, кількість на складі, зображення, парфумерні ноти, рейтинг і відгуки користувачів. Також передбачено кнопку додавання товару до кошика [18; 19; 21; 23; 26].

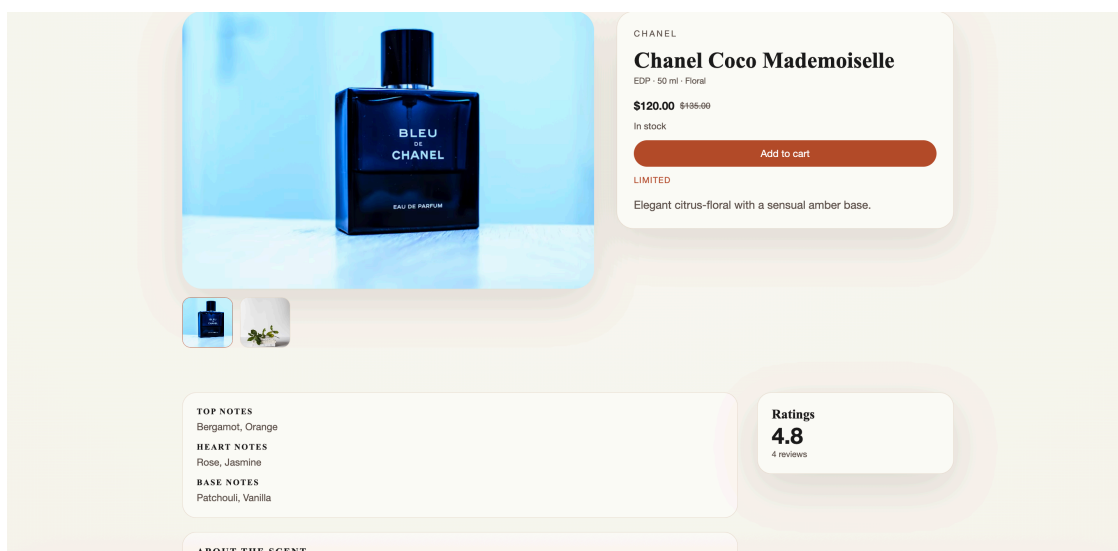


Рис. 4.4 Екранна форма сторінки товару з описом, нотами та відгуками

Окремо було реалізовано сторінку кошика. На цій сторінці користувач може переглянути додані товари, змінити їх кількість, видалити позиції та побачити загальну суму замовлення. Якщо кошик порожній, система відображає відповідне повідомлення та пропонує перейти до каталогу [18; 19; 21; 23; 26].

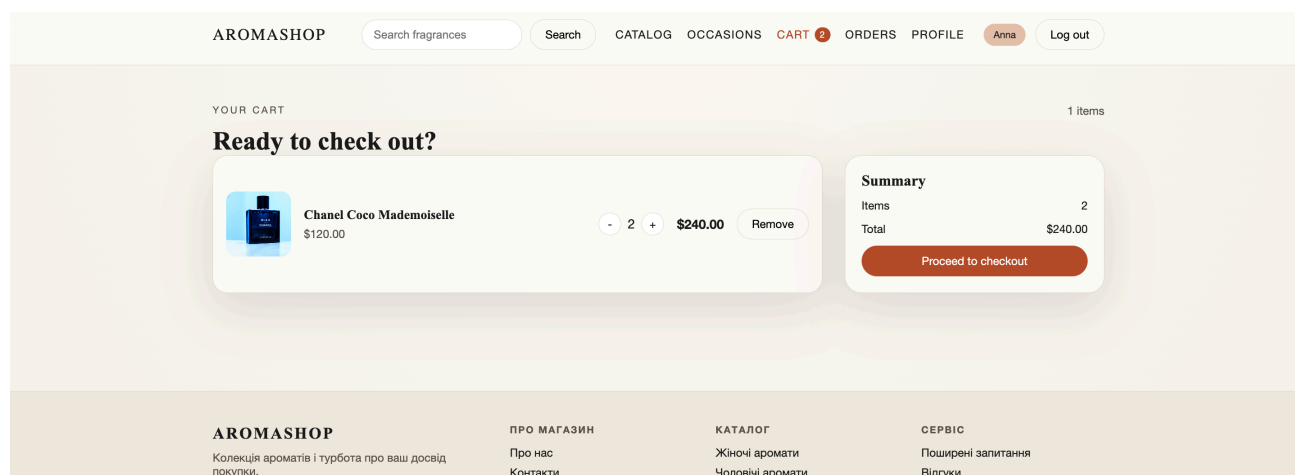


Рис. 4.5 Екранна форма кошика користувача

Сторінка оформлення замовлення містить форму для введення контактних даних і адреси доставки. Частина даних може автоматично підставлятися з профілю користувача, якщо вони вже збережені в системі. Після підтвердження замовлення frontend надсилає запит до backend, де створюється запис про замовлення [23].

AROMASHOP Search fragrances Search CATALOG OCCASIONS CART 2 ORDERS PROFILE Anna Log out

CHECKOUT 1 items

Confirm your order

Full name Phone Email

Shipping address

Place order

Order summary

Chanel Coco Mademoiselle	\$240.00
Total	\$240.00

AROMASHOP ПРО МАГАЗИН КАТАЛОГ СЕРВІС

Рис. 4.6 Екранна форма оформлення замовлення

Історія замовлень дозволяє користувачу переглядати попередні покупки. Для кожного замовлення відображаються дата створення, статус, загальна сума та можливість перейти до деталей. У деталях замовлення можна переглянути склад покупки, кількість товарів, ціну кожної позиції та контактні дані, вказані під час оформлення [18; 19; 21; 23; 26].

Таким чином, клієнтський web-інтерфейс реалізує основні сценарії роботи користувача з онлайн-магазином та забезпечує зв'язок із серверною частиною через REST API [26].

4.3 Реалізація серверної частини для роботи з товарами, брендами та користувачами

Серверна частина web-застосунку AromaShop реалізована на основі Spring Boot і має багаторівневу структуру. Основні рівні включають контролери, сервіси, репозиторії, доменні сутності та DTO. Така структура дозволяє розділити відповідальність між різними частинами backend і зробити код більш зрозумілим та зручним для супроводу [19].

Модуль товарів є одним із центральних модулів системи. Він забезпечує створення, редагування, видалення, перегляд і пошук товарів. Основною доменною сутністю цього модуля є ProductEntity. Вона містить поля id, name, slug, description,

shortDescription, brand, price, salePrice, stockQuantity, family, concentration, volumeMl, eanUpc, gs1Compliant, ifraCompliant, ifraNotice, images та notes [16; 17].

Для відображення товарів у каталозі використовується скорочена інформація: назва, бренд, ціна, об'єм, зображення, рейтинг і наявність. Для сторінки детального перегляду товару backend повертає розширені дані, включаючи опис, парфумерні ноти, зображення, відгуки та додаткові характеристики [18; 19; 21; 23; 26].

Пошук і фільтрація товарів реалізуються за параметрами, які надходять із клієнтської частини. До таких параметрів можуть належати назва товару, бренд, мінімальна та максимальна ціна, ароматичне сімейство, концентрація, об'єм і наявність. Backend обробляє ці параметри та повертає лише ті товари, які відповідають критеріям [18; 19; 21; 23; 26].

Модуль брендів реалізовано на основі сутності BrandEntity. Бренд містить id, name, description та logoUrl. Один бренд може бути пов'язаний із багатьма товарами. BrandServiceImpl забезпечує створення, редагування, видалення та отримання списку брендів. Цей модуль використовується як у публічній частині каталогу, так і в адміністративній панелі [18; 19; 21; 23; 26].

Модуль користувачів реалізовано через UserEntity та UserServiceImpl. UserEntity містить email, username, password, role, fullName, gender, phone, address і createdAt. Користувачі використовуються в модулях авторизації, профілю, кошика, замовлень і відгуків. Адміністратор має можливість переглядати список користувачів, створювати, оновлювати та видаляти записи [20].

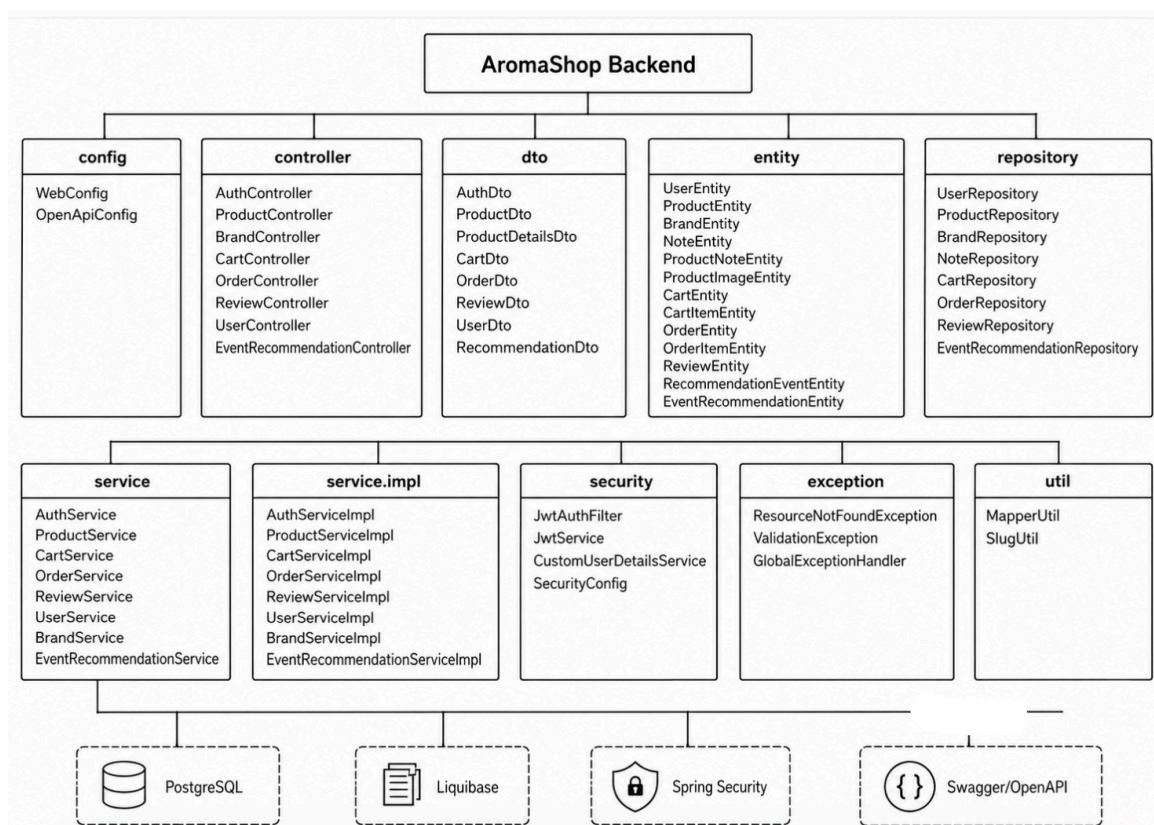


Рис. 4.8 Структура backend-проекту web-застосунку AromaShop

Для взаємодії клієнтської частини із серверною реалізовано REST API. Наприклад, для роботи з товарами можуть використовуватися такі endpoint-и: [26]

- GET /api/products — отримання списку товарів;
- GET /api/products/{id} — отримання товару за ідентифікатором;
- GET /api/products/slug/{slug} — отримання товару за slug;
- GET /api/products/search — пошук і фільтрація товарів;
- POST /api/products — створення товару адміністратором;
- PUT /api/products/{id} — редагування товару;
- DELETE /api/products/{id} — видалення товару.

Для роботи з брендами використовуються endpoint-и /api/brands, а для користувачів — /api/users. Доступ до адміністративних операцій обмежується за роллю користувача [18; 19; 21; 23; 26].

4.4 Реалізація кошика, замовлень, відгуків та історії покупок

Кошик є важливим функціональним модулем web-застосунку AromaShop, оскільки він забезпечує проміжний етап між вибором товару та оформленням замовлення. Кошик пов'язаний із конкретним авторизованим користувачем і містить список товарів, які користувач планує придбати [18; 19; 21; 23; 26].

Для реалізації кошика використовуються сутності `CartEntity` та `CartItemEntity`. `CartEntity` містить ідентифікатор, посилання на користувача, дату створення та список позицій. `CartItemEntity` містить посилання на кошик, товар і кількість. Така структура дозволяє одному користувачу мати власний кошик із декількома товарами [18; 19; 21; 23; 26].

`CartServiceImpl` містить методи `findByUserId()`, `addItemToCart()`, `updateCartItem()`, `removeItemFromCart()` та `checkout()`. Метод `addItemToCart()` додає товар до кошика. Якщо товар уже є в кошику, система може збільшити його кількість. Метод `updateCartItem()` дозволяє змінити кількість товару, а `removeItemFromCart()` — видалити позицію [18; 19; 21; 23; 26].

Замовлення створюється на основі кошика. Під час оформлення замовлення користувач вводить контактні дані, адресу доставки та підтверджує покупку. Backend отримує поточний кошик користувача, перевіряє наявність товарів, розраховує загальну суму, створює `OrderEntity` та набір `OrderItemEntity` [18; 19; 21; 23; 26].

`OrderEntity` містить `id`, `user`, `totalAmount`, `status`, `shippingAddress`, `fullName`, `phone`, `email`, `createdAt` та `items`. `OrderItemEntity` містить `order`, `product`, `quantity` та `unitPrice`. Збереження `unitPrice` у позиції замовлення є важливим, оскільки ціна товару може змінитися після оформлення замовлення, але історія покупки повинна зберігати фактичну ціну на момент придбання [18; 19; 21; 23; 26].

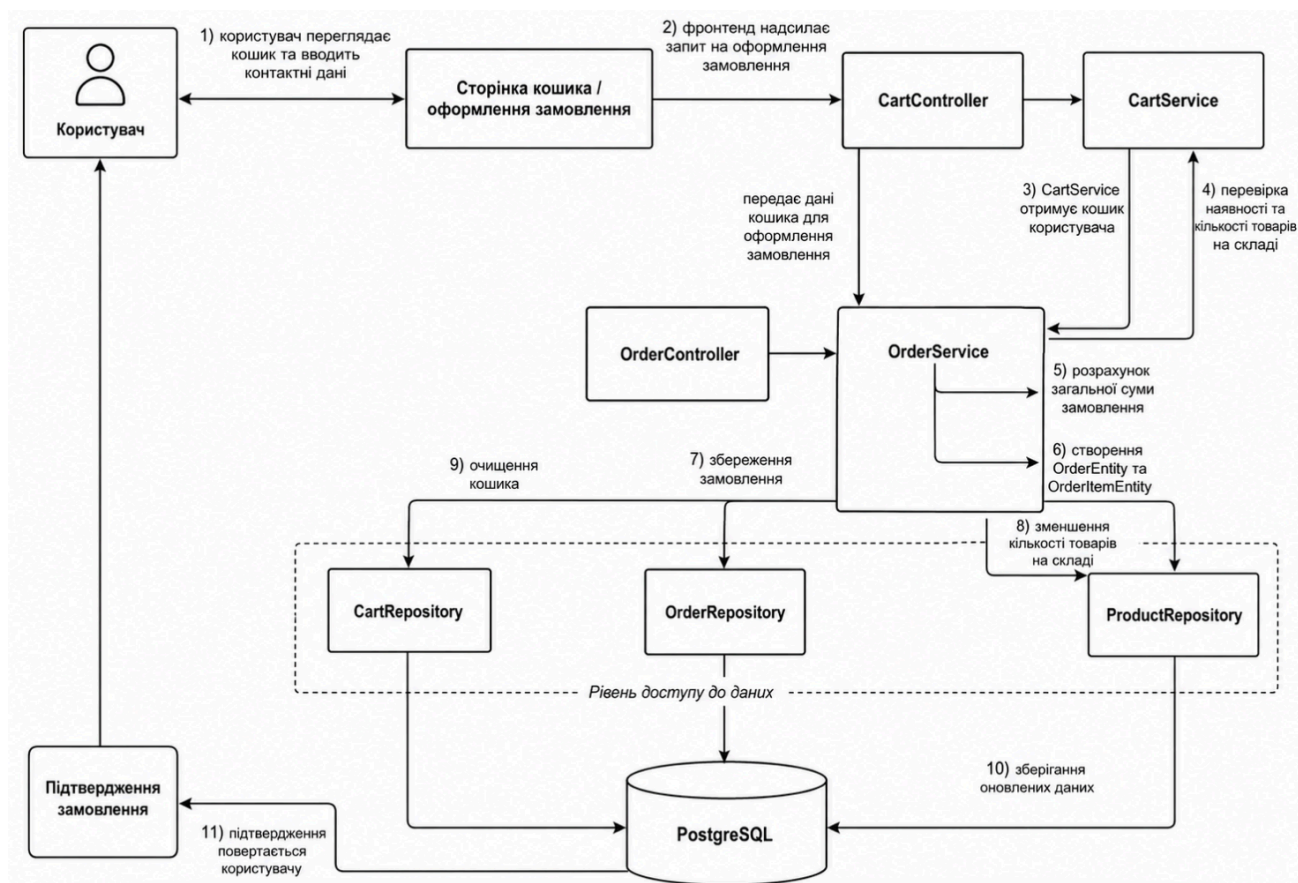


Рис. 4.10 Схема реалізації основної бізнес-логіки оформлення замовлення

Історія замовлень дозволяє користувачу переглядати всі власні покупки. Для цього frontend надсилає запит до backend, а backend повертає список замовлень, пов'язаних із поточним користувачем. При відкритті деталей замовлення система перевіряє, чи належить це замовлення саме цьому користувачу. Це запобігає перегляду чужих замовлень [23].

Модуль відгуків дозволяє користувачам залишати оцінки та коментарі до товарів. Для цього використовується ReviewEntity, яка містить id, user, product, rating, comment та createdAt. Відгук пов'язаний одночасно з користувачем і товаром [18; 19; 21; 23; 26].

ReviewServiceImpl містить методи getReviewsByProductId(), createReview() та deleteReview(). Метод getReviewsByProductId() використовується для відображення відгуків на сторінці товару. Метод createReview() дозволяє авторизованому користувачу залишити оцінку та коментар. Метод deleteReview()

може використовуватися для видалення відгуку адміністратором або самим автором, якщо така логіка передбачена [18; 19; 21; 23; 26].

Відгуки є важливим елементом саме для парфумерного магазину, оскільки вибір аромату є суб'єктивним. Коментарі інших користувачів можуть допомогти оцінити стійкість, шлейф, характер аромату та його відповідність опису. Тому модуль відгуків підвищує інформативність сторінки товару й допомагає користувачу прийняти рішення про покупку [18; 19; 21; 23; 26].

4.5 Реалізація функції «Підбір за подією» та адміністративної панелі

Функція «Підбір за подією» є однією з основних особливостей web-застосунку AromaShop. Вона призначена для того, щоб допомогти користувачу обрати аромат не за конкретною назвою чи брендом, а за ситуацією використання. Такий підхід є корисним для користувачів, які не мають чіткого уявлення про конкретний товар, але знають, для чого їм потрібен аромат [18; 19; 21; 23; 26].

На клієнтській частині реалізується окрема сторінка підбору. На ній користувач спочатку обирає подію: побачення, офіс, подарунок, кожен день або вечір. Після вибору події система відображає додаткові питання. Наприклад, для подарунка користувач може обрати отримувача, бюджет і бажаний стиль. Для офісу можуть уточнюватися легкість і ненав'язливість. Для вечора можуть враховуватися сезон, насиченість і бажаний характер аромату [18; 19; 21; 23; 26].

Після введення відповідей frontend формує запит до backend. Серверна частина обробляє вибрану подію та параметри, після чого звертається до бази даних і знаходить відповідні товари. При цьому враховуються характеристики аромату, наявність на складі, стиль, сезон, отримувач і додаткові параметри рекомендації [21; 23].

Для реалізації цієї функції використовуються таблиці `recommendation_events` та `event_recommendations`. Таблиця `recommendation_events` містить перелік доступних подій. Таблиця `event_recommendations` пов'язує подію з конкретними

товарами та містить пояснення, чому аромат підходить до обраного сценарію [18; 19; 21; 23; 26].

Результати підбору відображаються у вигляді списку товарів. Кожен товар містить назву, бренд, ціну, зображення, короткий опис і пояснення причини рекомендації. Наприклад, система може вказати, що аромат підходить для офісу через легкий характер і ненав'язливе звучання, або для вечора через насичені деревні чи східні ноти [18; 19; 21; 23; 26].

OCCASION FINDER

Зрозумілі, популярні та безпечні варіанти для різних адресатів.

Підбір аромату за подією

На побачення

Романтичні, теплі або виразні аромати для особистої зустрічі.

В офіс

Стримані аромати, які не переважають простір.

На подарунок

Зрозумілі, популярні та безпечні варіанти для різних адресатів.

На кожен день

Універсальні аромати для регулярного використання.

На вечір

Більш насичені аромати для вечірнього виходу та сезону.

Кому: Для чоловіка

Стиль: Солодкий

Бюджет від: 100

Бюджет до: 350

Підбрати аромат

Рекомендації

4 варіантів

MAISON FRANCIS KURKDJIAN
MFK Baccarat Rouge 540
EDP - 70 ml - Amber

Підбрано як подарунок: це зрозумілий і універсальний варіант з наявного каталогу.
Адресат: Для чоловіка. Стиль: Солодкий.

На подарунок Для чоловіка Солодкий

Amber EDP Рейтинг: 5.0 В наявності

\$325.00 [Деталі](#)

DIOR
Dior Sauvage
EDT - 100 ml - Aromatic

Підбрано як подарунок: це зрозумілий і універсальний варіант з наявного каталогу.
Адресат: Для чоловіка. Стиль: Солодкий.

На подарунок Для чоловіка Солодкий

Aromatic EDT Рейтинг: 4.0 В наявності

\$110.00 [Деталі](#)

Рис. 4.12 Екранна форма результатів підбору ароматів за подією

Адміністративна панель реалізована для керування основними даними системи. Вона доступна лише користувачам із роллю адміністратора. Через адміністративний інтерфейс можна керувати товарами, брендами, замовленнями та користувачами [18; 19; 21; 23; 26].

У панелі керування товарами адміністратор може створювати нові товари, редагувати наявні, змінювати ціну, опис, кількість на складі, бренд, ароматичне сімейство, концентрацію, об'єм, відповідність IFRA та GS1, а також видаляти товари. Це дозволяє підтримувати актуальний стан каталогу [16; 17].

Admin
PRODUCTS
BRANDS
ORDERS
USERS

Brands

Name: Logo URL:

Description:

Create brand

NAME	DESCRIPTION	ACTIONS
Chanel	Luxury French fashion house	Edit Delete
Dior	French luxury goods company	Edit Delete
Tom Ford	Modern couture and fragrance house	Edit Delete
Byredo	Swedish luxury fragrance house	Edit Delete
Jo Malone	British fragrance and lifestyle brand	Edit Delete
Maison Francis Kurkdjian	Parisian fragrance atelier	Edit Delete

Рис. 4.13 Екранна форма адміністративної панелі брендів

У панелі керування брендами адміністратор може додавати нові бренди, редагувати опис і логотип, а також видаляти записи, якщо вони не використовуються в товарах. Бренди є важливою частиною структури каталогу, оскільки користувачі часто шукають аромати саме за брендом [18; 19; 21; 23; 26].

У панелі керування замовленнями адміністратор може переглядати всі замовлення, відкривати їх деталі та змінювати статус. Наприклад, замовлення може мати статус нове, в обробці, відправлене, виконане або скасоване. Зміна статусу дозволяє користувачу відстежувати процес виконання покупки [18; 19; 21; 23; 26].

У панелі керування користувачами адміністратор може переглядати список зареєстрованих користувачів, їхні ролі, контактні дані та за потреби оновлювати або видаляти записи. Така функція потрібна для базового адміністрування системи [18; 19; 21; 23; 26].

Таким чином, функція «Підбір за подією» розширює стандартну логіку онлайн-магазину, а адміністративна панель забезпечує повноцінне керування даними системи.

4.6 Тестування основних функцій web-застосунку

Після реалізації основних модулів web-застосунку AromaShop було проведено тестування його функціональності. Метою тестування була перевірка відповідності системи поставленим вимогам, виявлення можливих помилок і підтвердження коректності роботи основних сценаріїв користувача та адміністратора [18; 19; 21; 23; 26].

Тестування проводилося для таких функціональних блоків:

- реєстрація та авторизація;
- перегляд каталогу товарів;
- пошук і фільтрація;
- перегляд сторінки товару;
- додавання товару до кошика;
- зміна кількості товарів у кошику;
- видалення товару з кошика;
- оформлення замовлення;
- перегляд історії замовлень;
- перегляд деталей замовлення;
- створення відгуку;
- підбір аромату за подією;
- адміністративне керування товарами;
- адміністративне керування брендами;
- адміністративне керування замовленнями;
- адміністративне керування користувачами.

Під час тестування авторизації перевірялося, чи може користувач успішно зареєструватися, увійти в систему, отримати JWT-токен і виконувати захищені запити. Також перевірялося, що користувач із роллю guest не має доступу до захищених дій, а користувач без ролі адміністратора не може створювати або редагувати товари [20].

Під час тестування каталогу перевірялося, чи відображаються товари, чи коректно працює пошук за назвою та брендом, чи застосовуються фільтри за ціною, брендом, сімейством, концентрацією та об'ємом. Також перевірялося, чи не відображаються некоректні або порожні результати без відповідного повідомлення [18; 19; 21; 23; 26].

Під час тестування сторінки товару перевірялося, чи відображаються всі необхідні характеристики: назва, бренд, опис, ціна, об'єм, концентрація, сімейство, ноти, зображення, рейтинг, відгуки та наявність. Також перевірялося, чи працює кнопка додавання до кошика [18; 19; 21; 23; 26].

Під час тестування кошика перевірялося додавання товару, повторне додавання того самого товару, зміна кількості, видалення позицій і перерахунок загальної суми. Окремо перевірялася ситуація, коли користувач намагається додати товар, якого немає в наявності [18; 19; 21; 23; 26].

Під час тестування оформлення замовлення перевірялося створення замовлення на основі кошика, збереження контактних даних, формування позицій замовлення, збереження ціни товару на момент покупки, очищення кошика після оформлення та оновлення залишків товарів [18; 19; 21; 23; 26].

Під час тестування відгуків перевірялося створення оцінки та коментаря, відображення відгуку на сторінці товару, зв'язок відгуку з користувачем і товаром, а також коректність відображення рейтингу [18; 19; 21; 23; 26].

Під час тестування функції «Підбір за подією» перевірялися всі основні сценарії: побачення, офіс, подарунок, кожен день і вечір. Для кожного сценарію перевірялося відображення уточнювальних питань, передавання параметрів до backend, отримання списку рекомендованих товарів і пояснення причини рекомендації [18; 19; 21; 23; 26].

Під час тестування адміністративної панелі перевірялося створення товару, редагування його характеристик, зміна кількості на складі, видалення товару, створення бренду, зміна статусу замовлення та перегляд списку користувачів [18; 19; 21; 23; 26].

Зведені результати тестування наведено в таблиці 4.1.

Таблиця 4.1

Результати тестування основних функцій web-застосунку AromaShop

№	Функція	Очікуваний результат	Фактичний результат	Статус
1	Реєстрація користувача	Новий користувач створюється в системі	Користувач успішно створюється	Виконано
2	Авторизація користувача	Користувач отримує JWT-токен	Токен успішно генерується	Виконано
3	Перегляд каталогу	Відображається список товарів	Каталог відкривається коректно	Виконано
4	Пошук товарів	Відображаються товари за запитом	Пошук працює за назвою та брендом	Виконано
5	Фільтрація товарів	Відображаються товари за параметрами	Фільтри застосовуються коректно	Виконано
6	Сторінка товару	Відображаються повні дані про аромат	Дані товару відображаються коректно	Виконано
7	Додавання до кошика	Товар додається до кошика	Позиція з'являється в кошику	Виконано
8	Зміна кількості в кошику	Кількість і сума оновлюються	Дані оновлюються коректно	Виконано
9	Оформлення замовлення	Створюється замовлення з товарами	Замовлення зберігається в базі даних	Виконано
10	Історія замовлень	Користувач бачить свої замовлення	Історія відображається коректно	Виконано
11	Залишення відгуку	Відгук додається до товару	Відгук відображається на сторінці товару	Виконано



Рис. 4.15 Схема тестування основних сценаріїв роботи web-застосунку

За результатами тестування було встановлено, що основні функції web-застосунку працюють відповідно до поставлених вимог. Користувач може пройти повний сценарій від перегляду каталогу до оформлення замовлення, а адміністратор може керувати ключовими даними системи. Функція «Підбір за подією» дозволяє отримати рекомендації товарів відповідно до обраної ситуації, що підвищує зручність вибору парфумерії [18; 19; 21; 23; 26].

Таким чином, реалізований web-застосунок AromaShop забезпечує основні можливості спеціалізованого онлайн-магазину парфумерії та може бути використаний як основа для подальшого розвитку системи. Надалі доцільно розширити функціональність за рахунок інтеграції онлайн-оплати, служб доставки, системи промокодів, списку обраних товарів, персоналізованих рекомендацій і розширеної аналітики продажів [18; 19; 21; 23; 26].

ВИСНОВКИ

У кваліфікаційній роботі було розглянуто задачу розробки web-застосунку для онлайн-продажу парфумерної продукції. У результаті виконання роботи було створено спеціалізовану систему AromaShop, яка поєднує стандартні можливості інтернет-магазину з предметно орієнтованими функціями для вибору ароматів [1; 2; 3; 4; 6].

На першому етапі було проведено аналіз предметної галузі онлайн-продажу парфумерії. Встановлено, що процес вибору ароматів в інтернет-магазині має низку особливостей, оскільки користувач не може фізично протестувати аромат перед покупкою. Через це важливими є повний опис товару, наявність парфумерної піраміди, характеристик аромату, відгуків, рейтингу, інформації про наявність і зручних засобів фільтрації. Також було визначено, що користувачі часто обирають аромат не лише за брендом або назвою, а за конкретною ситуацією використання, наприклад для офісу, побачення, подарунка, щоденного використання або вечірнього виходу [1; 2; 3; 4; 6].

У межах аналізу аналогів було розглянуто існуючі програмні рішення для онлайн-продажу парфумерної продукції, зокрема BROCARD, MAKEUP, ROZETKA та EVA.UA. Встановлено, що ці системи мають базові функції, але не забезпечують достатньо персоналізованого підбору ароматів за життєвою ситуацією, а також може мати перевантажений інтерфейс або надто широкий асортимент, не зосереджений саме на парфумерній продукції [1; 2; 3; 4; 6; 11].

Було сформовано функціональні та нефункціональні вимоги до web-застосунку AromaShop. До основних функціональних вимог належать реєстрація та авторизація користувачів, перегляд каталогу товарів, пошук і фільтрація, перегляд сторінки товару, робота з кошиком, оформлення замовлень, перегляд історії покупок, залишення відгуків, редагування профілю, адміністративне керування

товарами, брендами, користувачами та замовленнями, а також використання функції «Підбір за подією». Серед нефункціональних вимог було визначено швидкодію, адаптивність інтерфейсу, захист користувацьких даних, підтримку одночасної роботи користувачів, масштабованість і супроводжуваність системи [5; 10; 20].

Для реалізації web-застосунку було обґрунтовано вибір сучасного технологічного стеку. Серверну частину реалізовано з використанням Java 21, Spring Boot, Spring Web, Spring Security, JWT, Spring Data JPA та Hibernate. Для зберігання даних використано PostgreSQL, а для керування змінами структури бази даних — Liquibase. Клієнтську частину реалізовано за допомогою React, TypeScript, Vite, React Router і TanStack React Query. Для документування та тестування API використано Swagger/OpenAPI, а для контейнеризації — Docker і Docker Compose [18; 19; 20; 21; 22; 23].

Під час проектування було визначено загальну архітектуру системи, логічну структуру, модель предметної області та структуру бази даних. Також було розроблено ER-діаграму бази даних, діаграму варіантів використання, діаграму класів доменної моделі, діаграму послідовності оформлення замовлення та діаграму роботи функції «Підбір за подією» [21].

У процесі реалізації було створено основні модулі web-застосунку. Модуль авторизації забезпечує реєстрацію, вхід користувача в систему, генерацію JWT-токена та розмежування прав доступу. Модуль товарів відповідає за перегляд, пошук, фільтрацію, створення, редагування та видалення парфумерної продукції. Модуль кошика дозволяє користувачу додавати товари, змінювати їх кількість і видаляти позиції. Модуль замовлень забезпечує створення замовлення, збереження його складу, контактних даних і статусу. [20].

Особливістю розробленої системи є функція «Підбір за подією». Вона дозволяє користувачу вибрати ситуацію, для якої потрібен аромат, відповісти на

уточнювальні питання та отримати список відповідних товарів. Підбір виконується на основі внутрішньої бази даних товарів, їх характеристик, наявності на складі, стилю, сезону, типу отримувача та пояснення причини рекомендації. [21].

Було проведено тестування основних функцій web-застосунку AromaShop. Перевірено реєстрацію та авторизацію користувачів, перегляд каталогу, пошук, фільтрацію, сторінку товару, додавання товарів до кошика, зміну кількості товарів, оформлення замовлення, роботу функції «Підбір за подією» та адміністративне керування основними сутностями системи. За результатами тестування встановлено, що реалізовані функції працюють відповідно до поставлених вимог [20].

Таким чином, поставлена мета кваліфікаційної роботи була досягнута. У результаті виконання роботи було розроблено web-застосунок AromaShop, який забезпечує основні функції онлайн-продажу парфумерії, враховує специфіку предметної галузі та містить додатковий механізм підбору ароматів за подією [1; 2; 3; 4; 6].

Робота пройшла апробацію. За її результатами було опубліковано наступні тези доповідей:

1. Мельник В.О., Корецька В.О. Розробка онлайн-магазину парфумерії "AromaShop". Матеріали II Всеукраїнської науково-технічної конференції «Виклики та рішення в програмній інженерії», 26 листопада 2025 року, Київ, Державний університет інформаційно-комунікаційних технологій. Збірник тез. – К.: ДУІКТ, 2025, С.493-495.

2. Мельник В.О., Корецька В.О. Програмна система AromaShop для онлайн-продажу та підбору парфумерної продукції. Матеріали Всеукраїнської науково-технічної конференції «Технології цифрового розвитку: програмні системи та децентралізація», 1-3 травня 2026 року, Київ, ДУІКТ. Збірник тез. *(подано до друку)*.

ПЕРЕЛІК ПОСИЛАНЬ

1. Краус К. М., Краус Н. М., Манжура О. В. Електронна комерція та Інтернет-торгівля : навч.-метод. посіб. Київ : Аграр Медіа Груп, 2021. 454 с. URL: https://elibrary.kubg.edu.ua/id/eprint/37044/1/Kraus_Elektronna_komertsii_2021.pdf (дата звернення: 25.05.2026).
2. Карпенко М. Ю. Електронна комерція : конспект лекцій. Харків : ХНУМГ ім. О. М. Бекетова, 2021. 146 с. URL: <https://eprints.kname.edu.ua/58976/1/2020%20%D0%BF%D0%B5%D1%87%20129%D0%9B-073-%D0%9A%D0%BE%D0%BD%D1%81%D0%BF%D0%B5%D0%BA%D1%82%D0%95%D0%BB%D0%9A%D0%BE%D0%BC%D0%BC.pdf> (дата звернення: 25.05.2026).
3. Сак Т. В., Ховхалюк Д. О. Електронна торгівля в Україні: стан, тенденції, перспективи розвитку. Маркетинг і цифрові технології. 2020. Т. 4, № 3. С. 73–85. URL: <https://mdt-opu.com.ua/index.php/mdt/article/view/112> (дата звернення: 25.05.2026).
4. Рижкова Г. А. Особливості розвитку електронної торгівлі в Україні. Економічний вісник. 2024. № 4. С. 120–129. URL: https://ev.nmu.org.ua/docs/2024/4/EV20244_120-129.pdf (дата звернення: 25.05.2026).
5. Деякі аспекти правового регулювання електронної торгівлі. Юридичний науковий електронний журнал. 2024. № 6. С. 156–159. URL: https://www.lsej.org.ua/6_2024/35.pdf (дата звернення: 25.05.2026).
6. Шостак Л. В. Вплив електронної комерції на інновації бізнес-моделей в сучасних ринкових умовах. Економіка та суспільство. 2025. № 71. URL: <https://journals.kyiv.ua/index.php/economy/article/view/263> (дата звернення: 25.05.2026).

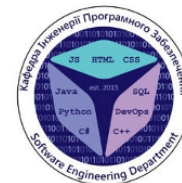
7. Дослідження українського ринку eCommerce 2024. Promodo. URL: <https://www.promodo.ua/ukrayinskiy-ecommerce-2024> (дата звернення: 25.05.2026).
8. Дослідження українського eCommerce: тренди та прогнози на 2024 рік. Promodo. URL: <https://www.promodo.ua/ukrayinskiy-ecommerce-2024-2> (дата звернення: 25.05.2026).
9. Дослідження та підсумки українського eCommerce за 1 півріччя 2025 року. Promodo. URL: <https://www.promodo.ua/ukrayinskiy-ecommerce-2025-1> (дата звернення: 25.05.2026).
10. 9 ключових правил UX в ecommerce. Brander. 2022. URL: <https://brander.ua/blog/9-klyuchovikh-pravil-ux-v-ecommerce> (дата звернення: 25.05.2026).
11. Інтернет-магазин BROCARD.UA. BROCARD. URL: <https://www.brocard.ua/ua/internet-store> (дата звернення: 25.05.2026).
12. BROCARD : офіційний інтернет-магазин парфумерії та косметики. URL: <https://www.brocard.ua/> (дата звернення: 25.05.2026).
13. MAKEUP : інтернет-магазин косметики та парфумерії. URL: <https://makeup.com.ua/ua/> (дата звернення: 25.05.2026).
14. Парфумерія в інтернет-магазині ROZETKA. URL: <https://rozetka.com.ua/ua/perfume/c2048522/> (дата звернення: 25.05.2026).
15. Парфумерія в інтернет-магазині EVA.UA. URL: <https://eva.ua/ua/217/parfjumerija/> (дата звернення: 25.05.2026).
16. IFRA Standards Library. International Fragrance Association. URL: <https://ifrafragrance.org/safe-use/library> (date of access: 25.05.2026).
17. GS1 General Specifications. GS1. URL: <https://www.gs1.org/standards/barcodes-epcrfid-id-keys/gs1-general-specifications> (date of access: 25.05.2026).
18. Java Platform, Standard Edition Documentation. Oracle. URL: <https://docs.oracle.com/en/java/javase/21/> (date of access: 25.05.2026).
19. Spring Boot Reference Documentation. VMware Tanzu. URL: <https://docs.spring.io/spring-boot/reference/index.html> (date of access: 25.05.2026).

20. Spring Security Reference. VMware Tanzu. URL: <https://docs.spring.io/spring-security/reference/> (date of access: 25.05.2026) [20].
21. PostgreSQL Documentation. The PostgreSQL Global Development Group. URL: <https://www.postgresql.org/docs/> (date of access: 25.05.2026).
22. Liquibase Documentation. Liquibase. URL: <https://docs.liquibase.com/> (date of access: 25.05.2026).
23. React Documentation. Meta Open Source. URL: <https://react.dev/> (date of access: 25.05.2026).
24. TypeScript Documentation. Microsoft. URL: <https://www.typescriptlang.org/docs/> (date of access: 25.05.2026).
25. Vite Guide. Vite. URL: <https://vite.dev/guide/> (date of access: 25.05.2026).
26. OpenAPI Specification. Swagger. URL: <https://swagger.io/specification/> (date of access: 25.05.2026).
27. Docker Compose Documentation. Docker Inc. URL: <https://docs.docker.com/compose/> (date of access: 25.05.2026) [27].

ДОДАТОК А. ДЕМОНСТРАЦІЙНІ МАТЕРІАЛИ



ДЕРЖАВНИЙ УНІВЕРСИТЕТ
ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНИХ ТЕХНОЛОГІЙ
НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ
КАФЕДРА ІНЖЕНЕРІЇ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ



Розробка онлайн-магазину парфумерії "AromaShop"

Виконав студент 4 курсу
групи ПД-44
Мельник Владислав Олександрович
Керівник роботи
канд.пед.наук, доц. Коренька Вікторія Олександрівна

Київ – 2026

ЗАДАЧІ КВАЛІФІКАЦІЙНОЇ РОБОТИ

1. Проаналізувати предметну галузь онлайн-продажу парфумерної продукції та визначити основні проблеми, пов'язані з пошуком, вибором і придбанням парфумерії через інтернет.
2. Дослідити існуючі програмні рішення для онлайн-продажу парфумерії, зокрема BROCARD, MAKEUP, ROZETKA та EVA.UA, визначити їх переваги й недоліки.
3. Сформувати функціональні та нефункціональні вимоги до web-застосунку «AromaShop».
4. Обґрунтувати вибір програмних засобів реалізації web-застосунку, зокрема Java, Spring Boot, PostgreSQL, React та TypeScript.
5. Спроектувати архітектуру клієнт-серверного web-застосунку для онлайн-магазину парфумерії.
6. Розробити структуру бази даних для зберігання інформації про користувачів, товари, бренди, категорії, кошики та замовлення.
7. Змоделювати основні бізнес-процеси та функціональність web-застосунку за допомогою UML-діаграм.
8. Реалізувати клієнтську частину web-застосунку з каталогом товарів, пошуком, фільтрацією, сторінкою товару, кошиком та оформленням замовлення.
9. Реалізувати серверну частину застосунку з обробкою запитів, бізнес-логікою, роботою з базою даних та адміністративним керуванням товарами і замовленнями.
10. Провести тестування основних сценаріїв роботи web-застосунку та перевірити відповідність реалізованого функціоналу поставленим вимогам.

МЕТА, ОБ'ЄКТ ТА ПРЕДМЕТ ДОСЛІДЖЕННЯ

Мета роботи: спрощення процесу вибору, пошуку та придбання парфумерної продукції за рахунок розробки web-застосунку AromaShop.

Об'єкт дослідження: процеси онлайн-продажу парфумерної продукції.

Предмет дослідження: засоби, методи та технології реалізації web-застосунку AromaShop для онлайн-продажу парфумерії.

АНАЛІЗ АНАЛОГІВ

<u>Показник</u>	<u>BROCARD</u>	<u>MAKEUP</u>	<u>ROZETKA</u>	<u>EVA.UA</u>	<u>AromaShop</u>
<u>Каталог парфумерії</u>	+	+	+	+	+
<u>Фільтрація за ціною/брендом/об'ємом</u>	+	+	+	+	+
<u>Поділ за ароматичним сімейством</u>	Є окремі групи ароматів	Групи ароматів подані не всюди	Залежить від продавця або категорії	Є базовий поділ за типами ароматів	Фільтр за сімейством: квіткові, деревні, цитрусові, східні тощо
<u>Відображення парфумерних нот</u>	Ноти подані не для всіх товарів	Ноти часто є в описі	-	Ноти подані вибірково	Ноти аромату винесені в окремий блок кожного товару
<u>Детальна сторінка товару</u>	+	+	+	+	+
<u>Відгуки та рейтинг</u>	+	+	+	+	+
<u>Оформлення замовлення</u>	+	+	+	+	+
<u>Підбір аромату за подією</u>	-	-	-	-	+
<u>Пояснення причини рекомендації</u>	-	-	-	-	+

ВИМОГИ ДО ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

Функціональні вимоги

1. Реєстрація та авторизація користувачів з урахуванням ролей.
2. Перегляд каталогу парфумерної продукції.
3. Пошук товарів за назвою та брендом.
4. Фільтрація товарів за брендом, ціною, об'ємом, концентрацією та ароматичним сімейством.
5. Перегляд детальної сторінки товару з описом, зображеннями, нотами, ціною, наявністю та відгуками.
6. Додавання товарів до кошика, зміна кількості та видалення позицій.
7. Оформлення замовлення із зазначенням контактних даних і адреси доставки.
8. Перегляд історії замовлень та деталей конкретного замовлення.
9. Залишення відгуків і оцінювання товарів авторизованими користувачами.
10. Редагування профілю користувача.
11. Адміністративне керування товарами, брендами, користувачами та замовленнями.
12. Формування рекомендацій ароматів із поясненням причини підбору.

Нефункціональні вимоги

1. Час завантаження сторінок — не більше 3 секунд.
2. Час виконання пошуку та фільтрації товарів — до 2 секунд.
3. Оформлення замовлення — не більше 4–5 основних кроків.
4. Підтримка одночасної роботи не менше 100 користувачів.
5. Адаптивність інтерфейсу для екранів від 360 px до 1920 px.
6. Зберігання паролів користувачів лише у захешованому вигляді.

5

ПРОГРАМНІ ЗАСОБИ РЕАЛІЗАЦІЇ

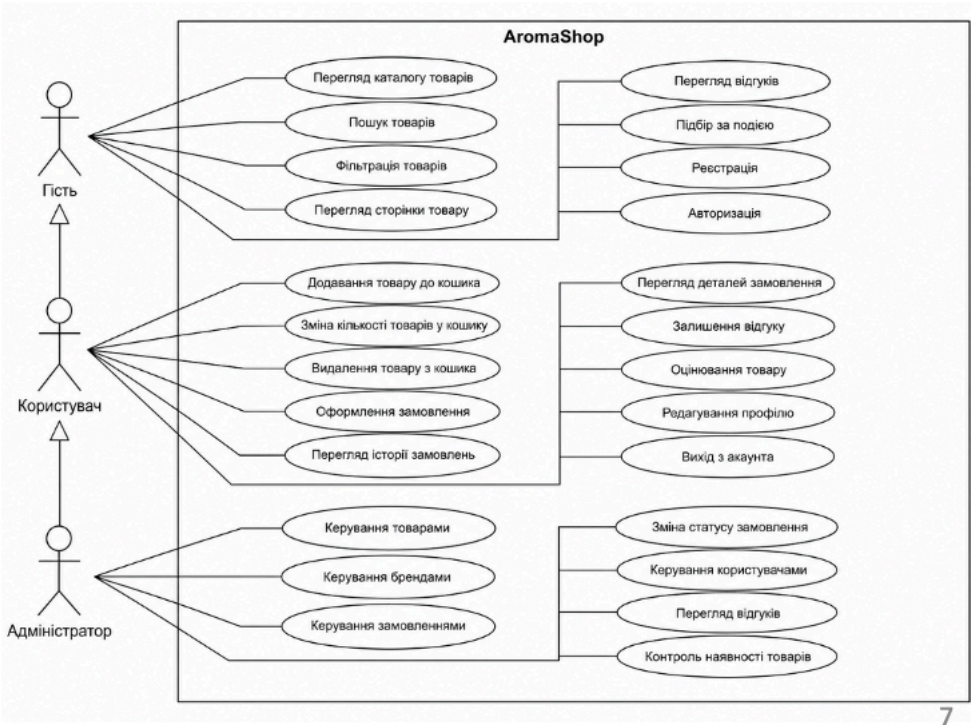


PostgreSQL

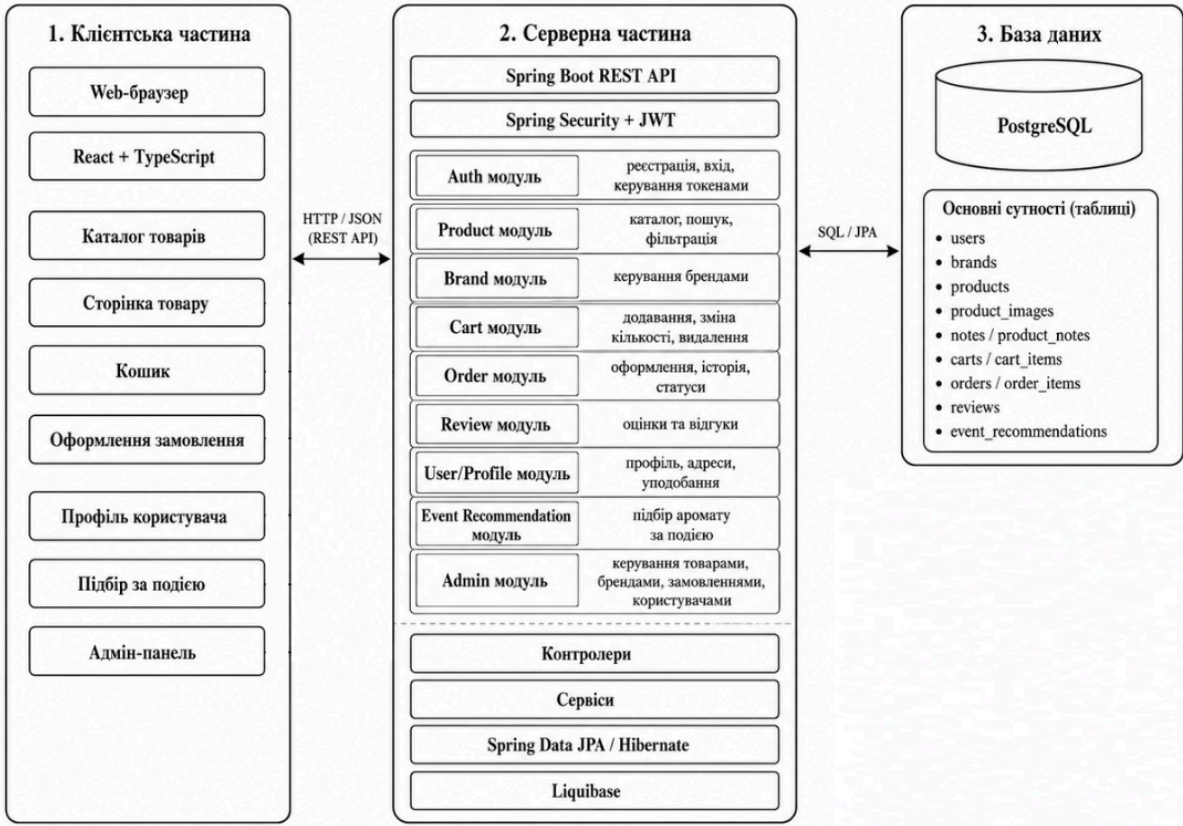


6

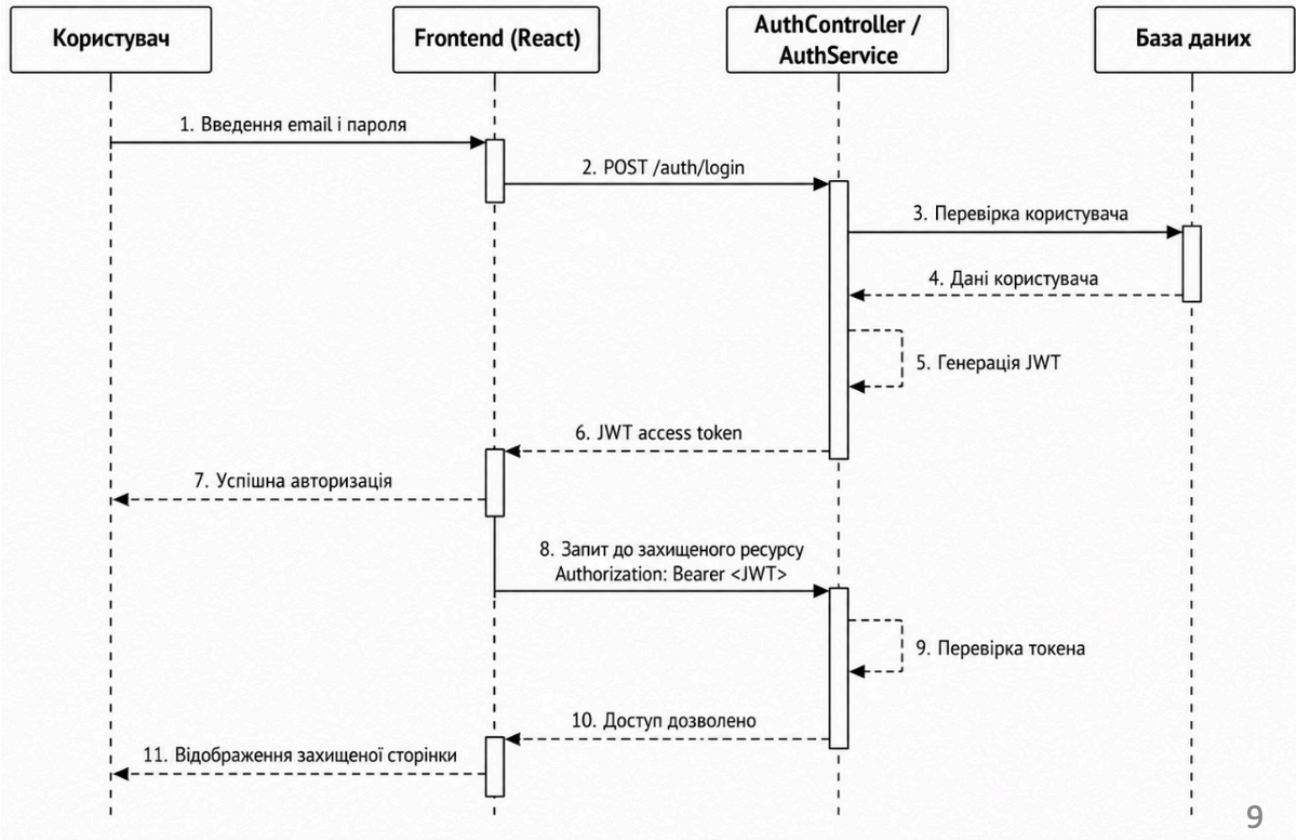
ДІАГРАМА ВАРІАНТІВ ВИКОРИСТАННЯ



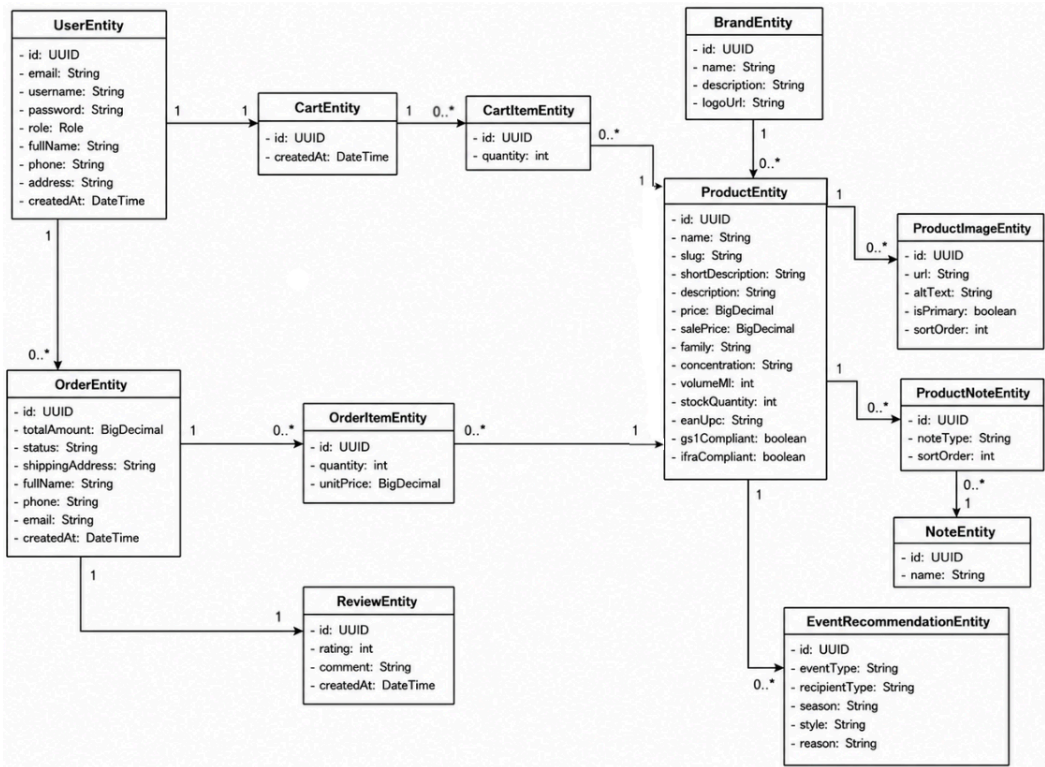
АРХІТЕКТУРА WEB-ЗАСТОСУНКУ AROMASHOP



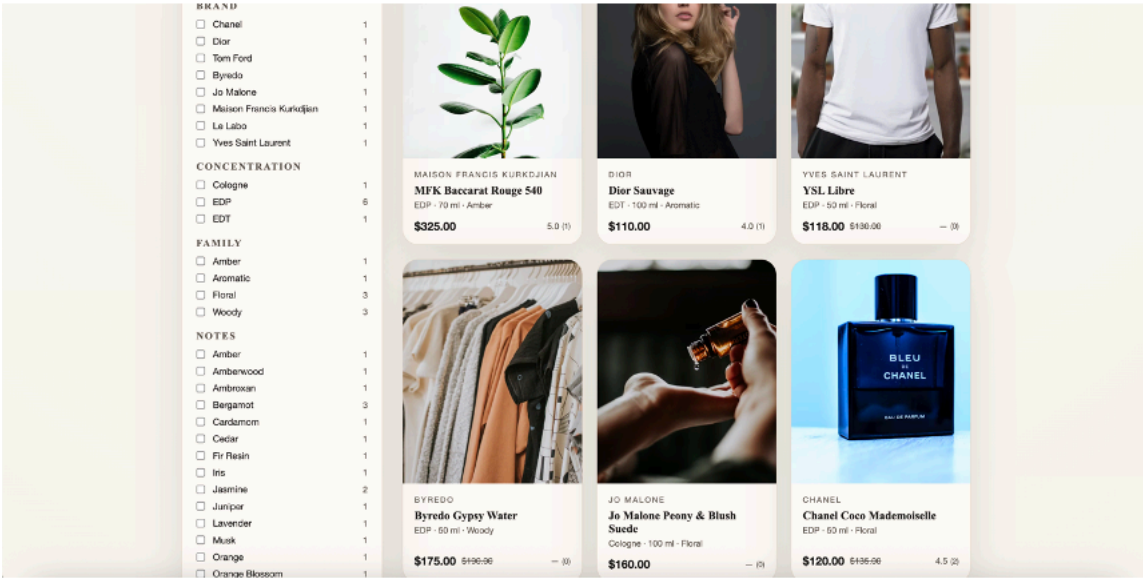
ДІАГРАМА ПОСЛІДОВНОСТІ АВТОРИЗАЦІЇ КОРИСТУВАЧА
ЗА ДОПОМОГОЮ JWT



ДІАГРАМА КЛАСІВ ДОМЕННІЙ МОДЕЛІ



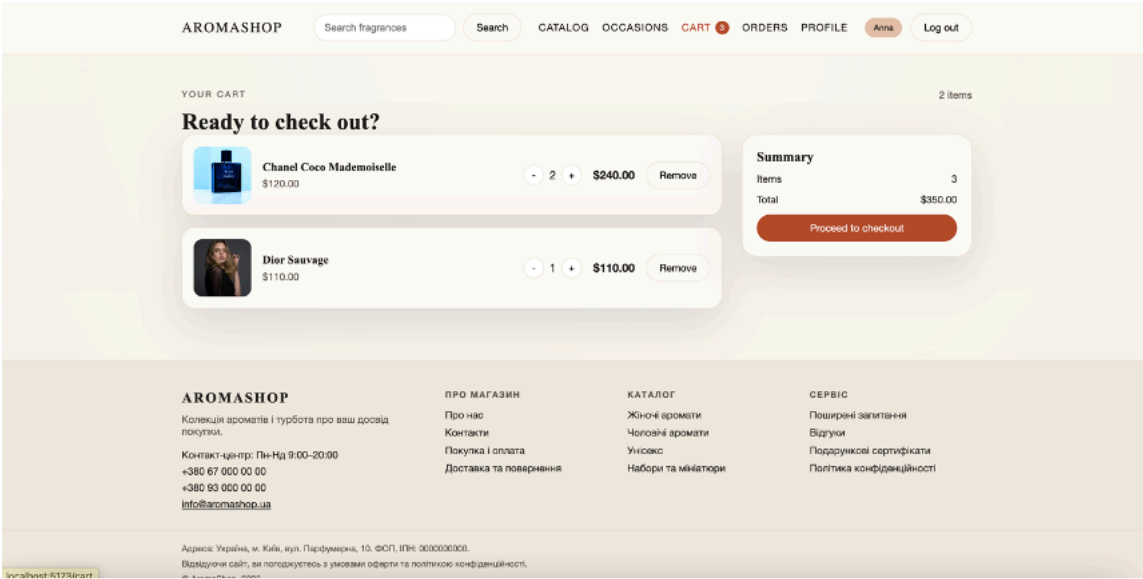
ЕКРАННІ ФОРМИ



Сторінка каталогу товарів

11

ЕКРАННІ ФОРМИ



Сторінка кошику користувача

12

ЕКРАННІ ФОРМИ

OCCASION FINDER

Романтичні, теплі або виразні аромати для особистої зустрічі.

Підбір аромату за подією

На побачення
Романтичні, теплі або виразні аромати для особистої зустрічі.

В офіс
Стримані аромати, які не переважають простір.

На подарунок
Зрозумілі, популярні та безпечні варіанти для різних адресатів.

На кожен день
Універсальні аромати для регулярного використання.

На вечір
Більш наочні аромати для важливого виходу та сезону.

Кому

Стиль

Бюджет від

Бюджет до

Для чоловіка

Романтичний

100

350

Підбрати аромат

Рекомендації

6 варіантів

MAISON FRANCIS KURKDJIAN
MFK Baccarat Rouge 540
EDP - 70 ml - Amber

Підбрано для побачення: аромат має більш емоційний, теплий або квітковий характер.
Адресат: Для чоловіка. Стиль: Романтичний.

На побачення Для чоловіка Романтичний

Amber EDP Рейтинг 5.0 В наявності

\$325.00

Деталі

CHANEL
Chanel Coco Mademoiselle
EDP - 50 ml - Floral

Підбрано для побачення: аромат має більш емоційний, теплий або квітковий характер.
Адресат: Для чоловіка. Стиль: Романтичний.

На побачення Для чоловіка Романтичний

Floral EDP Рейтинг 4.7 В наявності

\$120.00 ~~\$180.00~~

Деталі

Сторінка підбору товару за подією

13

ЕКРАННІ ФОРМИ

ADMIN

8 brands

Admin
PRODUCTS
BRANDS
ORDERS
USERS

Brands

Name

Logo URL

Description

Create brand

NAME	DESCRIPTION	ACTIONS
Chanel	Luxury French fashion house	Edit Delete
Dior	French luxury goods company	Edit Delete
Tom Ford	Modern couture and fragrance house	Edit Delete
Byredo	Swedish luxury fragrance house	Edit Delete
Jo Malone	British fragrance and lifestyle brand	Edit Delete
Maison Francis Kurkdjian	Parisian fragrances atelier	Edit Delete

Сторінка адміністративної панелі брендів

14

АПРОБАЦІЯ РЕЗУЛЬТАТІВ ДОСЛІДЖЕННЯ

Тези доповідей:

1. Мельник В.О., Корецька В.О. Розробка онлайн-магазину парфумерії "AromaShop". Матеріали II Всеукраїнської науково-технічної конференції «Виклики та рішення в програмній інженерії», 26 листопада 2025 року, Київ, Державний університет інформаційно-комунікаційних технологій. Збірник тез. – К.: ДУІКТ, 2025, С.493-495.

2. Мельник В.О., Корецька В.О. Програмна система AromaShop для онлайн-продажу та підбору парфумерної продукції. Матеріали Всеукраїнської науково-технічної конференції «Технології цифрового розвитку: програмні системи та децентралізація», 1-3 травня 2026 року, Київ, ДУІКТ. Збірник тез. (подано до друку).

16

ВИСНОВКИ

1. Проведено аналіз предметної галузі онлайн-продажу парфумерної продукції та визначено: необхідність детального опису товару, наявності парфумерних нот, концентрації, об'єму, відгуків, рейтингу та актуальної інформації про залишки.
2. Досліджено існуючі програмні засоби для онлайн-продажу парфумерії, зокрема BROCARD, MAKEUP, ROZETKA та EVA.UA. Визначено основні проблеми онлайн-вибору парфумерії: неможливість фізично протестувати аромат, складність вибору без знання бренду або назви, обмежена фільтрація та недостатня інформативність.
3. Сформовано функціональні та нефункціональні вимоги до web-застосунку AromaShop.
4. Спроектовано архітектуру web-застосунку, логічну структуру системи, базу даних.
5. Реалізовано web-застосунок AromaShop на основі Java Spring Boot, PostgreSQL, React і TypeScript. Система забезпечує роботу з каталогом парфумерії, кошиком, замовленнями, відгуками, користувацьким профілем та адміністративним керуванням товарами, брендами й замовленнями.
6. Розроблено функцію «Підбір за подією», яка дозволяє користувачу обрати аромат для конкретної ситуації: побачення, офісу, подарунка, щоденного використання або вечірнього виходу. За результатами вибору система формує список відповідних товарів із поясненням причини рекомендації.
7. Проведено функціональне тестування web-застосунку та перевірено коректність роботи всіх функцій системи. Реалізований функціонал відповідає поставленим функціональним та нефункціональним вимогам.

17

ДОДАТОК Б. ЛІСТИНГИ ПРОГРАМНИХ МОДУЛІВ

Listing 1. AromaShopApplication class

Text: The main class starts the Spring Boot application and enables web security and asynchronous processing.

```
@SpringBootApplication
@EnableWebSecurity
@EnableAsync
public class AromaShopApplication {

    public static void main(String[] args) {

SpringApplication.run(AromaShopApplication.class,
args);
    }
}
```

Listing 2. UserEntity class

Text: This entity represents a system user with authentication data, profile fields, and role.

```
@Entity
@Table(name = "users")
@Getter
@Builder(toBuilder = true)
@NoArgsConstructor
@AllArgsConstructor
public class UserEntity {

    @Id
    @GeneratedValue(strategy = GenerationType.UUID)
    private UUID id;

    @Column(nullable = false, unique = true)
    private String email;

    @Column(nullable = false)
    private String password;

    private String username;
    private String fullName;
    private String gender;
    private String phone;

    @Column(columnDefinition = "text")
    private String address;

    @Enumerated(EnumType.STRING)
    @Column(nullable = false)
    private Role role;

    private LocalDateTime createdAt;

    @PrePersist
    protected void onCreate() {
        this.createdAt = LocalDateTime.now();
    }
}
```

Listing 3. ProductEntity class

Text: This entity stores perfume product data, including brand, price, stock, fragrance family, images, and notes.

```
@Entity
@Table(name = "products")
@Getter
@Setter
@Builder(toBuilder = true)
@NoArgsConstructor
@AllArgsConstructor
public class ProductEntity {

    @Id
    @GeneratedValue(strategy = GenerationType.UUID)
    private UUID id;

    private String name;
    private String slug;

    @Column(columnDefinition = "TEXT")
    private String description;

    private String shortDescription;
    private String family;
    private String concentration;
    private BigDecimal volumeML;
    private String eanUpc;
    private Boolean gs1Compliant;
    private Boolean ifraCompliant;

    @Column(columnDefinition = "TEXT")
    private String ifraNotice;

    @ManyToOne
    @JoinColumn(name = "brand_id", nullable = false)
    private BrandEntity brand;

    private BigDecimal price;
    private BigDecimal salePrice;
    private String promoLabel;
    private int stockQuantity;

    @OneToMany(mappedBy = "product", cascade =
CascadeType.ALL, orphanRemoval = true)
    private List<ProductImageEntity> images = new
ArrayList<>();

    @OneToMany(mappedBy = "product", cascade =
CascadeType.ALL, orphanRemoval = true)
    private List<ProductNoteEntity> notes = new
ArrayList<>();

    private LocalDateTime createdAt;

    @PrePersist
    protected void onCreate() {
        this.createdAt = LocalDateTime.now();
    }
}
```

Listing 4. ProductNoteEntity class

Text: This entity connects a product with a perfume note and defines its pyramid level.


```

@Entity
@Table(name = "product_notes")
@Getter
@Builder(toBuilder = true)
@NoArgsConstructor
@AllArgsConstructor
public class ProductNoteEntity {
    @Id
    @GeneratedValue(strategy = GenerationType.UUID)
    private UUID id;

    @ManyToOne
    @JoinColumn(name = "product_id", nullable = false)
    private ProductEntity product;

    @ManyToOne
    @JoinColumn(name = "note_id", nullable = false)
    private NoteEntity note;

    @Enumerated(EnumType.STRING)
    private NoteType noteType;

    private int sortOrder;
}

```

Listing 5. CartEntity class

Text: This entity represents the shopping cart assigned to a registered user.

```

@Entity
@Table(name = "carts")
@Getter
@Builder(toBuilder = true)
@NoArgsConstructor
@AllArgsConstructor
public class CartEntity {
    @Id
    @GeneratedValue(strategy = GenerationType.UUID)
    private UUID id;

    @OneToOne
    @JoinColumn(name = "user_id", nullable = false,
        unique = true)
    private UserEntity user;

    @OneToMany(mappedBy = "cart", fetch =
        FetchType.EAGER)
    private List<CartItemEntity> items = new
        ArrayList<>();

    private LocalDateTime createdAt;

    @PrePersist
    protected void onCreate() {
        this.createdAt = LocalDateTime.now();
    }
}

```

Listing 6. OrderEntity class

Text: This entity stores customer orders, contact data, status, total price, and order items.

```

@Entity
@Table(name = "orders")
@Getter

```

```

@Builder(toBuilder = true)
@NoArgsConstructor
@AllArgsConstructor
public class OrderEntity {
    @Id
    @GeneratedValue(strategy = GenerationType.UUID)
    private UUID id;

    @ManyToOne
    @JoinColumn(name = "user_id", nullable = false)
    private UserEntity user;

    private BigDecimal total;
    private String status;

    @Column(columnDefinition = "text")
    private String address;

    private String fullName;
    private String phone;
    private String email;
    private LocalDateTime createdAt;

    @OneToMany(mappedBy = "order", cascade =
        CascadeType.ALL)
    private List<OrderItemEntity> items;

    @PrePersist
    protected void onCreate() {
        this.createdAt = LocalDateTime.now();
    }
}

```

Listing 7. ProductController class

Text: This controller provides endpoints for product search, product details, filtering, and admin product management.

```

@RestController
@RequestMapping("/api/v1/products")
@RequiredArgsConstructor
public class ProductController {

    private final ProductService productService;

    @GetMapping
    public ResponseEntity<List<ProductListDTO>>
        getAllProducts() {
        return
            ResponseEntity.ok(productService.listProducts());
    }

    @GetMapping("/{id}")
    public ResponseEntity<ProductDetailDTO>
        getProductById(@PathVariable UUID id) {
        return
            ResponseEntity.ok(productService.getProductDetails(id)
            );
    }

    @GetMapping("/slug/{slug}")
    public ResponseEntity<ProductDetailDTO>
        getProductBySlug(@PathVariable String slug) {

```

```

        return
        ResponseEntity.ok(productService.getProductDetailsBy
        Slug(slug));
    }

    @GetMapping("/search")
    public ResponseEntity<List<ProductListDTO>>
    searchProducts(
        @RequestParam(required = false) String name,
        @RequestParam(required = false) String brand,
        @RequestParam(required = false) List<String>
        families,
        @RequestParam(required = false) List<String>
        concentrations,
        @RequestParam(required = false) BigDecimal
        minPrice,
        @RequestParam(required = false) BigDecimal
        maxPrice) {
        return
        ResponseEntity.ok(productService.searchProducts(
            name, brand, null, null, concentrations,
            families,
            null, null, null, minPrice, maxPrice
        ));
    }

    @PreAuthorize("hasRole('ADMIN')")
    @PostMapping
    public ResponseEntity<ProductDetailDTO>
    createProduct(
        @Valid @RequestBody ProductUpsertRequest
        request) {
        return new
        ResponseEntity<>(productService.createProduct(request
        ), HttpStatus.CREATED);
    }

    @PreAuthorize("hasRole('ADMIN')")
    @PutMapping("/{id}")
    public ResponseEntity<ProductDetailDTO>
    updateProduct(
        @PathVariable UUID id,
        @Valid @RequestBody ProductUpsertRequest
        request) {
        return
        ResponseEntity.ok(productService.updateProduct(id,
        request));
    }

    @PreAuthorize("hasRole('ADMIN')")
    @DeleteMapping("/{id}")
    public ResponseEntity<Void>
    deleteProduct(@PathVariable UUID id) {
        productService.deleteProduct(id);
        return ResponseEntity.noContent().build();
    }
}

```

Listing 8. EventRecommendationController class
Text: This controller provides API endpoints for occasion-based perfume recommendations.

```

@RestController
@RequestMapping("/api/v1/recommendations")

```

```

@RequiredArgsConstructor
public class EventRecommendationController {

    private final EventRecommendationService
    eventRecommendationService;

    @GetMapping("/events")
    public ResponseEntity<List<EventOptionDTO>>
    getAvailableEvents() {
        return
        ResponseEntity.ok(eventRecommendationService.getAv
        ailableEvents());
    }

    @PostMapping("/event")
    public
    ResponseEntity<List<EventRecommendationDTO>>
    getRecommendationsByEvent(
        @Valid @RequestBody
        EventRecommendationRequest request) {
        return ResponseEntity.ok(

        eventRecommendationService.getRecommendationsByE
        vent(request)
        );
    }
}

```

Listing 9. EventRecommendationServiceImpl class
Text: This service selects perfumes for a chosen occasion using product availability, budget, family, concentration, season, and recipient type.

```

@Service
@RequiredArgsConstructor
public class EventRecommendationServiceImpl
implements EventRecommendationService {

    private static final int LIMIT = 8;

    private final ProductService productService;

    @Override
    public List<EventOptionDTO> getAvailableEvents()
    {
        return List.of(
            event("DATE", "На побачення"),
            event("OFFICE", "В офіс"),
            event("GIFT", "На подарунок"),
            event("EVERYDAY", "На кожен день"),
            event("EVENING", "На вечір")
        );
    }

    @Override
    public List<EventRecommendationDTO>
    getRecommendationsByEvent(
        EventRecommendationRequest request) {
        String eventType =
        normalize(request.getEventType(), "EVERYDAY");

        return productService.listProducts().stream()
        .filter(product -> product.getStockQuantity() >
        0)
    }
}

```

```

        .filter(product -> matchesBudget(product,
request))
        .filter(product -> matchesEvent(product,
eventType))
        .sorted(Comparator
            .comparing((ProductListDTO p) ->
                p.getRatingAverage() != null ?
p.getRatingAverage() : 0.0)
            .reversed())
        .limit(LIMIT)
        .map(product ->
EventRecommendationDTO.builder()
            .product(product)
            .reason(buildReason(eventType))
            .badges(List.of(eventType,
product.getFamily(), "В наявності"))
            .build())
        .toList();
    }

    private boolean matchesEvent(ProductListDTO
product, String eventType) {
        String family = product.getFamily();

        return switch (eventType) {
            case "DATE" -> List.of("Floral", "Amber",
"Woody").contains(family);
            case "OFFICE" -> List.of("Aromatic", "Floral",
"Woody").contains(family);
            case "GIFT" -> List.of("Floral", "Aromatic",
"Woody", "Amber").contains(family);
            case "EVENING" -> List.of("Amber", "Woody",
"Floral").contains(family);
            default -> true;
        };
    }

    private boolean matchesBudget(ProductListDTO
product, EventRecommendationRequest request) {
        BigDecimal price = product.getSalePrice() != null
            ? product.getSalePrice()
            : product.getPrice();

        if (request.getBudgetMin() != null &&
price.compareTo(request.getBudgetMin()) < 0) {
            return false;
        }

        return request.getBudgetMax() == null ||
price.compareTo(request.getBudgetMax()) <= 0;
    }

    private String buildReason(String eventType) {
        return switch (eventType) {
            case "DATE" -> "Підібрано для побачення.";
            case "OFFICE" -> "Підібрано для офісу.";
            case "GIFT" -> "Підібрано як подарунок.";
            case "EVENING" -> "Підібрано для вечора.";
            default -> "Підібрано на кожен день.";
        };
    }
}

```

```

    private EventOptionDTO event(String value, String
label) {
        return EventOptionDTO.builder()
            .value(value)
            .label(label)
            .description("Occasion preset")
            .recipientTypes(List.of())
            .seasons(List.of())
            .styles(List.of())
            .build();
    }

    private String normalize(String value, String fallback)
{
        return value == null || value.isBlank()
            ? fallback
            : value.trim().toUpperCase();
    }
}

```

Listing 10. AuthServiceImpl class

Text: This service handles user registration, login, password verification, and JWT token generation.

```

@Service
@RequiredArgsConstructor
public class AuthServiceImpl implements AuthService {

    private final UserRepository userRepository;
    private final JwtTokenProvider jwtTokenProvider;
    private final PasswordEncoder passwordEncoder;

    @Override
    public UserDTO registerUser(RegisterRequest
request) {
        if
(userRepository.existsByEmail(request.getEmail())) {
            throw new UserAlreadyExistsException("User
already exists");
        }

        UserEntity user =
userRepository.save(UserEntity.builder()
            .username(request.getUsername())
            .email(request.getEmail())

.password(passwordEncoder.encode(request.getPassword()))

            .role(Role.USER)
            .createdAt(LocalDateTime.now())
            .build());

        return mapToDto(user);
    }

    @Override
    public UserDTO loginUser(LoginRequest request) {
        UserEntity user = Optional

.ofNullable(userRepository.findByEmail(request.getEma
il()))

            .orElseThrow(() -> new
AuthenticationException("Invalid email or password"));
    }
}

```



```

        if
(!passwordEncoder.matches(request.getPassword(),
user.getPassword())) {
            throw new AuthenticationException("Invalid
email or password");
        }

        return mapToDto(user);
    }

    public String generateToken(UserDTO user) {
        Role role =
userRepository.findByEmail(user.getEmail()).getRole();
        return
jwtTokenProvider.createToken(user.getEmail(), role);
    }

    private UserDTO mapToDto(UserEntity user) {
        return UserDTO.builder()
            .id(user.getId())
            .email(user.getEmail())
            .username(user.getUsername())
            .role(user.getRole())
            .fullName(user.getFullName())
            .gender(user.getGender())
            .phone(user.getPhone())
            .address(user.getAddress())
            .build();
    }
}

```