

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ імені ІГОРЯ СІКОРСЬКОГО»
Факультет інформатики та обчислювальної техніки
(повна назва інституту/факультету)

Кафедра інформатики та програмної інженерії
(повна назва кафедри)

«До захисту допущено»

Завідувач кафедри

_____ Едуард ЖАРИКОВ
(підпис) (ім'я прізвище)

“ _____ ” _____ 2025 р.

Дипломний проєкт
на здобуття ступеня бакалавра
за освітньо-професійною програмою «Інженерія програмного забезпечення
інформаційних систем»
спеціальності «121 Інженерія програмного забезпечення»

на тему: Вебзастосунок підбору супермаркету за кошиком продуктів та
місцем розташування

Виконав студент IV курсу, групи ІП-15
(шифр групи)

Аджигельдієва Мадіна Алімівна
(прізвище, ім'я, по батькові) _____ (підпис)

Керівник ст. викл., Ковтунець О.В.
(посада, науковий ступінь, вчене звання, прізвище та ініціали) _____ (підпис)

Консультант ас., Шулькевич Т. В.
(посада, науковий ступінь, вчене звання, прізвище та ініціали) _____ (підпис)

Рецензент доцент кафедри ІСТ, к.т.н., доц., Галушко Д.О.
(посада, науковий ступінь, вчене звання, прізвище та ініціали) _____ (підпис)

Засвідчую, що у цьому дипломному проєкті
немає запозичень з праць інших авторів без
відповідних посилань.

Студент _____
(підпис)

Київ – 2025

Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»
Факультет інформатики та обчислювальної техніки
Кафедра інформатики та програмної інженерії
Рівень вищої освіти – перший (бакалаврський)
Спеціальність – 121 Інженерія програмного забезпечення
Освітньо-професійна програма – Інженерія програмного забезпечення
інформаційних систем

ЗАТВЕРДЖУЮ

Завідувач кафедри

_____ Едуард ЖАРИКОВ
(підпис) (ім'я прізвище)

“ _____ ” _____ 2025 р.

ЗАВДАННЯ
на дипломний проєкт студенту

Аджигельдієвій Мадіні Алімівні
(прізвище, ім'я, по батькові)

1. Тема проєкту Вебзастосунок підбору супермаркету за кошиком продуктів та місцем розташування

керівник проєкту Ковтунець Олесь Володимирович, старший викладач
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

затверджені наказом по університету від « 23 » травня 2025 р. № 1705-с

2. Термін подання студентом проєкту «16» червня 2025 року

3. Вихідні дані до проєкту: технічне завдання

4. Зміст пояснювальної записки

1) Передпроєктне обстеження предметної області: постановка завдання дипломного проєктування, аналіз предметної області, аналіз існуючих рішень, аналіз та моделювання бізнес-процесів.

2) Розроблення вимог до програмного забезпечення: варіанти використання програмного забезпечення, розроблення функціональних вимог, розроблення нефункціональних вимог, аналіз економічних показників програмного забезпечення, постановка завдання на розробку програмного забезпечення.

3) Конструювання та розроблення програмного забезпечення: архітектура програмного забезпечення, архітектурні рішення та обґрунтування вибору засобів розробки, конструювання програмного забезпечення, аналіз безпеки даних.

4) Аналіз якості та тестування програмного забезпечення: аналіз якості програмного забезпечення, опис процесів тестування, опис контрольного прикладу.

5) Розгортання та супровід програмного забезпечення: розгортання програмного забезпечення, супровід програмного забезпечення.

5. Перелік графічного матеріалу

1) Схема структурна варіантів використання

2) Схема структурна компонентів програмного забезпечення

3) Схема бази даних

6. Консультанти розділів проєкту

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв

7. Дата видачі завдання «15» березня 2025 року

Календарний план

№ з/п	Назва етапів виконання дипломного проєкту	Термін виконання етапів проєкту	Примітка
1	Вивчення рекомендованої літератури	15.03.2025	
2	Аналіз існуючих методів розв'язання задачі	15.04.2025	
3	Постановка та формалізація задачі	16.04.2025	
4	Розробка інформаційного забезпечення	23.04.2025	
5	Алгоритмізація задачі	24.04.2025	
6	Обґрунтування вибору використаних технічних засобів	25.04.2025	
7	Розробка програмного забезпечення	15.05.2025	
8	Налагодження програми	24.05.2025	
9	Виконання графічних документів	28.05.2025	
10	Оформлення пояснювальної записки	30.05.2025	
11	Подання ДП на попередній захист	04.06.2025	
12	Подання ДП рецензенту	10.06.2025	
13	Подання ДП на основний захист	16.06.2025	

Студент

(підпис)

Мадіна АДЖИГЕЛЬДІЄВА

(ініціали, прізвище)

Керівник

(підпис)

Олесь КОВТУНЕЦЬ

(ініціали, прізвище)

АНОТАЦІЯ

Пояснювальна записка дипломного проєкту складається з п'яти розділів, містить 66 таблиць, 46 рисунків та 22 джерела – загалом 98 сторінок.

Дипломний проєкт присвячений розробці вебзастосунку підбору супермаркету за кошиком продуктів та місцем розташування.

Мета – створення і надання зручного вебзастосунку для користувачів, які шукають доступні ціни на продукти в популярних супермаркетах поблизу свого місцезнаходження.

У розділі передпроєктного обстеження предметної області представлено аналіз предметної області, аналіз існуючих рішень, аналіз та моделювання бізнес-процесів.

У розділі розроблення вимог до програмного забезпечення представлено варіанти використання програмного забезпечення, функціональні та нефункціональні вимоги, аналіз економічних показників програмного забезпечення.

У розділі конструювання та розроблення програмного забезпечення представлено архітектуру розробленого програмного забезпечення, виконано конструювання програмного забезпечення, аналіз безпеки даних.

У розділі аналізу якості та тестування програмного забезпечення представлено аналіз якості програмного забезпечення, описано процеси тестування та контрольний приклад.

У розділі розгортання та супроводу програмного забезпечення описано процеси розгортання програмного забезпечення та надано інформацію щодо супроводу програмного забезпечення.

Програмне забезпечення впроваджено на віддаленому сервері платформи Vultr.

КЛЮЧОВІ СЛОВА: ВЕБЗАСТОСУНОК, КЛІЄНТ-СЕРВЕР, СУПЕРМАРКЕТИ, КОШИК ПРОДУКТІВ, ГЕОЛОКАЦІЯ.

ABSTRACT

The explanatory note of the diploma project consists of five sections, contains 66 tables, 46 figures and 22 sources – in total 98 pages.

The purpose of the diploma project is to develop a web application for selecting a supermarket by grocery basket and user's geographic location.

The goal is to create and provide a convenient web application for users looking for affordable prices for products in popular supermarkets near their location.

The pre-project scope analysis section presents the scope analysis, existing solutions analysis, business process analysis and modeling.

The section on developing software requirements presents software use cases, functional and non-functional requirements, and analysis of economic software metrics.

The section on software design and development presents the developed software architecture, performs software design, and data security analysis.

The section on software quality analysis and testing presents software quality analysis, describes test cases.

The section on software deployment and support describes the software deployment processes and provides information on software support.

The software is implemented on a remote server of the Vultr platform.

KEYWORDS: WEB APPLICATION, CLIENT-SERVER, SUPERMARKETS, GROCERY BASKET, GEOLOCATION

Факультет інформатики та обчислювальної техніки

Кафедра інформатики та програмної інженерії

“ЗАТВЕРДЖЕНО”

Завідувач кафедри

_____ Едуард ЖАРІКОВ

“ ____ ” _____ 2025 р.

**ВЕБЗАСТОСУНОК ПІДБОРУ СУПЕРМАРКЕТУ ЗА КОШИКОМ
ПРОДУКТІВ ТА МІСЦЕМ РОЗТАШУВАННЯ**

Технічне завдання

КПІ.ІП-1501.045440.01.91

“ПОГОДЖЕНО”

Керівник проєкту:

_____ Олесь КОВТУНЕЦЬ

Нормоконтроль:

_____ Тетяна ШУЛЬКЕВИЧ

Виконавець:

_____ Мадіна АДЖИГЕЛЬДІЄВА

Київ – 2025

ЗМІСТ

1 НАЙМЕНУВАННЯ ТА ГАЛУЗЬ ЗАСТОСУВАННЯ.....	3
2 ПІДСТАВА ДЛЯ РОЗРОБКИ.....	4
3 ПРИЗНАЧЕННЯ РОЗРОБКИ.....	5
4 ВИМОГИ ДО ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ.....	6
4.1 Вимоги до функціональних характеристик.....	6
4.1.1 Користувацького інтерфейсу.....	6
4.1.2 Облік користувачів:.....	13
4.1.2.1 Для користувача:.....	13
4.1.2.2 Для адміністратора системи:.....	13
4.1.2.3 Додаткові вимоги:.....	13
4.2 Вимоги до надійності.....	14
4.3 Умови експлуатації.....	14
4.3.1 Вид обслуговування.....	14
4.3.2 Обслуговуючий персонал.....	14
4.4 Вимоги до складу і параметрів технічних засобів.....	14
4.5 Вимоги до інформаційної та програмної сумісності.....	14
4.5.1 Вимоги до вхідних даних.....	15
4.5.2 Вимоги до вихідних даних.....	15
4.5.3 Вимоги до мови розробки.....	15
4.5.4 Вимоги до середовища розробки.....	15
4.5.5 Вимоги до представленню вихідних кодів.....	15
4.6 Вимоги до маркування та пакування.....	15
4.7 Вимоги до транспортування та зберігання.....	15
4.8 Спеціальні вимоги.....	15
5 ВИМОГИ ДО ПРОГРАМНОЇ ДОКУМЕНТАЦІЇ.....	16
5.1 Попередній склад програмної документації.....	16
5.2 Спеціальні вимоги до програмної документації.....	16
6 СТАДІЇ І ЕТАПИ РОЗРОБКИ.....	17
7 ПОРЯДОК КОНТРОЛЮ ТА ПРИЙМАННЯ.....	18

1 НАЙМЕНУВАННЯ ТА ГАЛУЗЬ ЗАСТОСУВАННЯ

Назва розробки: Вебзастосунок підбору супермаркету за кошиком продуктів та місцем розташування.

Галузь застосування:

Наведене технічне завдання поширюється на розробку вебзастосунку підбору супермаркету за кошиком продуктів та місцем розташування [AGG], котре використовується для знаходження оптимального супермаркету на основі кошика продуктів і місця розташування користувача. Даний застосунок корисний для всіх учасників процесу купування товарів. Покупцям допомагає робити доступні та вигідні покупки, продавцям служить помічником у пошуку клієнтів.

2 ПІДСТАВА ДЛЯ РОЗРОБКИ

Підставою для розробки вебзастосунку підбору супермаркету за кошиком продуктів та місцем розташування є завдання на дипломне проєктування, затверджене кафедрою інформатики та програмної інженерії Національного технічного університету України «Київський політехнічний інститут імені Ігоря Сікорського».

3 ПРИЗНАЧЕННЯ РОЗРОБКИ

Розробка призначена для спрощення процесу вибору супермаркету шляхом автоматизації порівняння цін продуктів з урахуванням місця розташування користувача.

Метою розробки є надання зручного вебзастосунку для користувачів, які шукають доступні ціни на продукти в популярних супермаркетах поблизу свого місцезнаходження.

4 ВИМОГИ ДО ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

4.1 Вимоги до функціональних характеристик

Програмне забезпечення повинно забезпечувати виконання наступних основних функцій:

4.1.1 Користувацького інтерфейсу

Для користувача:

- Список категорій продуктів.

На головній сторінці, де представлений список категорій продуктів, система повинна передбачати можливість фільтрувати по категорії для звуження пошуку потрібного продукту (рис. 4.1).

AGG

Корзина(0)

Фрукти, овочі Фруктові, овочеві снеки (0) Фрукти (12) Ягоди (0) Смузі і фреші (0) Овочі (17) Зелень і салати (14) Гриби (4) Соління (0) Горіхи і сухофрукти (4)	М'ясо М'ясо птиці (4) Свинина (0) Яловичина та телятина (0) Баранина та ягнятина (0) Кролятина (0) Фарш (0) Сало (0) Субпродукти (0) М'ясні напівфабрикати (0) М'ясо для шашлика та барбекю (0)	Риба Свіжа риба (4) Заморожена риба (0) Приготована та копчена риба (5) Копчена і в'ялена риба (2) Рибні напівфабрикати (0) Морепродукти та молоски (2) Устриці та лобстери (0) Крабові палички (0) Ікра (3) Заморожені равишки (0)
Ковбаси і м'ясні делікатеси Сосиски і сардельки (2) Ковбаси (6) М'ясні делікатеси (11) Хамон (0) М'ясні снеки (0) М'ясо-ковбасна нарізка (0)	Сири Сири з пліснявою (0) Крем-сири (0) Сири розсільні (фета, моцарела, інші) (0) Сири плавлені (0) Тверді і напівтверді сири (4) Набори сирів (0)	Хліб та випічка Хлібобулочні вироби (6) Випічка (11) Коржі, основа для піци (3) Сушка, хлібці, праники (7)

Рисунок 4.1 – Список категорій продуктів

- Фільтрація по категорії продуктів.

При виборі категорії система повинна забезпечувати перехід до списку продуктів, відфільтрованих за цією категорією (рис. 4.2).

AGG

Корзина(0)

Вода (215)

Напої / Вода

Негазована вода (0) Слабогазована вода (0) Сильногазована вода (0) Смакова вода (0)

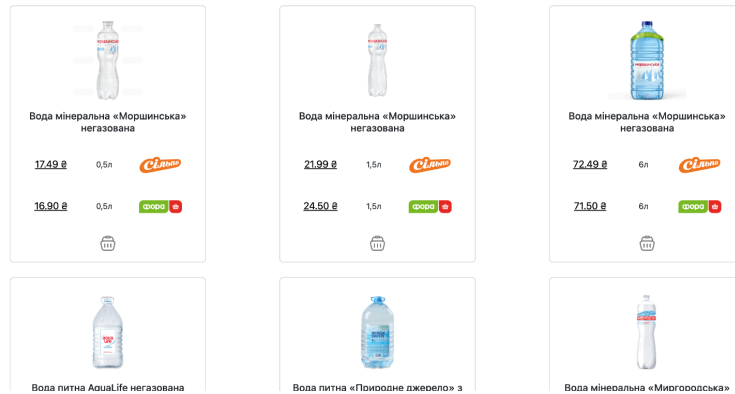


Рисунок 4.2 – Список продуктів, відфільтрованих за категорією
– Фільтрація по назві продукта.

На сторінці зі списком продуктів, відфільтрованих за категорією, система повинна передбачити фільтрацію за назвою продукту (рис. 4.3).

AGG

Корзина(0)

Вода (4)

Напої / Вода

Негазована вода (0) Слабогазована вода (0) Сильногазована вода (0) Смакова вода (0)

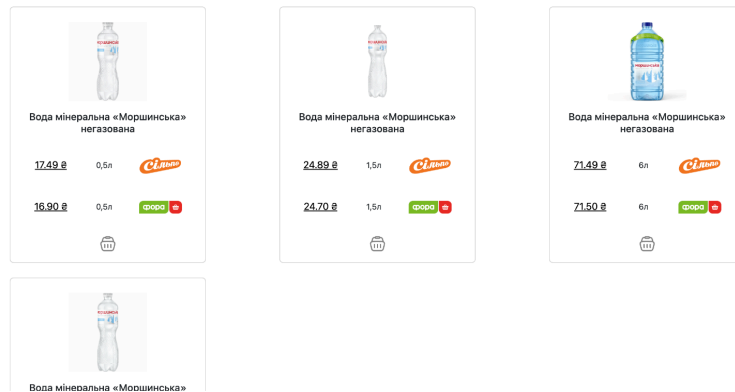


Рисунок 4.3 – Список продуктів, відфільтрованих за назвою
– Формування кошику продуктів.

Система повинна забезпечувати формування кошику продуктів (рис. 4.4) шляхом додавання потрібного продукту в кошик (рис. 4.5).

AGG

Корзина(1)

Вода (4)

Вода мінеральна «Моршинська» негазова

Напої / Вода

Негазована вода (0)

Слабогазована вода (0)

Сильногазована вода (0)

Смакова вода (0)

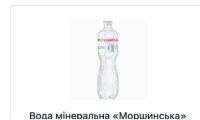
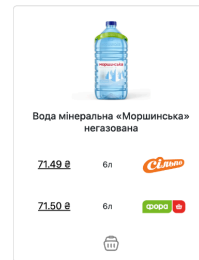
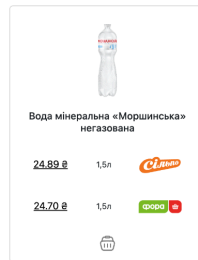
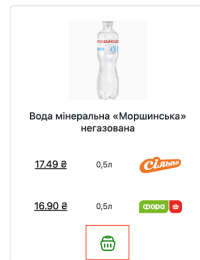


Рисунок 4.4 – Додавання продукту до кошика продуктів

AGG

Корзина(1)

Корзина продуктів

	Вода мінеральна «Моршинська» негазована	0,5л	- 1 +	 17.498	 16.908 
Загальна сума:				17.498	16.908
Найближчий магазин					

Найближчий магазин

Рисунок 4.5 – Формування кошику продуктів

– Порівняння загальної вартості кошику продуктів по супермаркетах. На сторінці кошику продуктів система повинна передбачити можливість змінювати кількість одиниць продукту з автоматичним підрахунком загальної суми, що буде використовуватись для порівняння цін по різних супермаркетах (рис. 4.6).

AGG

Корзина(1)

Корзина продуктів

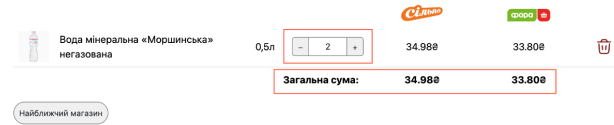


Рисунок 4.6 – Загальна вартість кошику продуктів по супермаркетах

– Пошук найближчого супермаркету.

Система повинна здійснювати пошук найближчого супермаркету, забезпечуючи можливість вибору на карті свого місце розташування. Після підтвердження вибору має бути відображена карта з позначеними найближчими супермаркетами і список, відсортований за зростанням відстані до них (рис. 4.7).

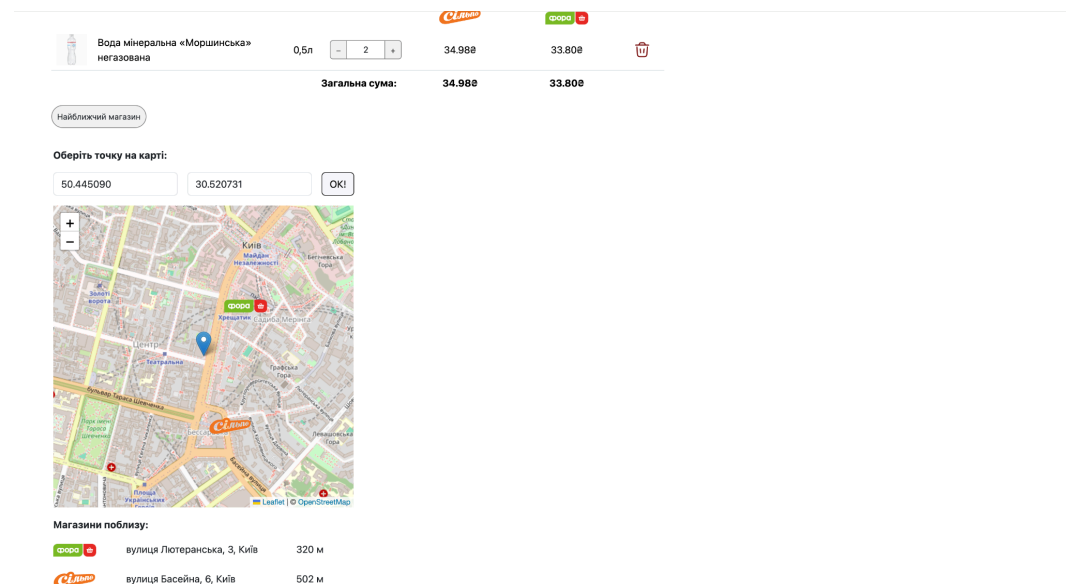


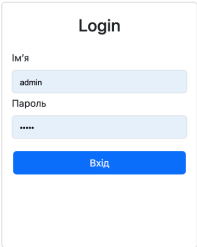
Рисунок 4.7 – Найближчий супермаркет

Для адміністратора системи:

– Вхід в систему.

Система повинна забезпечити можливість авторизації, що включає

введення імені користувача та пароля. Після чого має відбуватися автентифікація та вхід в систему (рис. 4.8).

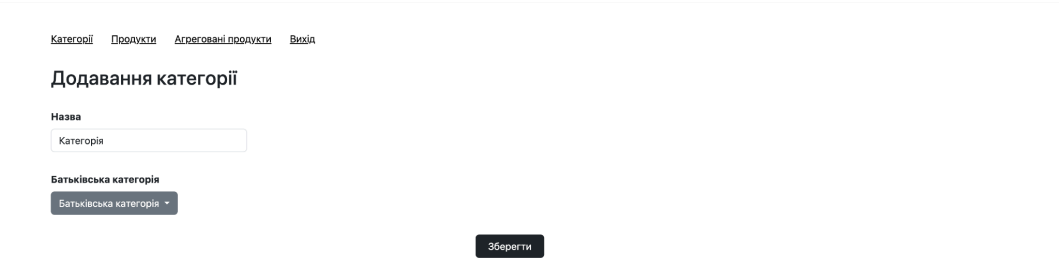


The image shows a 'Login' form. It has a title 'Login' at the top. Below it are two input fields: 'Ім'я' (Username) with the value 'admin' and 'Пароль' (Password) with masked characters '.....'. At the bottom is a blue button labeled 'Вхід' (Login).

Рисунок 4.8 – Вхід в систему

– Додавання нової категорії продуктів агрегатору.

Система повинна забезпечувати додавання нової категорії продуктів, що передбачає введення назви категорії та, за необхідності, вибору її батьківської категорії (рис. 4.9).

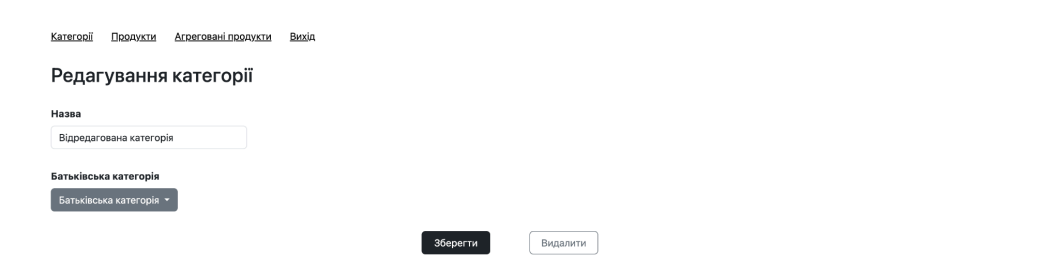


The image shows a web interface for 'Додавання категорії' (Adding category). It has a navigation bar with links: Категорії, Продукти, Агреговані продукти, and Вихід. The form has a 'Назва' (Name) field with the value 'Категорія'. Below it is a 'Батьківська категорія' (Parent category) dropdown menu with the selected value 'Батьківська категорія'. At the bottom is a 'Зберегти' (Save) button.

Рисунок 4.9 – Додавання нової категорії продуктів

– Редагування категорії продуктів агрегатору.

Система повинна передбачити редагування категорії продуктів шляхом зміни назви та/або батьківської категорії (рис. 4.10).



The image shows a web interface for 'Редагування категорії' (Editing category). It has a navigation bar with links: Категорії, Продукти, Агреговані продукти, and Вихід. The form has a 'Назва' (Name) field with the value 'Відредагована категорія'. Below it is a 'Батьківська категорія' (Parent category) dropdown menu with the selected value 'Батьківська категорія'. At the bottom are two buttons: 'Зберегти' (Save) and 'Видалити' (Delete).

Рисунок 4.10 – Редагування категорії продуктів

- Видалення категорії продуктів агрегатору.

На сторінці редагування категорії продуктів система повинна забезпечувати можливість її видалення, що зобов'язує підтвердження (рис. 4.11).

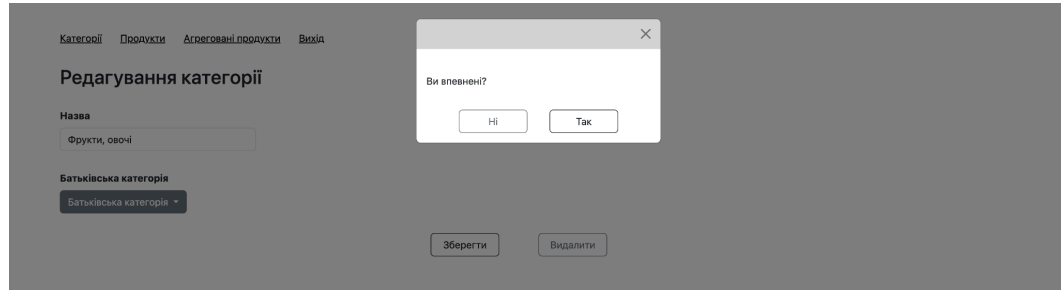


Рисунок 4.11 – Діалогове вікно для підтвердження видалення категорії продуктів

- Агрегація обраних продуктів з супермаркетів.

На сторінці зі списком продуктів система повинна передбачити вибір продуктів для агрегації (рис. 4.12).

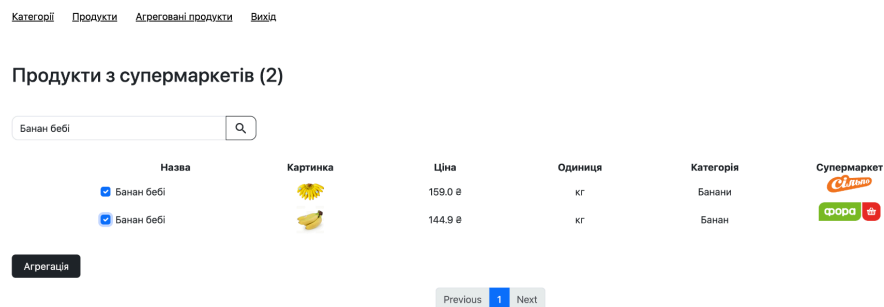


Рисунок 4.12 – Агрегація обраних продуктів з супермаркетів

- Додавання нового продукту в агрегатор.

Система повинна забезпечувати додавання нового продукту в агрегатор з можливістю введення назви, вибрати категорію та зображення продукту (рис. 4.13), інтегруючи його у список продуктів агрегатору.

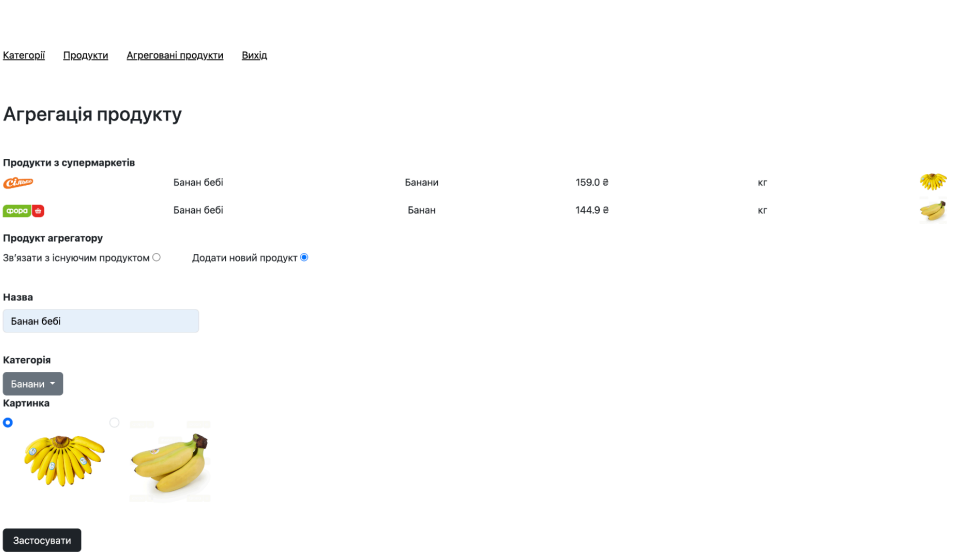


Рисунок 4.13 – Додавання нового продукту в агрегатор – Редагування продукту в агрегаторі.

Система повинна передбачити можливість редагування продукту шляхом зміни назви, категорії або зображення продукту. Даний функціонал також повинен надавати можливість видалення продукту з агрегатору, тобто відв’язати продукт супермаркету від продукту агрегатора (рис. 4.14).

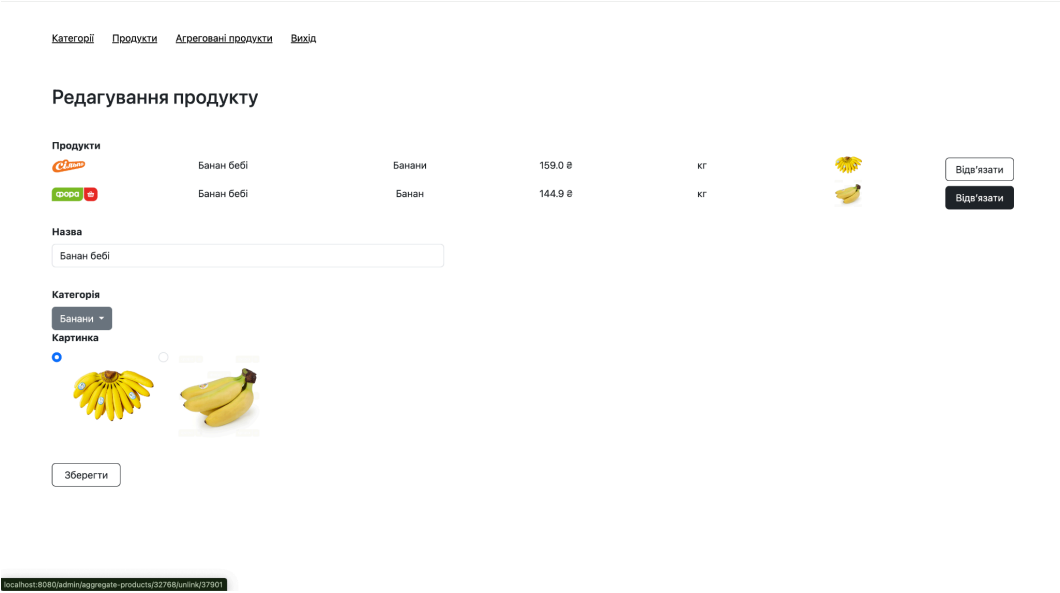


Рисунок 4.14 – Редагування продукту в агрегаторі

4.1.2 Облік користувачів:

4.1.2.1 Для користувача:

- пошук продукту;
- фільтрація по категорії продуктів;
- фільтрація по назві продукту;
- формування кошику продуктів;
- порівняння загальної вартості кошику продуктів по супермаркетах;
- пошук найближчого супермаркету.

4.1.2.2 Для адміністратора системи:

- вхід в систему;
- додавання нової категорії продуктів агрегатору;
- редагування категорії продуктів агрегатору;
- видалення категорії продуктів агрегатору;
- агрегація обраних продуктів з супермаркетів;
- додавання нового продукту в агрегатор;
- редагування продукту в агрегаторі;
- відв'язування продукту супермаркету від продукта агрегатору.

4.1.2.3 Додаткові вимоги:

- скрипт імпортування даних (категорій, продуктів, інформації о супермаркетах) з різних супермаркетів;
- скрипт автоагрегації:
 - 1) повна агрегація;
 - 2) часткова агрегація (категорія продуктів не може бути визначена);
- частота оновлення інформації о продуктах раз в 6 годин.

4.2 Вимоги до надійності

Передбачити контроль введення інформації та захист від некоректних дій користувача. Забезпечити цілісність інформації в базі даних.

4.3 Умови експлуатації

Умови експлуатації згідно СанПін 2.2.2.542 – 96.

4.3.1 Вид обслуговування

Додавання, редагування, видалення категорій та продуктів агрегатору.

4.3.2 Обслуговуючий персонал

Адміністратор вебзастосунку підбору супермаркету за кошиком продуктів та місцем розташування.

4.4 Вимоги до складу і параметрів технічних засобів

Мінімальна конфігурація технічних засобів:

- Server Cloud Compute;
- CPU & Storage Technology: Intel Regular Performance;
- операційна система: Ubuntu 22.04 LTS x64;
- розмір серверу: 25 Гб SSD.

Рекомендована конфігурація технічних засобів:

- операційна система: macOS Sonoma 14.6.1;
- процесор: Apple M1 Pro;
- об'єм ОЗП: 16 Гб;

4.5 Вимоги до інформаційної та програмної сумісності

Програмне забезпечення повинно працювати під управлінням операційних систем сімейства WIN32 (Windows XP, Windows NT і т.д.) або Unix.

4.5.1 Вимоги до вхідних даних

Вхідні дані не передбачені.

4.5.2 Вимоги до вихідних даних

Результат повинен бути представлений в форматі вебзастосунку.

4.5.3 Вимоги до мови розробки

Розробку виконати на мові програмування Java 17.

4.5.4 Вимоги до середовища розробки

Розробку виконати за допомогою середовища IntelliJ IDEA Ultimate 2024.3.1.

4.5.5 Вимоги до представленню вихідних кодів

Вихідний код програми має бути представлений у GitHub.

4.6 Вимоги до маркування та пакування

Вимоги до маркування та пакування не висуваються.

4.7 Вимоги до транспортування та зберігання

Вимоги до транспортування та зберігання не висуваються.

4.8 Спеціальні вимоги

- Скрипт пакування. Результат пакування – jar файл.
- Інструкція по запуску вебзастосунку та скриптов імпортування даних.

5 ВИМОГИ ДО ПРОГРАМНОЇ ДОКУМЕНТАЦІЇ

5.1 Попередній склад програмної документації

У склад супроводжувальної документації повинні входити наступні документи на аркушах формату А4:

- пояснювальна записка;
- технічне завдання;
- текст програми;
- програма та методика тестування;
- керівництво користувача;

Графічна частина повинна бути виконана на аркушах формату А3 та містити наступні документи:

- схема структурна варіантів використання;
- схема структурна компонентів програмного забезпечення;
- схема бази даних;

5.2 Спеціальні вимоги до програмної документації

Програмні модулі, котрі розробляються, повинні бути задокументовані, тобто тексти програм повинні містити всі необхідні коментарі.

6 СТАДІЇ І ЕТАПИ РОЗРОБКИ

№	Назва етапу	Строк	Звітність
1.	Вивчення рекомендованої літератури	15.03	
2.	Аналіз існуючих методів розв'язання задачі	15.04	
3.	Аналіз вимог до програмного забезпечення	21.04	Схема структурна варіантів використання
4.	Алгоритмізація задачі	24.04	
5.	Моделювання програмного забезпечення	01.05	
6.	Розробка архітектури програмного забезпечення	04.05	Схема архітектури програмного забезпечення, схема бази даних
7.	Розробка програмного забезпечення	15.05	Тексти програмного забезпечення
8.	Тестування програмного забезпечення	24.05	Тести, результати тестування
9.	Оформлення графічних документів	28.05	Графічний матеріал проєкту
10.	Оформлення пояснювальної записки	30.05	Пояснювальна записка

7 ПОРЯДОК КОНТРОЛЮ ТА ПРИЙМАННЯ

Тестування розробленого програмного продукту виконується відповідно до «Програми та методики тестування»

**Пояснювальна записка
до дипломного проєкту**

на тему: **Вебзастосунок підбору супермаркету за кошиком продуктів та
місцем розташування**

КП.ІП-1501.045440.02.81

Київ – 2025

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ.....	4
ВСТУП.....	5
1 ПЕРЕДПРОЄКТНЕ ОБСТЕЖЕННЯ ПРЕДМЕТНОЇ ОБЛАСТІ.....	6
1.1 Постановка завдання дипломного проєктування.....	6
1.2 Аналіз предметної області.....	6
1.3 Аналіз існуючих рішень.....	9
1.3.1 Аналіз відомих програмних продуктів.....	9
1.3.2 Аналіз відомих алгоритмічних та технічних рішень.....	14
1.4 Аналіз та моделювання бізнес-процесів.....	16
Висновки до розділу.....	18
2 РОЗРОБЛЕННЯ ВИМОГ ДО ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ.....	20
2.1 Варіанти використання програмного забезпечення.....	20
2.2 Розроблення функціональних вимог.....	20
2.3 Розроблення нефункціональних вимог.....	21
2.4 Аналіз економічних показників програмного забезпечення.....	22
2.5 Постановка завдання на розробку програмного забезпечення.....	26
Висновки до розділу.....	27
3 КОНСТРУЮВАННЯ ТА РОЗРОБЛЕННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ.....	28
3.1 Архітектура програмного забезпечення.....	28
3.2 Архітектурні рішення та обґрунтування вибору засобів розробки.....	30
3.3 Конструювання програмного забезпечення.....	31
3.3.1 Опис структури бази даних.....	31
3.4 Аналіз безпеки даних.....	36
Висновки до розділу.....	37
4 АНАЛІЗ ЯКОСТІ ТА ТЕСТУВАННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ.....	38

4.1 Аналіз якості ПЗ.....	38
4.2 Опис процесів тестування.....	39
4.3 Опис контрольного прикладу.....	48
Висновки до розділу.....	57
5 РОЗГОРТАННЯ ТА СУПРОВІД ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ.....	58
5.1 Розгортання програмного забезпечення.....	58
5.2 Супровід програмного забезпечення.....	60
Висновки до розділу.....	61
ВИСНОВКИ.....	63
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ.....	64
ДОДАТКИ.....	66
ДОДАТОК А. ЗВІТ ПОДІБНОСТІ.....	66
ДОДАТОК Б. ВАРІАНТИ ВИКОРИСТАННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ.....	70
ДОДАТОК В. ОПИС ФУНКЦІОНАЛЬНИХ ВИМОГ ДО ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ.....	84
ДОДАТОК Г. АНАЛІЗ ЕКОНОМІЧНИХ ПОКАЗНИКІВ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ.....	89

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

IDE	– Integrated Development Environment – інтегроване середовище розробки.
API	– Application programming interface, прикладний програмний Інтерфейс.
БД	– База даних.
ПЗ	– Програмне забезпечення.
HTML	– Hypertext Markup Language.
Java EE	– Java Platform, Enterprise Edition.
JVM	– Java Virtual Machine.
HTTP	– Hypertext Transfer Protocol.
URL	– Uniform Resource Locator.
JSON	– JavaScript Object Notation.
XML	– EXtensible Markup Language.
BPMN	– Business Process Model and Notation.
MVC	– Model-View-Controller.
COCOMO	– Constructive Cost Model.
AGG	– Вебзастосунок підбору супермаркету за кошиком продуктів та місцем розташування.

ВСТУП

З кожним роком все більше шукають зручні та швидкі способи придбання товарів онлайн. Одним із них є сайти-агрегатори. Вони збирають і відображають дані з різних джерел в одному місці: товари, новини тощо.

Агрегатори цін дозволяють заощадити час і знайти найвигіднішу пропозицію. Ручний пошук потрібного товару і порівняння вартості на нього в популярних інтернет-магазинах займе чимало часу.

Сучасні споживачі при виборі товарів, особливо продуктів харчування, орієнтуються не лише на порівняння цін окремих позицій. Вони враховують загальну вартість усього кошика покупок у певному магазині, а також зручність його розташування, які стають вирішальними факторами при прийнятті рішення про місце покупки.

Розроблений вебзастосунок реалізує функціонал підбору супермаркету за кошиком продуктів та місцем розташування в межах системи порівняння цін. Застосунок надає можливість порівняти ціни не тільки на певні продукти, а й загальну вартість кошику продуктів з необмеженої кількості супермаркетів. Для додавання нового супермаркету в систему необхідно реалізувати інтерфейс MarketAPI, виконати початкове імпортування продуктів. Для демонстрації можливостей вебзастосунку даний інтерфейс реалізовано для двох, популярних в Україні, супермаркетів – Сільпо та Фора.

Узагальнюючи, агрегатори залишаються актуальними та важливими інструментами для покупців та бізнесів, допомагаючи їм здійснювати більш інформативні та обґрунтовані рішення.

1 ПЕРЕДПРОЄКТНЕ ОБСТЕЖЕННЯ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Постановка завдання дипломного проєктування

Дипломне проєктування передбачає виконання наступних завдань:

- аналіз предметної області та опис її ключових бізнес-процесів, визначення загального завдання розробки;
- аналіз існуючих рішень обраного завдання розробки;
- аналіз та моделювання бізнес-процесів;
- розроблення функціональних та нефункціональних вимог до ПЗ;
- аналіз економічних показників ПЗ;
- постановка завдання на розробку ПЗ;
- розроблення архітектури ПЗ;
- конструювання та розроблення ПЗ;
- аналіз безпеки даних ПЗ;
- аналіз якості та тестування ПЗ;
- розгортання та супровід ПЗ;
- створення супроводжувальної документації до розробленого ПЗ.

1.2 Аналіз предметної області

Існує багато вебзастосунків для порівняння цін, які допомагають користувачам знаходити найкращі пропозиції. В Україні також відслідковується розвиток подібних сервісів, орієнтованих на українського споживача.

Сучасні напрямки розробок включають:

- інтеграцію з мобільними додатками: наприклад, для сканування штрих-кодів товарів для миттєвого порівняння цін [1];
- використання штучного інтелекту: для персоналізації рекомендацій та відстеження цін [2, 3];

- розширення функціоналу: наприклад, створення кошику покупок, а також навігації до магазинів з урахуванням місцезнаходження користувача для максимальної зручності.

Для звичайного покупця практично неможливе самостійне здійснення порівняння цін на продукти в усіх доступних магазинах. Саме тому виникає потреба в інструменті, який би автоматично аналізував вміст кошику користувача та пропонував оптимальний супермаркет поблизу з урахуванням місцезнаходження.

Розробка подібного застосунку охоплює такі ключові області, як: веброзробка та геоінформаційні системи.

Вимоги до сучасних вебдодатків наразі визначають високі стандарти, а очікування користувачів є досить значними. Очікується, що сучасні вебзастосунки будуть не лише надзвичайно доступними з будь-чого, але і функціональними на різних типах пристроїв та розмірах екранів.

Це означає, що вебзастосунки повинні бути адаптовані до різноманітних умов використання, включаючи роботу на мобільних телефонах, планшетах, ноутбуках та настільних комп'ютерах. Крім того, вони повинні враховувати різні швидкості Інтернет-з'єднань та забезпечувати ефективну роботу.

Важливою є також висока відповідність до стандартів безпеки, оскільки користувачі очікують, що їхні дані будуть належним чином захищені. Врахування цих аспектів є ключовим для успішного розвитку та прийняття сучасних вебзастосунків користувачами.

Також, важливим стандартом є кросплатформність служб, яка дозволяє працювати на основних мовах програмування в операційних системах Linux, macOS і Windows. Ця можливість розширює спектр можливостей як для розробки, так і для використання програм. Завдяки цій функціональності стає

можливим ефективно створення та розгортання програм на різних платформах, що сприяє більш широкому доступу та застосуванню вебзастосунків.

Для розробки сучасних застосунків можна використовувати різноманітні інструменти, починаючи від простих текстових редакторів та інтерфейсів командного рядка і закінчуючи багатофункціональними середовищами розробки. Автоматизація розгортання може дозволяти ефективно впроваджувати зміни та покращення у вебзастосунках як частину неперервного процесу розробки.

Враховуючи різноманітний спектр вимог у сучасній веброзробці, знаходження інструментів, які б забезпечували достатню підтримку та мали достатню кількість можливостей для розробки та розгортання застосунків, може бути складним. Однак, для розробки вебзастосунку агрегатору з розширеним функціоналом обрано мову програмування Java та розвинутий фреймворк для розробки – Spring Boot.

Spring Boot є відомим вибором в розробці вебдодатків, оскільки він надає обширний набір функцій, включає в себе інструменти для роботи з базами даних, вебсервісами, безпекою, а також має велику спільноту розробників та обширну документацію.

Вибір такого фреймворку, як Spring Boot, значно полегшує розробку та розгортання вебдодатків, забезпечуючи високий рівень надійності та ефективності вирішення багатьох завдань.

Геоінформаційні системи – це комп'ютерні системи, призначені для збору, зберігання, аналізу та відображення просторових даних з метою підтримки процесів прийняття рішень. Зокрема, такі системи дозволяють визначити найближчі до користувача магазини на основі координат, здійснювати фільтрацію супермаркетів за відстанню тощо [4].

Реалізація картографічної частини вебзастосунку базується на таких технологіях, як Google Directions API, Leaflet та OpenStreetMap.

Об'єднання Google Directions API для точної геолокації, Leaflet та OpenStreetMap для побудови інтерактивної карти та отримання картографічних даних дозволяє розробити ефективну та економічно доцільну частину вебдодатку.

1.3 Аналіз існуючих рішень

Проаналізуємо відомі на сьогодні вебзастосунки у даній області та технічні рішення, що допоможуть у реалізації AGG. Далі будуть розглянуті готові програмні рішення, допоміжні програмні засоби та засоби розробки.

1.3.1 Аналіз відомих програмних продуктів

Як аналоги для порівняння з вебзастосунком AGG обрано два відомих українських онлайн-сервісів для порівняння цін – hotline.ua [5] та ek.ua [6].

Сервіс Hotline працює як агрегатор комерційних пропозицій, надаючи користувачам такі можливості:

- перегляд характеристики та опису товару (рис. 1.1 – рис. 1.2);
- порівняння цін в магазинах (рис. 1.3);
- ознайомлення з відгуками магазинів (рис. 1.4);
- перегляд пропозицій магазинів на мапі (рис. 1.5).

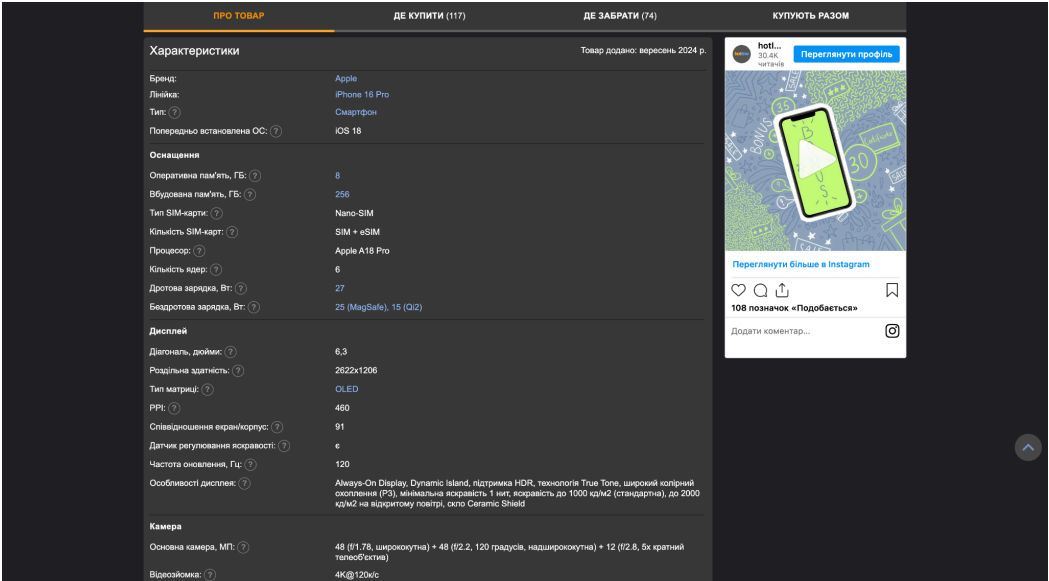


Рисунок 1.1 – Характеристики товару

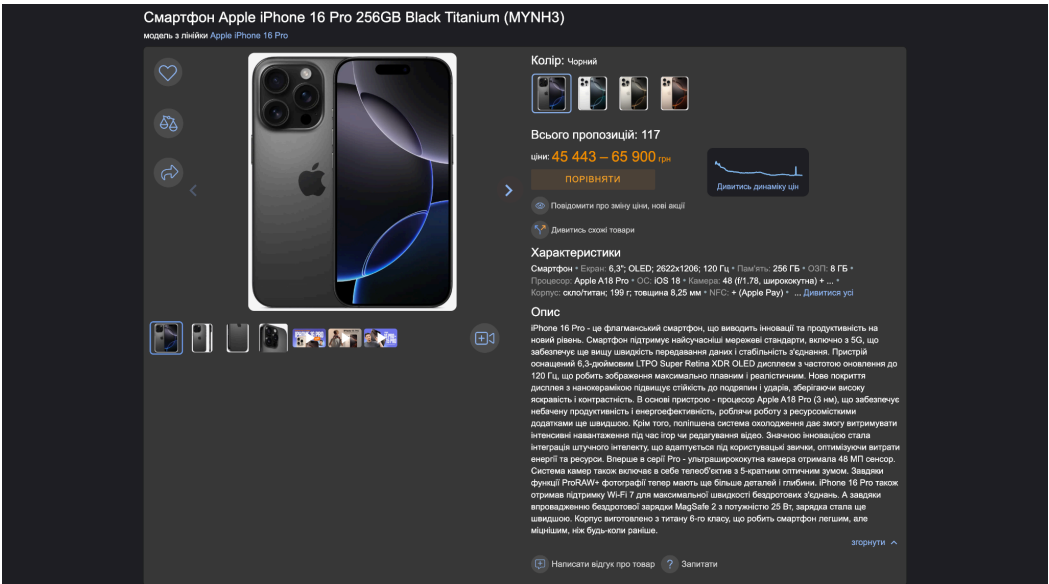


Рисунок 1.2 – Опису товару

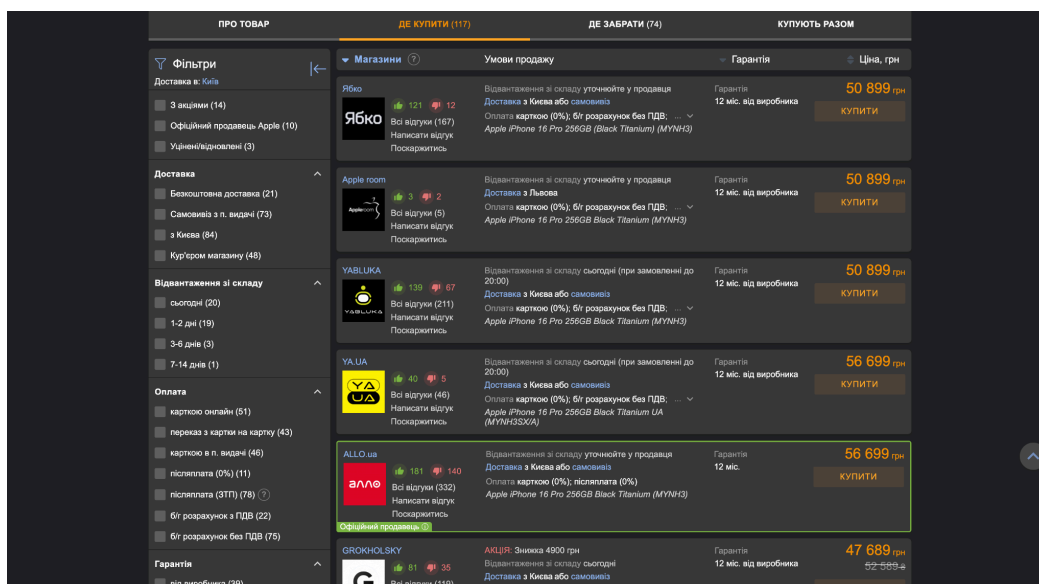


Рисунок 1.3 – Ціни в магазинах

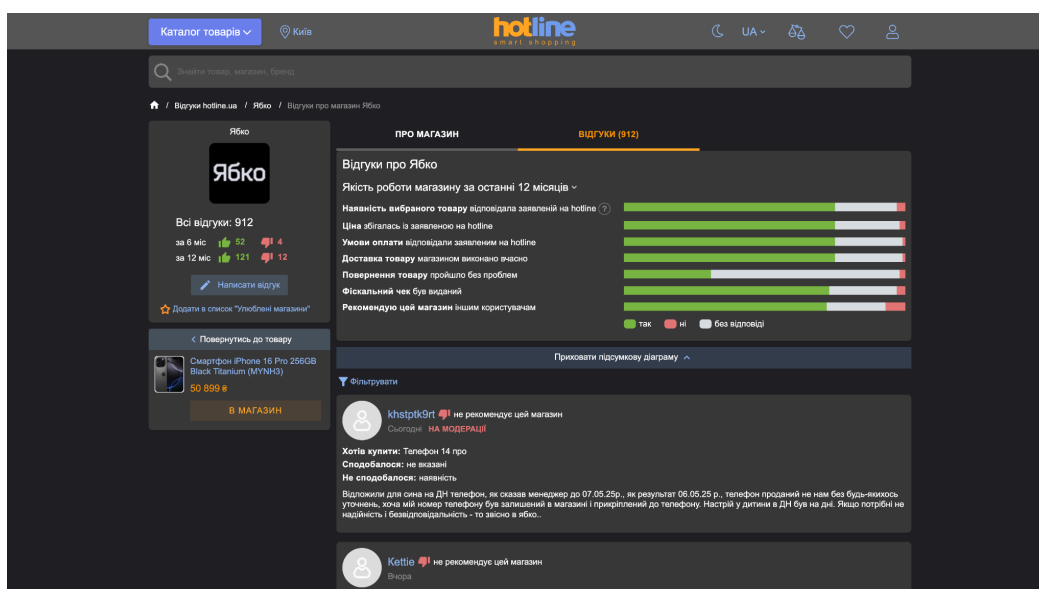


Рисунок 1.4 – Відгуки магазинів

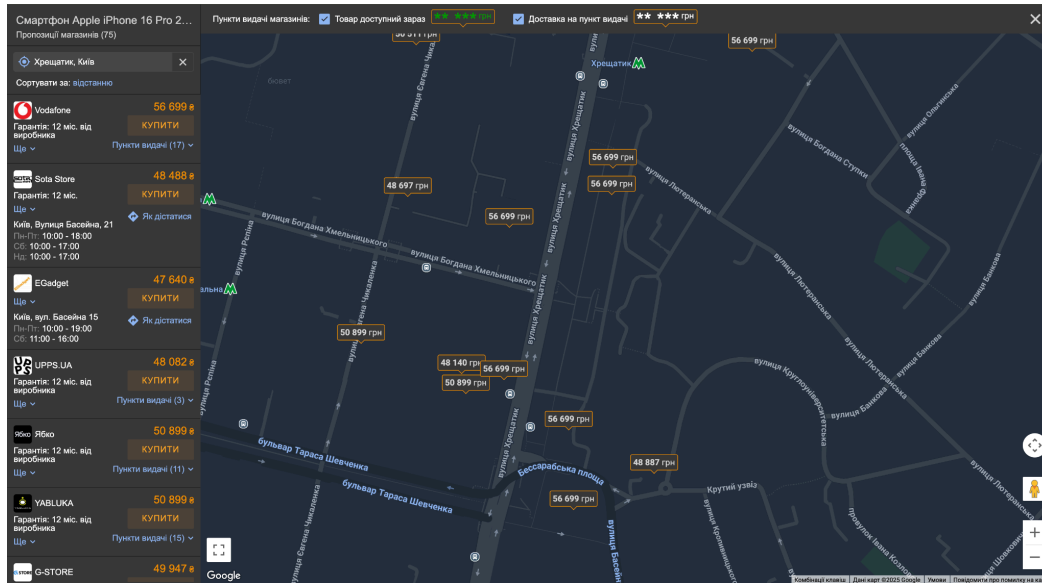


Рисунок 1.5 – Пропозиції магазинів на мапі

Нарівні з Hotline, вагомим застосунком на ринку агрегаторів цін є E-Katalog. Незважаючи на схожість у функціоналі, він має ряд унікальних можливостей, а саме:

- порівняння цін в магазинах за рейтингом (рис. 1.6);
- одночасне порівняння декількох товарів (рис. 1.7);
- ознайомлення з відгуками покупців товару (рис. 1.8).

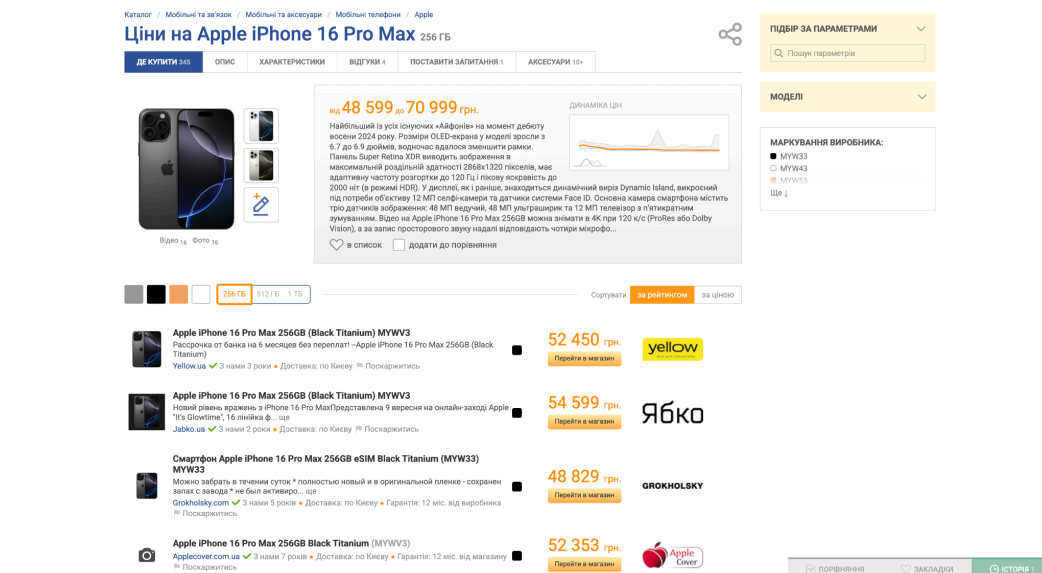


Рисунок 1.6 – Ціни в магазинах за рейтингом

Порівняння Apple iPhone 15 Pro Max 256 ГБ vs Apple iPhone 16 Pro Max 256 ГБ

Всі параметри	Apple iPhone 15 Pro Max 256 ГБ	Apple iPhone 16 Pro Max 256 ГБ
Відгуки	Ябло.ua → 53 499 грн. Grokholsky.com → 49 379 грн. Applecover.com.ua → 50 483 грн.	Yellow.ua → 52 450 грн. Ябло.ua → 54 599 грн. Grokholsky.com → 48 829 грн.
Дисплей	основний дисплей: 6.7" 2796x1290 460 ppi OLED (LTPO) 120 Гц HDR10 Ceramic Shield співвідношення дисплей/корпус: 90 % ТЕСТ ДИОМАНК (дисплей): 151	основний дисплей: 6.9" 2868x1320 459 ppi OLED (LTPO) 120 Гц HDR10+ Dolby Vision Ceramic Shield співвідношення дисплей/корпус: 92 % ТЕСТ ДИОМАНК (дисплей): 150

Рисунок 1.7 – Одночасне порівняння товарів

Відгуки про Apple iPhone 16 Pro Max 256 ГБ

Детальні відгуки: 340, Опис, Характеристики, Відгуки, Поставити запитання, Аксесуари.

Відгуки про Apple iPhone 16 Pro Max 256GB

Гарна модель, понравилось. Вік вражає потужністю продуктивністю завдяки новому процесору, швидкому режиму роботи та вражаючим камерам. Камера з новими функціями дозволяє робити чудові фотографії навіть у складних умовах освітлення. Екран AMOLED виглядає дуже яскраво та чітко, а підтримка 120 Гц робить прокручування та анімації неймовірно плавними. Завдяки великій кількості пам'яті, я маю можливість зберігати багато фото, відео та додатків без жодних проблем. В цілому, пристрій має дуже софійний вигляд і зручний у використанні. Але варто зазначити, що ціна все ж таки висока. Проте якщо ви шукаєте найкраще з технологій, цей смартфон однозначно вартий своїх грошей.

+ Потужна продуктивність
+ Відмінна камера
+ Яскравий і плавний екран
+ Великі можливості для зберігання даних
+ Прекрасна збірка і дизайн

- Висока ціна
Відсутність значних змін у дизайні порівняно з попередніми моделями

9. Оцінка користувачів Apple iPhone 16 Pro Max 256GB кілька місяців

Раджу оновлюватись після 14 айфона. Хто з 15 - почекайте наступного. Дуже красивий дизайн - очував що не відрізнятиметься від попереднього, але - відмінно краще

Рисунок 1.8 – Відгуки покупців

Розглянуті сервіси є корисними для порівняння цін на товари, однак їхня основна мета – не оптимізація процесу покупок шляхом формування кошика та пошуку найближчого магазину з урахуванням геолокації користувача.

Для порівняння розробленого вебзастосунку з аналогами можна скористатись таблицею 1.1.

Таблиця 1.1 – Порівняння з аналогами

Функціонал	AGG	Hotline	E-Katalog
Пошук продукту	+	+	+
Фільтрація по категорії продуктів	+	+	+
Фільтрація по назві продукту	+	+	+
Формування кошику продуктів	+	–	–
Порівняння загальної вартості кошику продуктів по супермаркетах	+	–	–
Пошук найближчого супермаркету	+	–	–

1.3.2 Аналіз відомих алгоритмічних та технічних рішень

Патерни архітектури є важливими для розробки програмного забезпечення, яке відповідає конкретним потребам та викликам. Найпоширенішими архітектурними паттернами програмного забезпечення є багаторівнева, мікросервісна та клієнт-серверна архітектури.

Багаторівневий паттерн підходить для структурування додатків, які можна розділити на групи, де кожна група має певну роль або відповідальність. Він поширений у корпоративних додатках, де спрощує залежності, покращує супроводжуваність, підтримує масштабування та розділення стеку технологій [7].

Мікросервісна архітектура забезпечує створення та керування

мікросервісами, які є невеликими, незалежними сервісами, що працюють разом у застосунку. Вони представляють різні патерни проектування, що охоплюють такі аспекти, як зв'язок сервісів, управління даними та обробка збоїв. Дотримуючись даного паттерну, розробники можуть створювати більш стійкі, масштабовані та зручні в обслуговуванні застосунки [8].

Клієнт-серверний паттерн архітектури має вирішальне значення в сучасному проектуванні систем, полегшуючи ефективний зв'язок між користувачами (клієнтами) і серверами в різноманітних додатках. Дана архітектура забезпечує масштабованість, безпеку та надійність взаємодії з користувачами та управління даними [9]. Саме тому його обрано для реалізації вебзастосунку AGG.

Розробка програмного забезпечення має відбуватися з використанням добре розвинутої платформи, що пропонує широкий спектр можливостей для роботи з вебзастосунками. Також важливо мати систему для встановлення пакетів або бібліотек з метою ефективного організування функціоналу.

Найпопулярнішими платформами, що задовольняють зазначені критерії, є Spring Framework [10] та JHipster. Виходячи із суб'єктивного досвіду роботи із зазначеними системами, обрано Spring Framework.

Spring Framework – фреймворк з відкритим кодом та контейнери з підтримкою інверсії керування для платформи Java. Основні особливості Spring Framework можуть бути використані будь-яким застосунком Java, але є розширення для створення вебдодатків на платформі Java EE.

Як фреймворк для полегшення написання функціоналу вебдодатку використано цілком розвинуте рішення від Spring – Spring Boot [11]. Це найбільший фреймворк, який призначений для допомоги у створенні додатків Spring продуктивного рівня з мінімальною кількістю налаштувань.

1.4 Аналіз та моделювання бізнес-процесів

Для опису бізнес-процесів програмного забезпечення використовується BPMN-модель (рис. 1.9).

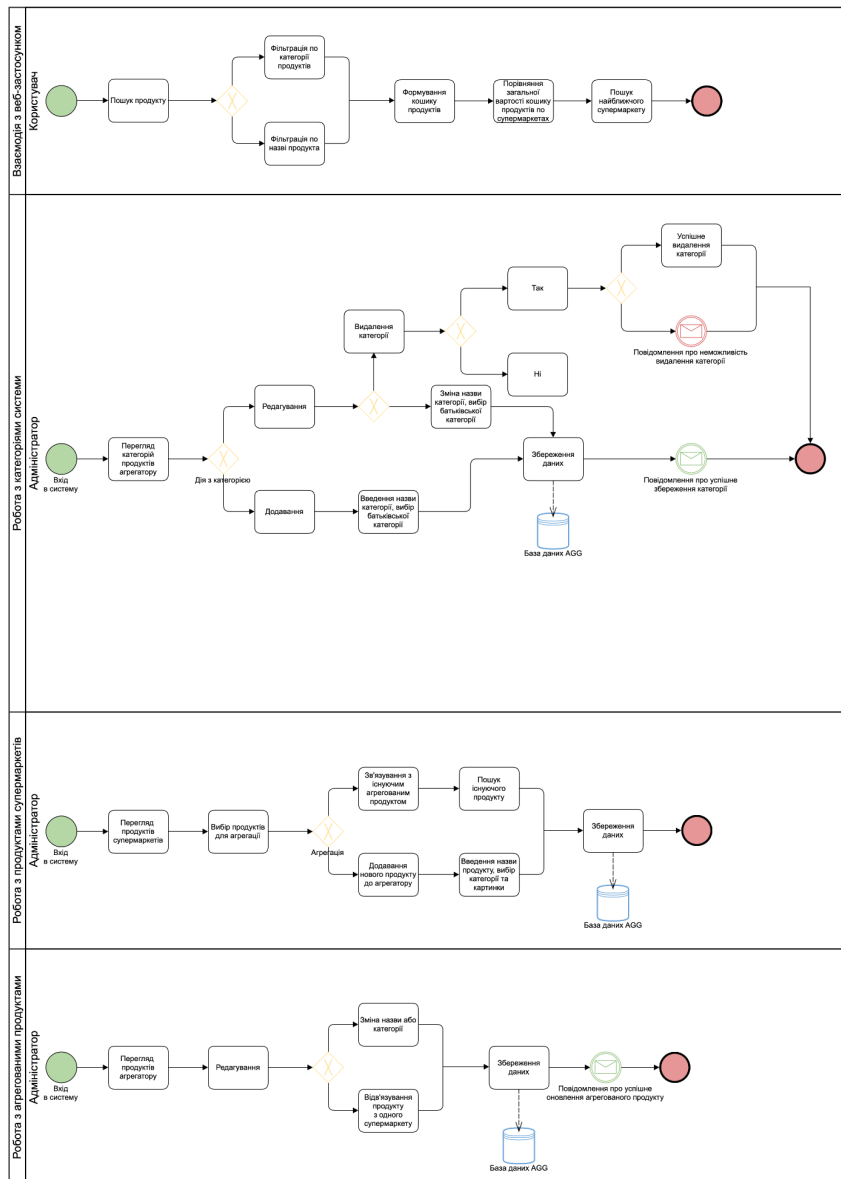


Рисунок 1.9 – Модель BPMN системи AGG

Опис послідовності взаємодії з вебзастосунком:

- користувач перебуває на сайті вебзастосунку AGG;
- користувач шукає продукт;
- якщо користувач має на меті знайти продукт за фільтрацією по

категоріях продуктів, звужуючи область пошуку, то він може обрати потрібну категорію і отримати перелік відповідних продуктів;

- якщо користувач має на меті знайти продукт за фільтрацією по назві продукту, то він може ввести назву продукту в поле пошуку і знайти конкретний продукт;
- користувач формує кошик продуктів, шляхом додавання знайденого продукту в кошик;
- користувач порівнює загальну вартість кошику продуктів по супермаркетах;
- користувач шукає найближчий супермаркет з урахуванням свого місця розташування.

Опис послідовності роботи з категоріями системи:

- адміністратор входить в систему AGG;
- адміністратор переглядає список категорій продуктів агрегатору;
- якщо адміністратор має на меті редагувати категорію, то він може обрати категорію, натиснувши на її назву, змінити назву та/або батьківську категорію, що приведе до оновлення даних в БД AGG;
- якщо адміністратор має на меті видалити категорію, то він може обрати категорію та видалити, натиснувши на відповідну кнопку, що приведе до видалення даних про обрану категорію в БД AGG;
- якщо адміністратор має на меті додати нову категорію, то він може ввести назву та вказати батьківську категорію, що приведе до додавання нової інформації в БД AGG.

Опис послідовності роботи з продуктами супермаркетів:

- адміністратор входить в систему AGG;
- адміністратор переглядає список продуктів з супермаркетів;
- адміністратор обирає продукти для агрегації;

- якщо адміністратор має на меті зв'язати з вже існуючим продуктом в системі агрегатору, то за допомогою пошуку є можливість знайти необхідний продукт та зв'язати з ним, що приведе до оновлення інформації в БД AGG;
- якщо адміністратор має на меті додати новий продукт в систему агрегатору, то він може ввести назву, обрати необхідну категорію та картинку, що приведе до додавання нової інформації в БД AGG.

Опис послідовності роботи з агрегованими продуктами:

- адміністратор входить в систему AGG;
- адміністратор переглядає список агрегованих продуктів;
- адміністратор обирає продукт для редагування;
- якщо адміністратор має на меті змінити назву та/або категорію продукту, то є можливість ввести іншу назву та/або обрати потрібну категорію, що приведе до оновлення інформації в БД AGG;
- якщо адміністратор має на меті відв'язати продукт з супермаркету від агрегованого, то, натискаючи на відповідну кнопку, продукт видаляється, що приведе до оновлення інформації в БД AGG.

Висновки до розділу

У даному розділі визначено завдання дипломного проєктування, аналіз предметної області розробленого вебзастосунку.

Проаналізовано готові програмні продукти, допоміжні програмні засоби та засоби розробки системи. В якості архітектурного рішення було обрано клієнт-серверний паттерн. В якості мови програмування було обрано Java. В якості допоміжного засобу було обрано потужний Spring Framework.

Проаналізовано бізнес-процеси системи AGG, які продемонстровані за допомогою BPMN нотації. Основними учасниками є адміністратор та

користувач, вони представлені у вигляді пулів з відповідними завданнями. Серед бізнес-процесів розглянуто взаємодію користувача з вебзастосунком, роботу адміністратора з категоріями та продуктами агрегатору, а також продуктами з супермаркетів.

2 РОЗРОБЛЕННЯ ВИМОГ ДО ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

2.1 Варіанти використання програмного забезпечення

Діаграма варіантів використання з ключовими акторами та основними функціями показана на рисунку 2.1.

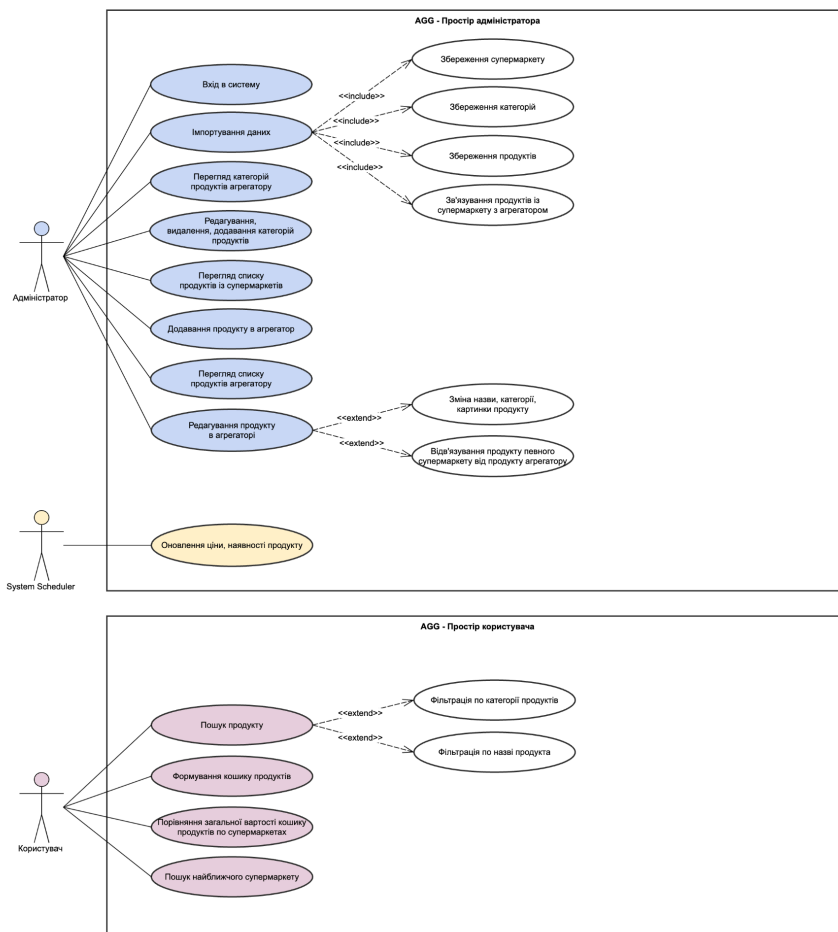


Рисунок 2.1 – Діаграма варіантів використання

В таблицях Б.1 – Б.21 наведені варіанти використання розробленого вебзастосунку AGG.

2.2 Розроблення функціональних вимог

Програмне забезпечення розділене на модулі. Кожен модуль має свій певний набір функцій. На рисунку 2.2 наведено загальну модель вимог, а в таблицях В.1 – В.24 наведений опис функціональних вимог до програмного

забезпечення. Матриця трасування вимог продемонстрована на рисунку 2.3.

ID	Summary	Code	Priority
1	Вхід в систему	FR-01	High
2	Імпортуння даних	FR-02	Medium
3	Збереження супермаркету	FR-03	Medium
4	Збереження категорій	FR-04	Medium
5	Збереження продуктів	FR-05	Medium
6	Перегляд категорій продуктів агрегатору	FR-06	Low
7	Редагування категорій продуктів агрегатору	FR-07	Medium
8	Видалення категорій продуктів агрегатору	FR-08	Medium
9	Додавання категорій продуктів агрегатору	FR-09	Medium
10	Перегляд списку продуктів з супермаркетів	FR-10	High
11	Додавання продукту в агрегатор	FR-11	High
12	Перегляд списку продуктів агрегатору	FR-12	Low
13	Зв'язування продуктів з супермаркету с агрегатором	FR-13	High
14	Редагування продукту в агрегаторі	FR-14	Medium
15	Зміна назви, категорії, картинки продукту	FR-15	Medium
16	Відв'язування продукту певного супермаркету від продукту агрегатору	FR-16	Medium
17	Оновлення ціни, наявності продуктів	FR-17	High
18	Пошук продукту	FR-18	High
19	Фільтрація по категорії продуктів	FR-19	Medium
20	Фільтрація по назві продукта	FR-20	Medium
21	Формування кошику продуктів	FR-21	High
22	Порівняння загальної вартості кошику продуктів по супермаркетах	FR-22	High
23	Введення координат місця розташування	FR-23	Medium
24	Перегляд найближчого супермаркету	FR-24	Medium

Рисунок 2.2 – Модель вимог у загальному вигляді

	FR-1	FR-2	FR-3	FR-4	FR-5	FR-6	FR-7	FR-8	FR-9	FR-10	FR-11	FR-12	FR-13	FR-14	FR-15	FR-16	FR-17	FR-18	FR-19	FR-20	FR-21	FR-22	FR-23	FR-24
UC-01	+																							
UC-02		+																						
UC-03			+																					
UC-04				+																				
UC-05					+																			
UC-06						+																		
UC-07							+	+	+															
UC-08										+														
UC-09											+													
UC-10												+												
UC-11													+											
UC-12														+										
UC-13															+									
UC-14																+								
UC-15																	+							
UC-16																		+						
UC-17																			+					
UC-18																				+				
UC-19																					+			
UC-20																						+		
UC-21																							+	+

Рисунок 2.3 – Матриця трасування вимог

2.3 Розроблення нефункціональних вимог

Нефункціональні вимоги задають атрибути якості системи. Найпоширеніші типи нефункціональних вимог є доступність,

продуктивність, надійність, зручність використання тощо [12]. Для системи AGG визначені наступні вимоги:

- продуктивність: система повинна забезпечувати відповідний рівень продуктивності для певних дій. Час відповіді системи не повинен перевищувати 1 секунду для більшості запитів;
- сумісність: система повинна бути сумісна з існуючими браузером;
- зручність використання: інтерфейс користувача системи повинен бути інтуїтивно зрозумілим та зручним для використання;
- безпека: система повинна гарантувати конфіденційність інформації та захищати від несанкціонованого доступу під час дій.

Вище вказані вимоги сприяють врахуванню важливих аспектів, що не лише забезпечують повноту функціоналу, але й гарантують якість програмного продукту.

2.4 Аналіз економічних показників програмного забезпечення

Розрахунок економічних показників здійснено за допомогою конструктивної моделі вартості COSOMO.

Конструктивна модель COSOMO – це спосіб оцінювання вартості розробки програмного забезпечення, розроблений Баррі Боемом [13]. COSOMO застосовується до трьох класів проєктів розробки ПЗ:

- органічний (до 25 тис. рядків коду);
- напіврозподілений (до 75 тис. рядків коду);
- вбудований (від 75 тис. рядків коду).

Вебзастосунок AGG характеризується відносно невеликим обсягом коду – 21319, досить простою предметною областю та не передбачає складної інтеграції з зовнішніми системами, що дозволяє віднести його до органічного класу проєктів розробки ПЗ.

Існує три типи моделі COCOMO:

- базовий;
- проміжний;
- деталізований.

Базова модель COCOMO – спосіб оцінити зусилля та людино-місяців, виходячи з розміру проекту, що вимірюється в тисячах рядків вихідного коду (KSLOC). Рівняння цієї моделі мають вигляд:

$$PM = a \cdot Size^b, \quad (2.1)$$

$$TM = c \cdot PM^d, \quad (2.2)$$

$$SS = \frac{PM}{TM}, \quad (2.3)$$

$$P = \frac{Size}{PM}, \quad (2.4)$$

де PM – трудомісткість (люд. · міс.);

TM – час розробки в календарних місяцях;

$Size$ – обсяг програмного продукту в тисячах рядків вихідного коду (KSLOC);

SS – середня чисельність персоналу;

P – продуктивність;

a, b, c, d – коефіцієнти, наведені в таблиці Г.1.

Розрахунок показників базової моделі COCOMO:

$$PM = 2,4 \cdot 21,319^{1,05} = 59,6 \text{ люд.} \cdot \text{міс.};$$

$$TM = 2,5 \cdot 59,6^{0,38} = 11,8 \text{ міс.};$$

$$SS = \frac{59,6}{11,8} = 5;$$

$$P = \frac{21,319}{59,6} = 0,35.$$

Проміжна модель COCOMO – розширення базової моделі для точного розрахунку оцінок, що включає набір факторів витрат, які згруповані за чотирма категоріями:

- характеристики продукту;
- характеристики апаратного забезпечення;
- характеристики персоналу;
- характеристики проєкту.

В таблиці Г.2 наведено значення кожного атрибуту вартості факторів витрат відповідно до ступеня значущості в розробленому проєкті.

Формула моделі проміжного рівня має вигляд:

$$PM = EAF \cdot a \cdot Size^b, \quad (2.5)$$

де PM – трудомісткість (люд. · міс.);

EAF – добуток обраних атрибутів вартості;

$Size$ – обсяг програмного продукту в тисячах рядків вихідного коду (KSLOC);

a, b – коефіцієнти, наведені в таблиці Г.3.

Час розробки розраховується за формулою (2.2).

Розрахунок показників проміжної моделі COCOMO:

$$EAF = 1,15 \cdot 1,00 \cdot 1,00 \cdot 1,00 \cdot 1,00 \cdot 1,00 \cdot 0,86 \cdot 1,00 \cdot 0,86 \cdot 1,00 \cdot 0,95 \cdot 1,00 \cdot 0,91 \cdot 1,10 = 0,8;$$

$$PM = 0,8 \cdot 3,2 \cdot 21,319^{1,05} = 63,6 \text{ люд.} \cdot \text{міс.};$$

$$TM = 2,5 \cdot 63,6^{0,38} = 12,1 \text{ міс.}$$

Методика COCOMO була вдосконалена і отримала назву COCOMO II. Результатом вдосконалення є поява двох стадій оцінки проєкту:

- попередня оцінка на початковій стадії;
- детальна оцінка на стадії після опрацювання архітектури.

Для розрахунку трудомісткості необхідно оцінити чинники масштабу та

множники трудомісткості. Чинники масштабу застосовуються на двох стадіях оцінки проєкту. Кількість і значення множників трудомісткості відрізняються для різних стадій оцінки проєкту.

В таблицях Г.4 – Г.6 наведено опис і значення кожного чиннику масштабу та множників трудомісткості для попередньої та детальної оцінки розробленого проєкту.

Формула оцінки трудомісткості проєкту в люд. · міс. має вигляд:

$$PM = EAF \cdot A \cdot Size^E, \quad (2.6)$$

де EAF – добуток обраних множників трудомісткості – 7 для попередньої оцінки; 17 для детальної оцінки;

A – 2,94 для попередньої оцінки; 2,45 для детальної оцінки;

$Size$ – обсяг програмного продукту в тисячах рядків вихідного коду (KSLOC);

E – обчислюється за формулою:

$$E = B + 0,01 \cdot \sum_{j=1}^5 SF_j, \quad (2.7)$$

де B – 0,91;

SF_j – чинники масштабу.

Час розробки проєкту для обох стадій розраховується за формулою:

$$TM = SCED \cdot C \cdot (PM_{NS})^{D+0,2 \cdot (E-B)}, \quad (2.8)$$

де $SCED$ – значення множника трудомісткості необхідного графіку розроблення проєкту;

C – 3,67;

PM_{NS} – розрахована трудомісткість проєкту без урахування множника;

D – 0,28.

Розрахунок попередньої оцінки на початковій стадії:

$$EAF = 1,00 \cdot 1,00 \cdot 0,83 \cdot 1,00 \cdot 1,00 \cdot 1,10 \cdot 1,00 = 0,9;$$

$$E = 0,91 + 0,01 \cdot 23,58 = 1,14;$$

$$PM = 0,9 \cdot 2,94 \cdot 21,319^{1,14} = 86,5 \text{ люд.} \cdot \text{міс.};$$

$$TM = 1,00 \cdot 3,67 \cdot 86,5^{0,28+0,2 \cdot (1,14-0,91)} = 13,9 \text{ міс.}$$

Розрахунок детальної оцінки на стадії після опрацювання архітектури:

$$EAF = 1,29 \cdot 1,10 \cdot 1,15 \cdot 1,12 \cdot 1,09 \cdot 1,09 \cdot 1,00 \cdot 0,73 \cdot 0,95 \cdot$$

$$\cdot 0,91 \cdot 1,00 \cdot 1,00 \cdot 1,00 \cdot 0,90 \cdot 1,22 \cdot 1,00 = 1,5;$$

$$E = 1,14;$$

$$PM = 1,5 \cdot 2,45 \cdot 21,319^{1,14} = 120,2 \text{ люд.} \cdot \text{міс.};$$

$$TM = 1,00 \cdot 3,67 \cdot 120,2^{0,28+0,2 \cdot (1,14-0,91)} = 15,4 \text{ міс.}$$

Результати розрахунків економічних показників проєкту AGG за різними типами моделей COCOMO показують зростання трудомісткості від 59,6 до 120,2 людино-місяців, а також часу розробки проєкту – від 11,8 до 15,4 календарних місяців. Деталізована модель COCOMO II демонструє найвищі значення, що пояснюється врахуванням обширного спектру чинників і множників трудомісткості.

2.5 Постановка завдання на розробку програмного забезпечення

Проєкт орієнтований на розробку вебзастосунку підбору супермаркету за кошиком продуктів та геолокації користувача в межах системи порівняння цін AGG.

Цільова аудиторія системи включає в себе користувачів, які порівнюють ціни на продукти в різних супермаркетах та активно шукають найвигідніші пропозиції неподалік свого розташування. Такий застосунок може бути корисним для будь-якої групи споживачів, які цінують ефективність, економію часу і грошей при здійсненні покупок у супермаркетах.

Метою розробки є надання зручного вебзастосунку для користувачів, які шукають доступні ціни на продукти в популярних супермаркетах поблизу свого місцезнаходження.

Висновки до розділу

У даному розділі надано варіанти використання системи у вигляді Use case діаграми та їх опис у вигляді таблиць.

Виконано аналіз функціональних та нефункціональних вимог до програмного забезпечення. Для розробки функціональних вимог надано модель вимог у загальному вигляді, табличний опис, а також матрицю трасування вимог.

Проаналізовано економічні показники програмного забезпечення, розрахунок яких здійснено за допомогою конструктивної моделі COCOMO. Результати показали зростання оцінки трудомісткості та часу розробки від базової до проміжної та деталізованої моделі COCOMO II.

За результатами розділу сформовано технічне завдання на розробку програмного забезпечення.

3 КОНСТРУЮВАННЯ ТА РОЗРОБЛЕННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

3.1 Архітектура програмного забезпечення

Розробка програмного забезпечення системи порівняння цін продуктів у супермаркетах ґрунтувалась на клієнт-серверній архітектурі.

Основними компонентами даного архітектурного патерну є набір серверів, набір клієнтів та мережа. Сервери надають інформацію або інші послуги програмам, які звертаються до них. Клієнти – використовують сервіси, що надаються серверами. Мережа – забезпечує взаємодію між клієнтами та серверами.

Отже, розроблюваний, в ході виконання дипломного проєкту, вебзастосунок складається з таких компонентів як Core, Spring Boot Standalone, Spring Scheduling, Spring Repository, Spring Boot Web.

Основою системи є розроблюваний набір інтерфейсів Market API, який дозволяє взаємодіяти універсальним чином з різними типами маркетів.

Кожен маркет має свою специфіку в отриманні інформації про продукти, категорії і т.д. Для кожного з інтегруючих маркетів реалізовано інтерфейс Market API, в якому приховано всю специфіку.

Інші підсистеми застосунку напряду не залежать від специфіки зовнішніх систем, однак залежать від одного інтерфейсу. Тому для підтримки нового супермаркету в системі необхідно реалізувати тільки даний інтерфейс.

Впроваджено підсистему автоматичного оновлення цін та наявності продукту. Це забезпечується завдяки запуску процесу оновлення із заданою періодичністю, користуючись можливостями модулю Spring Scheduling.

Інтеграція з Google Directions API [15] реалізована за допомогою універсального інтерфейсу Directions API, що приховує деталі взаємодії із зовнішнім сервісом, забезпечуючи незалежне отримання необхідної

інформації про маршрути. При необхідності є можливість інтеграції з іншим сервісом, шляхом реалізації даного інтерфейсу.

Інші підсистеми надають класичний MVC паттерн, для рівня представлення яких використовується сучасний серверний механізм Java-шаблонів – Thymeleaf [14].

Також присутній окремий Spring Standalone модуль, який використовується для початкового імпорту даних.

На рисунку 3.1 представлено загальну структуру програмного забезпечення.

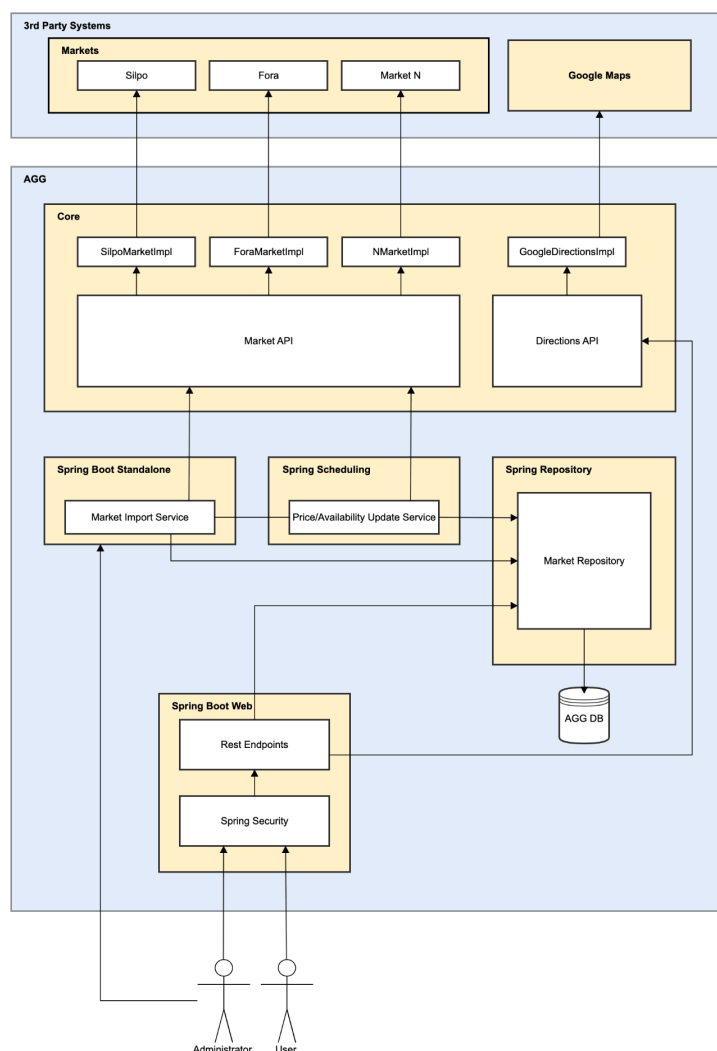


Рисунок 3.1 – Діаграма структури системи AGG

3.2 Архітектурні рішення та обґрунтування вибору засобів розробки

Однією з ключових складових вебзастосунку є інтеграція з Google Directions API. Directions API – це сервіс, який приймає HTTP-запит і повертає інформацію про маршрути між локаціями у форматі JSON або XML.

За допомогою Directions API є можливість отримати маршрути для декількох видів транспорту, таких як транзит, автомобільний, піший або велосипедний. Сервіс пропонує найоптимальніші маршрути, враховуючи наступні фактори:

- час у дорозі;
- відстань;
- кількість поворотів.

Вибір Google Directions API обумовлений його надійністю, точністю, гнучкістю, зручністю інтеграції у Spring Boot-застосунок завдяки наявності детальної документації та бібліотек для Java, а також широким географічним покриттям, що включає інформацію про розташування супермаркетів на всій території України.

При реалізації персоналізованого кошику продуктів використано Session Scope на серверному рівні. Дані кошика зберігаються в межах HTTP-сесії користувача на сервері, забезпечуючи кожному користувачу особистий кошик, який існує протягом його взаємодії з вебзастосунком.

Використання Session Scope обґрунтовано кількома перевагами: легкістю інтеграції завдяки підтримці Spring Boot [16], високою продуктивністю через зберігання даних у пам'яті сесії, можливістю надання індивідуального кошика навіть неавторизованим користувачам, а також посиленою безпекою за рахунок тимчасового зберігання даних, обмеженого тривалістю сесії.

Основним засобом розробки серверної частини є мова програмування

Java зі системою збірки Maven, відомі фреймворки від кампанії Spring та движок шаблонів Java Thymeleaf, який відмінно інтегрується зі Spring. Технологічних стек для клієнтської частини включає HTML, Bootstrap.

3.3 Конструювання програмного забезпечення

Важливий аспект в розробці даного програмного забезпечення є імпортування даних з супермаркетів. Імпортування даних складається з декількох етапів:

- збереження маркету в БД;
- збереження категорій продуктів в БД;
- збереження продуктів в БД;
- зв'язування продуктів маркету з продуктами агрегатору.

3.3.1 Опис структури бази даних

В якості системи управління базами даних використовується MySQL [17]. База даних серверу призначена для зберігання даних про маркет, категорії, продукти, а також про продукти агрегатору. Опис таблиць бази даних наведено у таблиці 3.1. Модель бази даних наведена на рисунку 3.2.

Таблиця 3.1 – Опис таблиць бази даних

Таблиця	Назва поля	Тип даних	Опис
markets	id	bigint	ідентифікаційний номер маркету
	domain_name	varchar(512)	доменне ім'я маркету

Продовження таблиці 3.1

Таблиця	Назва поля	Тип даних	Опис
	logo_url	varchar(1024)	логотип маркету, у вигляді URL
	name	varchar(512)	назва маркету
categories	id	bigint	ідентифікаційний номер категорії
	external_id	varchar(1024)	ідентифікаційний номер категорії з маркету
	is_imported	bit(1)	прапорець, для імпорту даних про категорії
	logo_url	varchar(1024)	логотип категорії, у вигляді URL
	market_id	bigint	ідентифікаційний номер маркету
	name	varchar(512)	назва категорії
	parent_id	bigint	ідентифікаційний номер батьківської категорії
product	id	bigint	ідентифікаційний номер продукту

Продовження таблиці 3.1

Таблиця	Назва поля	Тип даних	Опис
	category_id	bigint	ідентифікаційний номер категорії
	external_id	varchar(256)	ідентифікаційний номер продукту з маркету
	image_url	varchar(256)	картинка продукту, у вигляді URL
	is_available	bit(1)	прапорець наявності продукту в маркеті
	name	varchar(256)	назва продукту
	unit	varchar(128)	одиниця вимірювання продукту
	price	double	вартість продукту
	site_url	varchar(1024)	URL продукту з онлайн-маркету
	aggr_product_id	bigint	ідентифікаційний номер продукту для агрегації
	market_id	bigint	ідентифікаційний номер маркету

Продовження таблиці 3.1

Таблиця	Назва поля	Тип даних	Опис
aggr_products	id	bigint	ідентифікаційний номер агрегованого продукту
	category_id	bigint	ідентифікаційний номер категорії агрегатора
	image_url	varchar(256)	картинка продукту, у вигляді URL
	name	varchar(256)	назва агрегаційного продукту
	unit	varchar(256)	одиниця вимірювання агрегованого продукту

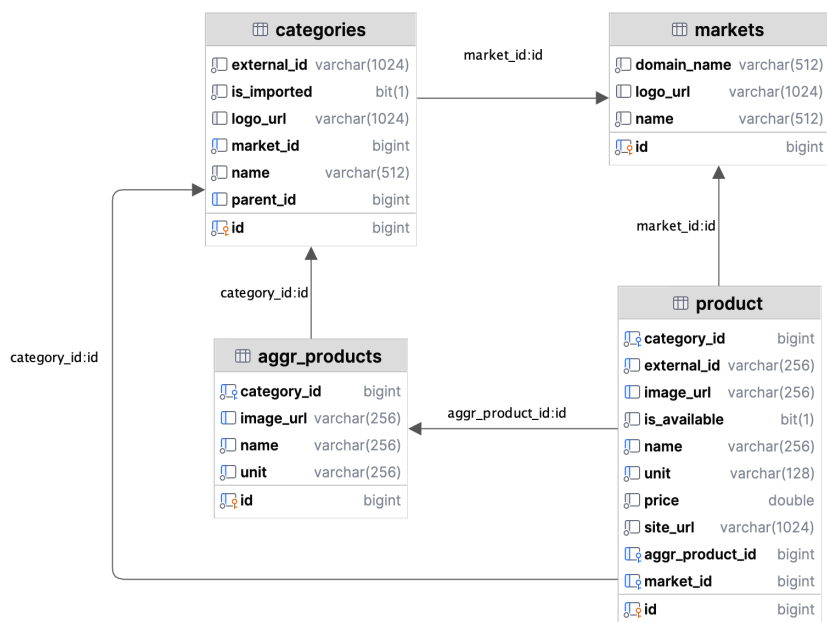


Рисунок 3.2 – Модель бази даних

Створено клас `MarketImportService`, призначений для імпортування та збереження даних з супермаркетів в БД. Опис основних методів даного класу наведено в таблиці 3.2.

Таблиця 3.2 – Опис основних методів класу `MarketImportService`

Назва методу	Сигнатура методу	Опис методу
<code>importMarket</code>	<code>importMarket(MarketApi marketApi)</code>	Забезпечує імпортування даних по API певного маркету.
<code>saveMarket</code>	<code>saveMarket(MarketApi marketApi)</code>	Забезпечує збереження маркету в базу даних системи.
<code>saveCategories</code>	<code>saveCategories(MarketApi marketApi)</code>	Забезпечує збереження категорій певного маркету в базу даних системи.
<code>saveProducts</code>	<code>saveProducts(MarketApi marketApi, boolean isImport)</code>	Забезпечує збереження продуктів з категорій певного маркету.
<code>updatePrice</code>	<code>updatePrice(MarketApi marketApi)</code>	Забезпечує оновлення інформації про вартість та наявність продуктів кожні 6 годин.

Опис утиліт, бібліотек та іншого стороннього програмного забезпечення, що використовується у розробці наведено в таблиці 3.3.

Таблиця 3.3 – Опис утиліт

№ п/п	Назва утиліти	Опис застосування
1	IntelliJ IDEA	Головне середовище розробки програмного забезпечення як клієнтської та серверної частини, так і для взаємодії з базою даних системи.

3.4 Аналіз безпеки даних

В розробленому вебзастосунку реалізовано систему авторизації та автентифікації для адміністратора системи AGG за допомогою Spring Security [18] на основі сервлетів. Підтримка сервлетів базується на фільтрах сервлетів (рис. 3.3). Клієнт надсилає запит до програми, контейнер створює FilterChain, який містить Filter екземпляри і Servlet, які повинні обробити HttpServletRequest, на основі шляху URI запиту.

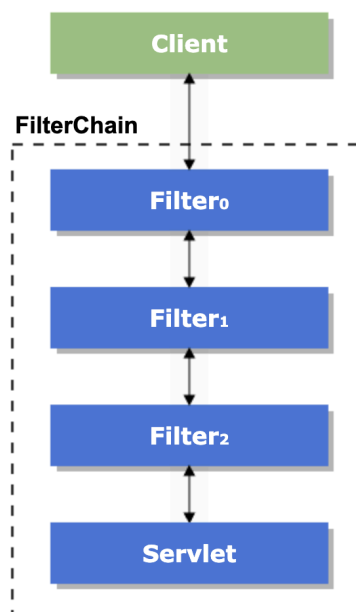


Рисунок 3.3 – Ланцюг фільтрів

Spring Security надає власний вбудований модуль входу для автентифікації адміністратора. Він перевіряє облікові дані адміністратора та забезпечує доступ до системи.

Висновки до розділу

За поставленою, в технічному завданні, задачею розроблено вебзастосунок підбору супермаркету за кошиком продуктів та геолокації користувача в межах системи порівняння цін AGG. Застосунок зберігає дані про наявність продукту та його ціни в супермаркетах, надає можливість користувачам зручного порівняння і додавання в кошик, а також допомагає аналізувати вміст кошику та знайти оптимальний супермаркет поблизу.

Розробка вебзастосунку засновано на клієнт-серверному архітектурному патерні, який складається з сервісу отримання даних з супермаркетів, адміністративного та користувацького простору, баз даних.

Для зберігання імпортованої інформації о маркетах, категоріях, продуктах з супермаркетів та агрегатору спроектовано та реалізовано базу даних з необхідними таблицями.

Описано основні допоміжні засоби для розробки системи, а також проаналізовано систему авторизації та автентифікації для адміністратора системи.

4 АНАЛІЗ ЯКОСТІ ТА ТЕСТУВАННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

4.1 Аналіз якості ПЗ

Як зазначено раніше, розробка програмного забезпечення виконувалась в IntelliJ Idea [19]. Останні версії IDE надають можливість виконання статичного аналізу коду за допомогою Qodana [20].

Qodana перевіряє код на відповідність заздалегідь встановленим критеріям якості:

- загальна кількість проблем;
- ступінь серйозності проблем;
- складність коду.

Результат виконання аналізу якості коду надає інформацію про актуальні проблеми, які не заважають працездатності програмного забезпечення (рис. 4.1 – рис. 4.2).

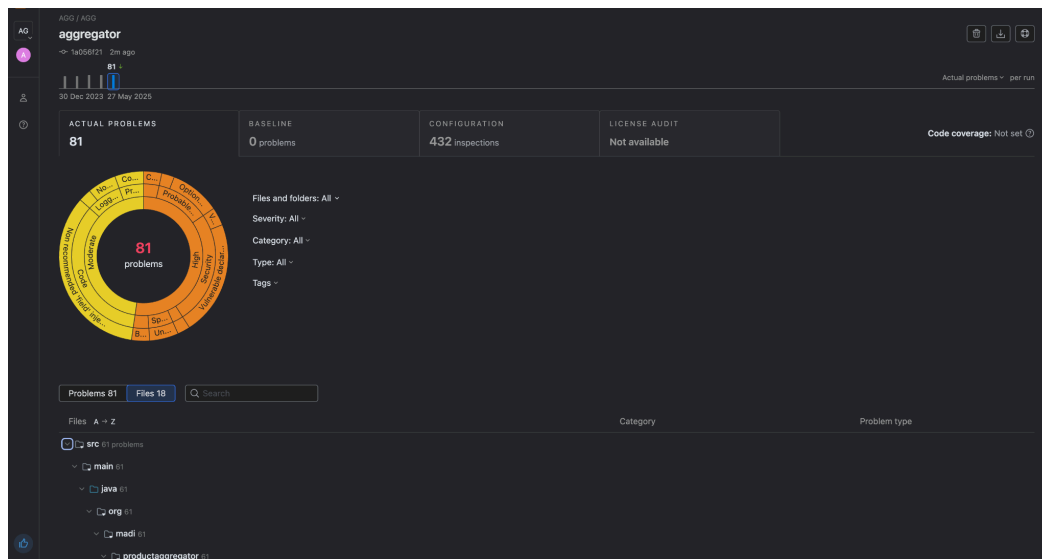


Рисунок 4.1 – Результат аналізу якості коду

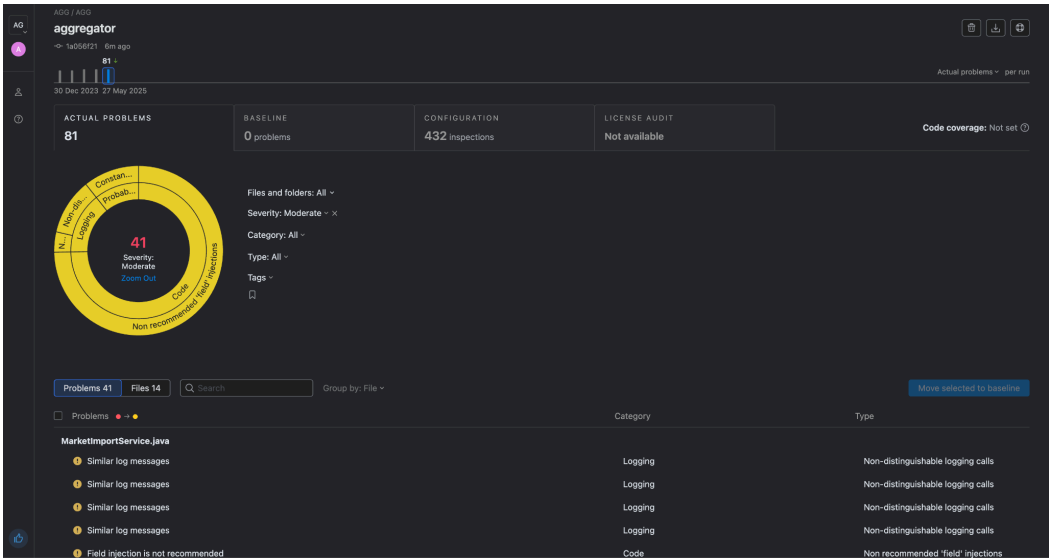


Рисунок 4.2 – Результат аналізу якості коду

4.2 Опис процесів тестування

Тестування виконується згідно документу «Програма та методика тестування».

Було виконане мануальне тестування програмного забезпечення, опис відповідних тестів наведено у таблицях 4.1 – 4.11.

Таблиця 4.1 – Тест 1.1

Тест	Фільтрація по категорії продуктів
Номер тесту	1.1
Початковий стан системи	Користувач перебуває на сайті.
Вхідні дані	—
Опис проведення тесту	Перейдено до сторінки зі списком категорій продуктів. Обрано одну з категорій, натиснувши на її назву. Перенаправлено до сторінки з результатами.

Продовження таблиці 4.1

Очікуваний результат	На сторінці відображаються продукти, що належать до обраної категорії.
Фактичний результат	Відповідає очікуваному результату.

Таблиця 4.2 – Тест 1.2

Тест	Фільтрація по назві продукта
Номер тесту	1.2
Початковий стан системи	Користувач перебуває на сайті.
Вхідні дані	Назва продукту.
Опис проведення тесту	Перейдено до сторінки зі списком продуктів певної категорії. Знайдено поле пошуку за назвою продукту. Введено назву продукту. Натиснуто на іконку з лупою. Отримано результат пошуку.
Очікуваний результат	На сторінці відображаються ті продукти, у назві яких міститься введений текст.
Фактичний результат	Відповідає очікуваному результату.

Таблиця 4.3 – Тест 2.1

Тест	Додавання продукту до кошика продуктів
Номер тесту	2.1
Початковий стан системи	Користувач перебуває на сайті.

Продовження таблиці 4.3

Вхідні дані	—
Опис проведення тесту	Перейдено до сторінки зі списком продуктів. Знайдено продукт, який потрібно додати до кошика продуктів. Натиснуто на іконку з кошиком. Перевірено зміну стану кошика (відображення кількості товарів, оновлення вмісту).
Очікуваний результат	При переході на сторінку кошика продуктів, доданий продукт присутній у списку з коректною назвою, одиницею вимірювання, кількістю (за замовчуванням 1) та цінами з різних супермаркетів. Доступні можливості зміни кількості продукту або його видалення з кошика.
Фактичний результат	Відповідає очікуваному результату.

Таблиця 4.4 – Тест 2.2

Тест	Оновлення загальної вартості кошику по супермаркетах при зміні кількості продукту в кошику
Номер тесту	2.2
Початковий стан системи	Користувач перебуває на сайті.
Вхідні дані	—

Продовження таблиці 4.4

Опис проведення тесту	Перейдено до сторінки з кошиком продуктів. Знайдено продукт, кількість якого потрібно змінити. Запам'ятовано поточну кількість цього продукту, його ціни за одиницю та поточну загальну суму кошика. Змінено кількість продукту на одиницю. Перевірено оновлення загальної вартості кошика продуктів.
Очікуваний результат	При зміні кількості продукту на одиницю, загальна вартість кошика має змінитися на ціну одиниці даного продукту. Перерахунок загальної вартості відбувається коректно для змінених позицій у кошику.
Фактичний результат	Відповідає очікуваному результату.

Таблиця 4.5 – Тест 3.1

Тест	Пошук найближчого супермаркету
Номер тесту	3.1
Початковий стан системи	Користувач перебуває на сайті.
Вхідні дані	Координати місця розташування.
Опис проведення тесту	Перейдено до сторінки з кошиком продуктів. Натиснуто на кнопку «Найближчий магазин». Обрано на карті місце розташування. Натиснуто на кнопку «ОК!». Завантажено карту та список супермаркетів.

Продовження таблиці 4.5

Очікуваний результат	На карті відображаються маркери, що позначають розташування найближчих супермаркетів. Під картою відображається відсортований, за зростанням відстані, список найближчих супермаркетів.
Фактичний результат	Відповідає очікуваному результату.

Таблиця 4.6 – Тест 4.1

Тест	Авторизація адміністратора в систему AGG
Номер тесту	4.1
Початковий стан системи	Адміністратор зареєстрований в системі.
Вхідні дані	Ім'я користувача, пароль.
Опис проведення тесту	Запущено веббраузер та перейдено на сторінку авторизації в систему. Введено вірне ім'я користувача у відповідне поле. Введено вірний пароль у відповідне поле. Натиснуто на кнопку «Вхід». Переконано, що перенаправлення на головну сторінку системи відбувається.
Очікуваний результат	Адміністратор успішно увійшов в систему і отримав доступ до особистого кабінету.
Фактичний результат	Відповідає очікуваному результату.

Таблиця 4.7 – Тест 4.2

Тест	Авторизація адміністратора в систему AGG
Номер тесту	4.2
Початковий стан системи	Адміністратор зареєстрований в системі.
Вхідні дані	Ім'я користувача, пароль.
Опис проведення тесту	Запущено веббраузер та перейдено на сторінку авторизації в систему. Введено невірне ім'я користувача у відповідне поле. Введено невірний пароль у відповідне поле. Натиснуто на кнопку «Вхід». Переконано, що система відображає повідомлення про некоректно введені дані та не надає доступ до особистого кабінету.
Очікуваний результат	Отримання повідомлення про некоректно введені дані.
Фактичний результат	Відповідає очікуваному результату.

Таблиця 4.8 – Тест 5.1

Тест	Пошук категорій продуктів агрегатору
Номер тесту	5.1
Початковий стан системи	Адміністратор перебуває в системі.
Вхідні дані	—

Продовження таблиці 4.8

Опис проведення тесту	На головній сторінці обрано та натиснуто на пункт меню «Категорії». Переконано, що перенаправлення на сторінку з переліком категорій продуктів відбулося. В полі пошуку введено назву потрібної категорії. Отримано результат пошуку.
Очікуваний результат	Пошук категорії продуктів агрегатору успішно відображає результат за введеним запитом.
Фактичний результат	Відповідає очікуваному результату.

Таблиця 4.9 – Тест 6.1

Тест	Додавання нової категорії продуктів до агрегатору
Номер тесту	6.1
Початковий стан системи	Адміністратор перебуває в системі.
Вхідні дані	Назва категорії, батьківська категорія (необов'язково).

Продовження таблиці 4.9

Опис проведення тесту	На головній сторінці обрано та натиснуто пункт меню «Категорії». Переконано, що перенаправлення на сторінку з переліком категорій продуктів відбулося. Натиснуто на кнопку «Додати». Переконано, що перенаправлення відбулося на сторінку з формою для введення даних про нову категорію. В відповідне поле введено назву категорії. З випадаючого списку категорій обрано необхідну батьківську категорію. Натиснуто на кнопку «Зберегти». Система повідомила про успішне додавання категорії.
Очікуваний результат	В систему агрегатору успішно додано нову категорію продуктів.
Фактичний результат	Відповідає очікуваному результату.

Таблиця 4.10 – Тест 6.2

Тест	Редагування категорії продуктів агрегатору
Номер тесту	6.2
Початковий стан системи	Адміністратор перебуває в системі.
Вхідні дані	Назва категорії, батьківська категорія (необов'язково).

Продовження таблиці 4.10

Опис проведення тесту	На головній сторінці обрано та натиснуто пункт меню «Категорії». Переконано, що перенаправлення на сторінку з переліком категорій продуктів відбулося. Обрано та натиснуто на назву категорії, яку потрібно редагувати. Переконано, що перенаправлення відбулося на сторінку для редагування категорії. Поля для введення назви та вибору батьківської категорії містять поточну інформацію. Змінено назву категорії. Натиснуто на кнопку «Зберегти». Система повідомила про успішне збереження даних категорії.
Очікуваний результат	В системі агрегатору успішно оновлено інформацію про назву категорії продуктів.
Фактичний результат	Відповідає очікуваному результату.

Таблиця 4.11 – Тест 6.3

Тест	Видалення категорії продуктів агрегатору
Номер тесту	6.3
Початковий стан системи	Адміністратор перебуває в системі.
Вхідні дані	—

Продовження таблиці 4.11

Опис проведення тесту	На головній сторінці обрано та натиснуто пункт меню «Категорії». Переконано, що перенаправлення на сторінку з переліком категорій продуктів відбулося. Обрано та натиснуто на назву категорії, яку потрібно видалити. Переконано, що перенаправлення відбулося на сторінку для редагування категорії. Натиснуто на кнопку «Видалити». На екрані з'явилося діалогове вікно для підтвердження видалення категорії. Натиснувши на кнопку «Так», категорія видаляється, система повідомляє про видалення категорії.
Очікуваний результат	В системі агрегатору видалено категорію продуктів.
Фактичний результат	Відповідає очікуваному результату.

4.3 Опис контрольного прикладу

Опис контрольного прикладу вебзастосунку зосереджено на його ключовому функціоналі і користувацькому інтерфейсі.

Для користувача:

– пошук продукту. Фільтрація по категорії продуктів:

1) очікуваний результат: список продуктів, відфільтрованих за категорією (рис. 4.3);

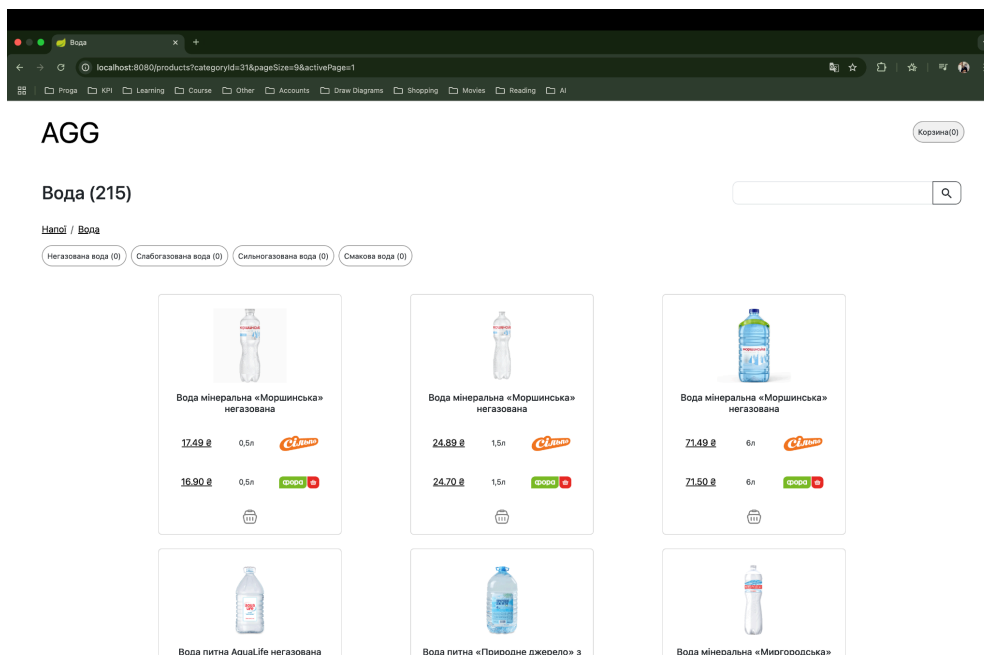


Рисунок 4.3 – Список продуктів, відфільтрованих за категорією

— пошук продукту. Фільтрація по назві продукта:

1) вхідні дані: назва продукту;

2) очікуваний результат: список продуктів, відфільтрованих за назвою (рис. 4.4);

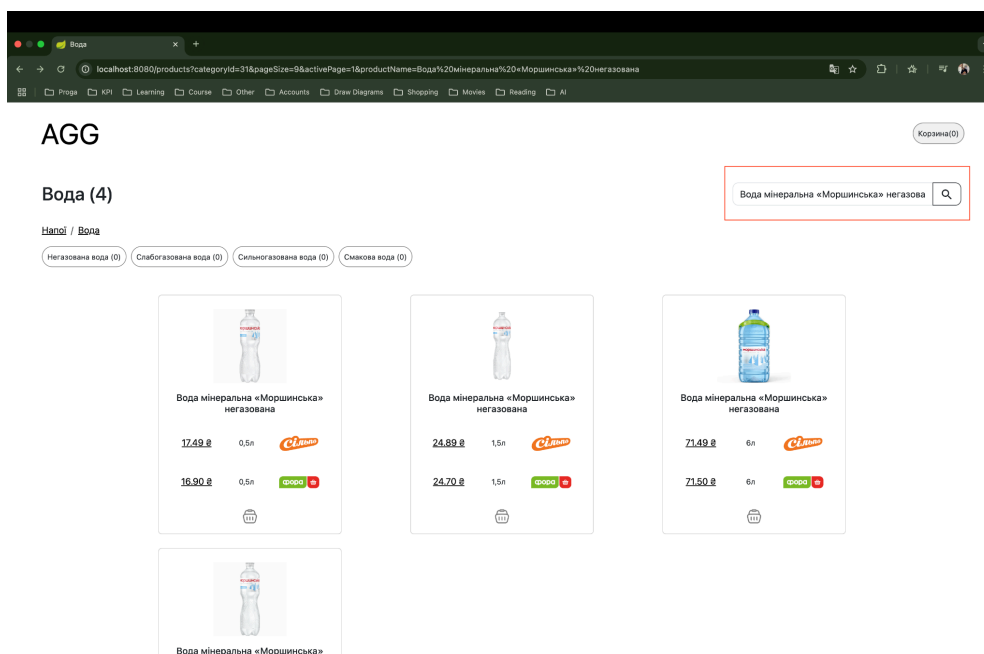


Рисунок 4.4 – Список продуктів, відфільтрованих за назвою

- формування кошику продуктів. Додавання продукту до кошика продуктів:

1) очікуваний результат: продукт успішно додано до кошика продуктів (рис. 4.5 – 4.6);

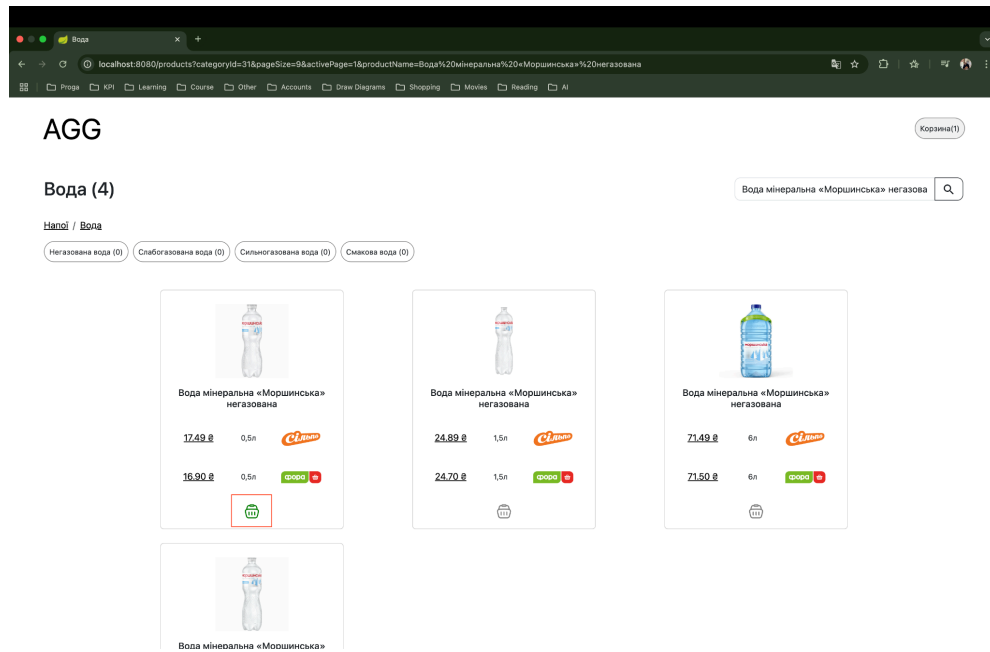


Рисунок 4.5 – Додавання продукту до кошика продуктів

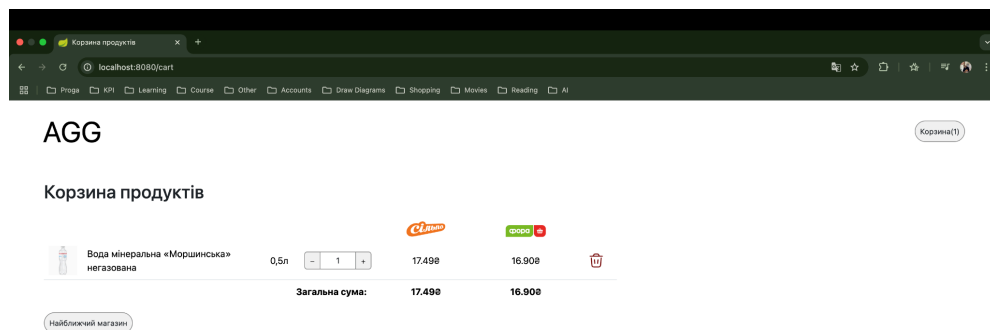


Рисунок 4.6 – Формування кошику продуктів

- оновлення загальної вартості кошику продуктів по супермаркетах:
- 1) очікуваний результат: перерахунок загальної вартості кошику продуктів при зміні кількості одиниць продуктів (рис. 4.7);

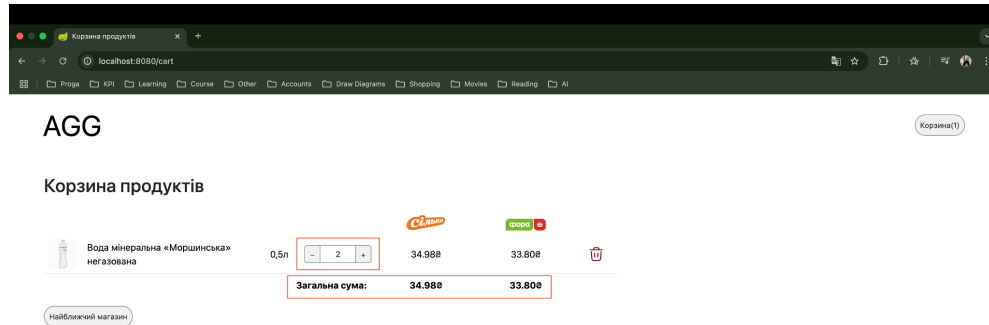


Рисунок 4.7 – Загальна вартість кошику продуктів по супермаркетах

– пошук найближчого супермаркету:

- 1) вхідні дані: координати місця розташування користувача;
- 2) очікуваний результат: карта з відображенням найближчих супермаркетів, список, відсортованих за зростанням відстані, супермаркетів (рис. 4.8).

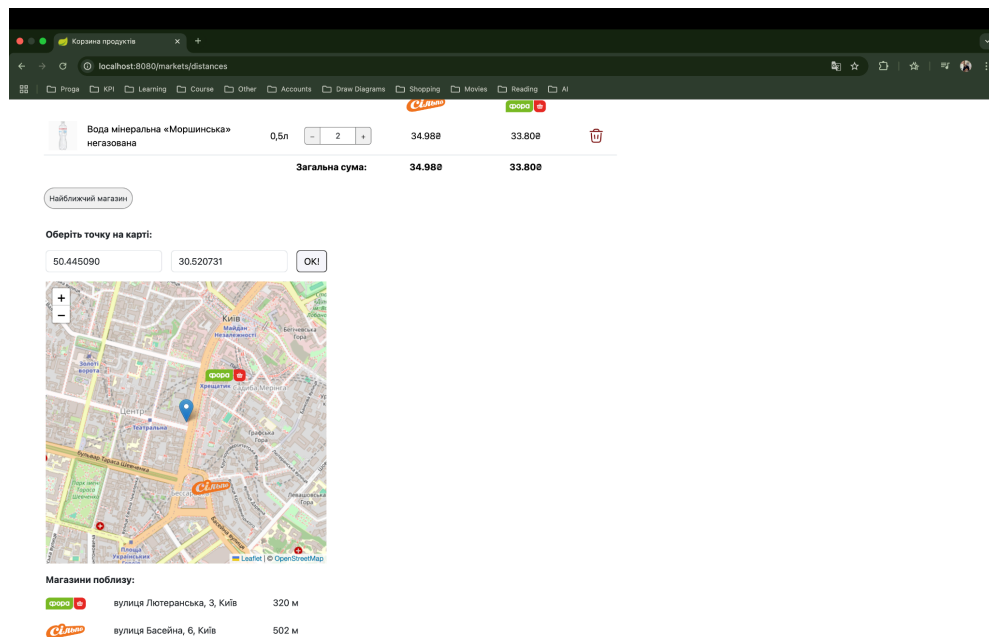


Рисунок 4.8 – Найближчий супермаркет

Для адміністратора системи:

– вхід в систему (рис. 4.9):

- 1) вхідні дані: ім'я користувача – admin, пароль – admin;

2) очікуваний результат: головна сторінка системи (рис. 4.10);

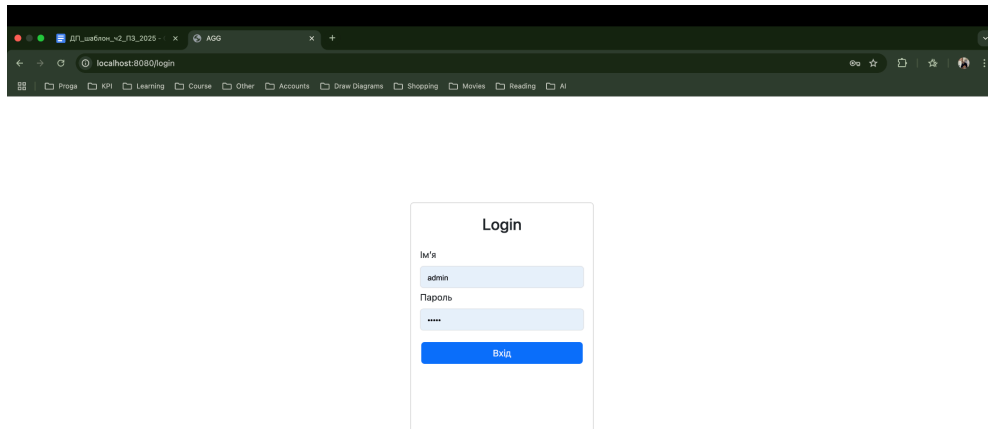


Рисунок 4.9 – Вхід в систему

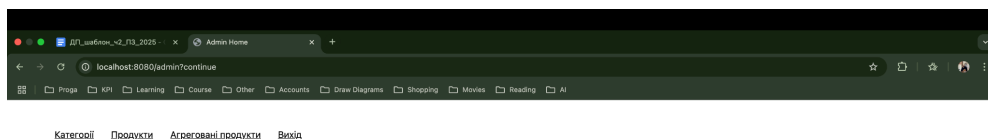


Рисунок 4.10 – Головна сторінка системи

- додавання нової категорії продуктів агрегатору (рис. 4.11):
 - 1) вхідні дані: назва категорії, батьківська категорія;
 - 2) очікуваний результат: повідомлення про успішне збереження нової категорії (рис. 4.12);

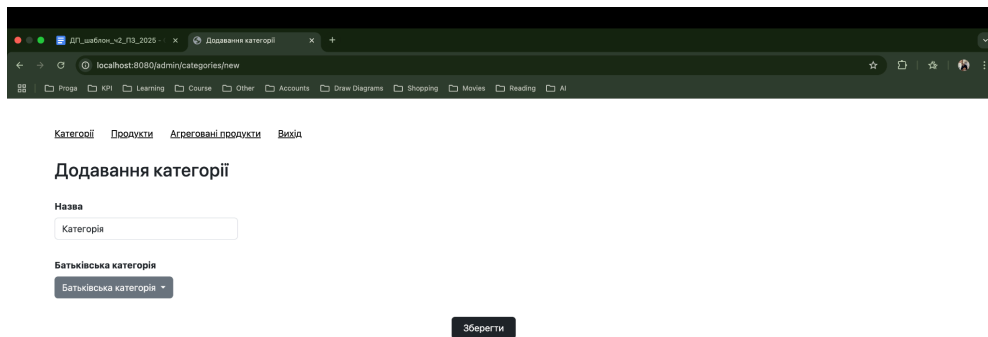


Рисунок 4.11 – Додавання нової категорії продуктів

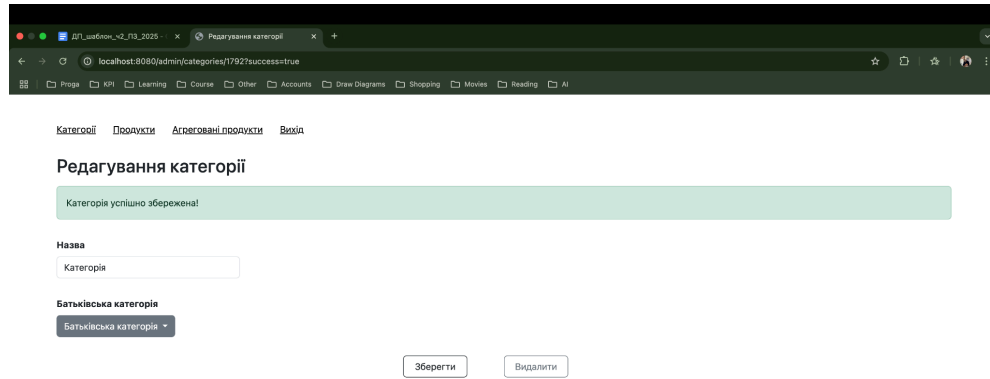


Рисунок 4.12 – Повідомлення про успішне збереження категорії продуктів

– редагування категорії продуктів агрегатору (рис. 4.13):

- 1) вхідні дані: назва категорії, батьківська категорія;
- 2) очікуваний результат: повідомлення про успішне збереження змін категорії (рис. 4.14);

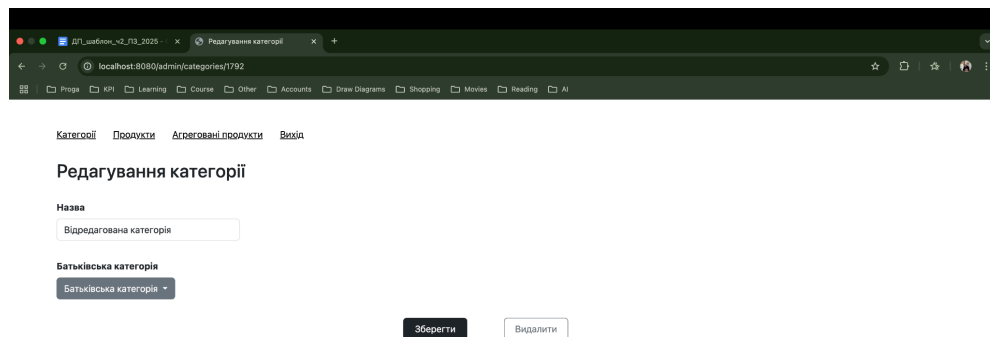


Рисунок 4.13 – Редагування категорії продуктів

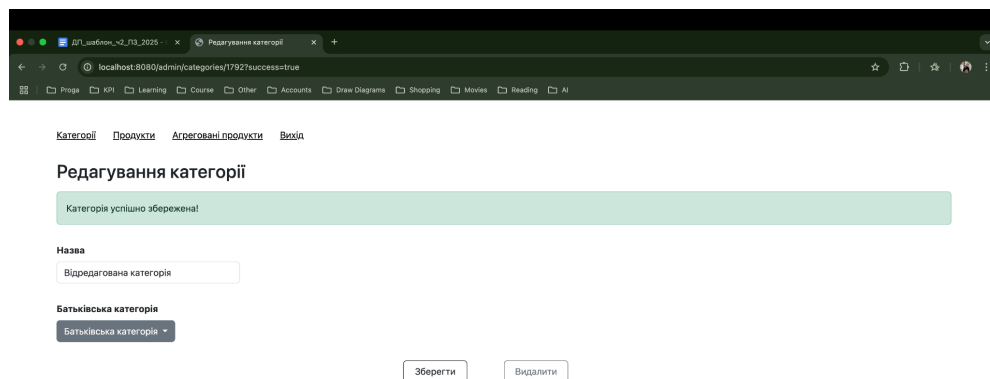


Рисунок 4.14 – Повідомлення про успішне збереження змін

– видалення категорії продуктів агрегатору:

- 1) очікуваний результат: діалогове вікно для підтвердження видалення (рис. 4.15); повідомлення про неможливе видалення категорії (рис. 4.16);

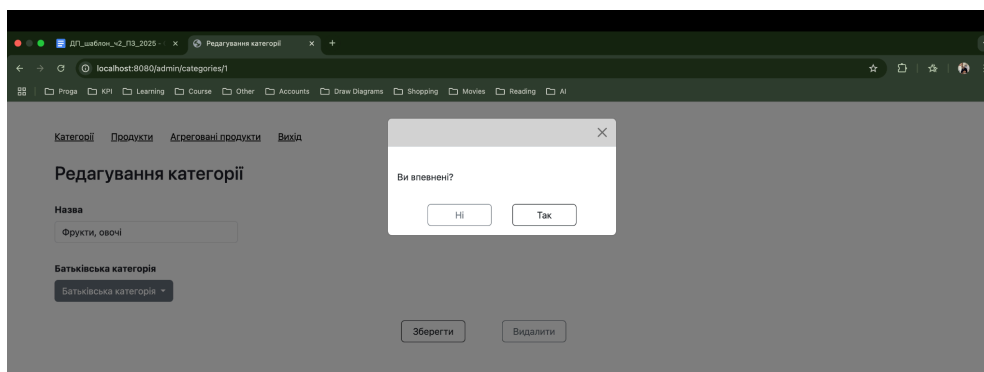


Рисунок 4.15 – Діалогове вікно для підтвердження видалення

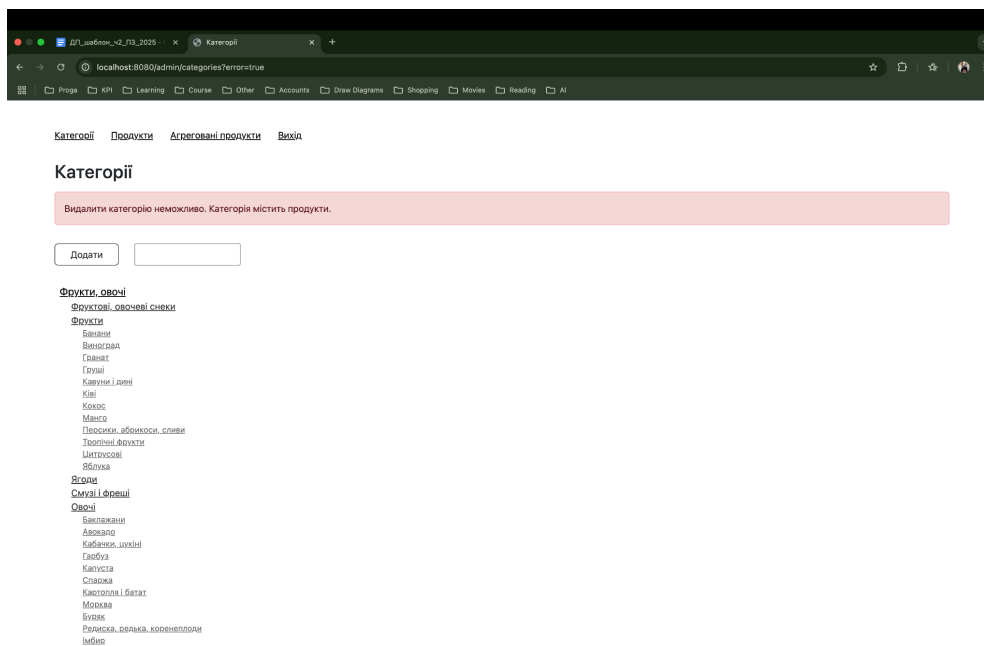


Рисунок 4.16 – Повідомлення про неможливе видалення категорії продуктів

– агрегація обраних продуктів з супермаркетів (рис. 4.17). Додавання нового продукту в агрегатор (рис. 4.18):

- 1) вхідні дані: назва продукту, категорія, картинка;
- 2) очікуваний результат: повідомлення про успішне зв'язування

продуктів (рис. 4.19);

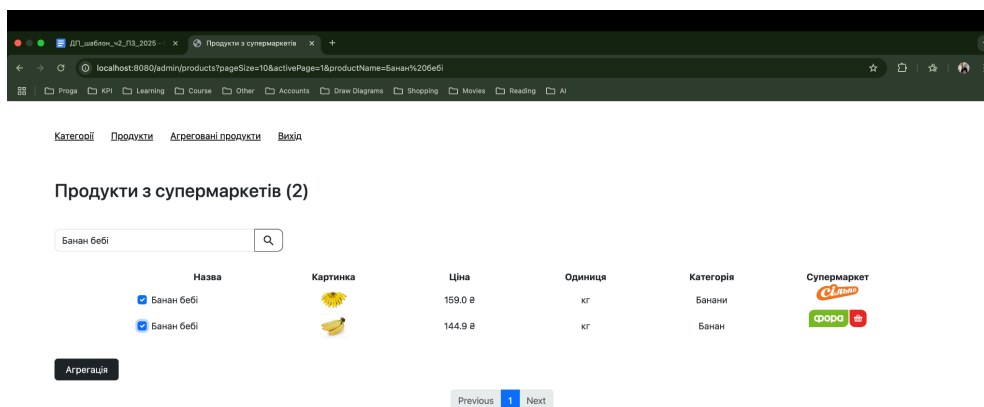


Рисунок 4.17 – Агрегація обраних продуктів з супермаркетів

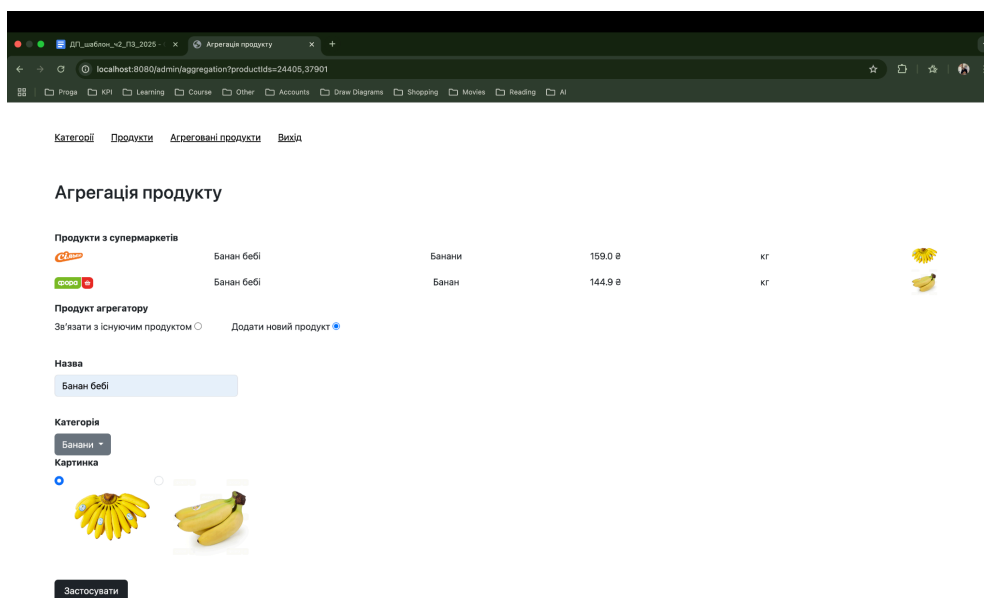
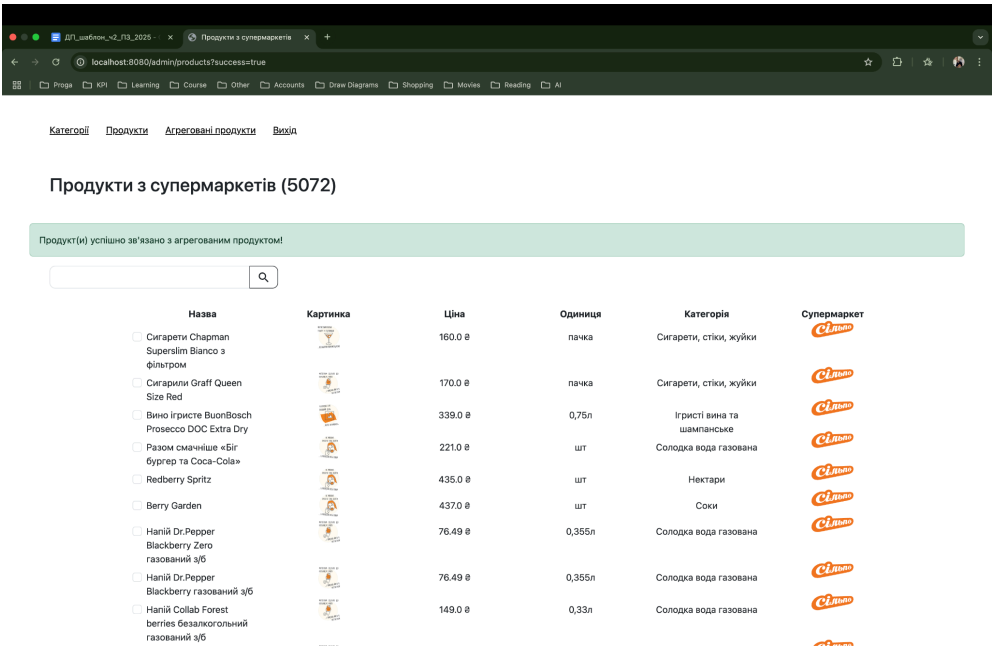


Рисунок 4.18 – Додавання нового продукту в агрегатор



- Рисунок 4.19 – Повідомлення про успішне зв’язування продуктів
- відв’язування продукту супермаркету від продукту агрегатора (рис. 4.20):
- 1) очікуваний результат: повідомлення про успішне оновлення продукту (рис. 4.21).

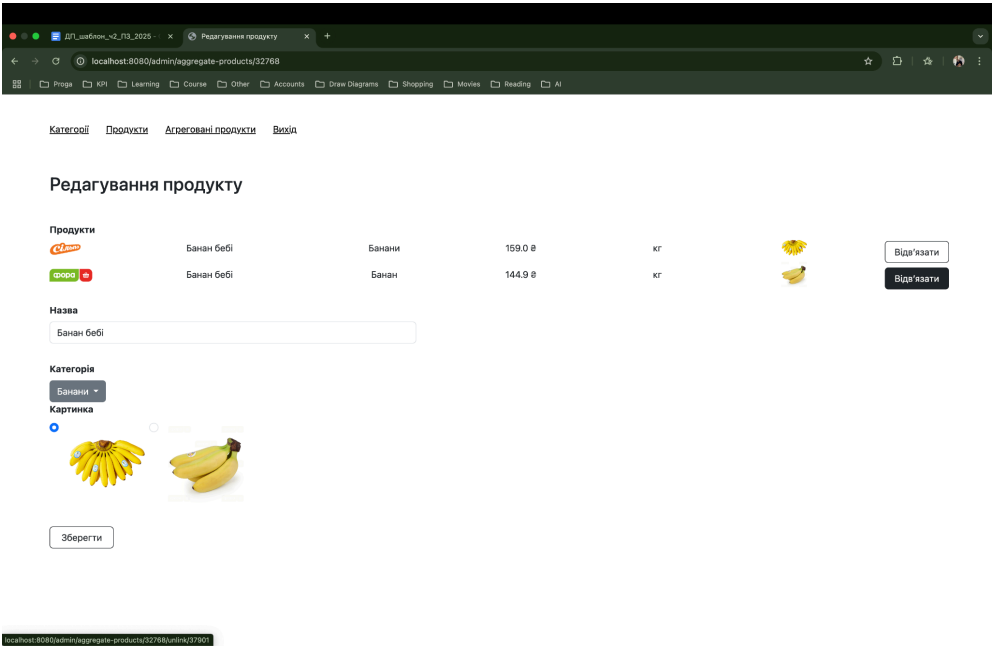


Рисунок 4.20 – Відв’язування продукту супермаркету від продукту агрегатора

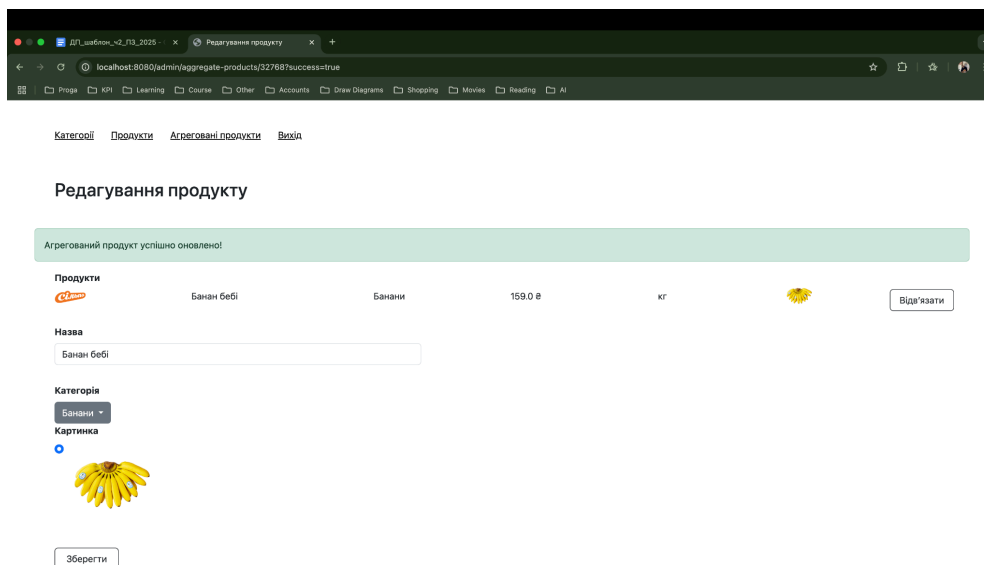


Рисунок 4.21 – Повідомлення про успішне оновлення продукту агрегатору

Висновки до розділу

В даному розділі виконано аналіз якості коду розробленого програмного забезпечення, результати якого вказують на наявність проблем, які не впливають на функціональність вебзастосунку.

Крім того, впроваджено ряд тест-кейсів, які дозволяють переконатися в працездатності ключового функціоналу програмного забезпечення. Тест-кейси покривають декілька модулів, а саме: фільтрація по категорії та назві продуктів, додавання продукту до кошика продуктів, оновлення загальної вартості кошику по супермаркетах при зміні кількості продукту в кошику, пошук найближчого супермаркету, авторизація адміністратора в систему AGG, додавання та редагування категорій агрегатору.

Описано контрольний приклад роботи вебзастосунку, який допоможе користувачам освоїти його ключові можливості.

5 РОЗГОРТАННЯ ТА СУПРОВІД ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

5.1 Розгортання програмного забезпечення

Для розгортання вебзастосунку, підбору супермаркету за кошиком продуктів та місцем розташування, AGG потрібно виконати наступні кроки:

- встановити Git [21] (рис. 5.1):

```
apt install git -y
```

```
root@root:~# apt install git -y
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
git is already the newest version (1:2.34.1-1ubuntu1.12).
git set to manually installed.
0 upgraded, 0 newly installed, 0 to remove and 11 not upgraded.
```

Рисунок 5.1 – Встановлення Git

- встановити MySQL сервер (рис. 5.2):

```
apt install mysql-server -y
```

```
root@root:~# apt install mysql-server -y
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  libfcgi-fast-perl libfcgi-pm-perl libclone-perl libencode-locale-perl
  libevent-pthreads-2.1-7 libfcgi-bin libfcgi-perl libfcgi0ldbl
  libhtml-parser-perl libhtml-tagset-perl libhtml-template-perl
  libhttp-date-perl libhttp-message-perl libio-html-perl
  liblwp-mediatypes-perl libmecab2 libprotobuf-lite23 libtimedate-perl
  liburi-perl mecab-ipadic mecab-ipadic-utf8 mecab-utils mysql-client-8.0
  mysql-client-core-8.0 mysql-common mysql-server-8.0 mysql-server-core-8.0
Suggested packages:
  libdata-dump-perl libipc-sharedcache-perl libbusiness-isbn-perl libwww-perl
  mailx tinyca
```

Рисунок 5.2 – Встановлення MySQL сервер

- створити базу даних (рис. 5.3):

```
mysql -e "create database aggDB"
```

```
root@root:~# mysql -e "create database aggDB"
```

Рисунок 5.3 – Створення бази даних

- змінити обліковий запис MySQL (рис. 5.4):

```
mysql -e "ALTER USER 'root'@'localhost' IDENTIFIED WITH
mysql_native_password BY 'rootroot';"
```

```
root@root:~# mysql -e "ALTER USER 'root'@'localhost' IDENTIFIED WITH mysql_native_password BY 'rootroot';"
```

Рисунок 5.4 – Зміна облікового запису MySQL

- встановити JDK 17 (рис. 5.5):
`apt install openjdk-17-jre --fix-missing -y`

```
root@root:~# apt install openjdk-17-jre --fix-missing -y
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  adwaita-icon-theme alsa-topology-conf alsa-ucm-conf at-spi2-core
  ca-certificates-java dconf-gsettings-backend dconf-service fontconfig
  fonts-dejavu-extra gsettings-desktop-schemas gtk-update-icon-cache
  hicolor-icon-theme humanity-icon-theme java-common libasound2
  libasound2-data libatk-bridge2.0-0 libatk-wrapper-java
  libatk-wrapper-java-jni libatk1.0-0 libatk1.0-data libatspi2.0-0
  libavahi-client3 libavahi-common-data libavahi-common3 libcairo-gobject2
  libcairo2 libcups2 libdatrie1 libdconf1 libfontenc1 libgail-common libgail18
```

Рисунок 5.5 – Встановлення JDK 17

- клонувати репозиторій [22] (рис. 5.6):
`git clone https://github.com/madinaadzhi/aggregator.git`

```
root@root:~# git clone https://github.com/madinaadzhi/aggregator.git
Cloning into 'aggregator'...
remote: Enumerating objects: 1041, done.
remote: Counting objects: 100% (1041/1041), done.
remote: Compressing objects: 100% (391/391), done.
remote: Total 1041 (delta 491), reused 1014 (delta 474), pack-reused 0 (from 0)
Receiving objects: 100% (1041/1041), 5.50 MiB | 17.81 MiB/s, done.
Resolving deltas: 100% (491/491), done.
```

Рисунок 5.6 – Клонування репозиторію

- змінити директорію (рис. 5.7):
`cd aggregator`

```
root@root:~# cd aggregator
```

Рисунок 5.7 – Зміна директорії

- зібрати Maven проєкт (рис. 5.8 – рис. 5.9):
`./mvnw package`

```
root@root:~/aggregator# ./mvnw package
Warning: JAVA_HOME environment variable is not set.
[INFO] Scanning for projects...
Downloading from central: https://repo.maven.apache.org/maven2/org/springframework/boot/spring-boot-starter-parent/3.0.5/spring-boot-starter-parent-3.0.5.pom
Downloaded from central: https://repo.maven.apache.org/maven2/org/springframework/boot/spring-boot-starter-parent/3.0.5/spring-boot-starter-parent-3.0.5.pom (12 kB at 65 kB/s)
Downloading from central: https://repo.maven.apache.org/maven2/org/springframework/boot/spring-boot-dependencies/3.0.5/spring-boot-dependencies-3.0.5.pom
Downloaded from central: https://repo.maven.apache.org/maven2/org/springframework/boot/spring-boot-dependencies/3.0.5/spring-boot-dependencies-3.0.5.pom (93 kB at 2.4 MB/s)
Downloading from central: https://repo.maven.apache.org/maven2/io/zipkin/brave/brave-bom/5.14.1/brave-bom-5.14.1.pom
Downloaded from central: https://repo.maven.apache.org/maven2/io/zipkin/brave/brave-bom/5.14.1/brave-bom-5.14.1.pom (12 kB at 735 kB/s)
Downloading from central: https://repo.maven.apache.org/maven2/io/zipkin/reporter2/zipkin-reporter-bom/2.16.3/zipkin-reporter-bom-2.16.3.pom
Downloaded from central: https://repo.maven.apache.org/maven2/io/zipkin/reporter2/zipkin-reporter-bom/2.16.3/zipkin-reporter-bom-2.16.3.pom (6.9 kB at 492 kB/s)
Downloading from central: https://repo.maven.apache.org/maven2/com/datastax/oss/java-driver-bom/4.15.0/java-driver-bom-4.15.0.pom
Downloaded from central: https://repo.maven.apache.org/maven2/com/datastax/oss/java-driver-bom/4.15.0/java-driver-bom-4.15.0.pom (4.1 kB at 296 kB/s)
Downloading from central: https://repo.maven.apache.org/maven2/io/dropwizard/metrics/metrics-bom/4.2.18/metrics-bom-4.2.18.pom
Downloaded from central: https://repo.maven.apache.org/maven2/io/dropwizard/metrics/metrics-bom/4.2.18/metrics-bom-4.2.18.pom (7.5 kB at 580 kB/s)
```

Рисунок 5.8 – Збірка Maven проєкту

вебзастосунку AGG.

При розробці серверної частини використовується потужна база даних MySQL для зберігання інформації про продукти, ціни, наявність та інші дані. Реалізовано механізми для отримання інформації про актуальні ціни та наявність продуктів через API.

При розробці клієнтської частини вебзастосунку вдосконалено зручний та інтуїтивно зрозумілий інтерфейс, який дозволяє користувачам легко ознайомитись з необхідною інформацією про продукти.

Для адміністратора системи забезпечено надійний механізм для контролю доступу до функціоналу за допомогою авторизації.

У випадку оновлення програмного забезпечення, наприклад, внесення змін в інтерфейс клієнтської частини або додавання нового функціоналу серверної частини, використовується система контролю версій Git для відстеження та керування кодом розробленого програмного забезпечення.

В результаті такого підходу користувачі отримують безпечний, стабільний та завжди оновлений вебзастосунок, а розробники можуть швидко впроваджувати новий функціонал та виправляти помилки. Це сприяє покращенню користувацького досвіду та забезпеченню безпеки та стабільності розробки.

Висновки до розділу

В даному розділі розглянуто покрокову інструкцію щодо розгортання розробленого програмного забезпечення підбору супермаркету за кошиком продуктів та місцем розташування.

Для розгортання вебзастосунку необхідно встановити Git, встановити MySQL сервер, створити базу даних, змінити обліковий запис MySQL, встановити JDK 17, клонувати репозиторій, перейти до необхідної директорії,

зібрати Maven проект, запустити вебзастосунок.

Для тестування розгортання програмного забезпечення використовувалась хмарна інфраструктура VULTR. Обрано Server Cloud Compute, CPU & Storage Technology – Intel Regular Performance, Server Location – Франкфурт, операційну систему Ubuntu 22.04 LTS x64, розмір серверу 25 Гб SSD.

Надано інформацію щодо підтримки програмного забезпечення у разі внесення деяких змін в розробку вебзастосунку.

ВИСНОВКИ

Розроблено вебзастосунок підбору супермаркету за кошиком продуктів та місцем розташування. Даний застосунок дозволяє оптимізувати процес покупок, враховуючи як цінову політику, так і зручність розташування супермаркетів.

Мета дипломного проєкту – надання зручного застосунку, що дозволить користувачам ефективно підбирати супермаркети на основі їхнього кошика продуктів та місця розташування – була досягнута. Розроблений вебзастосунок повністю відповідає поставленим вимогам і демонструє достатньо високий ступінь працездатності та зручності у використанні.

Виконано аналіз предметної області та існуючих рішень. Здійснено моделювання бізнес-процесів, сформовано функціональні та нефункціональні вимоги. Розраховано економічні показники програмного забезпечення за допомогою конструктивної моделі СОСОМО. Спроєктовано ефективну і надійну архітектуру програмного забезпечення, а також виконано його розробку, конструювання та аналіз безпеки даних. Проведено аналіз якості та тестування створеного вебзастосунку. Розроблено покрокову інструкцію щодо розгортання та супроводу програмного забезпечення.

Доцільність продовження розробки в даній предметній області є очевидною. Подальші напрямки розвитку включають: розширення функціоналу, збільшення бази даних супермаркетів та продуктів, впровадження елементів штучного інтелекту для персоналізованих рекомендацій тощо. Перспективи розвитку дозволяють створити повноцінний та конкурентоспроможний продукт, здатний стати затребуваним сервісом для широкого кола споживачів.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Перевірка ціни [Електронний ресурс] -
<https://play.google.com/store/apps/details?id=com.pricecheckscanner>.
2. Amazon [Електронний ресурс] - <https://www.amazon.com/>.
3. Camelcamelcamel [Електронний ресурс] -
<https://uk.camelcamelcamel.com/>.
4. Geographic Information System (GIS) [Електронний ресурс] -
<https://www.esri.com/en-us/what-is-gis/overview>.
5. Hotline Smart Shopping [Електронний ресурс] - <https://hotline.ua/>.
6. E-Katalog [Електронний ресурс] - <https://ek.ua/ua/>.
7. Layered Architecture Pattern in Java: Building Scalable and Maintable Applications [Електронний ресурс] -
<https://java-design-patterns.com/patterns/layered-architecture/>.
8. Microservices Design Patterns [Електронний ресурс] -
<https://www.geeksforgeeks.org/microservices-design-patterns/>.
9. Client-Server Architecture - System Design [Електронний ресурс] -
<https://www.geeksforgeeks.org/client-server-architecture-system-design/>.
10. Spring Framework [Електронний ресурс] -
<https://spring.io/projects/spring-framework>.
11. Spring Boot [Електронний ресурс] - <https://spring.io/projects/spring-boot>.
12. Нефункціональні вимоги: приклади, типи, підходи [Електронний ресурс] -
<https://training.qatestlab.com/blog/technical-articles/non-functional-requirements-examples-types-approaches/>.
13. Сидоров М.О., Родіонов П.Ю., Марченко О.І. Економіка ІТ-індустрії та підприємництво : практикум : навч. посіб. Київ : КПІ ім. Ігоря Сікорського, 2024. 100 с.

14. Thymeleaf [Електронний ресурс] - <https://www.thymeleaf.org/index.html>.
15. Directions API (Legacy) overview [Електронний ресурс] - <https://developers.google.com/maps/documentation/directions/overview>.
16. Quick Guide to Spring Bean Scopes [Електронний ресурс] - <https://www.baeldung.com/spring-bean-scopes>.
17. MySQL Documentation [Електронний ресурс] - <https://www.mysql.com/>.
18. Spring Security [Електронний ресурс] - <https://spring.io/projects/spring-security>.
19. IntelliJ IDEA JetBrains IDEs [Електронний ресурс] - <https://www.jetbrains.com/idea/>.
20. Qodana [Електронний ресурс] - <https://www.jetbrains.com/qodana/>.
21. Вступ – Про систему контролю версій [Електронний ресурс] - <https://git-scm.com/book/uk/v2/Вступ-Про-систему-контролю-версій>.
22. Репозиторій проекту [Електронний ресурс] - <https://github.com/madinaadzhi/aggregator.git>.

ДОДАТКИ

ДОДАТОК А. ЗВІТ ПОДІБНОСТІ

Дата звіту 5/30/2025
Дата редагування 5/30/2025

Документ прийнятий

Звіт подібності

метадані

Назва організації

National Technical University of Ukraine Igor Sikorskyi Kyiv Politech Institute

Заголовок

ІП-15_Аджигельдісва_ПЗ

Автор

Науковий керівник / Експерт

АджигельдісваКовтунець О.В.

підрозділ

ФІОТ, К-а інформатики та програмної інженерії

Обсяг знайдених подібностей

Коефіцієнт подібності визначає, який відсоток тексту по відношенню до загального обсягу тексту було знайдено в різних джерелах. Зверніть увагу, що високі значення коефіцієнта не автоматично означають плагіат. Звіт має аналізувати компетентна / уповноважена особа.

14.64%

14.64%

КП 1

10

Довжина фрази для коефіцієнта подібності 2

6373

Кількість слів

50587

Кількість символів

Тривога

У цьому розділі ви знайдете інформацію щодо текстових спотворень. Ці спотворення в тексті можуть говорити про МОЖЛИВІ маніпуляції в тексті. Спотворення в тексті можуть мати навмисний характер, але частіше характер технічних помилок при конвертації документа та його збереженні, тому ми рекомендуємо вам підходити до аналізу цього модуля відповідально. У разі виникнення запитань, просимо звертатися до нашої служби підтримки.

Заміна букв		1
Інтервали		0
Мікропробіли		0
Білі знаки		0
Парафрази (SmartMarks)		70

Подібності за списком джерел

Нижче наведений список джерел. В цьому списку є джерела із різних баз даних. Копіє тексту означає в якому джерелі він був знайдений. Ці джерела і значення Коефіцієнту Подібності не відображають прямого плагіату. Необхідно відкрити кожне джерело і проаналізувати зміст і правильність оформлення джерела.

10 найдовших фраз

Копіє тексту

ПОРЯДКОВИЙ НОМЕР	НАЗВА ТА АДРЕСА ДЖЕРЕЛА URL (НАЗВА БАЗИ)	КІЛЬКІСТЬ ІДЕНТИЧНИХ СЛІВ (ФРАГМЕНТІВ)	
1	ІП-15_Борисик_ПЗ 5/27/2025 National Technical University of Ukraine Igor Sikorskyi Kyiv Politech Institute (ФІОТ, К-а інформатики та програмної інженерії)	63 0.99 %	
2	https://ela.kpi.ua/server/api/core/bitstreams/61f75747-3203-4f93-8e68-9698fbec074/content	35 0.55 %	

3	Вебзастосунок з інтегрованим вебсервісом для тестування штучного інтелекту вирішення NP-Complete задачі проходження Саперу 3/16/2025 National Technical University of Ukraine Igor Sikorskyi Kyiv Politech Institute (National Technical University of Ukraine Igor Sikorskyi Kyiv Politech Institute)	35 0.55 %	
4	ІП-15_Борисик_ПЗ 5/27/2025 National Technical University of Ukraine Igor Sikorskyi Kyiv Politech Institute (ФІОТ, К-а інформатики та програмної інженерії)	32 0.50 %	
5	Вебзастосунок з інтегрованим вебсервісом для тестування штучного інтелекту вирішення NP-Complete задачі проходження Саперу 3/16/2025 National Technical University of Ukraine Igor Sikorskyi Kyiv Politech Institute (National Technical University of Ukraine Igor Sikorskyi Kyiv Politech Institute)	32 0.50 %	
6	https://ela.kpi.ua/bitstreams/36b08709-1403-4fe0-b623-3d35bebb19a9/download	25 0.39 %	
7	ІП-13_Макарчук_ПЗ 5/27/2025 National Technical University of Ukraine Igor Sikorskyi Kyiv Politech Institute (ФІОТ, К-а інформатики та програмної інженерії)	24 0.38 %	
8	ІП-13_Макарчук_ПЗ 5/27/2025 National Technical University of Ukraine Igor Sikorskyi Kyiv Politech Institute (ФІОТ, К-а інформатики та програмної інженерії)	23 0.36 %	
9	Вебзастосунок для допомоги у діагностиці захворювань у професійній медичній практиці 3/15/2025 National Technical University of Ukraine Igor Sikorskyi Kyiv Politech Institute (National Technical University of Ukraine Igor Sikorskyi Kyiv Politech Institute)	22 0.35 %	
10	Вебсервіс для перегляду відеоконтенту в жанрі аніме 3/16/2025 National Technical University of Ukraine Igor Sikorskyi Kyiv Politech Institute (National Technical University of Ukraine Igor Sikorskyi Kyiv Politech Institute)	19 0.30 %	
з домашньої бази даних (9.13 %)			
ПОРЯДКОВИЙ НОМЕР	ЗАГОЛОВОК	КІЛЬКІСТЬ ІДЕНТИЧНИХ СЛІВ (ФРАГМЕНТІВ)	
1	Вебзастосунок для допомоги у діагностиці захворювань у професійній медичній практиці 3/15/2025 National Technical University of Ukraine Igor Sikorskyi Kyiv Politech Institute (National Technical University of Ukraine Igor Sikorskyi Kyiv Politech Institute)	109 (12) 1.71 %	
2	Вебзастосунок з інтегрованим вебсервісом для тестування штучного інтелекту вирішення NP-Complete задачі проходження Саперу 3/16/2025 National Technical University of Ukraine Igor Sikorskyi Kyiv Politech Institute (National Technical University of Ukraine Igor Sikorskyi Kyiv Politech Institute)	97 (5) 1.52 %	
3	ІП-15_Борисик_ПЗ 5/27/2025 National Technical University of Ukraine Igor Sikorskyi Kyiv Politech Institute (ФІОТ, К-а інформатики та програмної інженерії)	95 (2) 1.49 %	
4	ІП-13_Макарчук_ПЗ 5/27/2025 National Technical University of Ukraine Igor Sikorskyi Kyiv Politech Institute (ФІОТ, К-а інформатики та програмної інженерії)	82 (5) 1.29 %	

5	Вебсервіс для перегляду відеоконтенту в жанрі аніме 3/16/2025 National Technical University of Ukraine Igor Sikorskyi Kyiv Politech Institute (National Technical University of Ukraine Igor Sikorskyi Kyiv Politech Institute)	33 (2) 0.52 %	
6	ІП-15_Химич_ПЗ 5/28/2025 National Technical University of Ukraine Igor Sikorskyi Kyiv Politech Institute (ФІОТ, К-а інформатики та програмної інженерії)	33 (3) 0.52 %	
7	Економіка ІТ-індустрії та підприємництво 3/16/2025 National Technical University of Ukraine Igor Sikorskyi Kyiv Politech Institute (National Technical University of Ukraine Igor Sikorskyi Kyiv Politech Institute)	31 (4) 0.49 %	
8	Веб-застосунок для віддаленого керування та моніторингу роботи систем очищення води 3/15/2025 National Technical University of Ukraine Igor Sikorskyi Kyiv Politech Institute (National Technical University of Ukraine Igor Sikorskyi Kyiv Politech Institute)	23 (3) 0.36 %	
9	ІП-11_Ляля_ПЗ 5/29/2025 National Technical University of Ukraine Igor Sikorskyi Kyiv Politech Institute (ФІОТ, К-а інформатики та програмної інженерії)	22 (2) 0.35 %	
10	ІП-11_Трикош_ПЗ 5/28/2025 National Technical University of Ukraine Igor Sikorskyi Kyiv Politech Institute (ФІОТ, К-а інформатики та програмної інженерії)	14 (2) 0.22 %	
11	ІП-13_Дем'янчук_ПЗ 5/30/2025 National Technical University of Ukraine Igor Sikorskyi Kyiv Politech Institute (ФІОТ, К-а інформатики та програмної інженерії)	13 (2) 0.20 %	
12	Автоматизована система розрахунку вартості програмного забезпечення 3/15/2025 National Technical University of Ukraine Igor Sikorskyi Kyiv Politech Institute (National Technical University of Ukraine Igor Sikorskyi Kyiv Politech Institute)	10 (1) 0.16 %	
13	Мобільний застосунок "Антологія інтелектуальних ігор-головоломок" 3/16/2025 National Technical University of Ukraine Igor Sikorskyi Kyiv Politech Institute (National Technical University of Ukraine Igor Sikorskyi Kyiv Politech Institute)	8 (1) 0.13 %	
14	Мобільний застосунок для керування Arduino через Wi-Fi 3/16/2025 National Technical University of Ukraine Igor Sikorskyi Kyiv Politech Institute (National Technical University of Ukraine Igor Sikorskyi Kyiv Politech Institute)	7 (1) 0.11 %	
15	Програмне забезпечення для електронної комерції на базі хмарних технологій 3/16/2025 National Technical University of Ukraine Igor Sikorskyi Kyiv Politech Institute (National Technical University of Ukraine Igor Sikorskyi Kyiv Politech Institute)	5 (1) 0.08 %	
з програми обміну базами даних (1.00 %)			
ПОРЯДКОВИЙ НОМЕР	ЗАГОЛОВОК	КІЛЬКІСТЬ ІДЕНТИЧНИХ СЛІВ (ФРАГМЕНТІВ)	
1	Файлова система.docx 2/21/2024 Kherson national technical university (Херсонський національний технічний університет)	42 (4) 0.66 %	

2	Lõputöö_Tanel_Oberg_155083IAPB.pdf 2/14/2024 Estonian Academic Database (Estonian University)	11 (1) 0.17 %	
3	Удосконалення рівняння регресії для оцінювання тривалості виконання робіт з розробки веб-застосунків з використанням методології Agile та розробка програми для її реалізації 10/27/2024 National University of Shipbuilding named after Admiral Makarov (National University of Shipbuilding named after Admiral Makarov)	6 (1) 0.09 %	
4	bitstream_fd19d1c4-02c4-45d1-a074-ec99b3990b2a 12/9/2024 National Technical University "Kharkiv Polytechnic Institute" students papers (National Technical University "Kharkiv Polytechnic Institute" students papers)	5 (1) 0.08 %	

з Інтернету (4.50 %)



ПОРЯДКОВИЙ НОМЕР	ДЖЕРЕЛО URL	КІЛЬКІСТЬ ІДЕНТИЧНИХ СЛІВ (ФРАГМЕНТІВ)	
1	https://ela.kpi.ua/bitstreams/36b08709-1403-4fe0-b623-3d35bebb19a9/download	82 (5) 1.29 %	
2	http://konkurs.khnu.km.ua/wp-content/uploads/sites/25/2020/06/DP3_TRYVALIST-ZADACH.pdf	54 (6) 0.85 %	
3	https://ela.kpi.ua/server/api/core/bitstreams/61f75747-3203-4f93-8e68-9698ffbec074/content	35 (1) 0.55 %	
4	https://ppt-online.org/499157	23 (3) 0.36 %	
5	https://lektcii.org/12-76734.html	17 (2) 0.27 %	
6	https://ag.marketing/prays-agregatori-yak-kanal-prodazhiv/	17 (2) 0.27 %	
7	https://ela.kpi.ua/bitstreams/0141cda0-d47b-48f0-8eee-d6f54e79b129/download	15 (2) 0.24 %	
8	http://eprints.library.odeku.edu.ua/5903/1/Ganев_Rozrobka_informatciynogo_agenty_B2019.pdf	14 (1) 0.22 %	
9	https://ela.kpi.ua/server/api/core/bitstreams/3c9fc65e-68a0-4735-ace1-b83404d3c3e5/content	11 (2) 0.17 %	
10	https://ela.kpi.ua/bitstreams/49e763f4-0650-4c6d-9f00-da1a9abb025b/download	11 (1) 0.17 %	
11	https://ela.kpi.ua/bitstreams/94676364-e423-4a0b-8a53-31353181de51/download	8 (1) 0.13 %	

Список прийнятих фрагментів (немає прийнятих фрагментів)

ПОРЯДКОВИЙ НОМЕР	ЗМІСТ	КІЛЬКІСТЬ ОДНАКОВИХ СЛІВ (ФРАГМЕНТІВ)	
------------------	-------	---------------------------------------	--

ДОДАТОК Б. ВАРІАНТИ ВИКОРИСТАННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

Таблиця Б.1 – Варіант використання UC-01

Use case ID	UC-01
Use case name	Вхід в систему
Goals	Увійти в систему AGG.
Actors	Адміністратор.
Trigger	Адміністратор бажає увійти в систему.
Pre-conditions	–
Flow of Events	Адміністратор запускає систему. Система пропонує ввести логін та пароль. Адміністратор вводить логін та пароль, натискає на кнопку «Вхід». Система перевіряє логін та пароль. Адміністратор перенаправляється на головну сторінку.
Extension	У випадку, якщо вхід в систему не відбувся (наприклад, через введення некоректних даних), система повідомляє про помилку.
Post-Condition	Вхід в систему, перехід на головну сторінку.

Таблиця Б.2 – Варіант використання UC-02

Use case ID	UC-02
Use case name	Імпортування даних
Goals	Імпортувати дані в систему AGG.
Actors	Адміністратор.
Trigger	Адміністратор бажає імпортувати дані нового супермаркету в систему AGG.

Продовження таблиці Б.2

Pre-conditions	–
Flow of Events	–
Extension	–
Post-Condition	Адміністратор імпортував дані нового супермаркету в систему агрегатору.

Таблиця Б.3 – Варіант використання UC-03

Use case ID	UC-03
Use case name	Збереження супермаркету
Goals	Зберегти дані супермаркету в систему AGG.
Actors	Адміністратор.
Trigger	Адміністратор бажає зберегти новий супермаркет в систему.
Pre-conditions	–
Flow of Events	–
Extension	–
Post-Condition	Адміністратор зберіг дані нового супермаркету в систему агрегатору.

Таблиця Б.4 – Варіант використання UC-04

Use case ID	UC-04
Use case name	Збереження категорій
Goals	Зберегти дані категорій доданого супермаркету в систему AGG.
Actors	Адміністратор.

Продовження таблиці Б.4

Trigger	Адміністратор бажає зберегти категорії супермаркету в систему.
Pre-conditions	Збереження супермаркету.
Flow of Events	–
Extension	–
Post-Condition	Адміністратор зберіг категорії доданого супермаркету в систему агрегатору.

Таблиця Б.5 – Варіант використання UC-05

Use case ID	UC-05
Use case name	Збереження продуктів
Goals	Зберегти дані продуктів доданого супермаркету в систему AGG.
Actors	Адміністратор.
Trigger	Адміністратор бажає зберегти інформацію про продукти супермаркету в систему.
Pre-conditions	Збереження супермаркету. Збереження категорій.
Flow of Events	–
Extension	–
Post-Condition	Адміністратор зберіг інформацію про продукти доданого супермаркету в систему агрегатору.

Таблиця Б.6 – Варіант використання UC-06

Use case ID	UC-06
Use case name	Перегляд категорій продуктів агрегатору
Goals	Переглянути категорії продуктів агрегатору.
Actors	Адміністратор.
Trigger	Адміністратор бажає переглянути категорії продуктів агрегатору.
Pre-conditions	Вхід в систему.
Flow of Events	На головній сторінці Адміністратор натискає на пункт меню «Категорії», яка перенаправляє на сторінку для перегляду категорій продуктів агрегатору. Адміністратор переглядає сторінку, яка містить категорії продуктів агрегатору.
Extension	–
Post-Condition	Адміністратор переглянув сторінку, яка містить категорії продуктів агрегатору.

Таблиця Б.7 – Варіант використання UC-07

Use case ID	UC-07
Use case name	Редагування, видалення, додавання категорій продуктів
Goals	Редагувати, видаляти, додавати категорію продуктів агрегатору.
Actors	Адміністратор.
Trigger	Адміністратор бажає редагувати/видалити/додати категорію продуктів агрегатору.
Pre-conditions	Наявність категорій продуктів агрегатору.

Продовження таблиці Б.7

Flow of Events	Адміністратор переглядає сторінку, яка містить категорії продуктів агрегатору. Адміністратор обирає та натискає на необхідну категорію, якщо потрібно редагувати або видалити. Адміністратор натискає на кнопку «Додати», якщо потрібно додати нову категорію.
Extension	При видаленні категорії система викидає діалогове вікно для підтвердження видалення. У випадку неможливого видалення система повідомляє Адміністратора.
Post-Condition	Адміністратор відредагував/видалив/додав категорію продуктів агрегатору.

Таблиця Б.8 – Варіант використання UC-08

Use case ID	UC-08
Use case name	Перегляд списку продуктів з супермаркетів
Goals	Переглянути список продуктів з супермаркетів.
Actors	Адміністратор.
Trigger	Адміністратор бажає переглянути список продуктів з супермаркетів.
Pre-conditions	Імпортування даних супермаркету. Збереження продуктів.
Flow of Events	На головній сторінці Адміністратор натискає на пункт меню «Продукти», яка перенаправляє на сторінку для перегляду списку продуктів з супермаркетів. Адміністратор переглядає сторінку, яка містить список продуктів з супермаркетів.
Extension	—

Продовження таблиці Б.8

Post-Condition	Адміністратор переглянув сторінку, яка містить список продуктів з супермаркетів.
----------------	--

Таблиця Б.9 – Варіант використання UC-09

Use case ID	UC-09
Use case name	Зв'язування продукту з супермаркету с агрегатором
Goals	Зв'язати продукти з супермаркету с агрегатором.
Actors	Адміністратор.
Trigger	Адміністратор бажає зв'язати продукти з супермаркету с агрегатором.
Pre-conditions	Імпортування даних супермаркету. Збереження продуктів.
Flow of Events	Адміністратор переглядає сторінку, яка містить список продуктів з супермаркетів. Адміністратор, за допомогою checkbox, обирає продукт з супермаркету, натискає на кнопку «Агрегація» та перенаправляється до сторінки для агрегації продукту. За допомогою radio button обирає відповідну дію для зв'язування продукту з супермаркету с агрегатором, ввівши назву існуючого продукту в агрегаторі в пошук та за результатами пошуку обравши необхідний продукт, натискає на кнопку «Застосувати».
Extension	—
Post-Condition	Адміністратор додав продукт з супермаркету до вже існуючого продукту в агрегаторі.

Таблиця Б.10 – Варіант використання UC-10

Use case ID	UC-10
Use case name	Додавання продукту в агрегаторі
Goals	Додати продукт в агрегаторі
Actors	Адміністратор.
Trigger	Адміністратор бажає додати продукт в агрегаторі.
Pre-conditions	Імпортування даних супермаркету. Збереження продуктів.
Flow of Events	Адміністратор переглядає сторінку, яка містить список продуктів з супермаркетів. Адміністратор, за допомогою checkbox, обирає продукти з супермаркетів, натискає на кнопку «Агрегація» та перенаправляється до сторінки для агрегації продукту. За допомогою radio button обирає відповідну дію для додавання нового продукту в агрегатор, ввівши необхідні дані про продукт, натискає на кнопку «Застосувати».
Extension	–
Post-Condition	Адміністратор додав продукт до агрегатору.

Таблиця Б.11 – Варіант використання UC-11

Use case ID	UC-11
Use case name	Перегляд списку продуктів агрегатору
Goals	Переглянути список продуктів агрегатору.
Actors	Адміністратор.
Trigger	Адміністратор бажає переглянути список продуктів агрегатору.
Pre-conditions	Зв'язування продуктів з супермаркету с агрегатором.

Продовження таблиці Б.11

Flow of Events	Адміністратор переглядає головну сторінку, натиснувши на пункт меню «Агреговані продукти» перенаправляється до сторінки для перегляду списку продуктів агрегатору.
Extension	–
Post-Condition	Адміністратор переглянув сторінку, зі списком продуктів агрегатору.

Таблиця Б.12 – Варіант використання UC-12

Use case ID	UC-12
Use case name	Редагування продукту в агрегаторі
Goals	Редагувати продукт в агрегаторі.
Actors	Адміністратор.
Trigger	Адміністратор бажає редагувати продукт агрегатору.
Pre-conditions	Наявність продуктів в системі агрегатору.
Flow of Events	Адміністратор переглядає сторінку зі списком продуктів агрегатору. Адміністратор, натиснувши на назву продукту, перенаправляється до сторінки для редагування продукту агрегатору. Адміністратор, після внесення змін певних властивостей продукту, натискає на кнопку «Зберегти».
Extension	–
Post-Condition	Адміністратор відредагував продукт агрегатору.

Таблиця Б.13 – Варіант використання UC-13

Use case ID	UC-13
Use case name	Зміна назви, категорії, картинки продукту

Продовження таблиці Б.13

Goals	Змінити назву та/або категорію та/або картинку продукту.
Actors	Адміністратор.
Trigger	Адміністратор бажає змінити назву та/або категорію та/або картинку продукту агрегатору.
Pre-conditions	—
Flow of Events	Адміністратор на сторінці для редагування продукту агрегатору. Адміністратор змінює назву та/або обирає іншу категорію та/або картинку продукту. Адміністратор, після внесення змін, натискає на кнопку «Зберегти».
Extension	—
Post-Condition	Адміністратор змінив назву та/або категорію та/або картинку продукту.

Таблиця Б.14 – Варіант використання UC-14

Use case ID	UC-14
Use case name	Відв'язування продукту певного супермаркету від продукту агрегатору
Goals	Відв'язати продукт певного супермаркету від продукту агрегатору
Actors	Адміністратор.
Trigger	Адміністратор бажає відв'язати продукт певного супермаркету від продукту агрегатору.
Pre-conditions	—

Продовження таблиці Б.14

Flow of Events	Адміністратор на сторінці для редагування продукту агрегатору натискає на кнопку «Відв'язати» напроти продукту з певного супермаркету. Адміністратор, після внесення змін, натискає на кнопку «Зберегти».
Extension	–
Post-Condition	Адміністратор відв'язав продукт певного супермаркету від продукту агрегатору.

Таблиця Б.15 – Варіант використання UC-15

Use case ID	UC-15
Use case name	Оновлення ціни, наявності продуктів
Goals	Оновити ціну, наявність продуктів.
Actors	System Scheduler
Trigger	Необхідність надання актуальної інформації.
Pre-conditions	–
Flow of Events	–
Extension	–
Post-Condition	Оновлено ціну, наявність продуктів.

Таблиця Б.16 – Варіант використання UC-16

Use case ID	UC-16
Use case name	Пошук продукту
Goals	Знайти продукт.
Actors	Користувач.
Trigger	Користувач бажає знайти продукт.

Продовження таблиці Б.16

Pre-conditions	Перебування на сайті.
Flow of Events	Користувач шукає продукт.
Extension	–
Post-Condition	Користувач знайшов необхідний продукт.

Таблиця Б.17 – Варіант використання UC-17

Use case ID	UC-17
Use case name	Фільтрація по категорії продуктів
Goals	Знайти продукт, фільтруючи його за категорією.
Actors	Користувач.
Trigger	Користувач бажає знайти продукт.
Pre-conditions	Перебування на сайті.
Flow of Events	Користувач, знаходячись на сторінці зі списком категорій, визначившись з вибором категорії, перейшовши до сторінки зі списком продуктів, що його цікавлять, і знаходить необхідний продукт.
Extension	–
Post-Condition	Користувач знайшов необхідний продукт.

Таблиця Б.18 – Варіант використання UC-18

Use case ID	UC-18
Use case name	Фільтрація по назві продукта
Goals	Знайти продукт, використовуючи пошук за назвою.
Actors	Користувач.

Продовження таблиці Б.18

Trigger	Користувач бажає знайти продукт.
Pre-conditions	Перебування на сайті.
Flow of Events	Користувач, знаходячись на сторінці зі списком продуктів, ввівши назву продукту в пошукове поле, і, натискаючи на іконку з лупою, знаходить необхідний продукт.
Extension	—
Post-Condition	Користувач знайшов необхідний продукт.

Таблиця Б.19 – Варіант використання UC-19

Use case ID	UC-19
Use case name	Формування кошику продуктів
Goals	Сформувати кошик продуктів.
Actors	Користувач.
Trigger	Користувач бажає додати знайдений продукт до кошика.
Pre-conditions	Перебування на сайті.
Flow of Events	Користувач, знайшовши необхідний продукт, і, натискаючи на відповідну кнопку, додає його до кошику. Повторне натискання перенаправляє на сторінку кошика.
Extension	—
Post-Condition	Кошик продуктів сформовано. Обрані продукти готові до подальших дій.

Таблиця Б.20 – Варіант використання UC-20

Use case ID	UC-20
Use case name	Порівняння загальної вартості кошику продуктів по супермаркетах
Goals	Порівняти загальну вартість кошику продуктів по супермаркетах для здійснення економної покупки.
Actors	Користувач.
Trigger	–
Pre-conditions	Перебування на сайті. Кошик не порожній.
Flow of Events	Користувач, відкривши сторінку з кошиком, в якому система автоматично відображає загальну вартість по наявних продуктах кожного супермаркету, аналізує підсумкові суми.
Extension	–
Post-Condition	Користувач порівняв загальну вартість кошику продуктів по супермаркетах для здійснення економної покупки.

Таблиця Б.21 – Варіант використання UC-21

Use case ID	UC-21
Use case name	Пошук найближчого супермаркету
Goals	Знайти найближчий супермаркет.
Actors	Користувач.
Trigger	Користувач бажає знайти найближчий супермаркет з урахуванням свого місця розташування.
Pre-conditions	Перебування на сайті.

Продовження таблиці Б.21

Flow of Events	Користувач, перейшовши до сторінки з кошиком продуктів, натиснувши на кнопку «Найближчий магазин» та, вводячи свої координати, а після, натиснувши на кнопку «ОК!», переглядає мапу зі знайденими супермаркетами та список супермаркетів по зростанню відстані.
Extension	—
Post-Condition	Користувач знайшов найближчий супермаркет з урахуванням свого місця розташування.

ДОДАТОК В. ОПИС ФУНКЦІОНАЛЬНИХ ВИМОГ ДО ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

Таблиця В.1 – Функціональна вимога FR-01

Назва	Вхід в систему
Опис	Система повинна надавати можливість отримати права Адміністратора, вводячи логін та пароль.

Таблиця В.2 – Функціональна вимога FR-02

Назва	Імпортування даних
Опис	Система повинна надавати можливість імпортувати дані про супермаркет.

Таблиця В.3 – Функціональна вимога FR-03

Назва	Збереження супермаркету
Опис	Система повинна надавати можливість зберігати дані про супермаркет.

Таблиця В.4 – Функціональна вимога FR-04

Назва	Збереження категорій
Опис	Система повинна надавати можливість зберігати дані категорій супермаркету.

Таблиця В.5 – Функціональна вимога FR-05

Назва	Збереження продуктів
Опис	Система повинна надавати можливість зберігати дані про продукти супермаркету.

Таблиця В.6 – Функціональна вимога FR-06

Назва	Перегляд категорій продуктів агрегатору
Опис	Система повинна надавати можливість переглядати категорії продуктів агрегатору.

Таблиця В.7 – Функціональна вимога FR-07

Назва	Редагування категорій продуктів агрегатору
Опис	Система повинна надавати можливість редагувати категорії продуктів агрегатору.

Таблиця В.8 – Функціональна вимога FR-08

Назва	Видалення категорій продуктів агрегатору
Опис	Система повинна надавати можливість видаляти категорії продуктів агрегатору.

Таблиця В.9 – Функціональна вимога FR-09

Назва	Додавання категорій продуктів агрегатору
Опис	Система повинна надавати можливість додавати категорії продуктів агрегатору.

Таблиця В.10 – Функціональна вимога FR-10

Назва	Перегляд списку продуктів з супермаркетів
Опис	Система повинна надавати можливість переглядати список продуктів з супермаркетів.

Таблиця В.11 – Функціональна вимога FR-11

Назва	Додавання продукту в агрегатор
Опис	Система повинна надавати можливість додавати продукт в агрегатор.

Таблиця В.12 – Функціональна вимога FR-12

Назва	Перегляд списку продуктів агрегатору
Опис	Система повинна надавати можливість переглядати список агрегованих продуктів.

Таблиця В.13 – Функціональна вимога FR-13

Назва	Зв'язування продуктів з супермаркету с агрегатором
Опис	Система повинна надавати можливість зв'язувати продукти з супермаркетів с агрегатором.

Таблиця В.14 – Функціональна вимога FR-14

Назва	Редагування продукту в агрегаторі
Опис	Система повинна надавати можливість редагувати продукти в агрегаторі.

Таблиця В.15 – Функціональна вимога FR-16

Назва	Зміна назви, категорії, картинки продукту
Опис	Система повинна надавати можливість змінити назву, категорію, картинку продукту

Таблиця В.16 – Функціональна вимога FR-16

Назва	Відв'язування продукту певного супермаркету від продукту агрегатору
Опис	Система повинна надавати можливість відв'язувати продукт певного супермаркету від продукту агрегатору, тобто видаляти.

Таблиця В.17 – Функціональна вимога FR-17

Назва	Оновлення ціни, наявності продуктів
Опис	Система повинна надавати актуальну інформацію про продукт, тобто забезпечити оновлення ціни, наявності продуктів

Таблиця В.18 – Функціональна вимога FR-18

Назва	Пошук продукту
Опис	Система повинна надавати можливість пошуку продукту.

Таблиця В.19 – Функціональна вимога FR-19

Назва	Фільтрація по категорії продуктів
Опис	Система повинна надавати можливість пошуку продукту, фільтруючи його за категорією.

Таблиця В.20 – Функціональна вимога FR-20

Назва	Фільтрація по назві продукта
Опис	Система повинна надавати можливість пошуку продукту, використовуючи пошук за назвою.

Таблиця В.21 – Функціональна вимога FR-21

Назва	Формування кошику продуктів
Опис	Система повинна надавати можливість формування кошику, шляхом додавання продукту в кошик продуктів.

Таблиця В.22 – Функціональна вимога FR-22

Назва	Порівняння загальної вартості кошику продуктів по супермаркетах
Опис	Система повинна надавати можливість порівняння загальної вартості кошику продуктів по різних супермаркетах.

Таблиця В.23 – Функціональна вимога FR-23

Назва	Введення координат місця розташування
Опис	Система повинна надавати можливість введення координат місця розташування.

Таблиця В.24 – Функціональна вимога FR-24

Назва	Пошук найближчого супермаркету
Опис	Система повинна надавати можливість пошуку найближчого супермаркету.

ДОДАТОК Г. АНАЛІЗ ЕКОНОМІЧНИХ ПОКАЗНИКІВ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

Таблиця Г.1 – Значення коефіцієнтів базової моделі СОСОМО залежно від класу проєкту

Клас проєкту	<i>a</i>	<i>b</i>	<i>c</i>	<i>d</i>
Органічний	2,4	1,05	2,5	0,38
Напіврозподілений	3,0	1,12	2,5	0,35
Вбудований	3,6	1,20	2,5	0,32

Таблиця Г.2 – Значення атрибуту вартості факторів витрат

Атрибут вартості	Рейтинг	Причина
Характеристики продукту		
Необхідна надійність ПЗ	1,15 (високий)	AGG працює з реальними даними про продукти у супермаркетах, тому від надійності залежить достовірність результатів.
Розмір БД додатка	1,00 (середній)	База даних містить інформацію про супермаркети, категорії та продукти, але її обсяг не перевищує меж, характерних для складних баз даних вебзастосунків.
Складність продукту	1,00 (середній)	Типовий рівень складності для сучасних вебзастосунків.

Продовження таблиці Г.2

Атрибут вартості	Рейтинг	Причина
Характеристики продукту		
Обмеження швидкодії при виконанні програми	n/a (низький)	Обмеження не мають критичного впливу.
Обмеження пам'яті	1,00 (середній)	Обмеження пам'яті не є жорсткими, оскільки для розробленого вебзастосунку системні вимоги є типовими.
Нестійкість оточення віртуальної машини	1,00 (середній)	AGG працює в середовищі Java Virtual Machine (JVM), яке може бути нестабільним.
Необхідний час відновлення	1,00 (середній)	Прийнятним є помірний час на відновлення роботи вебзастосунку.
Характеристики персоналу		
Аналітичні здібності	0,86 (високий)	Є важливим для ефективного проектування архітектури та вирішення відносно складних технічних завдань.
Досвід розробки	1,00 (середній)	Свідчить про наявність достатнього рівня навиків для реалізації проєкту.

Продовження таблиці Г.2

Атрибут вартості	Рейтинг	Причина
Здібності до розробки ПЗ	0,86 (високий)	Знання принципів розробки програмного забезпечення є обов'язковим.
Досвід використання віртуальних машин	1,00 (середній)	Досвід роботи з віртуальними машинами є достатнім для розгортання та підтримки вебзастосунку.
Досвід розробки на мовах програмування	0,95 (високий)	Володіння мовою програмування, за допомогою якої виконувалась реалізація AGG, є важливим для досягнення ефективності та якості програмного коду.
Характеристики проекту		
Застосування методів розробки ПЗ	1,00 (середній)	Використання структурованих підходів для реалізації вебзастосунку підвищує керованість розробки.
Використання інструментарію розробки ПЗ	0,91 (високий)	Застосування сучасних інструментів розробки покращує продуктивність розробника та якість кінцевого продукту.
Вимоги дотримання графіку розробки	1,10 (дуже високий)	Дотримання графіку розробки вказує на важливість своєчасного завершення проекту.

Таблиця Г.3 – Значення коефіцієнтів проміжного рівня залежно від класу проєкту

Клас проєкту	<i>a</i>	<i>b</i>
Органічний	3,2	1,05
Напіврозподілений	3,0	1,12
Вбудований	2,8	1,20

Таблиця Г.4 – Значення чиннику масштабу залежно від оцінки його рівня

Чинники масштабу, SF_j	Оцінка рівня чинника	Причина
PREC (Precedentedness)	3,72 (середній)	Використано знайомі технології та підходи веброзробки, однак присутні певні унікальні аспекти, які не мають повних аналогів.
FLEX (Development Flexibility)	4,05 (низький)	Самостійна розробка призвела до обмеження гнучкості процесу розробки.
RESL (Architecture / Risk Resolution)	5,65 (низький)	Ризики відомі та проаналізовані на 40%.

Продовження таблиці Г.4

Чинники масштабу, SF_j	Оцінка рівня чинника	Причина
TEAM (Team Cohesion)	5,48 (дуже низький)	Розробка вебзастосунку AGG виконувалась самостійно.
PMAT (Process Maturity)	4,68 (середній)	Можлива відсутність автоматизації деяких процесів, що може призвести до помилок і затримок.

Таблиця Г.5 – Значення множників трудомісткості залежно від оцінки їх рівня на початковій стадії

Множник трудомісткості	Оцінка рівня множника	Причина
PERS (Personnel Capability)	1,00 (середній)	Проект реалізовано з достатнім рівнем технічних навиків, але без глибокої спеціалізації в окремих сферах.
PREX (Personnel Experience)	1,00 (середній)	Наявний певний досвід в розробленні вебзастосунків.

Продовження таблиці Г.5

Множник трудомісткості	Оцінка рівня множника	Причина
RCPX (Product Reliability and Complexity)	0,83 (низький)	Розроблений продукт немає критичних вимог до високої складності бізнес-логіки.
RUSE (Developed for Reusability)	1,00 (середній)	Архітектура AGG характеризується модульною структурою, яка забезпечує легке інтегрування нових компонентів.
PDIF (Platform Difficulty)	1,00 (середній)	AGG працює на стандартній Java-платформі, що спрощує розробку.
FCIL (Facilities)	1,10 (низький)	Використано стандартний набір інструментів для розробки.
SCED (Required Development Schedule)	1,00 (високий)	До проєкту висуваються високі вимоги щодо термінів виконання.

Таблиця Г.6 – Значення множників трудомісткості залежно від оцінки їх рівня на стадії після опрацювання архітектури

Множник зусиль	Оцінка рівня множника	Причина
АСАР (Analyst Capability)	1,29 (середній)	Володіння базовими аналітичними навиками, достатніми для розуміння вимог і побудови архітектури.
АЕХР (Applications Experience)	1,10 (середній)	Досвід розробки схожих додатків дозволяє швидше адаптуватися до нових вимог і використовувати напрацювання з попередніх проектів.
РСАР (Programmer Capability)	1,15 (середній)	АГГ вимагає певних знань Java, принципів об'єктно-орієнтованого програмування та розуміння роботи баз даних.
РСОН (Personnel Continuity)	1,12 (середній)	Індивідуальне виконання може супроводжуватись змінами навантаження.
РЕХР (Platform Experience)	1,09 (середній)	Досвід роботи з платформою дозволяє ефективніше налаштовувати середовище розробки, усувати проблеми, пов'язані з конфігурацією, і оптимізувати продуктивність додатку.

Продовження таблиці Г.6

Множник зусиль	Оцінка рівня множника	Причина
LTEX (Language and Tool Experience)	1,09 (середній)	Є достатній досвід роботи з мовами програмування та інструментами для ефективної розробки.
RELY (Required Software Reliability)	1,00 (високий)	Передбачається стабільність роботи системи AGG.
DATA (Database Size)	n/a (низький)	Обсяг даних про продукти є значним, але не є надзвичайно великим.
CPLX (Software Product Complexity)	0.73 (низький)	Розробка не включає складних алгоритмів, особливих архітектурних рішень.
RUSE (Required Reusability)	0,95 (середній)	Необхідно дотримуватися певних стандартів розробки для забезпечення сумісності компонентів.
DOCU (Documentation Match to Life-Cycle Needs)	0,91 (середній)	Документація проєкту є достатньою для розробки та подальшої підтримки.

Продовження таблиці Г.6

Множник зусиль	Оцінка рівня множника	Причина
TIME (Execution Time Constraint)	1,00 (високий)	Для забезпечення комфортної роботи користувачів система повинна реагувати на запити в реальному часі.
STOR (Main Storage Constraint)	1,00 (високий)	Обробка великих об'єктів може призвести до перевищення доступної пам'яті.
PVOL (Platform Volatility)	1,00 (високий)	Регулярні оновлення Java, баз даних і інших технологій можуть вимагати додаткових зусиль для адаптації проекту.
TOOL (Use of Software Tools)	0,90 (дуже високий)	Використання інтегрованого середовища розробки для написання коду та налагодження, системи контролю версій - для управління змінами коду та співпраці в команді.
SITE (Multisite Development)	1,22 (низький)	Розробник системи AGG працює локально, таким чином, мінімувавши потенційні складнощі.

Продовження таблиці Г.6

Множник зусиль	Оцінка рівня множника	Причина
SCED (Required Development Schedule)	1,00 (високий)	До проєкту висуваються високі вимоги щодо термінів виконання.

Факультет інформатики та обчислювальної техніки
Кафедра інформатики та програмної інженерії

“ЗАТВЕРДЖЕНО”

Завідувач кафедри

_____ Едуард ЖАРІКОВ

“ ____ ” _____ 2025 р.

**ВЕБЗАСТОСУНОК ПІДБОРУ СУПЕРМАРКЕТУ ЗА КОШИКОМ
ПРОДУКТІВ ТА МІСЦЕМ РОЗТАШУВАННЯ**

Текст програми

КПІ.ІП-1501.045440.03.12

“ПОГОДЖЕНО”

Керівник проекту:

_____ Олесь КОВТУНЕЦЬ

Нормоконтроль:

_____ Тетяна ШУЛЬКЕВИЧ

Виконавець:

_____ Мадіна АДЖИГЕЛЬДІЄВА

Київ – 2025

Посилання на репозиторій з повним текстом програмного коду:

<https://github.com/madinaadzhi/aggregator.git>.

Файл MarketImportService.java

Реалізація функціональної задачі пакетного імпортування даних.

```
@Service
@Slf4j
public class MarketImportService {

    @Autowired
    private MarketRepository marketRepository;

    @Autowired
    private CategoryRepository categoryRepository;

    @Autowired
    private ProductRepository productRepository;

    public void importMarket(MarketApi marketApi) {
        String domainName = marketApi.getMarketInfo().getDomainName();
        log.info("{} Start importing ", domainName);

        MarketEntity market =
marketRepository.findMarketEntitiesByDomainName(domainName);
        if (market == null) {
            log.info("{} Saving market to DB ", domainName);
            saveMarket(marketApi);

            market =
marketRepository.findMarketEntitiesByDomainName(domainName);
        } else {
            log.info("Market already has been imported");
        }

        int categoriesCount =
categoryRepository.countByMarketId(market.getId());
        if (categoriesCount == 0) {
            log.info("{} Saving categories to DB ", domainName);

            saveCategories(marketApi);
            log.info("{} Done ", domainName);
            log.info("{} Saving products to DB ", domainName);
        } else {
            log.info("Categories already have been imported");
        }
    }
}
```

```

while (true) {
    try {
        saveProducts(marketApi, true);
        break;
    } catch (Exception e) {
        log.warn("Error occurred during import process. Waiting 1 minute
to retry. Details: {}", e.getMessage());
        try {
            Thread.sleep(60 * 1000);
        } catch (InterruptedException ex) {
        }
    }
}

log.info("[{}] Done ", domainName);
log.info("[{}] Add products to aggregator", domainName);
productRepository.linkProductWithAggrProduct(market.getId());
log.info("[{}] Done ", domainName);
}

public void updatePrice(MarketApi marketApi) {
    String domainName = marketApi.getMarketInfo().getDomainName();
    log.info("[{}] Updating price ", domainName);

    while (true) {
        try {
            saveProducts(marketApi, false);
            break;
        } catch (Exception e) {
            log.warn("Error occurred during price update. Waiting 1 minute
to retry. Details: {}", e.getMessage(), e);
            try {
                Thread.sleep(60 * 1000);
            } catch (InterruptedException ex) {
            }
        }
    }

    log.info("[{}] Done ", domainName);
}

private void saveMarket(MarketApi marketApi) {
    Market marketInfo = marketApi.getMarketInfo();
    MarketEntity marketEntity = new MarketEntity();
    marketEntity.setName(marketInfo.getName());
}

```

```

        marketEntity.setLogoUrl(marketInfo.getLogoUrl());
        marketEntity.setDomainName(marketInfo.getDomainName());
        marketRepository.save(marketEntity);
    }

    private void saveCategories(MarketApi marketApi) {
        Market marketInfo = marketApi.getMarketInfo();

        Long marketId =
marketRepository.findMarketEntitiesByDomainName(marketInfo.getDomainName()).get
Id();

        List<Category> categories = marketApi.getCategories();
        Map<String, Long> externalToInternalId = new HashMap<>();
        for (Category category : categories) {
            String externalId = category.getId();
            String name = category.getName();
            Long parentId = externalToInternalId.get(category.getParentId());
            String logoUrl = category.getLogoUrl();

            CategoryEntity categoryEntity = new CategoryEntity();

            categoryEntity.setName(name);

            categoryEntity.setParentId(parentId);
            categoryEntity.setMarketId(marketId);
            categoryEntity.setExternalId(externalId);
            categoryEntity.setLogoUrl(logoUrl);

            CategoryEntity saved = categoryRepository.save(categoryEntity);
            externalToInternalId.put(externalId, saved.getId());
        }
    }

    private void saveProducts(MarketApi marketApi, boolean isImport) {
        Market marketInfo = marketApi.getMarketInfo();

        MarketEntity marketEntity =
marketRepository.findMarketEntitiesByDomainName(marketInfo.getDomainName());

        List<CategoryEntity> categoriesForProcessing =
getCategories(marketEntity, isImport);

        log.info("[{}] Found {} categories ", marketInfo.getName(),
categoriesForProcessing.size());

        for (int i = 0; i < categoriesForProcessing.size(); i++) {
            CategoryEntity notProcessedCategory =
categoriesForProcessing.get(i);

```

```

        log.info("{} {} of {} category is being processed. Category Id: {}
",
        marketInfo.getName(), i + 1, categoriesForProcessing.size(),
notProcessedCategory.getId());

        List<Product> products =
marketApi.getProducts(notProcessedCategory.getExternalId());
        for (Product product : products) {
            String externalId = product.getId();
            String name = product.getName();
            double price = product.getPrice();
            Boolean isAvailable = product.isAvailable();
            String externalCategoryId = product.getCategoryId();
            String siteUrl = product.getSiteUrl();
            String imageUrl = product.getImageUrl();
            String unit = product.getUnit();

            if (!StringUtils.isNumeric(externalCategoryId)) {
                log.warn("{} is not a valid category id. ignoring product
{}", externalCategoryId, product.getName());
                continue;
            }

            if (isImport) {
                Long categoryId =
categoryRepository.getCategoryEntityByExternalIdAndMarketId(externalCategoryId,
marketEntity.getId()).getId();

                ProductEntity productEntity = new ProductEntity();
                productEntity.setExternalId(externalId);
                productEntity.setName(name);
                productEntity.setPrice(price);
                productEntity.setAvailable(isAvailable);
                productEntity.setCategoryId(categoryId);
                productEntity.setSiteUrl(siteUrl);
                productEntity.setImgUrl(imageUrl);
                productEntity.setMarketEntity(marketEntity);

                productEntity.setUnit(unit);

                try {
                    productRepository.save(productEntity);
                } catch (Exception e) {
                    log.info("{} could not persist product {} ",
                        marketInfo.getName(), productEntity, e);
                }
            } else {

```

```

        try {

            ProductEntity persistedProduct =
productRepository.findProductEntityByMarketEntityIdAndExternalId(marketEntity.g
etId(), externalId);

            if (persistedProduct != null) {
                if (!persistedProduct.getPrice().equals(price) ||
persistedProduct.isAvailable() == isAvailable) {
                    log.info("{} updating product {}. Changed
price from {} to {}. Changed availability from {} to {}",
                        marketInfo.getName(), externalId,
persistedProduct.getPrice(), price, persistedProduct.isAvailable(),
isAvailable);

                    persistedProduct.setPrice(price);
                    persistedProduct.setAvailable(isAvailable);
                    productRepository.save(persistedProduct);
                }
            } else {
                log.info("{} there is no product {} in the DB",
                    marketInfo.getName(), externalId);
            }
        } catch (Exception e) {
            log.info("{} could not update product {} ",
                marketInfo.getName(), externalId, e);
        }
    }

    notProcessedCategory.setImported(true);
    categoryRepository.save(notProcessedCategory);
}

}

private List<CategoryEntity> getCategories(MarketEntity marketEntity,
boolean isImport) {
    if (isImport) {
        return getNotImportedCategories(marketEntity.getId());
    } else {
        return
categoryRepository.getCategoryEntitiesByMarketId(marketEntity.getId());
    }
}

private List<CategoryEntity> getNotImportedCategories(Long marketId) {

```

```

        List<CategoryEntity> notImportedCategories =
categoryRepository.getCategoryEntitiesByMarketIdAndIsImportedFalse (marketId);
        List<CategoryEntity> result = new ArrayList<>();
        addToListWithNotImportedCategories(notImportedCategories, result);
        return result;
    }

    private void addToListWithNotImportedCategories(List<CategoryEntity>
notImportedCategories, List<CategoryEntity> result) {
        for (CategoryEntity notImportedCategory : notImportedCategories) {
            if (notImportedCategories.size() == 0) {
                addToListWithNotImportedCategories(notImportedCategories,
result);
            } else if (notImportedCategory.getChildCategories().size() == 0) {
                result.add(notImportedCategory);
            }
        }
    }
}

```

Файл SilpoMarketApi.java

Реалізація функціональної задачі пакетного імпортування даних.

```

@Service("SilpoMarketApi")
@Slf4j
public class SilpoMarketApi implements MarketApi {
    private static final String URL =
"https://api.catalog.ecom.silpo.ua/api/2.0/exec/EcomCatalogGlobal";

    @Override
    public Market getMarketInfo() {
        return new Market() {
            @Override
            public String getDomainName() {
                return "shop.silpo.ua";
            }

            @Override
            public String getName() {
                return "Silpo";
            }

            @Override

```

```

        public String getLogoUrl() {

            return
            "https://silpo.ua/pcs/logo.svg?id=Wed%20Nov%2001%202023%2014:47:48%20GMT+0200%2
            0(%D0%B7%D0%B0%20%D1%81%D1%85%D1%96%D0%B4%D0%BD%D0%BE%D1%94%D0%B2%D1%80%D0%BE%D
            0%BF%D0%B5%D0%B9%D1%81%D1%8C%D0%BA%D0%B8%D0%BC%20%D1%81%D1%82%D0%B0%D0%BD%D0%B4
            %D0%B0%D1%80%D1%82%D0%BD%D0%B8%D0%BC%20%D1%87%D0%B0%D1%81%D0%BE%D0%BC)";

        }

        @Override
        public String getSiteUrl() {
            return "https://shop.silpo.ua/";
        }

        @Override
        public Double getLatitude() {
            return 50.441310;
        }

        @Override
        public Double getLongitude() {
            return 30.521989;
        }

        @Override
        public String getAddress() {
            return "вулиця Басейна, 6, Київ";
        }
    };
}

@Override
public List<Category> getCategories() {
    try {
        String silpoCategoriesJson = getSilpoCategoriesJson();
        return parseCategories(silpoCategoriesJson);
    } catch (IOException e) {
        throw new RuntimeException(e);
    }
}

public String getSilpoCategoriesJson() throws IOException {
    return Jsoup.connect(URL)

```



```

        log.info("{} Making request: categoryId - {}, from - {}, to -
        {}, ", getMarketInfo().getName(), categoryId, from, to);
        String json = getProductPageByCategoryId(categoryId, from, to);
        List<Product> productList = SilpoUtils.getProduct(json);
        fetchedProductCnt += productList.size();
        products.addAll(productList);
        if (iteration == 1) {
            JSONObject jsonObject = new JSONObject(json);
            expectedProductCnt = jsonObject.getInt("itemsCount");
            log.info("{} Category {} contains {} products ",
getMarketInfo().getName(), categoryId, expectedProductCnt);
        }
        if (fetchedProductCnt == expectedProductCnt) {
            break;
        }
        iteration++;
    }
    return products;
} catch (IOException e) {
    throw new RuntimeException(e);
}
}
}
}

```

Файл ForaMarketApi.java

Реалізація функціональної задачі пакетного імпортування даних.

```

@Service("ForaMarketApi")
@Slf4j
public class ForaMarketApi implements MarketApi {

    private static final String URL =
"https://api.catalog.ecom.fora.ua/api/2.0/exec/EcomCatalogGlobal";

    @Override
    public Market getMarketInfo() {
        return new Market() {
            @Override
            public String getDomainName() {
                return "shop.fora.ua";
            }

            @Override
            public String getName() {

```

```

        return "Fora";
    }

    @Override
    public String getLogoUrl() {
        return
        "https://upload.wikimedia.org/wikipedia/commons/thumb/9/92/%D0%9B%D0%BE%D0%B3%D
        0%BE%D1%82%D0%B8%D0%BF_%D0%BC%D0%B5%D1%80%D0%B5%D0%B6%D1%96_%D1%81%D1%83%D0%BF%
        D0%B5%D1%80%D0%BC%D0%B0%D1%80%D0%BA%D0%B5%D1%82%D1%96%D0%B2_%D0%A4%D0%BE%D1%80%
        D0%B0.svg/1200px-%D0%9B%D0%BE%D0%B3%D0%BE%D1%82%D0%B8%D0%BF_%D0%BC%D0%B5%D1%80%
        D0%B5%D0%B6%D1%96_%D1%81%D1%83%D0%BF%D0%B5%D1%80%D0%BC%D0%B0%D1%80%D0%BA%D0%B5%
        D1%82%D1%96%D0%B2_%D0%A4%D0%BE%D1%80%D0%B0.svg.png";
    }

    @Override
    public String getSiteUrl() {
        return "https://shop.fora.ua/";
    }

    @Override
    public Double getLatitude() {
        return 50.446526;
    }

    @Override
    public Double getLongitude() {
        return 30.523066;
    }

    @Override
    public String getAddress() {
        return "вулиця Лютеранська, 3, Київ";
    }
};
}

@Override
public List<Category> getCategories() {
    try {
        String foraCategoriesJson = getForaCategoriesJson();
        return parseCategories(foraCategoriesJson);
    } catch (IOException e) {
        throw new RuntimeException(e);
    }
}

```

```

    }
}

public String getForaCategoriesJson() throws IOException {
    return Jsoup.connect(URL)
        .header("content-type", "application/json")

        .requestBody("{\"method\":\"GetCategories\",\"data\":{\"deliveryType\":2,\"filialId\":\"310\",\"merchantId\":2}}")
        .method(org.jsoup.Connection.Method.POST)
        .ignoreContentType(true)
        .execute().body();
}

private List<Category> parseCategories(String categoriesJson) throws
JSONException {
    JSONObject jsonObject = new JSONObject(categoriesJson);
    JSONArray items = jsonObject.getJSONArray("tree");
    List<Category> foraCategories = new ArrayList<>();
    for (int i = 0; i < items.length(); i++) {
        String externalId =
String.valueOf(items.getJSONObject(i).get("id"));
        String name = items.getJSONObject(i).getString("name");
        String parentId =
String.valueOf(items.getJSONObject(i).get("parentId"));
        String logoUrl =
String.valueOf(items.getJSONObject(i).get("mobileAppIconPath"));
        if (parentId.equals("null")) {
            parentId = null;
        }
        foraCategories.add(new ForaCategory(externalId, name, parentId,
logoUrl));
    }
    return foraCategories;
}

@Override
public List<Product> getProducts(String categoryId) {
    log.info("[{}] Start importing products ", getMarketInfo().getName());
    try {
        List<Product> products = new ArrayList<>();
        int iteration = 1;
        int expectedProductCnt = 0;

```

```

        int fetchedProductCnt = 0;
        while (true) {
            int from = (iteration - 1) * 30 + 1;
            int to = iteration * 30;

            log.info("[{}] Making request: categoryId - {}, from - {}, to -
{} ", getMarketInfo().getName(), categoryId, from, to);

            String json = ForaUtils.getProductPageByCategoryId(categoryId,
from, to);

            List<Product> productList = ForaUtils.getProduct(json);
            fetchedProductCnt += productList.size();
            products.addAll(productList);
            if (iteration == 1) {
                JSONObject jsonObject = new JSONObject(json);
                expectedProductCnt = jsonObject.getInt("itemsCount");
                log.info("[{}] Category {} contains {} products ",
getMarketInfo().getName(), categoryId, expectedProductCnt);
            }
            if (fetchedProductCnt == expectedProductCnt) {
                break;
            }
            iteration++;
        }
        return products;
    } catch (IOException e) {
        throw new RuntimeException(e);
    }
}
}

```

Файл v08__data_preaggregation.sql

Реалізація функціональної задачі автоматизованої агрегації товарів.

```

insert into aggr_products (category_id, image_url, name, unit)
select distinct category_id, image_url, name, unit from product
where market_id = (select id from markets where domain_name = 'shop.silpo.ua')
and image_url not like '%default%';

update aggr_products ap
set ap.category_id =
    (select agg_cat.id as agg_cat_id
     FROM (
         select c.id, c.name, c.external_id from categories c where
market_id = 2) as silpo_cat

```

```

        join (select c.id, c.name, c.external_id from categories c
where market_id = 1) as agg_cat on silpo_cat.external_id = agg_cat.external_id
        where silpo_cat.id = ap.category_id);

update product p
set p.aggr_product_id = (select ap.id from aggr_products ap where ap.image_url
= p.image_url and p.unit = ap.unit )
where p.aggr_product_id is null;

```

Файл PriceUpdater.java

Реалізація функціональної задачі інтеграції зовнішніх сервісів для отримання актуальної інформації про товари.

```

@Service
public class PriceUpdater {
    public static final int SIX_HOURS = 6 * 60 * 60 * 1000;
    @Autowired
    @Qualifier("SilpoMarketApi")
    private MarketApi silpoMarketApi;
    @Autowired
    @Qualifier("ForaMarketApi")
    private MarketApi foraMarketApi;
    @Autowired
    private MarketImportService marketImportService;

    @Scheduled(fixedDelay = SIX_HOURS, initialDelay = SIX_HOURS)
    public void scheduleFixedDelayTask() {
        marketImportService.updatePrice(silpoMarketApi);
        marketImportService.updatePrice(foraMarketApi);
    }
}

```

Файл GoogleDirectionsApi.java

Реалізація функціональної задачі підбору найближчого супермаркету на основі геолокації користувача та його кошика товарів.

```

@Service
public class GoogleDirectionsImpl implements DirectionsApi {
    @Autowired
    private MarketRepository marketRepository;

    @Autowired

```

```

@Qualifier("ForaMarketApi")
private MarketApi foraMarketApi;

@Autowired
@Qualifier("SilpoMarketApi")
private MarketApi silpoMarketApi;

@Override
public List<MarketDist> findMarkets(double lat, double lng) throws
IOException {
    List<MarketDist> foundMarkets = new ArrayList<>();

    Iterable<MarketEntity> entities = marketRepository.findAll();
    List<Market> markets = new ArrayList<>();
    for (MarketEntity entity : entities) {
        if
(entity.getEntityName().equals(foraMarketApi.getMarketInfo().getEntityName()))
        {
            Market fora = foraMarketApi.getMarketInfo();
            markets.add(fora);
        } else if
(entity.getEntityName().equals(silpoMarketApi.getMarketInfo().getEntityName()))
        {
            Market silpo = silpoMarketApi.getMarketInfo();
            markets.add(silpo);
        }
    }
    for (Market market : markets) {
        int dist = getDistance(lat, lng, market.getLatitude(),
market.getLongitude());
        foundMarkets.add(new MarketDist(market, dist));
    }

    foundMarkets.sort(Comparator.comparingInt(MarketDist::getDistance));
    return foundMarkets;
}
}

```

Файл CartService.java

Реалізація функціональної задачі підбору найближчого супермаркету на основі геолокації користувача та його кошика товарів.

```

@Service
@RequiredArgsConstructor
public class CartService implements CartManager {
    private final Cart cart;
    private final AggregatorProductRepository aggregatorProductRepository;

    @Override
    public Set<CartItem> getCart() {
        return cart.getCart();
    }

    @Override
    public void addProd(Long aggregateProductId, Integer quantity) {
        Optional<AggregatorProductEntity> aggregatorProduct =
aggregatorProductRepository.findById(aggregateProductId);
        cart.getCart().add(new CartItem(aggregatorProduct.get(), quantity));
    }

    @Override
    public void updateProdQuantity(Long aggregateProductId, Integer quantity) {
        Optional<AggregatorProductEntity> aggregatorProduct =
aggregatorProductRepository.findById(aggregateProductId);
        if (aggregatorProduct.isPresent()) {
            CartItem productFromCart = cart.getCart().stream()
                .filter(item ->
item.getAggregatorEntity().getId().equals(aggregateProductId))
                .findFirst()
                .orElse(null);
            if (productFromCart != null) {
                productFromCart.setQuantity(quantity);
            } else {
                cart.getCart().add(new CartItem(aggregatorProduct.get(),
quantity));
            }
        }
    }

    @Override
    public void removeProd(Long aggregateProductId) {

```

```

        cart.getCart().removeIf(cartItem ->
cartItem.getAggregatorEntity().getId().equals(aggregateProductId));
    }
}

```

Файл cart.html

Реалізація функціональної задачі підбору найближчого супермаркету на основі геолокації користувача та його кошика товарів.

```

<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Корзина продуктів</title>

    <link
href="https://cdn.jsdelivrivr.net/npm/bootstrap@5.3.2/dist/css/bootstrap.min.css"
rel="stylesheet"

integrity="sha384-T3c6CoIi6uLrA9TneNEoa7RxnatzjcDSCmG1MXxSR1GAsXEV/Dwwykc2MPK8M
2HN" crossorigin="anonymous">

    <link
rel="stylesheet"
href="https://unpkg.com/leaflet@1.9.4/dist/leaflet.css"
integrity="sha256-p4NxAoJBhIIN+hmNHzRCf9tD/miZyoHS5obTRR9BMY="
crossorigin=""/>
    <script src="https://unpkg.com/leaflet@1.9.4/dist/leaflet.js"
integrity="sha256-20nQCchB9co0qIjJZRGuk2/Z9VM+kNiyxNV1lvTlZBo="
crossorigin=""></script>
    <style>
        input::-webkit-outer-spin-button,
        input::-webkit-inner-spin-button {
            -webkit-appearance: none;
        }

        .total-row {
            font-weight: bold;
            border-top: 1px solid #dee2e6;
        }
    </style>
</head>
<body>
<div th:insert="~{aggregate-header}:: header}" style="margin-left:
-1.5rem"></div>

```

```

<h2 style="margin-left: 3.8rem; margin-top: 1rem; margin-bottom: 1.5rem">Корзина продуктів</h2>
<table class="table table-borderless" style="margin-left: 3.8rem; width: 1000px" id="cartTable">
  <thead>
    <tr>
      <th style="width: 50px"></th>
      <th style="width: 300px"></th>
      <th style="width: 70px"></th>
      <th style="width: 130px"></th>
      <th class="text-center market-column" th:each="market, iterStat : ${markets}"
        th:data-market-index="${iterStat.index}" style="width: 170px">
        
      </th>
      <th style="width: 50px"></th>
    </tr>
  </thead>
  <tbody>
    <tr th:each="cartItem, itemStat : ${cart}"
      th:data-row-index="${itemStat.index}">
      <td class="align-middle">
      </td>
      <td class="align-middle"><a
        th:text="${cartItem.aggregatorEntity.name}"></a></td>
      <td class="align-middle text-center"><a
        th:text="${cartItem.aggregatorEntity.unit}"></a></td>
      <td class="align-middle">
        <form th:action="@{/cart/updateProduct}" method="post" th:id="'form'
          + ${cartItem.aggregatorEntity.id}">
          <input type="hidden" name="productId"
            th:value="${cartItem.aggregatorEntity.id}"/>
          <div class="input-group input-group-sm" style="max-width: 120px;
            margin: 0 auto;">
            <button class="submit" type="button"
              onclick="decrQuantity(this)"
              style="border-top-left-radius: 5px; border-bottom-left-radius: 5px; border: 1px
                solid grey; background-color: #f0f1f2;">
              <svg xmlns="http://www.w3.org/2000/svg" width="14"
                height="14" fill="currentColor"
                class="bi bi-dash" viewBox="0 0 16 16">

```

```

<path d="M4 8a.5.5 0 0 1 .5-.5h7a.5.5 0 0 1 0
1h-7A.5.5 0 0 1 4 8"/>
</svg>
</button>
<input name="quantity" min="1" type="number" step="1"
th:value="${cartItem.quantity}"
class="form-control text-center quantity-input"
onchange="submitQuantityForm(this)"
style="border-color: grey">
<button class="submit" type="button"
onclick="incQuantity(this)"
style="border-top-right-radius: 5px; border-bottom-right-radius: 5px; border:
1px solid grey; background-color: #f0f1f2;">
<svg xmlns="http://www.w3.org/2000/svg" width="14"
height="14" fill="currentColor"
class="bi bi-plus" viewBox="0 0 16 16">
<path d="M8 4a.5.5 0 0 1 .5.5v3h3a.5.5 0 0 1 0
1h-3v3a.5.5 0 0 1-1 0v-3h-3a.5.5 0 0 1 0-1h3v-3A.5.5 0 0 1 8 4"/>
</svg>
</button >
</div>
</form>
</td>
<td class="align-middle text-center"
th:each="market, iterStat : ${markets}"
th:data-market-index="${iterStat.index}"
<div
th:with="productEntity=${cartItem.getProductEntity(market.id)}">
<a th:if="${productEntity == null or
!productEntity.isAvailable}"
class="product-price text-danger text-decoration-none"
data-price="0">Немає в наявності</a>
<a th:if="${productEntity != null and
productEntity.isAvailable}"
th:text="${productEntity.price + '€'}"
th:attr="data-price=${productEntity.price}"
class="product-price d-none"></a>
<div th:if="${productEntity != null and
productEntity.isAvailable}" class="product-total">
<span class="total-price"></span><span>€</span>
</div>
</div>
</td>

```

```

<td class="align-middle text-center">
    <form th:id="'form' + ${cartItem.aggregatorEntity.id}"
th:action="@{/cart/removeProduct}" method="post">
        <input type="hidden" name="productId"
th:value="${cartItem.aggregatorEntity.id}"/>
        <button type="submit" class="btn remove-product-btn"
style="border: none; background: none"
th:id="'btn' + ${cartItem.aggregatorEntity.id}">
            <svg th:id="'icon' + ${cartItem.aggregatorEntity.id}"
viewBox="0 0 24 24" fill="none"
xmlns="http://www.w3.org/2000/svg" width="30"
height="30"
style="transition: color 0.3s ease;">
                <g id="SVGRepo_bgCarrier" stroke-width="0"></g>
                <g id="SVGRepo_tracerCarrier" stroke-linecap="round"
stroke-linejoin="round"></g>
                <g id="SVGRepo_iconCarrier">
                    <path fill-rule="evenodd" clip-rule="evenodd"
d="M10.3094 2.25002H13.6908C13.9072 2.24988
14.0957 2.24976 14.2737 2.27819C14.977 2.39049 15.5856 2.82915 15.9146
3.46084C15.9978 3.62073 16.0573 3.79961 16.1256 4.00494L16.2373 4.33984C16.2562
4.39653 16.2616 4.41258 16.2661 4.42522C16.4413 4.90933 16.8953 5.23659 17.4099
5.24964C17.4235 5.24998 17.44 5.25004 17.5001 5.25004H20.5001C20.9143 5.25004
21.2501 5.58582 21.2501 6.00004C21.2501 6.41425 20.9143 6.75004 20.5001
6.75004H3.5C3.08579 6.75004 2.75 6.41425 2.75 6.00004C2.75 5.58582 3.08579
5.25004 3.5 5.25004H6.50008C6.56013 5.25004 6.5767 5.24998 6.59023
5.24964C7.10488 5.23659 7.55891 4.90936 7.73402 4.42524C7.73863 4.41251 7.74392
4.39681 7.76291 4.33984L7.87452 4.00496C7.94281 3.79964 8.00233 3.62073 8.08559
3.46084C8.41453 2.82915 9.02313 2.39049 9.72643 2.27819C9.90445 2.24976 10.093
2.24988 10.3094 2.25002ZM9.00815 5.25004C9.05966 5.14902 9.10531 5.04404
9.14458 4.93548C9.1565 4.90251 9.1682 4.86742 9.18322 4.82234L9.28302
4.52292C9.37419 4.24941 9.39519 4.19363 9.41601 4.15364C9.52566 3.94307 9.72853
3.79686 9.96296 3.75942C10.0075 3.75231 10.067 3.75004 10.3553
3.75004H13.6448C13.9331 3.75004 13.9927 3.75231 14.0372 3.75942C14.2716 3.79686
14.4745 3.94307 14.5842 4.15364C14.605 4.19363 14.626 4.2494 14.7171
4.52292L14.8169 4.82216L14.8556 4.9355C14.8949 5.04405 14.9405 5.14902 14.992
5.25004H9.00815Z"
fill="#7e1515"></path>
                    <path d="M5.91509 8.45015C5.88754 8.03685 5.53016
7.72415 5.11686 7.7517C4.70357 7.77925 4.39086 8.13663 4.41841 8.54993L4.88186
15.5017C4.96736 16.7844 5.03642 17.8205 5.19839 18.6336C5.36679 19.4789 5.65321
20.185 6.2448 20.7385C6.8364 21.2919 7.55995 21.5308 8.4146 21.6425C9.23662
21.7501 10.275 21.7501 11.5606 21.75H12.4395C13.7251 21.7501 14.7635 21.7501

```

```

15.5856 21.6425C16.4402 21.5308 17.1638 21.2919 17.7554 20.7385C18.347 20.185
18.6334 19.4789 18.8018 18.6336C18.9638 17.8206 19.0328 16.7844 19.1183
15.5017L19.5818 8.54993C19.6093 8.13663 19.2966 7.77925 18.8833 7.7517C18.47
7.72415 18.1126 8.03685 18.0851 8.45015L17.6251 15.3493C17.5353 16.6971 17.4713
17.6349 17.3307 18.3406C17.1943 19.025 17.004 19.3873 16.7306 19.6431C16.4572
19.8989 16.083 20.0647 15.391 20.1552C14.6776 20.2485 13.7376 20.25 12.3868
20.25H11.6134C10.2626 20.25 9.32255 20.2485 8.60915 20.1552C7.91715 20.0647
7.54299 19.8989 7.26958 19.6431C6.99617 19.3873 6.80583 19.025 6.66948
18.3406C6.52892 17.6349 6.46489 16.6971 6.37503 15.3493L5.91509 8.45015Z"
    fill="#7e1515"></path>
    <path d="M9.42546 10.2538C9.83762 10.2125 10.2052
10.5133 10.2464 10.9254L10.7464 15.9254C10.7876 16.3376 10.4869 16.7051 10.0747
16.7463C9.66256 16.7875 9.29503 16.4868 9.25381 16.0747L8.75381 11.0747C8.7126
10.6625 9.01331 10.295 9.42546 10.2538Z"
    fill="#7e1515"></path>
    <path d="M14.5747 10.2538C14.9869 10.295 15.2876
10.6625 15.2464 11.0747L14.7464 16.0747C14.7052 16.4868 14.3376 16.7875 13.9255
16.7463C13.5133 16.7051 13.2126 16.3376 13.2538 15.9254L13.7538 10.9254C13.795
10.5133 14.1626 10.2125 14.5747 10.2538Z"
    fill="#7e1515"></path>
  </g>
</svg>
</button>
</form>
</td>
</tr>
<tr class="total-row">
  <td colspan="4" class="text-end pe-3">Загальна сума:</td>
  <td class="text-center" th:each="market, iterStat : ${markets}">
    <span id="marketTotal" th:id="'marketTotal' +
${iterStat.index}">0</span>
  </td>
<td></td>
</tr>
</tbody>
</table>
<p class="d-inline-flex gap-1">
  <a class="me-2 badge rounded-pill bg-none text-dark"
data-bs-toggle="collapse" href="#marketDist" role="button"
aria-expanded=true
aria-controls="marketDist"
style="display: flex; border: 1px solid grey; background-color: #f0f1f2;
height: 2.3rem; align-items: center;

```

```

        text-decoration: none; font-weight: normal; font-size: small;
margin-left: 3.8rem">Найближчий магазин</a>
</p>
<div class="collapse" th:classappend="{expanded ? 'show' : ''}"
id="marketDist">
    <div class="card card-body" style="border: none">
        <b style="margin-left: 3rem">Оберіть точку на карті:</b>
        <form id="coords-form" style="margin-left: 3rem; margin-top: 1rem;
margin-bottom: 1rem"
            th:action="@{/markets/distances}" method="post">
            <div class="row g-3">
                <div class="col-auto">
                    <input type="text" class="form-control" id="lat" name="lat"
placeholder="Широта"
                        th:value="{lat != null ? lat : '50.445090'}">
                </div>
                <div class="col-auto">
                    <input type="text" class="form-control" id="lng" name="lng"
placeholder="Довгота"
                        th:value="{lng != null ? lng : '30.520731'}">
                </div>
                <div class="col-auto">
                    <button type="submit" class="btn btn-light mb-3"
style="--bs-btn-border-color: #000000;
                                --bs-btn-hover-border-color: #000000;
--bs-btn-active-border-color: #000000;
                                --bs-btn-disabled-border-color: #000000;">OK!
                </button>
            </div>
        </div>
        <div id="map" style="height: 490px; width: 490px;"></div>
    </form>
    <div th:if="{nearbyMarkets}">
        <b style="margin-left: 3rem; margin-top: 1rem">Магазини поблизу:</b>
        <div th:each="marketDist : {nearbyMarkets}" class="row"
style="width: 850px; margin-top: 1rem">
            <div class="col" style="max-width: 55px; margin-left: 3rem;
margin-top: 0.5rem">
                
            </div>
            <div class="col" style="margin-left: 4rem">
                <a th:text="{marketDist.market.address}"></a>

```

```

        </div>
        <div class="col" style="margin-left: -8rem">
            <span th:text="{marketDist.distance + ' M'}"></span>
        </div>
    </div>
</div>
</div>
</div>
<script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.2/dist/js/bootstrap.bundle.min.
js"

integrity="sha384-C6RzsynM9kWDrMNeT87bh95OGNyZPhcTNXj1NW7RuBCsyN/o0jlpcV8Qyq46c
DfL"

crossorigin="anonymous"></script>
</body>
</html>

```

Факультет інформатики та обчислювальної техніки
Кафедра інформатики та програмної інженерії

“ЗАТВЕРДЖЕНО”

Завідувач кафедри

_____ Едуард ЖАРІКОВ

“ ____ ” _____ 2025 р.

**ВЕБЗАСТОСУНОК ПІДБОРУ СУПЕРМАРКЕТУ ЗА КОШИКОМ
ПРОДУКТІВ ТА МІСЦЕМ РОЗТАШУВАННЯ**

Програма та методика тестування

КПІ.ІП-1501.045440.04.51

“ПОГОДЖЕНО”

Керівник проєкту:

_____ Олесь КОВТУНЕЦЬ

Нормоконтроль:

_____ Тетяна ШУЛЬКЕВИЧ

Виконавець:

_____ Мадіна АДЖИГЕЛЬДІЄВА

Київ – 2025

ЗМІСТ

1 ОБ'ЄКТ ВИПРОБУВАНЬ	3
2 МЕТА ТЕСТУВАННЯ	4
3 МЕТОДИ ТЕСТУВАННЯ	5
4 ЗАСОБИ ТА ПОРЯДОК ТЕСТУВАННЯ	6

1 ОБ'ЄКТ ВИПРОБУВАНЬ

Об'єктом випробування є розроблене програмне забезпечення, а саме: вебзастосунок підбору супермаркету за кошиком продуктів та місцем розташування.

Тестування виконується для всього функціоналу клієнтської та серверної частин реалізованої системи.

Програмне забезпечення є веборієнтованим і доступне з віддаленого серверу. Використання вебзастосунку можливе з будь-якого браузеру, що підтримує виконання скриптів мовою JavaScript.

Розробка вебзастосунку виконувалась у браузері Google Chrome 136.0.7103.114, тестування роботи – Safari 19618.3.11.11.5.

2 МЕТА ТЕСТУВАННЯ

Метою тестування є наступне:

- перевірка правильності роботи програмного забезпечення відповідно до функціональних вимог;
- перевірка збереження даних;
- перевірка сумісності вебзастосунку з останніми версіями сучасних браузерів (Google Chrome, Safari);
- знаходження проблем, помилок і недоліків з метою їх усунення;
- перевірка зручності графічного інтерфейсу.

3 МЕТОДИ ТЕСТУВАННЯ

Для тестування програмного забезпечення використовуються такі методи:

- динамічне тестування – застосовується в процесі виконання програми. Коректність програмного засобу перевіряється на певній кількості тестів. При прогоні кожного з них збираються та аналізуються дані про проблеми та помилки в роботі програми;
- системне тестування – перевіряється усе програмне забезпечення в цілому;
- мануальне тестування – тестування без використання автоматизації, тест-кейси пише особа, що тестує програмне забезпечення.

4 ЗАСОБИ ТА ПОРЯДОК ТЕСТУВАННЯ

Тестування виконується мануально з використанням наскрізного тестування з метою знаходження помилок та недоліків як у функціональній частині програмного забезпечення, так і в зручності користування. Порядок проведення тестування буде наступним:

- динамічне тестування на відповідність функціональним вимогам;
- тестування на мобільних пристроях з різною роздільною здатністю екрану;
- тестування на мобільних пристроях з різною версією операційної системи;
- тестування на виведення повідомлень про помилку, коли це необхідно;
- тестування зміни орієнтації екрану;
- тестування інтерфейсу користувача;
- тестування зручності використання;
- тестування в різних браузерах.

Факультет інформатики та обчислювальної техніки
Кафедра інформатики та програмної інженерії

“ЗАТВЕРДЖЕНО”

Завідувач кафедри

_____ Едуард ЖАРІКОВ

“ ____ ” _____ 2025 р.

**ВЕБЗАСТОСУНОК ПІДБОРУ СУПЕРМАРКЕТУ ЗА КОШИКОМ
ПРОДУКТІВ ТА МІСЦЕМ РОЗТАШУВАННЯ**

Керівництво користувача

КПІ.ІП-1501.045440.05.34

“ПОГОДЖЕНО”

Керівник проекту:

_____ Олесь КОВТУНЕЦЬ

Нормоконтроль:

_____ Тетяна ШУЛЬКЕВИЧ

Виконавець:

_____ Мадіна АДЖИГЕЛЬДІЄВА

Київ – 2025

ЗМІСТ

1 ПРИЗНАЧЕННЯ ПРОГРАМИ	3
2 ПІДГОТОВКА ДО РОБОТИ З ПРОГРАМНИМ ЗАБЕЗПЕЧЕННЯМ	4
2.1 Системні вимоги для коректної роботи	4
2.2 Завантаження застосунку	4
2.3 Перевірка коректної роботи	4
3 ВИКОНАННЯ ПРОГРАМИ	5

1 ПРИЗНАЧЕННЯ ПРОГРАМИ

«AGG» – це вебзастосунок, призначений для знаходження оптимального супермаркету на основі кошика продуктів і місця розташування користувача.

2 ПІДГОТОВКА ДО РОБОТИ З ПРОГРАМНИМ ЗАБЕЗПЕЧЕННЯМ

2.1 Системні вимоги для коректної роботи

Для успішної роботи даного застосунку необхідне виконання наступних вимог:

- програмне забезпечення повинно працювати під управлінням операційних систем сімейства WIN32 (Windows XP, Windows NT і т.д.) або Unix;
- мінімальні вимоги до складу і параметрів технічних засобів:
 - 1) Server Cloud Compute;
 - 2) CPU & Storage Technology – Intel Regular Performance;
 - 3) Операційна система Ubuntu 22.04 LTS x64;
 - 4) Розмір серверу 25 Гб SSD.
- наявність доступу до Інтернету.

2.2 Завантаження застосунку

Покрокова інструкція збірки та виконання програмного забезпечення доступна в репозиторії вебзастосунку.

2.3 Перевірка коректної роботи

Після збірки та запуску програмного забезпечення в пошукову стрічку вводиться відповідне посилання і виконується пошук. Після завантаження вебзастосунку, користувач побачить сторінку зі списком категорій агрегатору.

3 ВИКОНАННЯ ПРОГРАМИ

При запуску вебзастосунку користувачу буде відображено головну сторінку, де представлено список категорій продуктів. Дана сторінка дозволяє фільтрувати по категорії для звуження пошуку потрібного продукту (рис. 3.1).

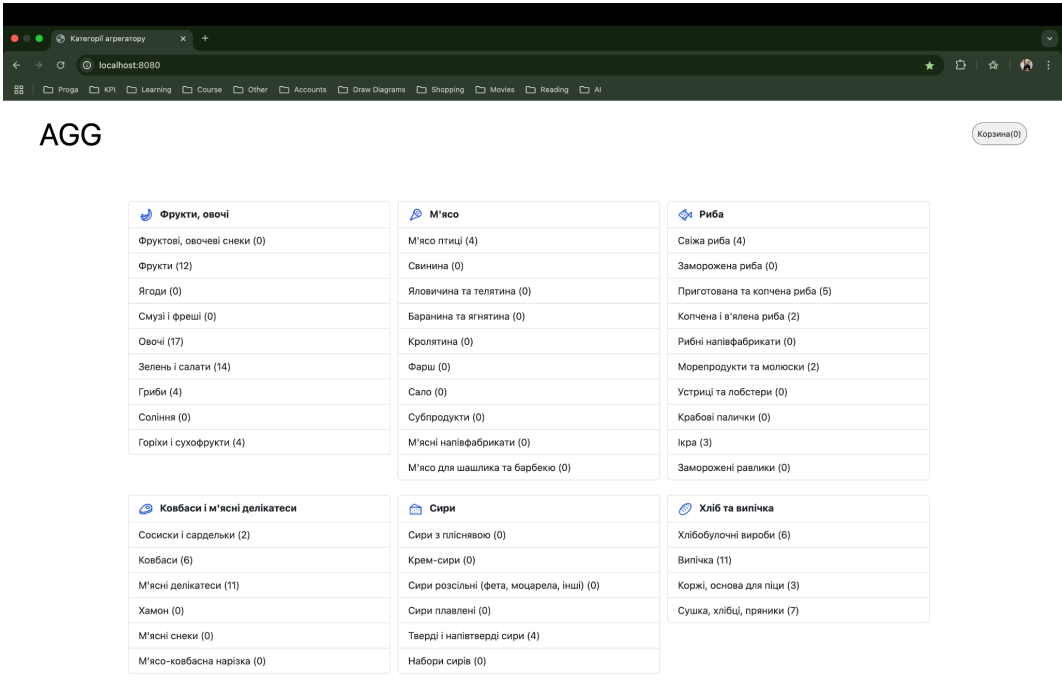


Рисунок 3.1 – Список категорій продуктів

При натисканні на обрану категорію здійснюється перехід до списку продуктів, відфільтрованих за цією категорією (рис. 3.2).

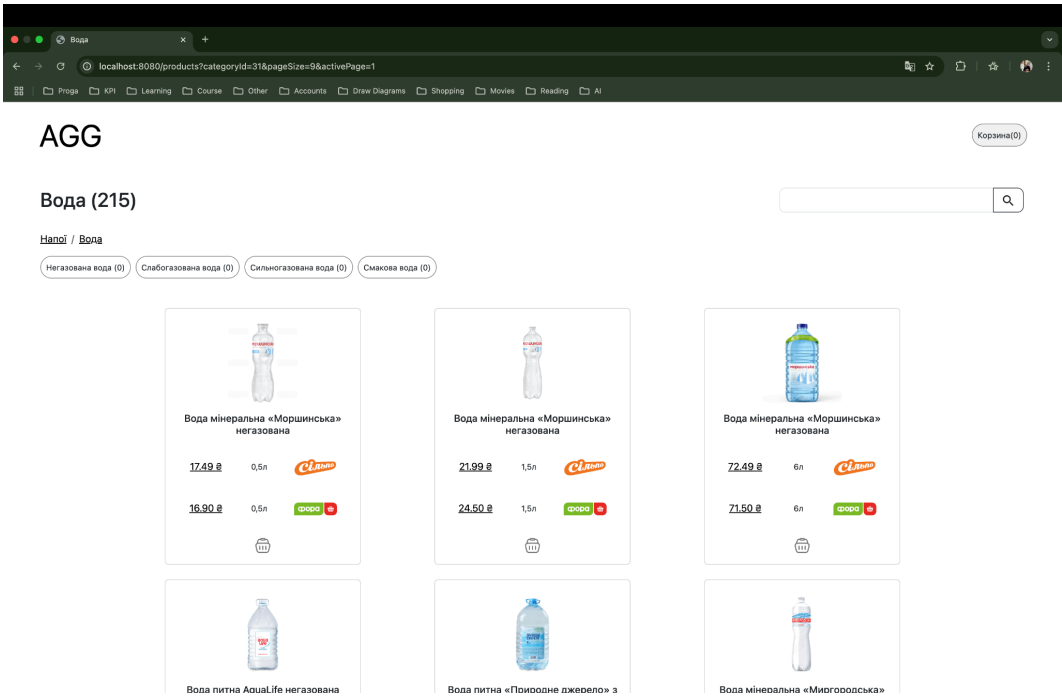


Рисунок 3.2 – Список продуктів, відфільтрованих за категорією

На сторінці зі списком продуктів, відфільтрованих за категорією, доступна функція фільтрації за назвою продукту (рис. 3.3).

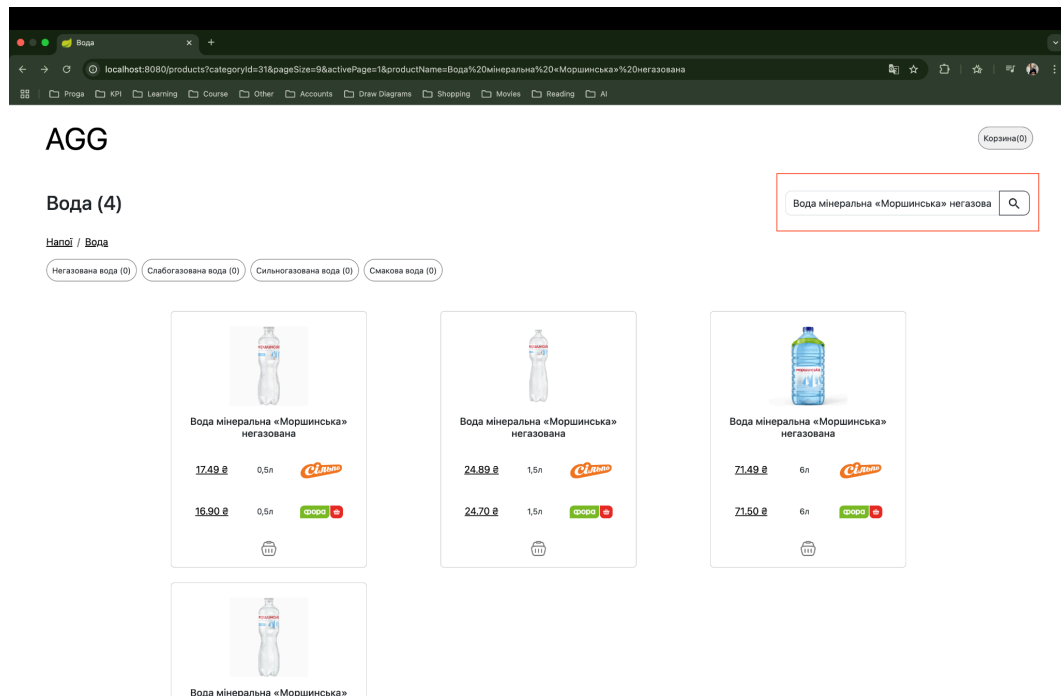


Рисунок 3.3 – Список продуктів, відфільтрованих за назвою

Передбачено формування кошику продуктів. При знаходженні потрібного продукту натискання на іконку кошика додає його в корзину (рис. 3.4). Повторне натискання на ту саму іконку призводить перехід безпосередньо до корзини продуктів (рис. 3.5).

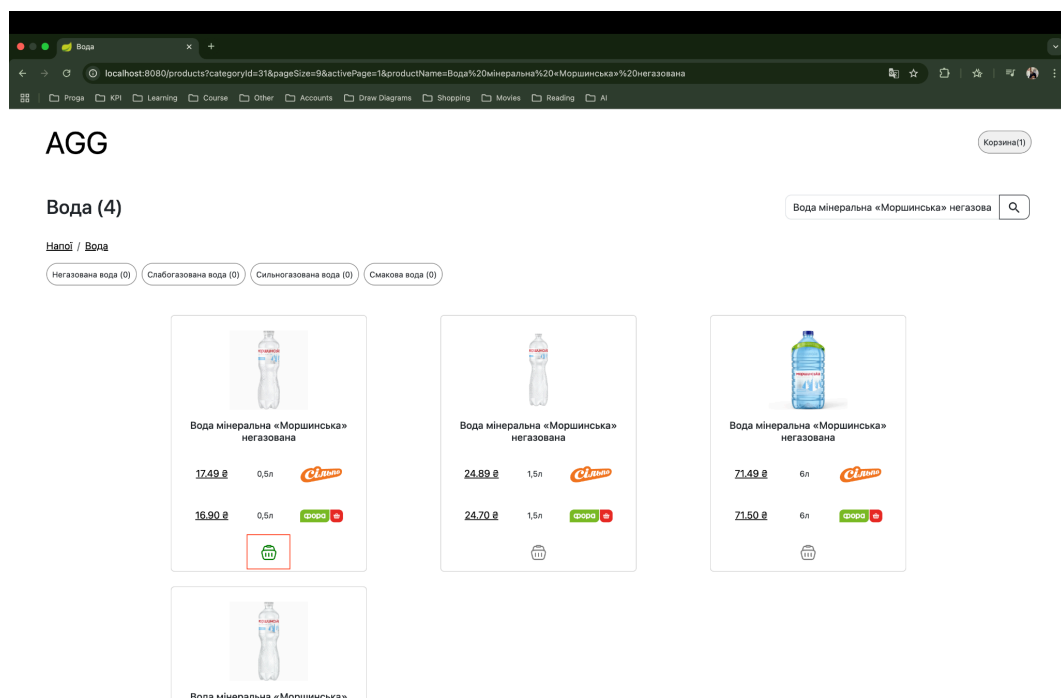


Рисунок 3.4 – Додавання продукту до кошика продуктів

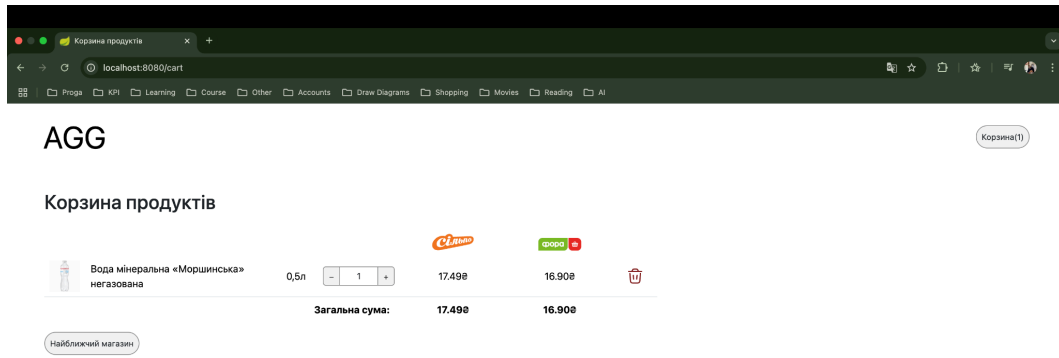


Рисунок 3.5 – Формування кошику продуктів

На сторінці кошика продуктів є можливість коригувати кількість одиниць продукту і система автоматично підраховує та надає загальну суму для порівняння цін у різних супермаркетах (рис. 3.6).

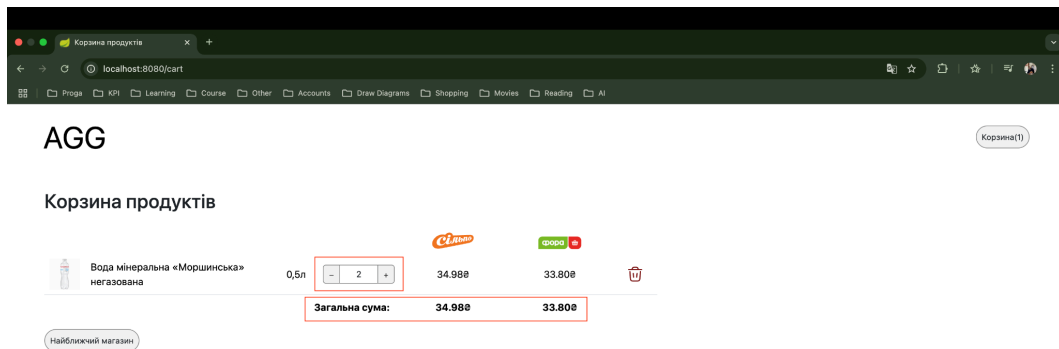


Рисунок 3.6 – Загальна вартість кошику продуктів по супермаркетах

Пошук найближчого супермаркету виконується шляхом вибору свого місця розташування на карті та натискання кнопки «ОК!». Після цього

відображається карта з позначеними найближчими супермаркетами і список, відсортований за зростанням відстані до них (рис. 3.7).

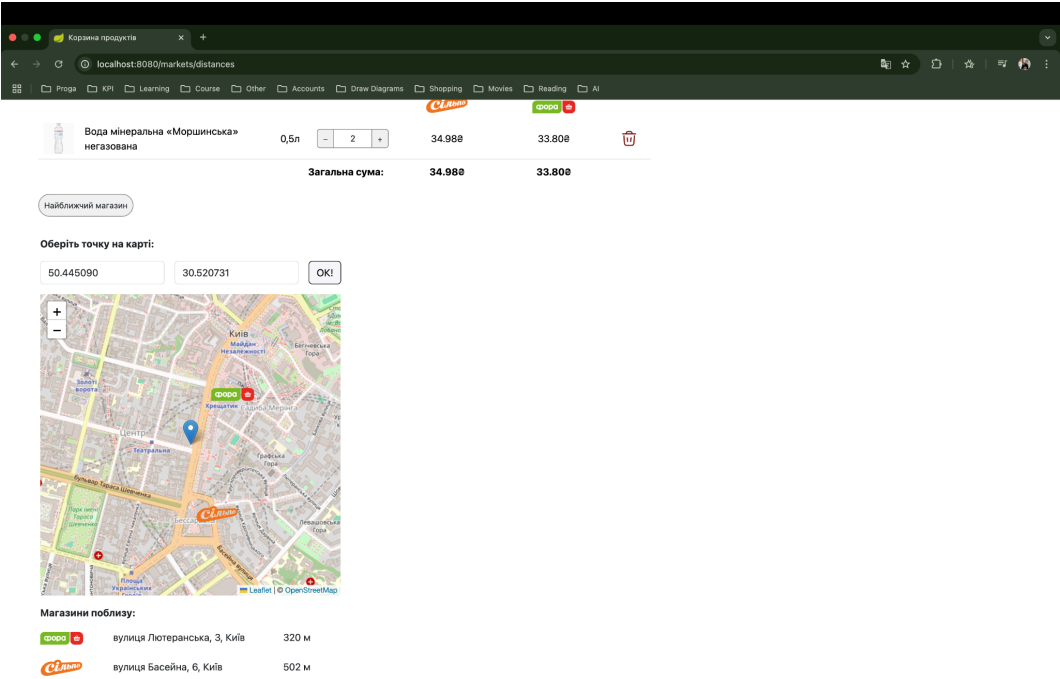


Рисунок 3.7 – Найближчий супермаркет

Факультет інформатики та обчислювальної техніки
Кафедра інформатики та програмної інженерії

“ЗАТВЕРДЖЕНО”

Завідувач кафедри

_____ Едуард ЖАРІКОВ

“ ____ ” _____ 2025 р.

**ВЕБЗАСТОСУНОК ПІДБОРУ СУПЕРМАРКЕТУ ЗА КОШИКОМ
ПРОДУКТІВ ТА МІСЦЕМ РОЗТАШУВАННЯ**

Графічний матеріал

КПІ.ІП-1501.045440.06.99

“ПОГОДЖЕНО”

Керівник проекту:

_____ Олесь КОВТУНЕЦЬ

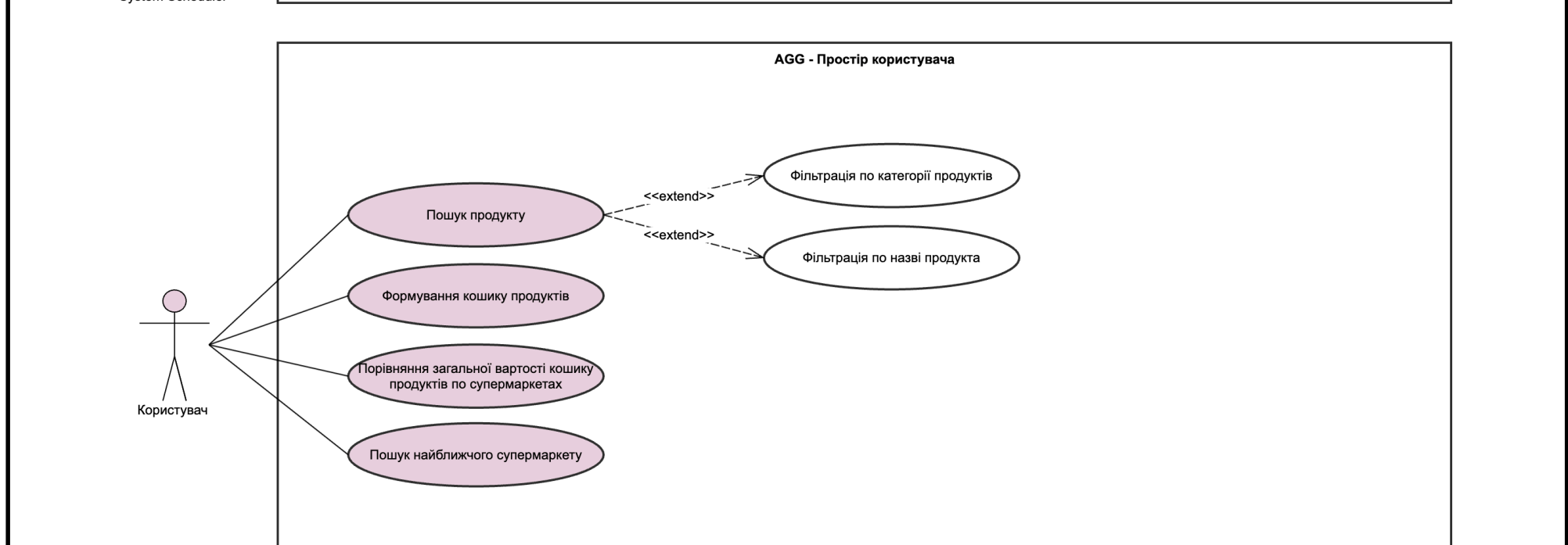
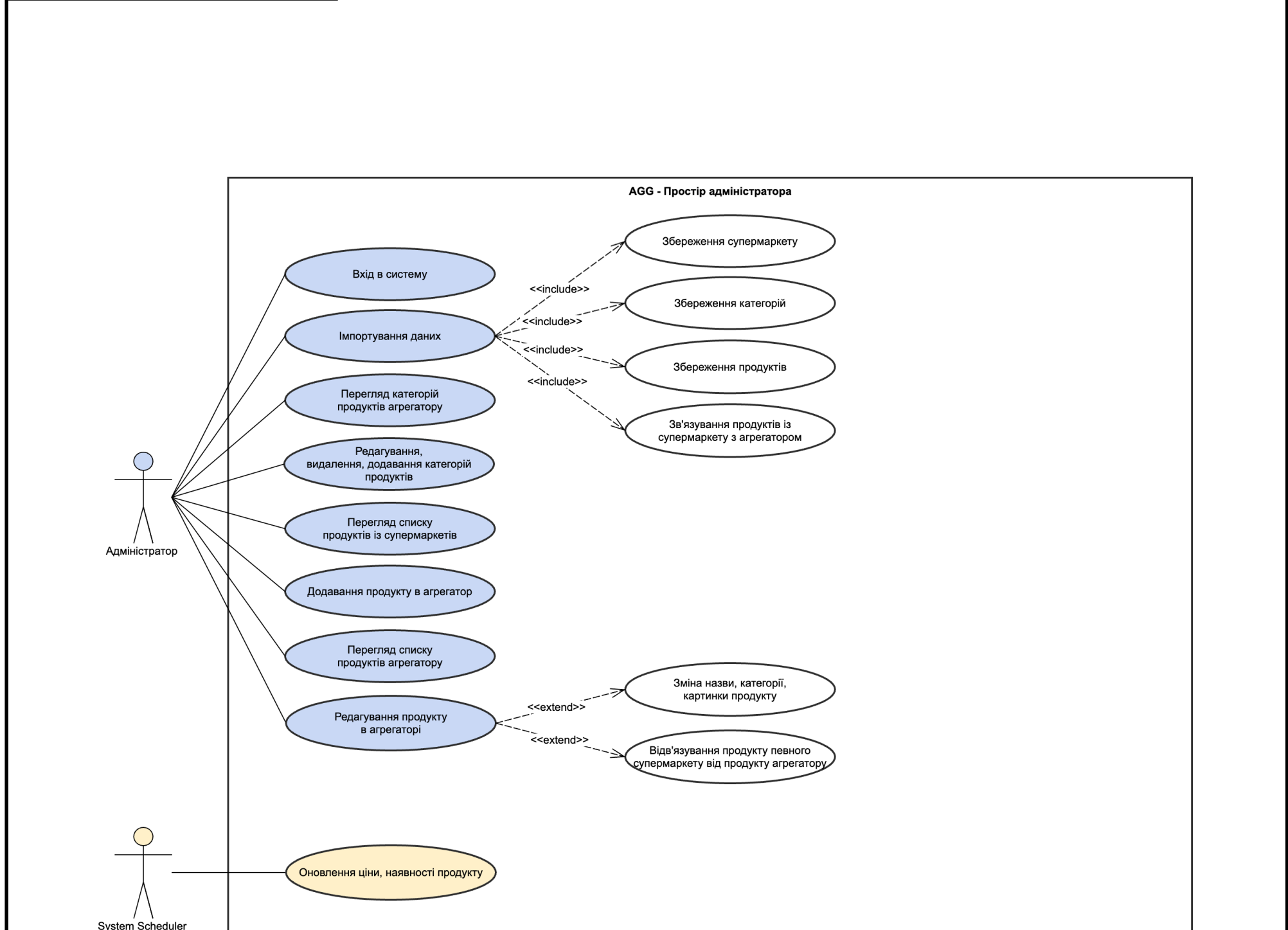
Нормоконтроль:

_____ Тетяна ШУЛЬКЕВИЧ

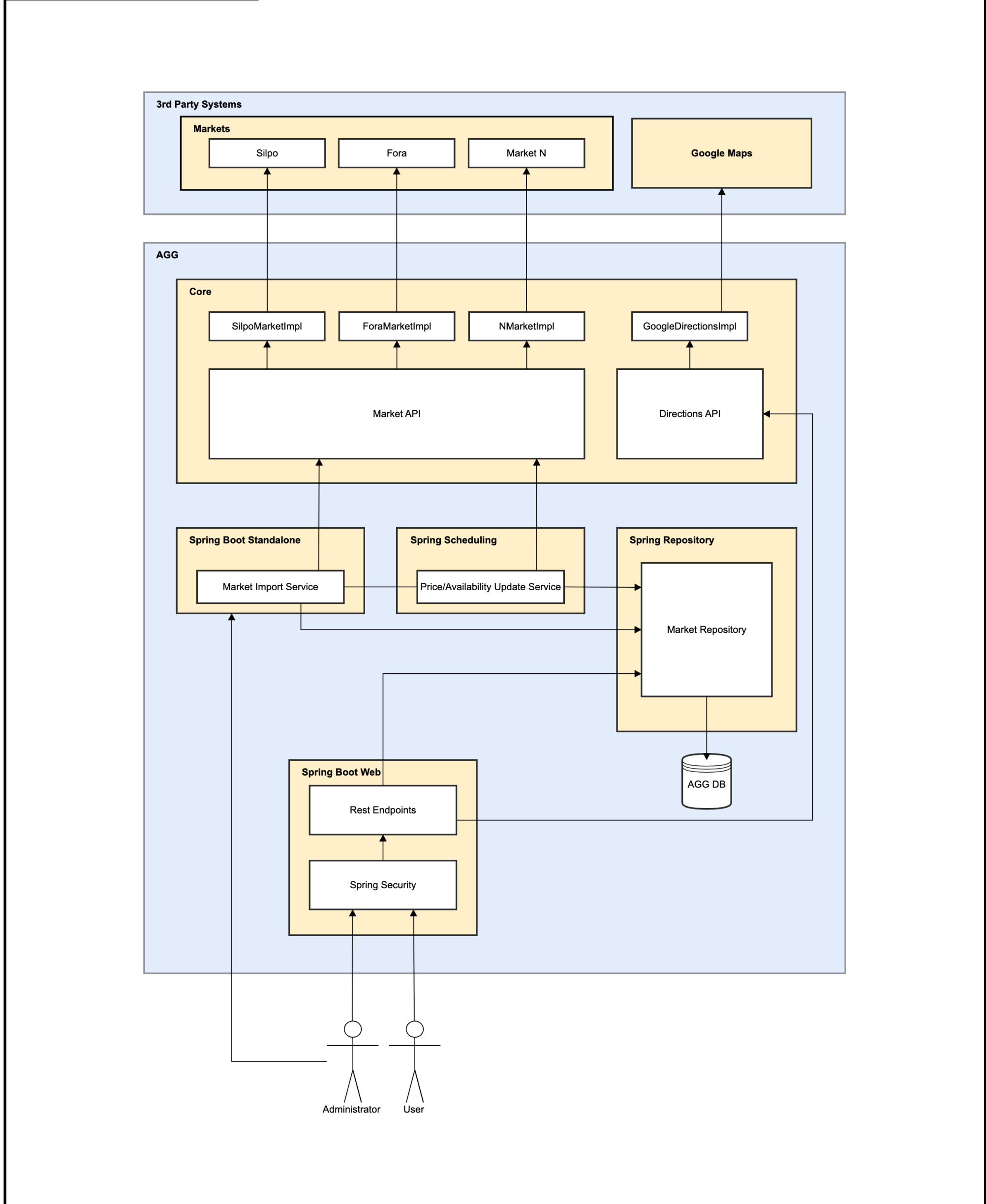
Виконавець:

_____ Мадіна АДЖИГЕЛЬДІЄВА

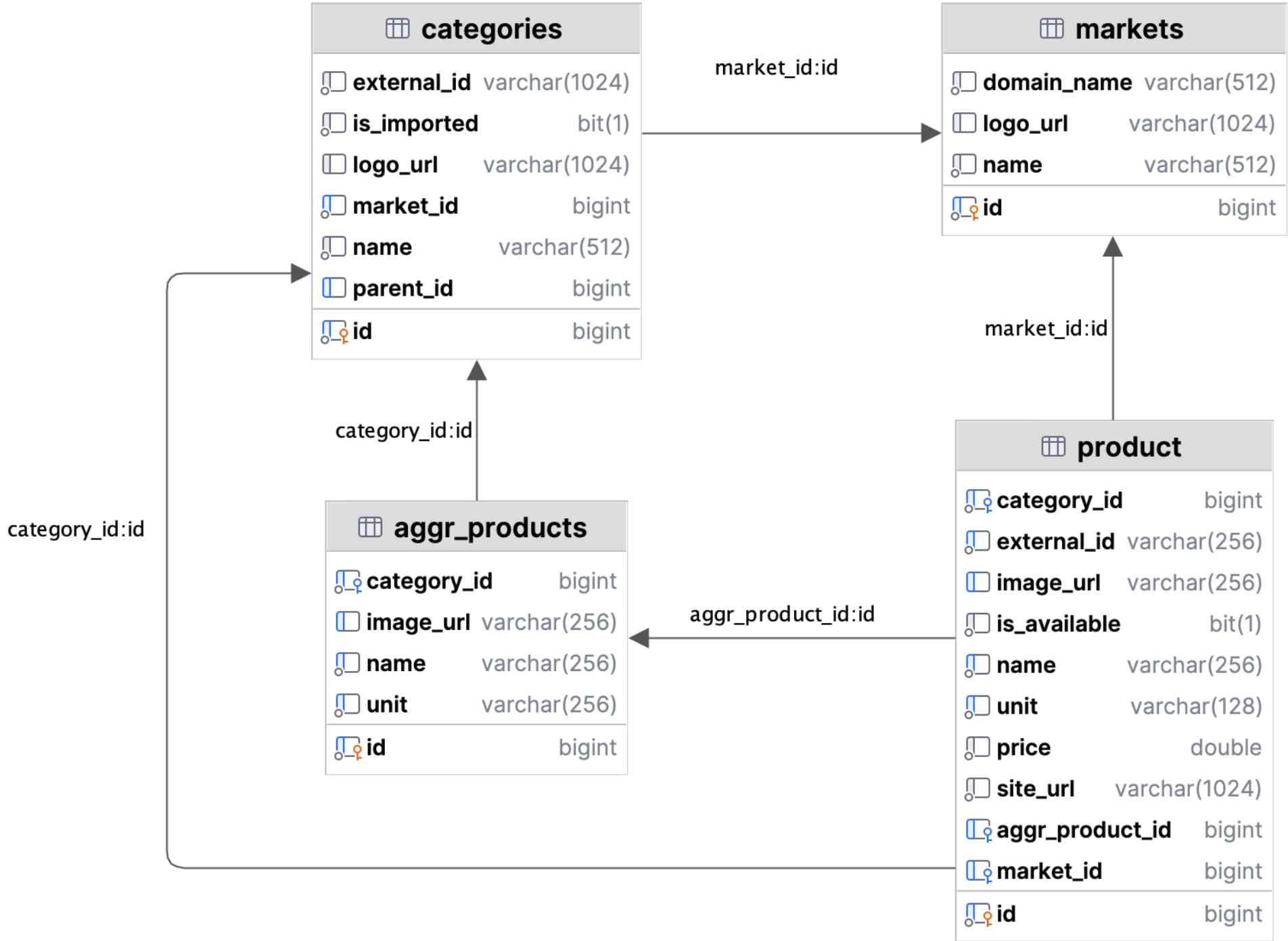
Київ – 2025



					КПІ.ІП-1501.045440.06.99.ССВ										
Зм.	Арк.	№ докум.	Підп.	Дата	Схема структурна варіантів використання						Лит.		Маса	Масштаб	
Розробив		Аджигельдієва М.А.													
Перевірів		Ковтунець О.В.													
Т. контр.															
Н. контр.		Шулькевич Т.В.			Вебзастосунок підбору супермаркету за кошиком продуктів та місцем розташування						Аркуш 1		Аркушів 1		
Затвердив		Жаріков Е.В.													
КПІ ім.Ігоря Сікорського Кафедра ІПІ гр. ІП-15															



					КПІ.ІП-1501.045440.06.99.CCM				
					Схема структурна компонентів програмного забезпечення	Лит.		Маса	Масштаб
Зм.	Арк.	№ докум.	Підп.	Дата					
Розробив		Аджигельдієва М.А.							
Перевірив		Ковтунець О.В.							
Т. контр.						Аркуш 1		Аркушів 1	
					Вибзастосунок підбору супермаркету за кошиком продуктів та місцем розташування	КПІ ім.Ігоря Сікорського Кафедра ІПІ гр. ІП-15			
Н. контр.		Шулькевич Т.В.							
Затвердив		Жаріков Е.В.							



					КПІ.ІП-1501.045440.06.99.СБД								
					Схема бази даних				Літера		Маса	Масштаб	
Зм.	Арк.	№ документа	Підпис	Дата									
Розробив		Аджигельдієва М.А.											
Перевірів		Ковтунець О.В.											
Т. контр.									Аркуш 1		Аркушів 1		
Н. контр.		Шулькевич Т.В.			Вебзастосунок підбору супермаркету за кошиком продуктів та місцем розташування				КПІ ім.Ігоря Сікорського Кафедра ІПІ гр. ІП-15				
Затвердив		Жаріков Е.В.											