

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

ПОЯСНЮВАЛЬНА ЗАПИСКА

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

«Інтернет магазин кави з використанням мікросервісів»

Виконав студент групи ІПЗ-20-1
ТКАЧУК Микита Андрійович

Керівник роботи:
МІНТІЙ Ірина Сергіївна

Консультант:
БАКАЛЮК Тетяна Анатоліївна

Рецензент:
МОРОЗОВ Андрій Васильович

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

«ЗАТВЕРДЖУЮ»

Зав. кафедри інженерії
програмного забезпечення

Тетяна Вакалюк

«26» лютого 2024р.

ЗАВДАННЯ
на кваліфікаційну роботу

Здобувач вищої освіти: **ТКАЧУК Микита Андрійович**

Керівник роботи: **к.п.н., доц., доц. каф. ІІЗ МІНТІЙ Ірина Сергіївна**

Тема роботи: **«Інтернет магазин кави з використанням мікросервісів»**,
затверджена Наказом закладу вищої освіти від **«26» лютого 2024 р., №74/с**
Вихідні дані для роботи: **Інтернет магазин кави з використанням мікросервісів**

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

Розділ	Консультант	Завдання видав	Завдання прийняв
1	Мінтій І.С.	02.03.2024	02.03.2024
2	Мінтій І.С.	04.04.2024	04.04.2024
3	Мінтій І.С.	02.05.2024	02.05.2024

РЕФЕРАТ

Випускна кваліфікаційна робота бакалавра складається з вебдодатку краудфандингової платформи та пояснювальної записки. Пояснювальна записка до випускної роботи містить 70 сторінок, 51 ілюстрацій та 33 таблиць.

Дипломний проєкт присвячений інтернет магазину для продажу кави з використанням мікросервісів.

Метою дипломного проєкту є підвищення ефективності продажів і можливістю розвивати проєкт у майбутньому.

Об'єктом дослідження є інтернет сайт для продажу товарів.

Предметом дослідження є підвищення ефективності продажів і можливістю розвивати проєкт у майбутньому завдяки мікросервісам.

У першому розділі було розглянуто аналіз вимог до програмного забезпечення, вивчено змістовний опис і аналіз предметної області, а також здійснено дослідження відомих технологій і успішних ІТ проєктів.

В другому розділі було розглянуто моделювання та конструювання програмного забезпечення, включаючи архітектуру програмного забезпечення та аналіз безпеки даних.

В третьому розділі було розглянуто обрані метрики для аналізу якості програмного забезпечення та процеси його тестування.

В четвертому розділі було розглянуто впровадження та супровід програмного забезпечення, а також процеси його розгортання та підтримки.

КЛЮЧОВІ СЛОВА: ІНТЕРНЕТ МАГАЗИН, МІКРОСЕРВІСИ, БАЗА ДАНИХ, УПРАВЛІННЯ.

					ІПЗ.КР.Б-121-23-ПЗ			
Змн.	Арк.	№ докум.	Підпис	Дата	Інтернет магазин кави з використанням мікросервісів	Літ.	Арк.	Аркушів
Розроб.		Ткачук М.А						
Керівник		Мінтій І.С.					3	95
						ФІКТ Гр. ІПЗ-20-1		
Н. контр.								
Зав. каф.								

ABSTRACT

The explanatory note of the diploma project consists of four sections and includes 33 tables, 51 figures, and 32 references, totaling 95 pages.

The diploma project is devoted to an online store for the sale of coffee using microservices.

The goal of the diploma project is to increase sales efficiency and the opportunity to develop the project in the future.

The object of the study is an Internet site for the sale of goods.

The subject of the research is the improvement of sales efficiency and the possibility of developing the project in the future thanks to microservices.

The first section covers the analysis of software requirements, a comprehensive description and analysis of the subject area, as well as research on known technologies and successful IT projects.

The second section discusses software modeling and construction, including software architecture and data security analysis.

The third section examines selected metrics for software quality analysis and its testing processes.

The fourth section addresses the implementation and maintenance of software, as well as deployment and support processes.

KEYWORDS: INTERNET-SHOP, BROWSER, MICROSERVICES, DATABASE, MANAGEMENT.

		Ткачук М.А			ІІЗ.КР.Б-121-24-ІІЗ	Арк.
		Мінтій І.С				4
Змн.	Арк.	№ докум.	Підпис	Дата		

ЗМІСТ

РЕФЕРАТ	3
ABSTRACT	4
ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ	6
ВСТУП.....	7
РОЗДІЛ 1. АНАЛІЗ НАПРЯМКІВ ЗАСТОСУВАННЯ ІНТЕРНЕТ МАГАЗИНУ З ВИКОРИСТАННЯМ МІКРОСЕРВІСІВ	9
1.1 Постановка задачі.....	9
1.2 Аналіз аналогів програмного продукту	12
1.3 Обґрунтування вибору інструментальних засобів та вимоги до апаратного забезпечення	20
Висновки до першого розділу.....	23
РОЗДІЛ 2. ПРОЕКТУВАННЯ ВЕБ-ДОДАТКУ	24
2.1 Визначення варіантів використання та об'єктно-орієнтованої структури системи	24
2.2 Проектування бази даних	30
2.3 Проектування та реалізація алгоритмів роботи системи.....	35
2.4 Архітектура програмного комплексу та його розгортання.....	37
2.5 Реалізація функціоналу інтернет магазину з використанням мікросервісів.	39
Висновки до другого розділу.	51
РОЗДІЛ 3 ІНТЕРФЕЙС ТА ПОРЯДОК РОБОТИ З ВЕБ-ДОДАТКОМ	52
3.1 Порядок встановлення та налаштування параметрів платформи.....	52
3.2 Структура інтерфейсу для користувача інтернет-магазину кави CoffeeShop	53
3.3 Структура інтерфейсу для адміністратора інтернет-магазину кави CoffeeShop	59
3.4 Тестування платформи.....	64
Висновки до третього розділу.....	71
ВИСНОВКИ.....	72
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ.....	73

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

API – прикладний програмний інтерфейс

HTML – HyperText Markup Language

JS – JavaScript

БД – база даних

СУБД – система управління базами даних

ASP – Active Server Pages

ПЗ – програмне забезпечення

SDK – набір для розробки програмного забезпечення

MVC – схема розділення даних додатків і керуючої логіки на три окремі компоненти Model-View-Controller

CRUD – основні операції, що використовуються для управління даними (Create, Read, Update, Delete)

		Ткачук М.А			ІПЗ.КР.Б-121-24-ІПЗ	Арк.
		Мінтій І.С				6
Змн.	Арк.	№ докум.	Підпис	Дата		

ВСТУП

Актуальність теми полягає у створенні інтернет-магазину з використанням мікросервісів та сучасних технологій ASP.NET Core, Azure і Service Bus обумовлена зростаючою популярністю онлайн-магазинів і необхідністю забезпечення високої надійності і масштабованості таких систем.

У сучасних умовах, коли все більше споживачів вибирають зручність онлайн-покупок, онлайн-шопінг стане основним інструментом продажу товарів.

Зростаюче суперництво, яке зростає в комерційному веб-секторі, змушує установи застосовувати креативні стратегії надання послуг, підвищення продуктивності та стабільності операцій. Концепція мікросервісів може створювати архітектури, які є гнучкими, масштабованими та простими в управлінні системами, які швидко пристосовуються до мінливих потреб.

Доступність і зручність онлайн-магазину дуже потрібні споживачам у сучасну епоху — можливість робити покупки будь-коли та будь-де — це розкіш. Коли інтернет-магазин інтегрується з платіжними системами, службами доставки та будь-якими іншими послугами, які можуть знадобитися споживачеві під час процесу покупки, це створює комплексний і зручний досвід покупки. Зростаюча потреба в продуктах вищого рівня підкреслює важливість таких програм: вони дозволяють споживачам легко та швидко зробити вибір щодо продуктів, яким вони віддають перевагу, і в ефективний спосіб.

Метою випускної кваліфікаційної роботи бакалавра є розробка онлайн-магазину із використанням мікросервісів -надати користувачам ефективний інструмент для покупки товарів онлайн, який включає в себе управління каталогом продукції, обробку замовлень, інтеграцію з платіжними системами і службами доставки. Додаток повинен підвищити зручність і якість обслуговування клієнтів, збільшити продажі і підвищити ефективність роботи магазину.

Завдання роботи обумовлює наступні завдання:

- Проведення аналізу вимог до системи інтернет-магазину та оцінка сучасних підходів до розробки онлайн-магазинів.

		Ткачук М.А			ІПЗ.КР.Б-121-24-ІПЗ	Арк.
		Мінтій І.С				
Змн.	Арк.	№ докум.	Підпис	Дата		7

- Визначення архітектури та загальної структури системи, включаючи мікросервісну архітектуру.
- Обґрунтування та вибір засобів реалізації системи, таких як ASP.NET Core, Azure і Service Bus.
- Розробка інформаційного та програмного забезпечення системи, включаючи управління каталогом продукції та обробку замовлень.
- Проектування структурних складових та алгоритмів роботи системи для інтеграції з платіжними системами та службами доставки.
- Реалізація програмного комплексу, зокрема розробка веб-орієнтованої підсистеми для управління інтернет-магазином з використанням мікросервісів.
- Тестування та оптимізація системи для забезпечення високої продуктивності та масштабованості.
- Запуск системи на хмарній платформі Azure та організація технічної підтримки і обслуговування.

Об'єктом дослідження в даному випадку є мова програмування C# та її розширення ASP.NET CORE, а також підхід до архітектури мікросервісів, а також хмарні послуги Azure та служби обміну повідомленнями на службовій шині. За допомогою ASP.NET Core Ви можете створювати високопродуктивні масштабовані веб-додатки, а мікросервіси забезпечують гнучкість і незалежність окремих компонентів системи. Azure надає потужні інструменти для хмарного зберігання та обробки даних, а сервісна шина забезпечує надійну передачу повідомлень між компонентами системи.

Предметом дослідження є використання мови програмування C# ASP.NET Core, мікросервіси, Azure та сервісна шина для розробки онлайн-магазинів. Дослідження охоплює аспекти ефективності та зручності цих технологій для створення функціонального та інтерактивного користувацького інтерфейсу та забезпечення надійної роботи серверної частини програми, а також інтеграції з іншими сервісами та системами для повного управління процесом покупки товарів онлайн.

		Ткачук М.А			ІПЗ.КР.Б-121-24-ІПЗ	Арк.
		Мінтій І.С				8
Змн.	Арк.	№ докум.	Підпис	Дата		

РОЗДІЛ 1. АНАЛІЗ НАПРЯМКІВ ЗАСТОСУВАННЯ ІНТЕРНЕТ МАГАЗИНУ З ВИКОРИСТАННЯМ МІКРОСЕРВІСІВ

1.1 Постановка задачі

Ця робота присвячена створенню інтернет-магазину, побудованого на основі мікросервісної архітектури, що включає такі сервіси, як купони, відстеження кошику, систему лояльності та інші. Сучасні інформаційні технології швидко розвиваються, і з кожним роком з'являється все більше платформ для електронної комерції, які можуть покращувати різні аспекти процесу покупок онлайн. Постійно розробляються нові програмні продукти, що охоплюють дедалі більше аспектів діяльності електронної комерції. Перші програмні продукти для інтернет-магазинів були спрямовані на автоматизацію процесів продажу з метою підвищення ефективності виконання певних завдань. Згодом розробка програмного забезпечення стала не лише потребою великих корпорацій, але й важливою частиною малого та середнього бізнесу. Програмісти почали оптимізувати та автоматизувати процеси у власних магазинах. В результаті з'явилися платформи, орієнтовані на підвищення ефективності продажів, включаючи ті, що допомагають відстежувати кошики, керувати купонами та здійснювати інші функції.

Мікросервісна архітектура - це архітектура, де є кілька мікросервісів або API, які слабо пов'язані між собою і можуть бути розгорнуті незалежно. Вони працюють разом, щоб створити повне кінцеве рішення. [\[1\]](#)

Основна відмінність від монолітної архітектури полягає в тому, що в монолітному додатку існує одне велике рішення, де всі компоненти тісно пов'язані і інтегровані.

Кожен мікросервіс відповідає за свою власну функціональність. Наприклад, у додатку для електронної комерції ми можемо мати окремі мікросервіси для продуктів, купонів, замовлень, кошиків, відправлення електронних листів та API для винагород, який відповідає за відстеження всіх винагород, які отримує користувач при розміщенні замовлень. Також може

		Ткачук М.А			ІПЗ.КР.Б-121-24-ІПЗ	Арк.
		Мінтій І.С				9
Змн.	Арк.	№ докум.	Підпис	Дата		

бути мікросервіс для аутентифікації та авторизації, що використовує .NET Identity для взаємодії з іншими мікросервісами.

Оскільки мікросервіси не тісно пов'язані, вони можуть використовувати різні технології. Один мікросервіс може використовувати SQL Server [2] для бази даних, тоді як інший може використовувати MongoDB або NoSQL. Вони також можуть бути написані на різних мовах програмування, таких як C# [3], JavaScript, Java тощо.

Інтернет-магазини, побудовані на основі мікросервісної архітектури, корисні з різних причин:

- **Масштабованість.**
 - Мікросервіси дозволяють легко масштабувати окремі компоненти системи, що підвищує продуктивність. Наприклад, можна збільшити ресурси для сервісу, який обробляє найбільше навантаження, без впливу на інші сервіси.
- **Незалежність розгортання.**
 - Оскільки мікросервіси є незалежними один від одного, їх можна розгортати, оновлювати і масштабувати окремо, що зменшує ризик збоїв у всій системі під час оновлень.
- **Гнучкість у виборі технологій.**
 - Кожен мікросервіс може бути реалізований з використанням найбільш підходящих технологій та інструментів для конкретного завдання, що дозволяє використовувати різні мови програмування та бази даних.
- **Простота в підтримці та розробці.**
 - Мікросервіси спрощують процес розробки та підтримки, оскільки кожен сервіс є меншим і легшим для розуміння. Команди можуть працювати над окремими сервісами незалежно одна від одної.
- **Підвищена стійкість.**
 - Відмови одного мікросервісу не впливають на роботу інших сервісів, що підвищує загальну стійкість системи. Можна швидко

		Ткачук М.А			ІПЗ.КР.Б-121-24-ІПЗ	Арк.
		Мінтій І.С				10
Змн.	Арк.	№ докум.	Підпис	Дата		

відновити роботу окремого сервісу без тривалих простоїв для всієї системи.

- **Швидкість розробки.**

- Мікросервісна архітектура дозволяє командам розробників працювати паралельно над різними сервісами, що прискорює процес розробки і впровадження нових функцій.

- **Реюзабельність компонентів.**

- Мікросервіси можуть бути повторно використані в різних проєктах або частинах системи, що зменшує час та зусилля, необхідні для розробки нових додатків.

- **Покращена продуктивність і ефективність.**

- Оптимізація окремих мікросервісів дозволяє досягти кращої продуктивності та ефективності системи в цілому.

- **Автоматизація DevOps. [\[4\]](#)**

- Легше впроваджувати процеси CI/CD (безперервна інтеграція та безперервна доставка) для окремих мікросервісів, що дозволяє швидше і частіше випускати оновлення і нові функції.

- **Зручність у тестуванні.**

- Окремі мікросервіси легше тестувати, що дозволяє швидше виявляти і виправляти помилки. Можна здійснювати юніт-тестування, інтеграційне тестування та енд-то-енд тестування для кожного сервісу окремо.

Після проведення аналізу, були визначені ключові вимоги до інтерфейсу та функціональності інтернет-магазину. Такий магазин має бути не вимогливим до системи для максимального охоплення потенційних користувачів. Детальний опис аналізу формування вимог до магазину, його функціональності, безпеки даних та зручності використання подано в наступних розділах дипломної роботи.

Інтернет-магазини, побудовані на основі мікросервісної архітектури, виникли внаслідок поєднання кількох ключових факторів: зростання

		Ткачук М.А			ІПЗ.КР.Б-121-24-ІПЗ	Арк.
		Мінтій І.С				
Змн.	Арк.	№ докум.	Підпис	Дата		11

використання інтернету та потреби в ефективному управлінні різними аспектами онлайн-торгівлі. Початкові версії таких магазинів з'явилися на початку 2010-х років, коли інтернет став більш поширеним та доступним. У цей період в основному використовувалися прості системи, які дозволяли користувачам вести облік покупок і керувати запасами. Однак із зростанням потужності комп'ютерів та розвитком інтернет-технологій, з'явилися більш функціональні платформи, які надавали можливості не лише для ведення продажів, а й для інтеграції з банківськими системами, управління маркетинговими кампаніями, відстеження поведінки користувачів тощо.

Поняття мікросервісної архітектури стало все більш актуальним у сучасному світі, де швидкість змін зростає, а потреба у гнучкості та надійності стає важливішою. Мікросервіси дозволяють розробникам створювати незалежні модулі, які можуть бути легко оновлені або замінені без впливу на всю систему.

Сьогодні інтернет-магазини, які використовують мікросервіси, стали не просто зручними платформами для онлайн-покупок, а й важливими інструментами для розвитку бізнесу і підвищення продуктивності. Вони постійно вдосконалюються і розвиваються, щоб відповідати зростаючим потребам сучасних користувачів. Основні вимоги до таких інтернет-магазинів включають легкий доступ, надійність, масштабованість і простоту використання для всіх категорій користувачів.

1.2 Аналіз аналогів програмного продукту

На момент написання цієї роботи було знайдено кілька інтернет-магазинів, які пропонують широкий асортимент кави та супутніх товарів. Серед відомих прикладів можна виділити такі магазини, як Ekava [5], Coffeum [6], Coffeeok [7], Xkava [8] та CoffeUa [9]. Кожен з них має свої особливості та підходи до ведення бізнесу, що дозволяє їм залишатися конкурентоспроможними на ринку.

		Ткачук М.А			ІПЗ.КР.Б-121-24-ІПЗ	Арк.
		Мінтій І.С				12
Змн.	Арк.	№ докум.	Підпис	Дата		

Ekava – пропонує широкий асортимент кави, акції та швидку доставку. Інтерфейс сайту зручний і дозволяє швидко знаходити необхідні товари.

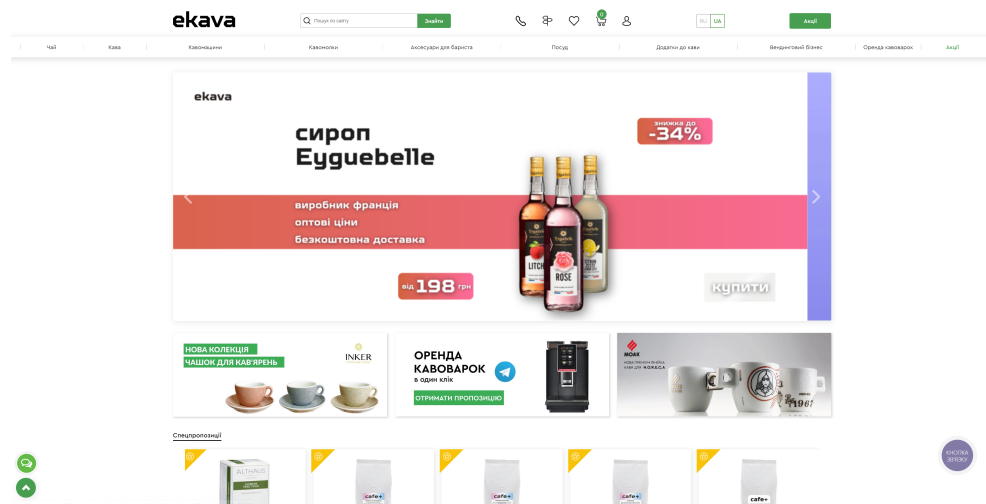


Рис. 1.1. Інтерфейс Ekava

Переваги:

- **Зручний інтерфейс:** Ekava має простий і зрозумілий інтерфейс, що дозволяє швидко знаходити потрібні товари, робити замовлення і відстежувати доставку.
- **Широкий асортимент кави:** Ekava пропонує великий вибір кави різних сортів і виробників, що дозволяє задовольнити смаки найвибагливіших клієнтів.
- **Популярні акції та знижки:** Магазин регулярно проводить акції та надає знижки на певні товари, що приваблює нових клієнтів і стимулює постійних покупців.
- **Швидка доставка:** Ekava забезпечує оперативну доставку замовлень, що є важливим фактором для задоволення клієнтів.
- **Висока якість продукції:** Ekava співпрацює з надійними постачальниками, що гарантує якість кожного продукту.

Недоліки:

- **Обмежений вибір супутніх товарів:** Асортимент супутніх товарів, таких як аксесуари для кави та кавомашини, може бути обмеженим.
- **Відсутність програми лояльності:** Немає програми лояльності, яка б стимулювала повторні покупки та підвищувала лояльність клієнтів.

		Ткачук М.А			ІПЗ.КР.Б-121-24-ІПЗ	Арк.
		Мінтій І.С				
Змн.	Арк.	№ докум.	Підпис	Дата		13

- **Можливість затримки продукції:** Незалежні продавці часто стикаються з конкуренцією з боку самого Amazon, який може пропонувати аналогічні товари за нижчими цінами.

Coffeum — відомий своєю капсульною кавою та аксесуарами для кавоварок. Сайт спеціалізується на продуктах для кавових машин, що робить його популярним серед любителів капсульної кави.

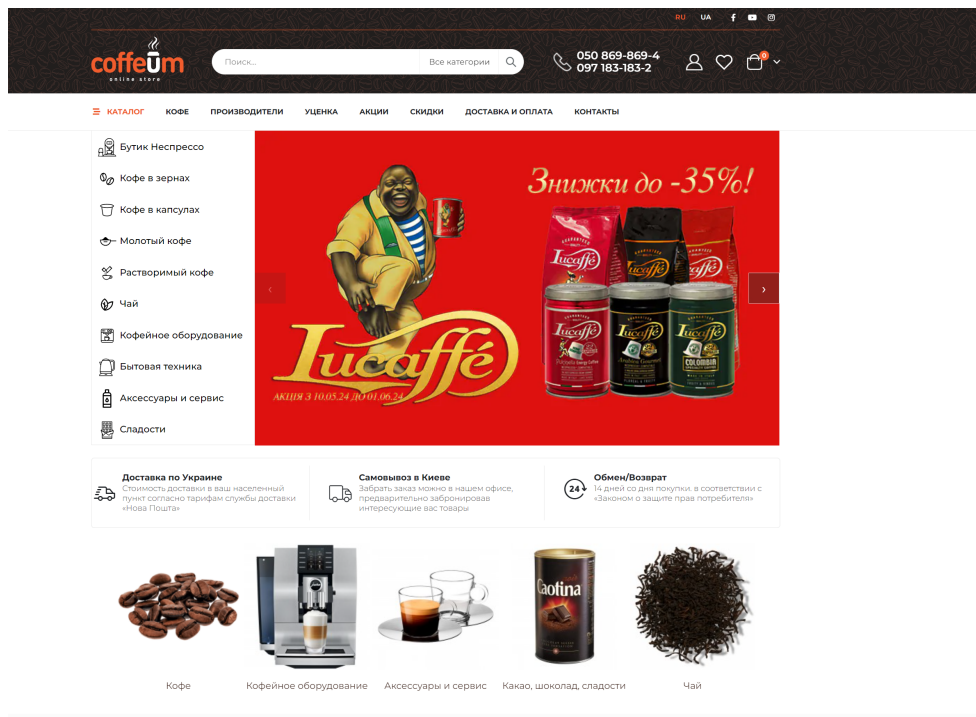


Рис. 1.2. Інтерфейс Coffeum

Переваги:

- **Інтуїтивний інтерфейс:** Coffeum має простий і зрозумілий інтерфейс, який дозволяє легко знаходити товари, брати участь в аукціонах і робити покупки через функцію "Купити зараз".
- **Великий вибір капсульної кави:** Coffeum пропонує широкий асортимент капсул для кавоварок різних виробників, що дозволяє клієнтам вибирати з безлічі варіантів.
- **Аксесуари для кавоварок:** Магазин має великий вибір аксесуарів для кавоварок, включаючи фільтри, молочники та інші корисні дрібниці.
- **Висока якість продукції:** Coffeum відомий своєю високоякісною

		Ткачук М.А			ІПЗ.КР.Б-121-24-ІПЗ	Арк.
		Мінтій І.С				14
Змн.	Арк.	№ докум.	Підпис	Дата		

продукцією, що забезпечує задоволення клієнтів.

- **Підтримка нових моделей кавоварок:** Магазин регулярно оновлює асортимент відповідно до нових моделей кавоварок на ринку.

Недоліки:

- **Обмежений вибір традиційної кави:** Асортимент традиційної кави може бути обмеженим, що може не задовольнити потреби всіх клієнтів.
- **Відсутність доставки в деякі регіони:** Магазин може не надавати доставку у всі регіони, що обмежує доступність продукції.
- **Високі ціни на аксесуари:** Деякі аксесуари можуть мати високі ціни, що може відлякати частину клієнтів.

Coffeeok – спеціалізується на продажу кави, чаю та кавомашин, пропонуючи багато супутніх товарів. Цей магазин має широкий асортимент продукції, що задовольняє різні потреби споживачів.

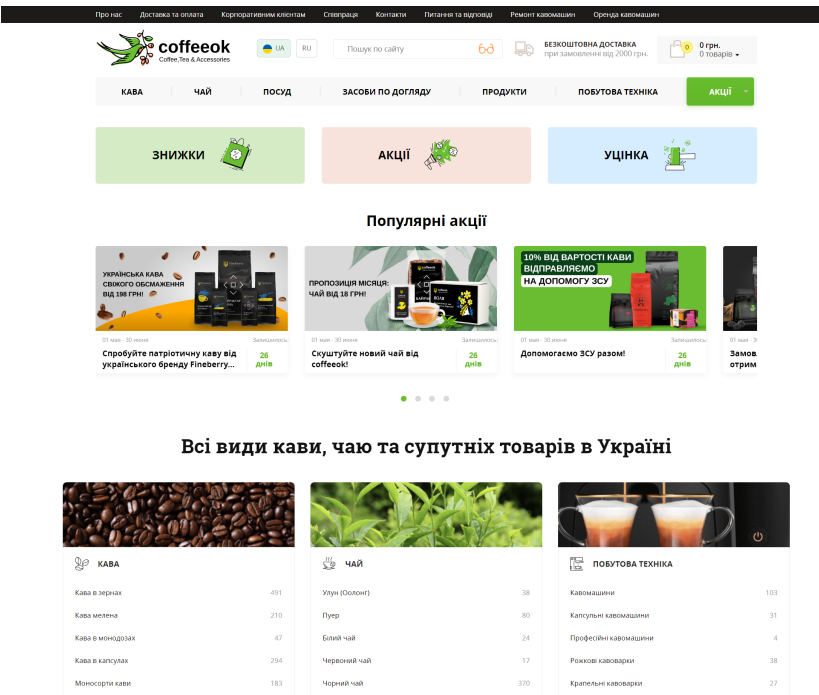


Рис. 1.3. Інтерфейс Coffeeok

Переваги:

- **Інтуїтивний інтерфейс:** Coffeeok має простий і зрозумілий інтерфейс, який дозволяє легко знаходити товари, створювати списки для збереження і налаштовувати профіль.

- **Широкий асортимент кави, чаю та кавомашин:** Coffeook пропонує великий вибір кави, чаю та кавомашин різних брендів, що дозволяє клієнтам знайти продукцію на свій смак.
- **Велика кількість супутніх товарів:** Магазин також пропонує широкий асортимент супутніх товарів, таких як аксесуари для кавоварок, фільтри, чашки та інше.
- **Швидка обробка замовлень:** Магазин забезпечує швидку обробку та відправку замовлень, що важливо для задоволення клієнтів.
- **Доступні ціни:** Coffeook пропонує конкурентні ціни на свою продукцію, що робить її доступною для широкого кола покупців.

Недоліки:

- **Відсутність програми лояльності:** Coffeook не має програми лояльності, що могло б стимулювати повторні покупки.
- **Відгуки користувачів можуть бути обмеженими:** Кількість відгуків користувачів може бути недостатньою для об'єктивної оцінки продукції.
- **Можливі проблеми з підтримкою:** Деякі користувачі можуть стикатися з труднощами у вирішенні питань через службу підтримки.

Хкава – акцентує на підвищенні якості споживання кави та має широкий вибір продукції. Магазин пропонує високоякісні продукти, які задовольняють найвибагливіших споживачів.

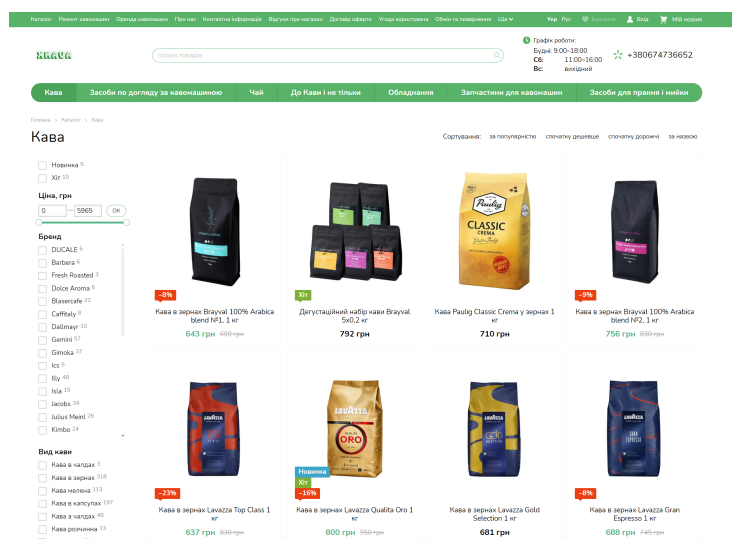


Рис. 1.3. Інтерфейс Хкава

		Ткачук М.А			ІПЗ.КР.Б-121-24-ІПЗ	Арк.
		Мінтій І.С				16
Змн.	Арк.	№ докум.	Підпис	Дата		

Переваги:

- **Інтуїтивний інтерфейс:** Хкава має простий і зрозумілий інтерфейс, який дозволяє легко знаходити товари, створювати списки для збереження і налаштовувати профіль.
- **Висока якість кавових продуктів:** Магазин пропонує великий вибір кавових продуктів, включаючи зернову, мелену та капсульну каву.
- **Висока якість продукції:** Хкава відома своєю високоякісною кавою та іншими продуктами, що забезпечує задоволення клієнтів.
- **Професійні консультанти:** Хкава надає професійні консультації щодо вибору кави та обладнання, що допомагає клієнтам прийняти правильне рішення.
- **Екологічна відповідальність:** Магазин підтримує екологічні ініціативи та пропонує продукти з екологічно чистих матеріалів.
- **Інноваційні продукти:** Хкава регулярно оновлює асортимент новими інноваційними продуктами, що задовольняють сучасні вимоги споживачів.

Недоліки:

- **Високі ціни на деякі продукти:** Ціни на деякі продукти можуть бути високими, що може відлякувати частину клієнтів.
- **Відсутність регулярних акцій:** Магазин не завжди проводить регулярні акції та знижки, що може вплинути на привабливість для покупців.
- **Можливі проблеми з доступністю:** У деяких регіонах можуть виникати проблеми з доступністю певних продуктів

CoffeUa – пропонує все необхідне для приготування кави вдома, від кавомашин до посуду. Магазин спеціалізується на забезпеченні всіх потреб любителів кави.

		Ткачук М.А			ІПЗ.КР.Б-121-24-ІПЗ	Арк.
		Мінтій І.С				
Змн.	Арк.	№ докум.	Підпис	Дата		17

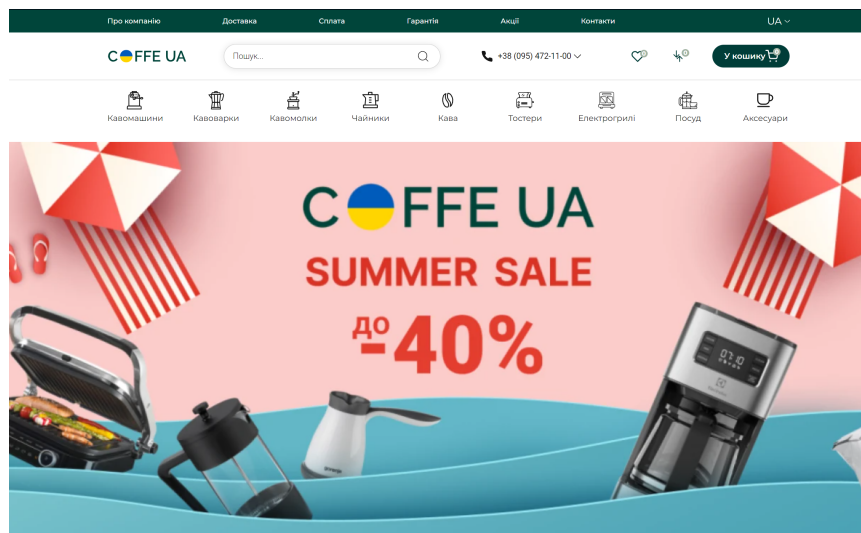


Рис. 1.3. Інтерфейс CoffeUa

Переваги:

- **Інтуїтивний інтерфейс:** CoffeUa має простий і зрозумілий інтерфейс, який дозволяє легко знаходити товари, створювати списки для збереження і налаштовувати профіль.
- **Комплексний асортимент для приготування кави вдома:** CoffeUa пропонує все необхідне для приготування кави вдома, включаючи кавомашини, фільтри, чашки та інші аксесуари.
- **Висока якість продукції:** Магазин відомий своєю високоякісною продукцією, що забезпечує задоволення клієнтів.
- **Розширений вибір кавомашин:** Магазин пропонує широкий асортимент кавомашин різних брендів та моделей, що дозволяє знайти оптимальний варіант для кожного покупця.
- **Підтримка клієнтів:** CoffeUa забезпечує високий рівень підтримки клієнтів, допомагаючи вирішувати будь-які питання, що виникають.

Недоліки:

- **Обмежена кількість фізичних магазинів:** CoffeUa може мати обмежену кількість фізичних магазинів, що обмежує можливість особистого огляду продукції перед покупкою.
- **Відсутність програми лояльності:** Магазин не має програми лояльності,

		Ткачук М.А			ІПЗ.КР.Б-121-24-ІПЗ	Арк.
		Мінтій І.С				
Змн.	Арк.	№ докум.	Підпис	Дата		18

яка б стимулювала повторні покупки та підвищувала лояльність клієнтів.

- **Висока ціна на деякі товари:** Деякі товари можуть мати високі ціни, що може відлякувати частину клієнтів.
- **Можливі затримки з доставкою:** У певних регіонах можуть виникати затримки з доставкою, що може викликати незадоволення клієнтів.

Таблиця 1.1

Характеристика	Ekava	Coffeum	Coffeeok	Xkava	CoffeUa
Масштабованість	+	+-	+	+-	+
Незалежність розгортання	+-	+	+	+	+
Гнучкість у виборі технологій	+	+-	-	+	+
Простота в підтримці та розробці	+	+	+	+-	+
Підвищена стійкість	+-	+	+	+-	+
Швидкість розробки	-	+-	+	+-	-
Реюзабельність компонентів	+	+	+	+	+
Покращена продуктивність і ефективність	+	+	+	+	+
Зручність у тестуванні	+	-	-	+	+

Аналіз показав, що інтернет-магазини на основі мікросервісної архітектури дозволяють компаніям ефективніше керувати своїми ресурсами, забезпечують високу доступність сервісів та швидко адаптуються до змін на ринку. Використання мікросервісів забезпечує гнучкість, надійність та продуктивність, що є ключовими факторами для успішної розробки та впровадження інтернет-магазину.

Мікросервісна архітектура, завдяки своїм унікальним особливостям, пропонує значні переваги для розвитку інтернет-магазинів. Перш за все, вона дозволяє компаніям ефективніше керувати своїми ресурсами. Кожен мікросервіс функціонує як незалежна одиниця, що дозволяє розробникам краще контролювати та оптимізувати використання ресурсів для кожної конкретної функції системи. Це означає, що навіть при високому навантаженні на окремі частини системи, інші її частини залишаються стабільними та продуктивними.

Крім того, мікросервісна архітектура забезпечує високу доступність сервісів. Завдяки незалежності мікросервісів, відмова одного з них не впливає на роботу інших. Це підвищує загальну надійність системи і дозволяє швидко відновлювати роботу окремих сервісів у разі їх збою. У результаті, користувачі отримують стабільний доступ до функціональності інтернет-магазину, що позитивно впливає на їх досвід та задоволення.

1.3 Обґрунтування вибору інструментальних засобів та вимоги до апаратного забезпечення

Для створення сучасного, високопродуктивного та масштабованого інтернет-магазину на основі мікросервісів було обрано низку потужних інструментів та технологій, кожен з яких має свої унікальні переваги. Ось детальний опис вибраних інструментів та технологій, а також обґрунтування їх вибору:

.NET Core Microservices (.NET 8 MVC) [\[10\]](#) обрано для розробки мікросервісів завдяки його високій продуктивності, кросплатформеності та широкій підтримці. .NET Core дозволяє легко створювати і розгортати мікросервіси, що працюють незалежно один від одного, забезпечуючи гнучкість та масштабованість системи. Використання .NET 8 MVC забезпечує:

- **Високу продуктивність:** .NET Core оптимізований для роботи в умовах високого навантаження.
- **Кросплатформеність:** Підтримка різних операційних систем (Windows, Linux, macOS).
- **Широку підтримку:** Велика спільнота розробників та регулярні оновлення від Microsoft.

.NET API з аутентифікацією та авторизацією використовується для захисту доступу до ресурсів інтернет-магазину. Використання .NET Identity дозволяє впровадити рольову авторизацію, що забезпечує контроль доступу до різних частин системи на основі ролей користувачів. Це рішення надає:

- **Безпеку:** Захист доступу до важливих даних та функцій.

		Ткачук М.А			ІПЗ.КР.Б-121-24-ІПЗ	Арк.
		Мінтій І.С				20
Змн.	Арк.	№ докум.	Підпис	Дата		

- **Безпеку:** Захист доступу до важливих даних та функцій.
- **Гнучкість:** Легке налаштування та інтеграція з існуючими системами

Асинхронна та синхронна комунікація між мікросервісами реалізована для забезпечення ефективної взаємодії між різними сервісами системи. Це включає:

- **Асинхронна комунікація:** Використання для підвищення продуктивності та зниження затримок, наприклад, обробка великих обсягів даних або тривалих операцій.
- **Синхронна комунікація:** Використання для негайного обміну даними у випадках, коли це необхідно, наприклад, при запитах на отримання даних в реальному часі.

Azure Service Bus – Topics та Queues [\[11\]](#) Azure Service Bus обрано для надійної обробки повідомлень між мікросервісами. Використання Azure Service Bus забезпечує:

- **Масштабованість:** Легке масштабування системи за рахунок ефективного оброблення повідомлень.
- **Високу доступність:** Гарантована доставка повідомлень навіть при збої окремих компонентів системи.
- **Стійкість до збоїв:** Надійне управління чергами повідомлень, що знижує ризики втрати даних.

Шлюзи в мікросервісах та впровадження шлюзу Ocelot [\[12\]](#) дозволяють централізовано керувати маршрутизацією та балансуванням навантаження між різними мікросервісами. Використання Ocelot забезпечує:

- **Гнучкість:** Легке налаштування маршрутів та схем аутентифікації.
- **Централізоване керування:** Управління маршрутизацією та безпекою в одному місці.
- **Балансування навантаження:** Розподіл навантаження між сервісами для підвищення продуктивності.

Swagger Open API використовується для документування та тестування

		Ткачук М.А			ІПЗ.КР.Б-121-24-ІПЗ	Арк.
		Мінтій І.С				
Змн.	Арк.	№ докум.	Підпис	Дата		21

API, що забезпечує:

- **Зручність для розробників:** Легка генерація документації API, що спрощує підтримку системи.
- **Тестування:** Можливість тестування API безпосередньо з документації.
- **Сумісність:** Підтримка стандартів Open API, що спрощує інтеграцію з іншими системами.

N-Layer архітектура з патерном Repository дозволяє структурувати код таким чином, що кожен шар системи (презентаційний, бізнес-логіки, доступу до даних) є незалежним та легко підтримуваним. Це забезпечує:

- **Чітку структурування коду:** Кожен шар відповідає за свою частину логіки.
- **Легкість підтримки:** Зміни в одному шарі мінімально впливають на інші шари.
- **Реюзабельність:** Можливість повторного використання компонентів коду.

ASP.NET Core Web Application з Bootstrap 5 [13] обрано для створення інтерфейсу користувача. Використання Bootstrap 5 забезпечує:

- **Адаптивний дизайн:** Інтерфейс добре виглядає на різних пристроях (мобільні телефони, планшети, комп'ютери).
- **Швидкість розробки:** Bootstrap 5 містить готові компоненти для швидкого створення сучасного інтерфейсу.
- **Сумісність:** Підтримка різних браузерів та платформ.

Entity Framework Core з базою даних SQL Server використовується для збереження даних. Використання цих технологій забезпечує:

- **Зручну роботу з базою даних:** Entity Framework Core дозволяє використовувати LINQ-запити для маніпуляції даними, що спрощує розробку.
- **Надійність:** SQL Server забезпечує високу продуктивність і

		Ткачук М.А			ІПЗ.КР.Б-121-24-ІПЗ	Арк.
		Мінтій І.С				22
Змн.	Арк.	№ докум.	Підпис	Дата		

надійність збереження даних.

- **Масштабованість:** Легке масштабування бази даних у міру зростання потреб бізнесу.

Висновки до першого розділу.

У першому розділі роботи було розглянуто актуальність теми розробки інтернет-магазину на основі мікросервісної архітектури з використанням сучасних технологій, таких як .NET Core Microservices, .NET API з аутентифікацією та авторизацією, рольова авторизація за допомогою .NET Identity, асинхронна та синхронна комунікація між мікросервісами, Azure Service Bus – Topics та Queues, впровадження шлюзів за допомогою Ocelot, реалізація Swagger Open API, N-Layer архітектура з патерном Repository, ASP.NET Core Web Application з Bootstrap 5 та Entity Framework Core з SQL Server.

Було проаналізовано еволюцію подібних технологій і їх застосування в електронній комерції, від базових рішень до сучасних складних архітектур. Висвітлено переваги та причини популярності мікросервісної архітектури, включаючи її гнучкість, масштабованість та надійність.

Проведений аналіз показав, що інтернет-магазини на основі мікросервісів є необхідними інструментами для розвитку бізнесу та підвищення продуктивності у сучасному світі. Вони дозволяють компаніям ефективніше керувати своїми ресурсами, забезпечувати високу доступність сервісів та швидко адаптуватися до змін на ринку.

На основі проведеного аналізу були сформульовані основні вимоги до програмного продукту для інтернет-магазину, які включають доступність, надійність, масштабованість та простоту використання для різних категорій користувачів. Ці вимоги стануть основою для подальшої розробки системи, забезпечуючи її ефективність та відповідність сучасним потребам бізнесу.

		Ткачук М.А			ІПЗ.КР.Б-121-24-ІПЗ	Арк.
		Мінтій І.С				23
Змн.	Арк.	№ докум.	Підпис	Дата		

РОЗДІЛ 2. ПРОЕКТУВАННЯ ВЕБ-ДОДАТКУ

2.1 Визначення варіантів використання та об'єктно-орієнтованої структури системи

Користувач інтернет-магазину кави матиме можливість ознайомитися з різноманітним асортиментом кавових сортів та аксесуарів зручно та інтуїтивно, отримуючи доступ до великої кількості продуктів. Інтернет-магазин спрощує взаємодію користувача з платформою, надаючи зручний інтерфейс та забезпечуючи легку навігацію.

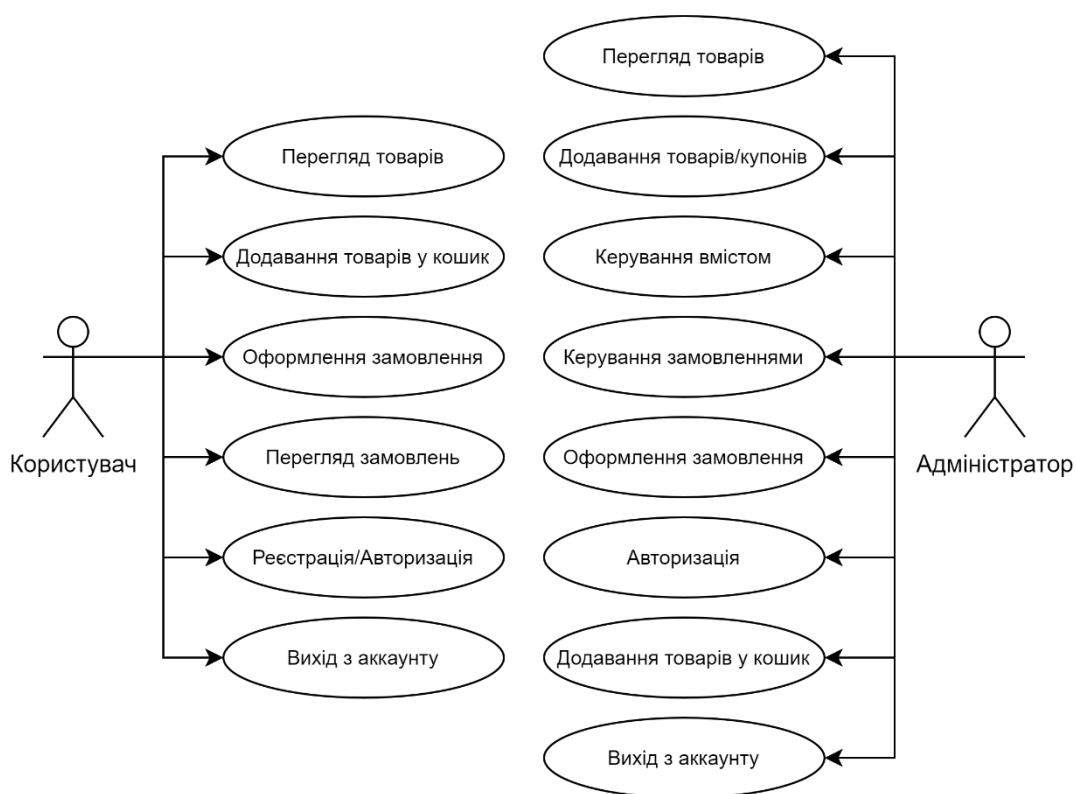


Рис. 2.1. Діаграма варіантів використання

Опис діаграми варіантів використання наведено в таблицях 2.1 – 2.10.

Таблиця 2.1

Опис прецеденту «Реєстрація»	
Назва варіанту використання	Реєстрація
Основні актори	Користувач
Короткий опис	Користувачі можуть створити аккаунт

Таблиця 2.2

Опис прецеденту «Авторизація»	
Назва варіанту використання	Авторизація
Основні актори	Користувач, Адміністратор
Короткий опис	Користувачі можуть увійти у аккаунт

Таблиця 2.3

Опис прецеденту «Перегляд замовлень»	
Назва варіанту використання	Перегляд замовлень
Основні актори	Користувач, Адміністратор
Короткий опис	Користувачі переглядати свої замовлення (Адміністратор всі замовлення)

Таблиця 2.4

Опис прецеденту «Оформлення замовлення»	
Назва варіанту використання	Оформлення замовлення
Основні актори	Користувач, Адміністратор
Короткий опис	Користувачі можуть оформлювати своє замовлення з товарів які знаходяться у кошику

Таблиця 2.5

Опис прецеденту «Додавання товарів у кошик»	
Назва варіанту використання	Додавання товарів у кошик
Основні актори	Користувач, Адміністратор
Короткий опис	Користувачі можуть додавати/видаляти товари до кошика

Таблиця 2.6

Опис прецеденту «Перегляд товарів»	
Назва варіанту використання	Перегляд товарів
Основні актори	Користувач, Адміністратор
Короткий опис	Користувачі можуть ознайомитись з наявним товаром на сайті

Таблиця 2.7

Опис прецеденту «Вихід з акаунту»	
Назва варіанту використання	Вихід з акаунту користувача
Основні актори	Користувач, Адміністратор
Короткий опис	Користувач натискає кнопку виходу з акаунту

Таблиця 2.8

Опис прецеденту «Додавання товарів»	
Назва варіанту використання	Додавання товарів
Основні актори	Адміністратор
Короткий опис	Адміністратор може додавати нові товари

Таблиця 2.9

Опис прецеденту «Додавання купонів»	
Назва варіанту використання	Додавання купонів
Основні актори	Адміністратор
Короткий опис	Адміністратор може додавати нові купони

Опис прецеденту «Керування вмістом»	
Назва варіанту використання	Редагування контенту
Основні актори	Адміністратор
Короткий опис	Адміністратор може редагувати товари, видаляти купони та товари, керувати замовленнями

Основна ціль розробки інтернет-магазину на основі мікросервісів полягає в наданні користувачам зручної та ефективної платформи для здійснення покупок онлайн. Такий інтернет-магазин повинен спростити процес купівлі та продажу товарів, забезпечуючи користувачам високу продуктивність і надійність. Він має включати функції аутентифікації та авторизації, асинхронної та синхронної комунікації між мікросервісами, використання Azure Service Bus, шлюзи Ocelot, Swagger Open API, N-Layer архітектуру з патерном Repository, а також сучасний інтерфейс на базі ASP.NET Core Web Application з Bootstrap 5 та Entity Framework Core з SQL Server.

Основна мета розробки такого інтернет-магазину – забезпечення високої продуктивності, безпеки та зручності для користувачів. Система дозволить швидко обробляти замовлення, підтримувати безперебійну роботу аукціонів і продажів, а також надавати персоналізовані рекомендації на основі поведінки користувачів.

Вимоги користувачів:

Внутрішній користувач – клієнт:

1. Робота з акаунтом користувача
 - 1.1 Реєстрація акаунту користувача
 - 1.2 Авторизація користувача
2. Робота з товарами та замовленнями
 - 2.1 Перегляд товарів
 - 2.2 Додавання товарів у кошик

		Ткачук М.А			ІПЗ.КР.Б-121-24-ІПЗ	Арк.
		Мінтій І.С				27
Змн.	Арк.	№ докум.	Підпис	Дата		

2.3 Оформлення замовлення

2.4 Перегляд замовлень

2.5 Використання купонів

3. Вихід з акаунту користувача

Внутрішній користувач – Адміністратор:

1. Робота з акаунтами користувачів

1.1 Створення акаунтів користувачів

1.2 Редагування акаунтів користувачів

1.3 Видалення акаунтів користувачів

2. Управління товарами

2.1 Перегляд товарів

2.2 Додавання нових товарів

2.3 Редагування інформації про товари

2.4 Видалення товарів з каталогу

3. Управління замовленнями

3.1 Перегляд замовлень

3.2 Обробка замовлень

3.3 Відстеження замовлень

4. Управління купонами та акціями

4.1 Створення нових купонів

4.2 Видалення купонів

5. Безпека та доступ

5.1 Управління ролями та правами доступу

5.2 Моніторинг безпеки

6. Комунікація з користувачами

6.1 Відправка повідомлень

6.2 Управління зворотним зв'язком

Функціональні вимоги:

1. Система повинна надавати можливість користувачеві авторизуватись в системі за допомогою пошти.

		Ткачук М.А			ІПЗ.КР.Б-121-24-ІПЗ	Арк.
		Мінтій І.С				
Змн.	Арк.	№ докум.	Підпис	Дата		28

2. Система повинна надавати можливість перегляду товарів.
3. Система повинна надавати можливість додавання товарів у кошик.
4. Система повинна надавати можливість оформлення замовлень.
5. Система повинна надавати можливість перегляду замовлень.
6. Система повинна надавати можливість додавання та управління купонами.
7. Система повинна надавати можливість управління вмістом (товари, купони).
8. Система повинна надавати можливість управління замовленнями.
9. Система повинна надавати можливість користувачеві виходити з раніше авторизованого акаунту.

Не функціональні вимоги:

1. **Інтуїтивність інтерфейсу:** Інтерфейс користувача повинен бути інтуїтивно зрозумілим та легким у використанні. Він повинен мати чітку структуру та бути дружнім до користувача, відповідаючи сучасним стандартам дизайну.

2. **Портативність:** Система має бути сумісною з різними веб-браузерами на десктопних та мобільних пристроях.

3. **Стабільність:** Надійність та стабільність системи є невід'ємною частиною її функціонування. Мета полягає в тому, щоб забезпечити безперервну роботу без аварій та помилок, що забезпечить користувачам безперебійний та стабільний досвід використання програмного забезпечення.

Системні вимоги:

Для успішної роботи даного застосунку необхідне виконання наступних вимог:

- Наявність пристрою з доступом до інтернету;
- Сучасний веб-браузер (Chrome, Firefox, Safari, Edge);
- Розмір екрана від 320px для мобільних пристроїв до 1920px і вище для десктопів.

		Ткачук М.А			ІПЗ.КР.Б-121-24-ІПЗ	Арк.
		Мінтій І.С				29
Змн.	Арк.	№ докум.	Підпис	Дата		

2.2 Проектування бази даних

Для збереження файлів та інформації у веб-застосунку використовується Entity Framework Core разом із базою даних SQL Server. Вибір цього поєднання обґрунтований наступними аспектами та можливостями:

Структура даних:

- Інформація зберігається у таблицях, які складаються з рядків та стовпців, що дозволяє ефективно організовувати та структурувати дані.
- Таблиці можуть бути пов'язані між собою через ключові поля (первинні та зовнішні ключі), що дозволяє створювати зв'язки типу "один до багатьох", "багато до багатьох" тощо.
- Використання нормалізації допомагає мінімізувати дублювання інформації та оптимізувати структуру бази даних, забезпечуючи більш ефективне управління даними.

Запити:

- SQL Server підтримує мову запитів T-SQL (Transact-SQL) [\[14\]](#), що дозволяє виконувати фільтри, сортування та обмеження кількості результатів.
- Підтримка складних запитів з об'єднанням таблиць (JOIN), агрегацією даних (GROUP BY, HAVING) та підзапитами (subqueries) дозволяє отримувати потрібні дані з різних таблиць і виконувати комплексні аналітичні завдання.

Безпека:

- SQL Server використовує правила безпеки на основі виразів для управління доступом до даних.
- Можливість визначення ролей і прав доступу дозволяє точно контролювати, хто може читати або записувати дані в базу.
- Шифрування даних забезпечує захист конфіденційної інформації, гарантуючи, що дані будуть зберігатися у безпечному вигляді.

Аутентифікація:

- SQL Server інтегрується з такими системами аутентифікації, як Active Directory, що дозволяє зручно керувати доступом користувачів.

		Ткачук М.А			ІПЗ.КР.Б-121-24-ІПЗ	Арк.
		Мінтій І.С				
Змн.	Арк.	№ докум.	Підпис	Дата		30

• Підтримка Windows Authentication та SQL Server Authentication надає різноманітні варіанти налаштування доступу до бази даних.

Проект складається з 7 баз даних, для кожного API окрема

Таблиця 2.11

Опис бази даних «CoffeeShop_Auth» та її таблиць

AspNetUsers		
Поле	Тип	Призначення
Id	nvarchar	Ідентифікатор
UserName	nvarchar	Ім'я користувача
Email	nvarchar	Пошта користувача
PasswordHash	nvarchar	Зашифрований пароль користувача
PhoneNumber	nvarchar	Номер телефону користувача
Name	nvarchar	Справжнє ім'я користувача

Таблиця 2.12

AspNetRoles		
Поле	Тип	Призначення
Id	nvarchar	Ідентифікатор
Name	nvarchar	Назва ролі
NormalizedName	nvarchar	Назва ролі у стандартному вигляді
ConcurrencyStamp	nvarchar	Штамп паралельності

Опис бази даних «CoffeeShop_Coupon» та її таблиць

Coupons		
Поле	Тип	Призначення
CouponId	int	Ідентифікатор
CouponCode	nvarchar	Назва купону
DiscountAmount	float	Кількість знижки в ціновому еквіваленті
MinAmount	int	Мінімальне значення для застосування купону

Таблиця 2.14

Опис бази даних «CoffeeShop_Email» та її таблиць

EmailLoggers		
Поле	Тип	Призначення
Id	int	Ідентифікатор
Email	nvarchar	Пошта на яку треба надіслати повідомлення
Message	nvarchar	Текст повідомлення
EmailSent	datetime	Дата надсилання

Таблиця 2.15

Опис бази даних «CoffeeShop_Order» та її таблиць

OrderDetails		
Поле	Тип	Призначення
OrderDetailsId	int	Ідентифікатор
OrderHeaderId	int	Ідентифікатор замовлення
ProductId	int	Ідентифікатор товару
Count	int	Кількість
ProductName	nvarchar	Назва продукту
Price	float	Ціна

Таблиця 2.16

OrderHeaders		
Поле	Тип	Призначення
OrderHeaderId	int	Ідентифікатор
UserId	int	Ідентифікатор користувача
CouponCode	nvarchar	Назва купону
Discount	int	Знижка
OrderTotal	float	Кінцева ціна
Name	float	Ім'я користувача
Phone	nvarchar	Номер телефону
Email	nvarchar	Пошта
OrderTime	datetime	Дата та час замовлення
Status	nvarchar	Статус замовлення
PaymentIntentId	nvarchar	Ідентифікатор оплати
StripeSessionId	nvarchar	Ідентифікатор сесії оплати

Таблиця 2.17

Опис бази даних «CoffeeShop_Product» та її таблиць

Products		
Поле	Тип	Призначення
ProductId	int	Ідентифікатор
Name	nvarchar	Назва товару
Price	float	Ціна товару
Description	nvarchar	Опис
CategoryName	nvarchar	Назва категорії
ImageUrl	nvarchar	Адреса зображення
ImageLocalPath	nvarchar	Шлях до зображення

Опис бази даних «CoffeeShop_Reward» та її таблиць

Rewards		
Поле	Тип	Призначення
Id	int	Ідентифікатор
UserId	nvarchar	Ідентифікатор користувача
RewardsDate	datetime	Дата та час отримання бонусів
RewardActivity	int	Кількість бонусів
OrderId	int	Ідентифікатор замовлення

Таблиця 2.19

Опис бази даних «CoffeeShop_ShoppingCart» та її таблиць

CartDetails		
Поле	Тип	Призначення
CartDetailsId	int	Ідентифікатор
CartHeaderId	int	Ідентифікатор кошика
ProductId	int	Ідентифікатор товару
Count	int	Кількість

Таблиця 2.20

Опис бази даних «CoffeeShop_ShoppingCart» та її таблиць

CartDetails		
Поле	Тип	Призначення
CartHeaderId	int	Ідентифікатор
UserId	nvarchar	Ідентифікатор користувача
CouponCode	nvarchar	Назва купону

2.3 Проектування та реалізація алгоритмів роботи системи

Застосунок є простим у використанні та має всі основні функції, що дозволяють користувачам легко взаємодіяти з платформою. Алгоритм роботи з додатком (рис. 2.3), а також описано детально його пункти.

Користувач потрапляє на сторінку авторизації.

- Користувач може зайти за допомогою пошти.

Запит Користувача обробляється API Шлюзом

- На головній сторінці користувач може переглядати товари.

Перевірка автентифікації користувача.

- Система перевіряє, чи є користувач автентифікованим. Якщо ні, користувачу повертається повідомлення про необхідність автентифікації.
- Якщо користувач автентифікований, запит передається далі.

Перевірка ролей користувача.

- Система перевіряє ролі користувача для визначення доступу до певних функцій.
- Якщо користувач не має відповідних прав, запит завершується з повідомленням про відсутність доступу.

Обробка запиту відповідним мікросервісом.

- Виконуються необхідні бізнес-логіка та операції, пов'язані з запитом користувача.
- Наприклад, отримання списку товарів, обробка замовлення або керування акаунтом користувача.

Взаємодія з базою даних.

- Мікросервіс виконує необхідні операції з базою даних через Entity Framework Core.
- Можливі операції включають створення, читання, оновлення та видалення даних.

Формування відповіді та повернення результату.

- Мікросервіс формує відповідь на запит користувача та повертає її

		Ткачук М.А			ІПЗ.КР.Б-121-24-ІПЗ	Арк.
		Мінтій І.С				
Змн.	Арк.	№ докум.	Підпис	Дата		35

через API Шлюз.

- API Шлюз відправляє сформовану відповідь користувачу

Відповідь на запит відображається користувачу

- Користувач отримує відповідь, яка може містити дані про товари, підтвердження замовлення або повідомлення про успішне виконання операції.

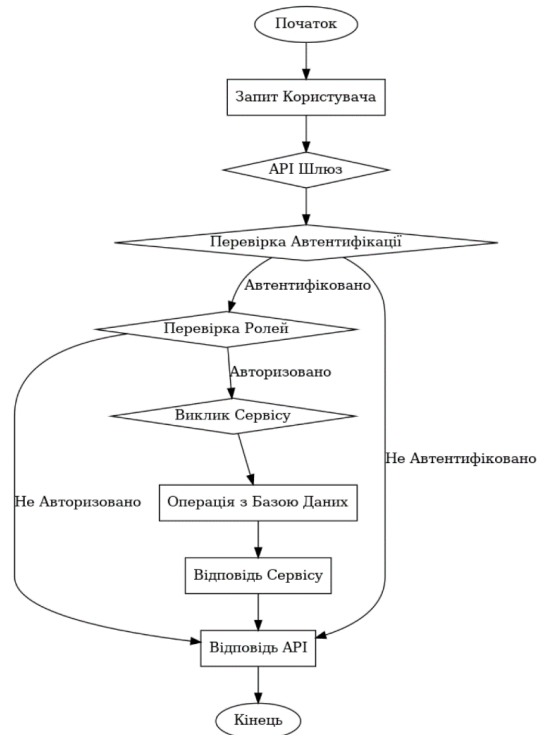


Рис. 2.3. Діаграма активності

Важливою особливістю цієї системи є те, що під кожен мікросервіс створюється окрема база даних, яка не залежить від інших мікросервісів. Це забезпечує незалежність і гнучкість в управлінні даними, а також дозволяє використовувати різні технології для різних мікросервісів, хоча в даному випадку використовується MSSQL для всіх баз даних.

Переваги окремих баз даних:

- **Незалежність:** Мікросервіси можуть розвиватися незалежно один від одного без впливу на інші компоненти системи.
- **Масштабованість:** Легше масштабувати окремі мікросервіси відповідно до їхніх специфічних потреб.

- **Гнучкість:** Можливість використання різних технологій для різних мікросервісів. В даному випадку, всі мікросервіси використовують MSSQL, але при потребі це може бути змінено.
- **Зниження ризиків:** Помилки в одному мікросервісі не впливають на інші мікросервіси.

2.4 Архітектура програмного комплексу та його розгортання.

Діаграма розгортання ілюструє фізичну структуру та взаємодію компонентів MVC додатку для інтернет-магазину кави, побудованого з використанням мікросервісів. Вона показує розташування та взаємозв'язки між серверними та клієнтськими частинами системи, відображаючи шляхи обробки та передачі даних. На діаграмі використано два типи ліній для позначення різних типів комунікації: суцільні лінії для синхронної комунікації та штрихові лінії для асинхронної комунікації.

Компоненти системи:

- **Gateway (Порт: 7777):**
 - **Функція:** Точка входу для зовнішніх запитів до системи.
 - **Опис:** Забезпечує маршрутизацію запитів до відповідних мікросервісів та контролює доступ до них. Використовує синхронну комунікацію з іншими мікросервісами для забезпечення швидкої відповіді.
- **ShoppingCart (Порт: 7003):**
 - **Функція:** Управління кошиком покупок користувачів.
 - **Опис:** Обробляє додавання товарів до кошика, видалення товарів та оформлення замовлення. Взаємодіє синхронно з Gateway для обробки запитів користувачів.
- **Coupon (Порт: 7001):**
 - **Функція:** Використання купонів зі знижками під час покупок.

		Ткачук М.А			ІПЗ.КР.Б-121-24-ІПЗ	Арк.
		Мінтій І.С				
Змн.	Арк.	№ докум.	Підпис	Дата		37

- **Опис:** Відповідає за перевірку та застосування знижок до замовлень. Підключається до Gateway для отримання інформації про купони та знижки.
- **Product (Порт: 7000):**
 - **Функція:** Надання інформації про доступні товари.
 - **Опис:** Містить дані про товари інтернет-магазину, дозволяє переглядати та шукати продукти. Використовує синхронну комунікацію з Gateway для надання швидких відповідей на запити.
- **Identity (Порт: 7002):**
 - **Функція:** Автентифікація та авторизація користувачів.
 - **Опис:** Забезпечує безпеку доступу до функціоналу системи та керує обліковими записами користувачів. Синхронно взаємодіє з Gateway для перевірки автентифікації користувачів.
- **Service Bus:**
 - **Функція:** Обмін повідомленнями між мікросервісами та іншими компонентами системи.
 - **Опис:** Забезпечує асинхронну комунікацію та обробку подій. Використовується для надсилання повідомлень про зміну стану замовлень, оновлення статусу користувачів тощо.
- **Order (Порт: специфічний):**
 - **Функція:** Обробка замовлень.
 - **Опис:** Відповідає за створення та управління замовленнями. Використовує асинхронну комунікацію з Service Bus для отримання та обробки подій, пов'язаних із замовленнями.
- **Email (Порт: специфічний):**
 - **Функція:** Відправка електронних листів.
 - **Опис:** Відправляє підтвердження замовлень, сповіщення про акції та інші важливі повідомлення користувачам.

		Ткачук М.А			ІПЗ.КР.Б-121-24-ІПЗ	Арк.
		Мінтій І.С				
Змн.	Арк.	№ докум.	Підпис	Дата		38

Використовує асинхронну комунікацію з Service Bus для обробки подій від інших мікросервісів.

- **Rewards (Порт: специфічний):**

- **Функція:** Управління бонусною програмою.
- **Опис:** Відстежує та нараховує бонусні бали користувачам за здійснені покупки. Використовує асинхронну комунікацію з Service Bus для обробки подій, пов'язаних з бонусами.

- **Типи комунікації:**

- **Суцільна лінія:** Позначає синхронну комунікацію, що передбачає негайний обмін даними між компонентами.
- **Штрихова лінія:** Позначає асинхронну комунікацію, що використовується для обміну даними через Service Bus без негайного очікування відповіді.

Діаграма (рис. 2.4) наочно демонструє, як різні компоненти системи взаємодіють один з одним, використовуючи як синхронні, так і асинхронні методи комунікації для забезпечення надійності та ефективності роботи інтернет-магазину кави

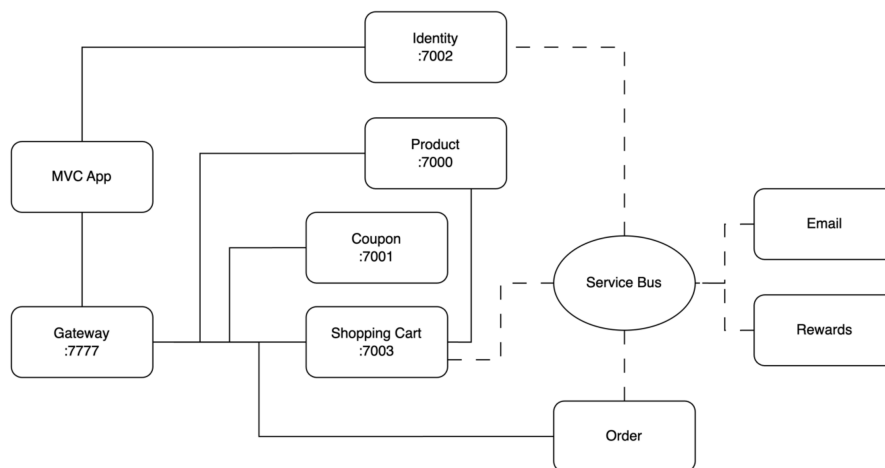


Рис. 2.4 Діаграма компонентів

2.5 Реалізація функціоналу інтернет магазину з використанням мікросервісів.

Метод **Index** – отримання списку всіх продуктів та відображення їх на головній сторінці.

```

public async Task<IActionResult> Index()
{
    List<ProductDto>? list = new();
    ResponseDto? response = await _productService.GetAllProductsAsync();
    if (response != null && response.IsSuccess)
    {
        list = JsonConvert.DeserializeObject<List<ProductDto>>(Convert.ToString(response.Result));
    }
    else
    {
        TempData["error"] = response?.Message;
    }
    return View(list);
}

```

Рис. 2.5 Метод **Index**

Цей код використовується для отримання всіх продуктів з бази даних через сервіс **IProductService** та відображення їх на головній сторінці. Якщо запит вдалий, продукти десеріалізуються та передаються до представлення для відображення. У разі помилки, повідомлення про помилку зберігається у **TempData**.

Метод **ProductDetails** (GET) – отримання деталей продукту за ідентифікатором та відображення їх на сторінці деталей продукту.

```

[Authorize]
public async Task<IActionResult> ProductDetails(int productId)
{
    ProductDto model = new();
    ResponseDto? response = await _productService.GetProductByIdAsync(productId);
    if (response != null && response.IsSuccess)
    {
        model = JsonConvert.DeserializeObject<ProductDto>(Convert.ToString(response.Result));
    }
    else
    {
        TempData["error"] = response?.Message;
    }
    return View(model);
}

```

Рис. 2.6 Метод **ProductDetails** (GET)

Цей код використовується для отримання деталей продукту за його ідентифікатором через сервіс **IProductService** та відображення їх на сторінці деталей продукту. Якщо запит вдалий, деталі продукту десеріалізуються та передаються до представлення для відображення. У разі помилки, повідомлення про помилку зберігається у **TempData**.

Метод **ProductDetails** (POST) – додавання продукту до кошика.

```
[Authorize]
[HttpPost]
[ActionName("ProductDetails")]
public async Task<IActionResult> ProductDetails(ProductDto productDto)
{
    CartDto cartDto = new CartDto()
    {
        CartHeader = new CartHeaderDto
        {
            UserId = User.Claims.Where(u => u.Type ==
JwtClaimTypes.Subject)?.FirstOrDefault()?.Value
        }
    };
    CartDetailsDto cartDetails = new CartDetailsDto()
    {
        Count = productDto.Count,
        ProductId = productDto.ProductId,
    };
    List<CartDetailsDto> cartDetailsDtos = new() { cartDetails };
    cartDto.CartDetails = cartDetailsDtos;
    ResponseDto? response = await _cartService.UpsertCartAsync(cartDto);
    if (response != null && response.IsSuccess)
    {
        TempData["success"] = "Товар додано до кошика";
        return RedirectToAction(nameof(Index));
    }
    else
    {
        TempData["error"] = response?.Message;
    }
    return View(cartDto);
}
```

Рис. 2.7 Метод **ProductDetails** (POST)

Цей код використовується для додавання продукту до кошика. Користувач заповнює деталі продукту, які потрібно додати до кошика, та відправляє запит. Запит обробляється сервісом **ICartService**, і якщо він вдалий, користувача перенаправляють на головну сторінку з повідомленням про успішне додавання товару до кошика. У разі помилки, повідомлення про помилку зберігається у **TempData**.

Метод **Login** (GET) – відображення сторінки входу.

```
[HttpGet]
public IActionResult Login()
{
    LoginRequestDto loginRequestDto = new();
    return View(loginRequestDto);
}
```

Рис. 2.8 Метод **Login** (GET)

		Ткачук М.А			ІПЗ.КР.Б-121-24-ІПЗ	Арк.
		Мінтій І.С				
Змн.	Арк.	№ докум.	Підпис	Дата		41

Цей код використовується для відображення сторінки входу. Метод створює новий об'єкт **LoginRequestDto** і передає його до представлення для відображення форми входу.

Метод **Login (POST)** – обробка запиту на вхід.

```
[HttpPost]
public async Task<IActionResult> Login(LoginRequestDto model)
{
    ResponseDto response = await _authService.LoginAsync(model);
    if (response != null && response.IsSuccess)
    {
        LoginResponseDto loginResponseDto =
        JsonConvert.DeserializeObject<LoginResponseDto>(Convert.ToString(response.Result));
        await SignInUser(loginResponseDto);
        _tokenProvider.SetToken(loginResponseDto.Token);
        return RedirectToAction("Index", "Home");
    }
    else
    {
        TempData["error"] = response.Message;
        return View(model);
    }
}
```

Рис. 2.9 Метод **Login (POST)**

Цей код використовується для обробки запиту на вхід. Він викликає метод **LoginAsync** сервісу автентифікації та, у випадку успішного входу, виконує метод **SignInUser** для автентифікації користувача у системі. Також токен автентифікації зберігається у провайдері токенів для подальшого використання. У разі помилки, повідомлення про помилку зберігається у **TempData**.

		Ткачук М.А			ІПЗ.КР.Б-121-24-ІПЗ	Арк.
		Мінтій І.С				
Змн.	Арк.	№ докум.	Підпис	Дата		42

Метод **Register** (POST) – обробка запиту на реєстрацію.

```
[HttpPost]
public async Task<IActionResult> Register(RegistrationRequestDto model)
{
    ResponseDto result = await _authService.RegisterAsync(model);
    ResponseDto assignRole;
    if (result != null && result.IsSuccess)
    {
        if (string.IsNullOrEmpty(model.Role))
        {
            model.Role = SD.RoleCustomer;
        }
        assignRole = await _authService.AssignRoleAsync(model);
        if (assignRole != null && assignRole.IsSuccess) {
            TempData["success"] = "Реєстрація успішна";
            return RedirectToAction(nameof(Login));
        }
    }
    else
    {
        TempData["error"] = result.Message;
    }
    var roleList = new List<SelectListItem>()
    {
        new() {Text= SD.RoleAdmin, Value= SD.RoleAdmin},
        new() {Text= SD.RoleCustomer, Value= SD.RoleCustomer}
    };
    ViewBag.RoleList = roleList;
    return View(model);
}
```

Рис. 2.10 Метод **Register** (POST)

Цей код використовується для обробки запиту на реєстрацію нового користувача. Метод викликає **RegisterAsync** сервісу автентифікації для створення нового користувача, а потім призначає роль користувачу. У разі успішної реєстрації та призначення ролі, користувача перенаправляють на сторінку входу. У разі помилки, повідомлення про помилку зберігається у **TempData**.

		Ткачук М.А			ІПЗ.КР.Б-121-24-ІПЗ	Арк.
		Мінтій І.С				
Змн.	Арк.	№ докум.	Підпис	Дата		43

Метод **SignInUser** – автентифікація користувача у системі.

```
private async Task SignInUser(LoginResponseDto login)
{
    var handler = new JwtSecurityTokenHandler();
    var jwt = handler.ReadJwtToken(login.Token);
    var identity = new ClaimsIdentity(CookieAuthenticationDefaults.AuthenticationScheme);
    identity.AddClaim(new Claim(JwtRegisteredClaimNames.Email,
        jwt.Claims.FirstOrDefault(u => u.Type == JwtRegisteredClaimNames.Email).Value));
    identity.AddClaim(new Claim(JwtRegisteredClaimNames.Sub,
        jwt.Claims.FirstOrDefault(u => u.Type == JwtRegisteredClaimNames.Sub).Value));
    identity.AddClaim(new Claim(JwtRegisteredClaimNames.Name,
        jwt.Claims.FirstOrDefault(u => u.Type == JwtRegisteredClaimNames.Name).Value));
    identity.AddClaim(new Claim(ClaimTypes.Name,
        jwt.Claims.FirstOrDefault(u => u.Type == JwtRegisteredClaimNames.Email).Value));
    identity.AddClaim(new Claim(ClaimTypes.Role,
        jwt.Claims.FirstOrDefault(u => u.Type == "role").Value));
    var principal = new ClaimsPrincipal(identity);
    await HttpContext.SignInAsync(CookieAuthenticationDefaults.AuthenticationScheme, principal);
}
```

Рис. 2.11 Метод **SignInUser**

Цей код використовується для автентифікації користувача у системі після успішного входу. Метод розбирає JWT токен [15], створює об'єкт **ClaimsIdentity** з відповідними клеймами (email, роль тощо) та виконує автентифікацію користувача, використовуючи Cookie Authentication.

Метод **Checkout** (POST) – обробка оформлення замовлення.

```
[HttpPost] [ActionName("Checkout")]
public async Task<IActionResult> Checkout(CartDto cartDto)
{
    CartDto cart = await LoadCartDtoBasedOnLoggedInUser();
    cart.CartHeader.Phone = cartDto.CartHeader.Phone;
    cart.CartHeader.Email = cartDto.CartHeader.Email;
    cart.CartHeader.Name = cartDto.CartHeader.Name;
    var response = await _orderService.CreateOrder(cart);
    OrderHeaderDto orderHeaderDto =
        JsonConvert.DeserializeObject<OrderHeaderDto>(Convert.ToString(response.Result));
    if (response != null && response.IsSuccess)
    {
        var domain = Request.Scheme + "://" + Request.Host.Value + "/";
        StripeRequestDto stripeRequestDto = new()
        {
            ApprovedUrl = domain + "cart/Confirmation?orderId=" + orderHeaderDto.OrderHeaderId,
            CancelUrl = domain + "cart/checkout", OrderHeader = orderHeaderDto
        };
        var stripeResponse = await _orderService.CreateStripeSession(stripeRequestDto);
        StripeRequestDto stripeResponseResult =
            JsonConvert.DeserializeObject<StripeRequestDto>(Convert.ToString(stripeResponse.Result));
        Response.Headers.Add("Location", stripeResponseResult.StripeSessionUrl);
        return new StatusCodeResult(303);
    }
    return View();
}
```

Рис. 2.12 Метод **Checkout** (POST)

		Ткачук М.А			ІПЗ.КР.Б-121-24-ІПЗ	Арк.
		Мінтій І.С				
Змн.	Арк.	№ докум.	Підпис	Дата		44

Цей код використовується для обробки оформлення замовлення. Метод оновлює дані замовлення користувача, створює замовлення та перенаправляє користувача до Stripe для оплати. У разі успішного створення замовлення, користувач перенаправляється на сторінку підтвердження замовлення.

Метод **GetAll** – отримання всіх замовлень з можливістю фільтрації за статусом.

```
[HttpGet]
public IActionResult GetAll(string status)
{
    IEnumerable<OrderHeaderDto> list;
    string userId = "";
    if (!User.IsInRole(SD.RoleAdmin))
    {
        userId = User.Claims.Where(u => u.Type == JwtRegisteredClaimNames.Sub)?.FirstOrDefault()?.Value;
    }
    ResponseDto response = _orderService.GetAllOrder(userId).GetAwaiter().GetResult();
    if(response != null && response.IsSuccess)
    {
        list = JsonConvert.DeserializeObject<List<OrderHeaderDto>>(Convert.ToString(response.Result));
        switch (status)
        {
            case "approved":
                list = list.Where(u=>u.Status == SD.Status_Approved);
                break;
            case "readyforpickup":
                list = list.Where(u => u.Status == SD.Status_ReadyForPickup);
                break;
            case "cancelled":
                list = list.Where(u => u.Status == SD.Status_Cancelled);
                break;
            default:
                break;
        }
    }
    else
    {
        list = new List<OrderHeaderDto>();
    }
    return Json(new {data = list});
}
```

Рис. 2.13 Метод **GetAll**

Цей код використовується для отримання всіх замовлень з можливістю фільтрації за статусом. Метод викликає сервіс для отримання всіх замовлень для поточного користувача або адміністратора і фільтрує їх за вказаним статусом. Потім результати повертаються у вигляді JSON.

		Ткачук М.А			ІПЗ.КР.Б-121-24-ІПЗ	Арк.
		Мінтій І.С				
Змн.	Арк.	№ докум.	Підпис	Дата		45

Метод **SendAsync** – відправка асинхронних HTTP запитів.

```
public async Task<ResponseDto> SendAsync(RequestDto requestDto, bool withBearer = true)
{
    try
    {
        HttpClient client = _httpClientFactory.CreateClient("CoffeeShopAPI");
        HttpRequestMessage message = new();
        if(requestDto.ContentType == ContentType.MultipartFormData)
        {message.Headers.Add("Accept", "*/*");}
        else
        {message.Headers.Add("Accept", "application/json");}
        if (withBearer)
        {var token = _tokenProvider.GetToken();
            message.Headers.Add("Authorization", $"Bearer {token}");}
        message.RequestUri = new Uri(requestDto.Url); //Get request
        if(requestDto.ContentType == ContentType.MultipartFormData)
        {var content = new MultipartFormDataContent();
            foreach(var prop in requestDto.Data.GetType().GetProperties())
            {var value = prop.GetValue(requestDto.Data);
                if(value is FormFile) {var file = (FormFile)value;
                    if(file != null)
                        {content.Add(new StreamContent(file.OpenReadStream()), prop.Name, file.FileName);}}
                else{content.Add(new StringContent(value == null ? "" : value.ToString(), prop.Name);}}
            message.Content = content;}
        else{if (requestDto.Url != null)
            {message.Content = new StringContent(JsonConvert.SerializeObject(requestDto.Data), Encoding.UTF8,
"application/json"); //serialize}}
            HttpResponseMessage? apiResponse = null;
            switch (requestDto.ApiType)
            {case ApiType.POST: message.Method = HttpMethod.Post; break;
                case ApiType.DELETE:message.Method = HttpMethod.Delete;break;
                case ApiType.PUT:message.Method = HttpMethod.Put;break;
                default:message.Method = HttpMethod.Get;break;}
            apiResponse = await client.SendAsync(message);
            switch (apiResponse.StatusCode)
            {case HttpStatusCode.NotFound:return new() { IsSuccess = false, Message = "Not Found" };
                case HttpStatusCode.Forbidden:return new() { IsSuccess = false, Message = "Access Denied" };
                case HttpStatusCode.Unauthorized:return new() { IsSuccess = false, Message = "Unauthorized" };
            case HttpStatusCode.InternalServerError:return new() { IsSuccess = false, Message = "Internal Server Error" };
                default: var apiContent = await apiResponse.Content.ReadAsStringAsync();
                    var apiResponseDto = JsonConvert.DeserializeObject<ResponseDto>(apiContent);
                    return apiResponseDto;}}
            catch (Exception ex)
            {var dto = new ResponseDto{Message = ex.Message.ToString(),IsSuccess = false};return dto;}
    }
}
```

Рис. 2.14 Метод **SendAsync**

Цей код використовується для відправки асинхронних HTTP запитів до API сервера. Метод створює запит, додає заголовки та токени, серіалізує дані і відправляє запит. Він підтримує різні типи запитів, такі як GET, POST, PUT, DELETE, та обробляє відповіді від сервера, повертаючи об'єкт **ResponseDto**.

1. Визначення клієнта та повідомлення

		Ткачук М.А			ІПЗ.КР.Б-121-24-ІІЗ	Арк.
		Мінтій І.С				
Змн.	Арк.	№ докум.	Підпис	Дата		46

- Метод починається з визначення клієнта **HttpClient** та створення повідомлення **HttpRequestMessage**. Він перевіряє тип контенту (**ContentType**) і додає відповідний заголовок **Accept**.

2. Додавання токєну

- Якщо параметр **withBearer** встановлено як **true**, метод додає токен авторизації в заголовок запиту.

3. Встановлення URI запиту

- Метод встановлює URI для запиту на основі властивості **Url** з об'єкта **RequestDto**.

4. Обробка контенту запиту

- Якщо тип контенту **MultipartFormData**, метод створює **MultipartFormDataContent** і додає кожен властивість з об'єкта **Data** в контент запиту. Якщо властивість є файлом, вона додається як **StreamContent**.
- В інших випадках дані серіалізуються в JSON і додаються до запиту як **StringContent**.

5. Встановлення методу запиту

- Метод встановлює тип HTTP методу (**POST**, **DELETE**, **PUT**, **GET**) на основі властивості **ApiType** з об'єкта **RequestDto**.

6. Відправка запиту

- Метод відправляє запит за допомогою **SendAsync** і обробляє відповідь, перевіряючи статусний код відповіді. Відповідно до статусного коду, метод повертає об'єкт **ResponseDto** з відповідним повідомленням про помилку або успіх.

7. Обробка виключень

- Якщо під час виконання запиту виникає виключення, метод ловить його та повертає об'єкт **ResponseDto** з повідомленням про помилку та встановленим прапором **IsSuccess** у **false**.

		Ткачук М.А			ІПЗ.КР.Б-121-24-ІПЗ	Арк.
		Мінтій І.С				
Змн.	Арк.	№ докум.	Підпис	Дата		47

Цей метод забезпечує універсальний механізм для відправки HTTP запитів до API сервера, що дозволяє взаємодіяти з різними кінцевими точками API, обробляючи різні типи запитів і відповідей.

Метод **AddAppAuthetication**— додавання JWT аутентифікації до веб-додатка.

```
public static WebApplicationBuilder AddAppAuthetication(this WebApplicationBuilder builder)
{
    var settingsSection = builder.Configuration.GetSection("ApiSettings");
    var secret = settingsSection.GetValue<string>("Secret");
    var issuer = settingsSection.GetValue<string>("Issuer");
    var audience = settingsSection.GetValue<string>("Audience");
    var key = Encoding.ASCII.GetBytes(secret);
    builder.Services.AddAuthentication(x =>
    {
        x.DefaultAuthenticateScheme = JwtBearerDefaults.AuthenticationScheme;
        x.DefaultChallengeScheme = JwtBearerDefaults.AuthenticationScheme;
    }).AddJwtBearer(x =>
    {
        x.TokenValidationParameters = new TokenValidationParameters
        {
            ValidateIssuerSigningKey = true,
            IssuerSigningKey = new SymmetricSecurityKey(key),
            ValidateIssuer = true,
            ValidIssuer = issuer,
            ValidAudience = audience,
            ValidateAudience = true
        };
    });
    return builder;
}
```

Рис. 2.12 Метод **AddAppAuthetication**

Цей код використовується для додавання JWT аутентифікації до веб-додатка. Метод зчитує налаштування для JWT із конфігурації додатка, налаштовує параметри аутентифікації та додає відповідні служби до контейнера залежностей.

Метод **AzureServiceBusConsumer** [16] – обробка повідомлень у черзі за допомогою Azure Service Bus.

```
public AzureServiceBusConsumer(IConfiguration configuration, RewardService rewardService)
{
    _rewardService = rewardService;
    _configuration = configuration;
    serviceBusConnectionString = _configuration.GetValue<string>("ServiceBusConnectionString");
    orderCreatedTopic = _configuration.GetValue<string>("TopicAndQueueNames:OrderCreatedTopic");
    orderCreatedRewardSubscription =
    _configuration.GetValue<string>("TopicAndQueueNames:OrderCreated_Rewards_Subscription");
    var client = new ServiceBusClient(serviceBusConnectionString);
    _rewardProcessor = client.CreateProcessor(orderCreatedTopic, orderCreatedRewardSubscription);
}
```

Рис. 2.12 Метод **AzureServiceBusConsumer**

Цей клас використовується для обробки повідомлень у черзі за допомогою Azure Service Bus. Основні кроки включають ініціалізацію клієнта Service Bus, налаштування процесора повідомлень і обробку отриманих повідомлень.

Метод **CreateStripeSession** – створення сесії для Stripe оплати.

```
[Authorize]
[HttpPost("CreateStripeSession")]
public async Task<ResponseDto> CreateStripeSession([FromBody] StripeRequestDto stripeRequestDto)
{
    try
    {
        var options = new SessionCreateOptions
        {
            SuccessUrl = stripeRequestDto.ApprovedUrl,
            CancelUrl = stripeRequestDto.CancelUrl,
            LineItems = new List<SessionLineItemOptions>(),
            Mode = "payment",
        };
        var DiscountsObj = new List<SessionDiscountOptions>()
        {
            new SessionDiscountOptions
            {
                Coupon = stripeRequestDto.OrderHeader.CouponCode
            }
        };
        foreach (var item in stripeRequestDto.OrderHeader.OrderDetails)
        {
            var sessionLineItem = new SessionLineItemOptions
            {
                PriceData = new SessionLineItemPriceDataOptions
                {
                    UnitAmount = (long)(item.Price * 100),
                    Currency = "uah",
                    ProductData = new SessionLineItemPriceDataProductDataOptions
                    {
                        Name = item.Product.Name
                    },
                    Quantity = item.Count
                }
            };
            options.LineItems.Add(sessionLineItem);
        }
        if (stripeRequestDto.OrderHeader.Discount > 0)
        {
            options.Discounts = DiscountsObj;
        }
        var service = new SessionService();
        Session session = service.Create(options);
        stripeRequestDto.StripeSessionUrl = session.Url;
        OrderHeader orderHeader = _db.OrderHeaders.First(u => u.OrderHeaderId == stripeRequestDto.OrderHeader.OrderHeaderId);
        orderHeader.StripeSessionId = session.Id;
        _db.SaveChanges();
        _response.Result = stripeRequestDto;
    }
    catch (Exception ex)
    {
        _response.Message = ex.Message;
        _response.IsSuccess = false;
    }
    return _response;
}
```

Рис. 2.12 Метод **CreateStripeSession**

Цей код використовується для створення Stripe [\[17\]](#) сесії для оплати. Метод отримує дані замовлення з вхідного об'єкта **StripeRequestDto**, налаштовує параметри сесії, включаючи список товарів, знижки, і URLs для успіху та

		Ткачук М.А			ІПЗ.КР.Б-121-24-ІПЗ	Арк.
		Мінтій І.С				
Змн.	Арк.	№ докум.	Підпис	Дата		49

скасування. Далі, метод створює сесію за допомогою Stripe API, зберігає ідентифікатор сесії в базі даних і повертає результат у вигляді JSON.

Метод **ValidateStripeSession** – перевірка сесії для Stripe оплати.

```
[Authorize]
[HttpPost("ValidateStripeSession")]
public async Task<ResponseDto> ValidateStripeSession([FromBody] int orderHeaderId)
{
    try
    {
        OrderHeader orderHeader = _db.OrderHeaders.First(u => u.OrderHeaderId == orderHeaderId);
        var service = new SessionService();
        Session session = service.Get(orderHeader.StripeSessionId);
        var paymentIntentService = new PaymentIntentService();
        PaymentIntent paymentIntent = paymentIntentService.Get(session.PaymentIntentId);
        if(paymentIntent.Status == "succeeded")
        {
            //paymend is okay
            orderHeader.PaymentIntentId = paymentIntent.Id;
            orderHeader.Status = SD.Status_Approved;
            _db.SaveChanges();
            RewardsDto rewardsDto = new()
            {
                OrderId = orderHeader.OrderHeaderId,
                RewardsActivity = Convert.ToInt32(orderHeader.OrderTotal),
                UserId = orderHeader.UserId
            };
            string topicName = _configuration.GetValue<string>("TopicAndQueueNames:OrderCreatedTopic");
            await _messageBus.PublishMessage(rewardsDto, topicName);
            _response.Result = _mapper.Map<OrderHeaderDto>(orderHeader);
        }
    }
    catch (Exception ex)
    {
        _response.Message = ex.Message;
        _response.IsSuccess = false;
    }
    return _response;
}
```

Рис. 2.12 Метод **ValidateStripeSession**

Цей код використовується для перевірки Stripe сесії для оплати. Метод отримує ідентифікатор замовлення, витягує відповідне замовлення з бази даних, перевіряє статус платежу за допомогою Stripe API. Якщо платіж успішний, оновлює статус замовлення, створює об'єкт нагороди та публікує повідомлення за допомогою message bus. Результати повертаються у вигляді JSON.

		Ткачук М.А			ІПЗ.КР.Б-121-24-ІПЗ	Арк.
		Мінтій І.С				
Змн.	Арк.	№ докум.	Підпис	Дата		50

Висновки до другого розділу.

В процесі проектування інтернет-магазину з використанням мікросервісів були проаналізовані всі варіанти використання програмного продукту відповідно до вимог програмного забезпечення. На основі отриманого результату була спроектована структура платформи. Проектування включало моделювання структури проекту та взаємодії між його складовими частинами.

Також було описано базу даних для даних, які будуть зберігатися на серверній частині системи, таких як користувачі, продукти, замовлення та категорії.

Було описано алгоритм роботи з додатком та детально розглянуті основні етапи взаємодії користувачів із системою, включаючи автентифікацію, авторизацію, обробку замовлень та керування кошиком.

Крім того, був наведений опис основного функціоналу додатку – це створення та управління акаунтами користувачів, додавання продуктів до кошика, оформлення замовлень, застосування купонів, та інші функції, які забезпечують повноцінне функціонування інтернет-магазину.

Таким чином, другий розділ роботи демонструє комплексний підхід до проектування інтернет-магазину з використанням сучасних технологій мікросервісів, що забезпечує його масштабованість, надійність та зручність у використанні.

		Ткачук М.А			ІПЗ.КР.Б-121-24-ІПЗ	Арк.
		Мінтій І.С				
Змн.	Арк.	№ докум.	Підпис	Дата		51

РОЗДІЛ 3 ІНТЕРФЕЙС ТА ПОРЯДОК РОБОТИ З ВЕБ-ДОДАТКОМ

3.1 Порядок встановлення та налаштування параметрів платформи

Встановлення та налаштування інструментів вимагає розгортання всіх необхідних компонентів та налаштування як кожного елементу окремо, так і загальної системи в цілому. Для опису архітектури курсу було створено діаграму розгортання (рис. 3.1).

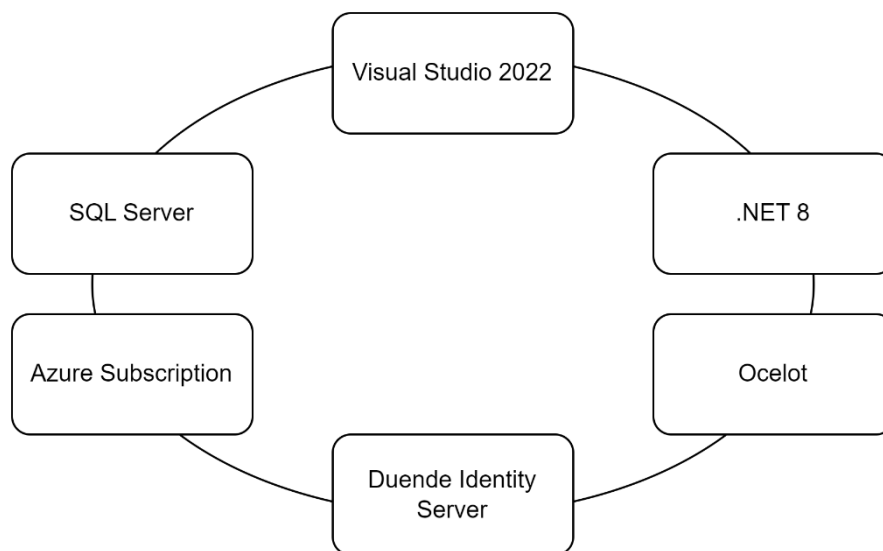


Рис. 3.1. Діаграма розгортання інструментів

Опис процесу розгортання.

Метою є розгортання платформи, яка включає налаштування Visual Studio, .NET 8, SQL Server та Azure Service Bus для забезпечення ефективної роботи проєкту. Інструменти інтегруються для забезпечення розробки, тестування та розгортання додатків з використанням різних служб та технологій.

Етапи розгортання

1. Налаштування Visual Studio та .NET 8:

- Завантажте Visual Studio 2022 preview версію для використання .NET 8.
- Встановіть .NET 8 SDK.

2. Встановлення SQL Server

- Завантажте та встановіть SQL Server.
- Встановіть SQL Server Management Studio для управління базою

даних.

- Налаштуйте базу даних для проєкту, створивши необхідні таблиці та зв'язки між ними.

3. Підписка на Azure

- Зареєструйтеся на Azure.
- Використовуйте Azure Service Bus для комунікації між API.
- Налаштуйте Azure Service Bus, створивши необхідні топіки та черги для вашого проєкту.
- Врахуйте можливі витрати близько \$10 на місяць, або використайте безкоштовний кредит від Azure.

4. Інтеграція з GitHub репозиторієм

5. Налаштування конфігураційних файлів

- Відредагуйте файли конфігурацій проєкту, такі як appsettings.json, для відповідності вашим локальним налаштуванням та підключенню до SQL Server і Azure Service Bus.
- Переконайтеся, що всі ключі та з'єднання правильно налаштовані.

6. Запуск проєкту

- Відкрийте проєкт у Visual Studio.
- Запустіть проєкт у режимі відладки, переконавшись, що всі сервіси працюють правильно.
- Тестуйте функціональність проєкту, перевіряючи зв'язок між компонентами та коректність виконання операцій.

З вищенаведеними кроками ви зможете налаштувати всі необхідні інструменти для роботи над проєктом, включаючи Visual Studio, SQL Server та Azure, інтегровані з кодом з GitHub репозиторію. Це забезпечить ефективну роботу проєкту та полегшить процес його розгортання та тестування.

3.2 Структура інтерфейсу для користувача інтернет-магазину кави CoffeeShop

		Ткачук М.А			ІПЗ.КР.Б-121-24-ІПЗ	Арк.
		Мінтій І.С				
Змн.	Арк.	№ докум.	Підпис	Дата		53

На сьогоднішній день на сторінку сайту можна зайти по посиланню, або запустивши його локально.

Після того як користувач заїде на сторінку на головній сторінці (рис. 3.2) він побачить вітання, там асортимент, який є у цьому магазині.

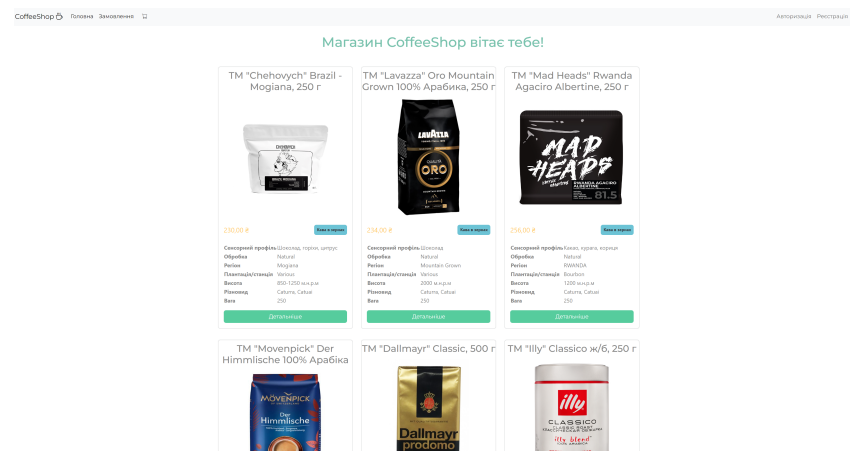


Рисунок 3.2 – Головна сторінка

На цій сторінці користувач може авторизуватись через вже створений обліковий запис (рис 3.3) або зареєструватися (рис 3.4). Після успішної авторизації користувач буде перенаправлений на головну сторінку.

Авторизація

Авторизуватися

Нема акаунту?

Рисунок 3.3 - Сторінка авторизації

Реєстрація

Зареєструватися

Рисунок 3.4 - Сторінка реєстрації

На цій сторінці відображається інформація про товар, а саме його назва, категорія, опис та ціна (рис. 3.5). Також користувач може обрати потрібну

кількість товару (по замовчуванню це 1 одиниця) та додати його до кошика, або повернутися до каталогу товарів.



Рисунок 3.5 – Сторінка детальної інформації про товар

Коли користувач обрав потрібний для себе товар, він може перейти на сторінку «Кошик», де зможе подивитися весь товар який він обрав, а саме його назву, опис, кількість та ціну. Також користувач бачить суму замовлення (рис. 3.6) і може використати купон, який надасть йому знижку, як це зображено на рисунку 3.7. Якщо все добре, він може перейти далі до оформлення замовлення (рис. 3.8), або натиснути на кнопку «Відстеження кошику», яка надасть йому можливість отримувати сповіщення на електронну пошту у разі, якщо на товар у його кошику відбудуться зміни, або повернутися назад до покупок.

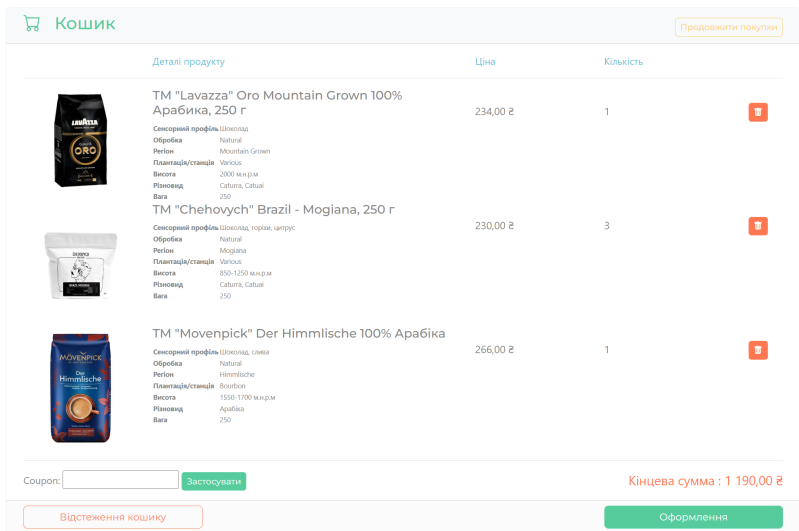


Рисунок 3.6 – Сторінка «Кошик»

Кошик

Продовжити покупки

Деталі продукту	Ціна	Кількість	
 ТМ "Lavazza" Oro Mountain Grown 100% Арабіка, 250 г Сенсорний профіль Шоколад Обробка Natural Регіон Mountain Grown Платформа/станція Various Висота 2000 м.н.р.м Різнovid Catutta, Catbali Вага 250	234,00 ₴	1	
 ТМ "Chehovych" Brazil - Mogiana, 250 г Сенсорний профіль Шоколад, горіхи, цитрус Обробка Natural Регіон Mogiana Платформа/станція Various Висота 850-1250 м.н.р.м Різнovid Catutta, Catbali Вага 250	230,00 ₴	3	
 ТМ "Movenpick" Der Himmlische 100% Арабіка Сенсорний профіль Шоколад, слива Обробка Natural Регіон Himmlische Платформа/станція Bourbon Висота 1550-1700 м.н.р.м Різнovid Арабіка Вага 250	266,00 ₴	1	

Сoupon: 20OFF

Прибрати

Кінцева сума : 1 170,00 ₴

Знижка до замовлення : 20,00 ₴

Відстеження кошику

Оформлення

Рисунок 3.7 – Сторінка «Кошик» з використанням купону

Підсумок замовлення

Повернутися до кошику

Повне ім'я...

Email...

Номер телефону...

Ми доставляємо замовлення протягом 1-2 діб!

Деталі продукту	Ціна	Кількість
ТМ "Lavazza" Oro Mountain Grown 100% Арабіка, 250 г	234,00 ₴	1
ТМ "Chehovych" Brazil - Mogiana, 250 г	230,00 ₴	3
ТМ "Movenpick" Der Himmlische 100% Арабіка	266,00 ₴	1
Кінцева сума : 1 170,00 ₴		
Знижка до замовлення : 20,00 ₴		
Перейти до оплати		

Рисунок 3.8 – Сторінка «Оформлення замовлення»

На сторінці «Оформлення замовлення» (рис. 3.9) користувач повинен заповнити всі поля інформацією про себе (рис. 3.10), для можливості переходу до сторінки оплатити замовлення та в майбутньому вдалого отримання свого замовлення.

Підсумок замовлення

Повернутися до кошику

Микита Ткачук

nikitatkachuk6@gmail.com

+380 66 155 144 5

Ми доставляємо замовлення протягом 1-2 діб!

Деталі продукту	Ціна	Кількість
TM "Lavazza" Oro Mountain Grown 100% Арабіка, 250 г	234,00 ₴	1
TM "Chehovych" Brazil - Mogiana, 250 г	230,00 ₴	3
TM "Movenpick" Der Himmlische 100% Арабіка	266,00 ₴	1
Кінцева сума : 1 170,00 ₴ Знижка до замовлення : 20,00 ₴		

Перейти до оплати

Рисунок 3.9 – Сторінка «Оформлення замовлення» з заповненими полями

Якщо користувач все зробив коректно, він буде перенаправлений на сторінку оплати платіжної системи Stripe, на якій у нього буде можливість побачити детальну інформацію про своє замовлення та варіанти його сплати

← CoffeeShop TEST MODE

Pay CoffeeShop

UAH 1,170.00

TM "Lavazza" Oro Mountain Grown 100% Арабіка, 250 г	UAH 234.00
Qty 1	
TM "Chehovych" Brazil - Mogiana, 250 г	UAH 690.00
Qty 3	
	UAH 230.00 each
TM "Movenpick" Der Himmlische 100% Арабіка	UAH 266.00
Qty 1	
Subtotal	UAH 1,190.00
20OFF	-UAH 20.00
UAH 20.00 off	
Total due	UAH 1,170.00

Powered by stripe

Terms Privacy

link VISA 1111

Or pay with card

Email [Continue with Link](#)

nikitatkachuk6@gmail.com

Card information

1234 1234 1234 1234

MM / YY

CVC

Cardholder name

Full name on card

Country or region

Ukraine

Pay

Рисунок 3.10 – Сторінка оплати

Після успішної оплати, користувач буде перенаправлений на сторінку «Дякую за замовлення» на якій побачить номер свого замовлення, як це зображено на рисунку 3.11

		Ткачук М.А			ІПЗ.КР.Б-121-24-ІПЗ	Арк.
		Мінтій І.С				
Змн.	Арк.	№ докум.	Підпис	Дата		57

Вітаю! Ви на крок ближче до смачної кави.

Ми доставляємо замовлення протягом 1-2 діб!

Номер замовлення - 1011



Рисунок 3.11 – Сторінка «Дякую за замовлення»

Також користувач може подивитися свої замовлення, скористатися фільтруванням, та отримати інформацію про свої замовлення на сторінці «Замовлення» (рис. 3.12). Також він може подивитися більш детальну інформацію про обране замовлення, як це зображено на рисунку 3.13

Список замовлень

Підтверджено

Готовий до відправлення

Скасовано

Всі

10

entries per page

Search:

ID	Email	Ім'я	Номер телефону	Статус	Сума	
1011	nikitatkachuk6@gmail.com	Микита Ткачук	+380 66 155 144 5	Approved	1170	

Showing 1 to 1 of 1 entry

«

<

1

>

»

Рисунок 3.12– Сторінка «Замовлення»

Інформація про замовлення

Повернутися до списку

Ім'я

Микита Ткачук

Номер телефону

+380 66 155 144 5

Email

nikitatkachuk6@gmail.com

Дата замовлення

04-Jun-24

Статус замовлення

Approved

Деталі замовлення

Статус замовлення - Approved

TM "Lavazza" Oro Mountain Grown 100% Арабіка, 250 г

234,00 ₪

Price : 234,00 ₪

Quantity : 1

TM "Chehovych" Brazil - Mogiana, 250 г

690,00 ₪

Price : 230,00 ₪

Quantity : 3

TM "Movenpick" Der Himmlische 100% Арабіка

266,00 ₪

Price : 266,00 ₪

Quantity : 1

Кінцева ціна

1 170,00 ₪

Рисунок 3.13 – Сторінка «Деталі замовлення»

		Ткачук М.А			ІПЗ.КР.Б-121-24-ІПЗ	Арк.
		Мінтій І.С				
Змн.	Арк.	№ докум.	Підпис	Дата		58

3.3 Структура інтерфейсу для адміністратора інтернет-магазину кави CoffeeShop

На сайті передбачена авторизація під користувачем Адміністратор, окрім можливостей звичайного користувача, йому доступно керування асортиментом, купонами, відслідковування та модерування замовленнями (рис. 3.14)

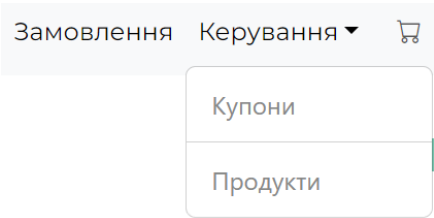


Рисунок 3.14 – Панель «Керування»

Адміністратор має можливість передивлятися асортимент, додавати новий, редагувати або видаляти його.

Список продуктів				
Створити продукт				
Назва	Категорія	Ціна		
TM "Chehovych" Brazil - Mogiana, 250 r	Кава в зернах	230,00 ₴	☒	☐
TM "Lavazza" Oro Mountain Grown 100% Арабіка, 250 r	Кава в зернах	234,00 ₴	☒	☐
TM "Mad Heads" Rwanda Agaciro Albertaine, 250 r	Кава в зернах	256,00 ₴	☒	☐
TM "Movenpick" Der Himmlische 100% Арабіка	Кава в зернах	266,00 ₴	☒	☐
TM "Dallmayr" Classic, 500 r	Кава в зернах	274,00 ₴	☒	☐
TM "illy" Classico ж/б, 250 r	Кава в зернах	311,00 ₴	☒	☐
TM "Lazzaro" Cartel Бразилія, 1 шт	Дріп Кава	15,00 ₴	☒	☐
TM "Chehovych" Ethiopia - Yirgacheffe, 12 r	Дріп кава	38,00 ₴	☒	☐
TM "Lazzaro" Cartel Ефіопія, 1 шт	Дріп Кава	20,00 ₴	☒	☐
TM "Chehovych" Kenya - Ruka Chui, 12 r	Дріп кава	40,00 ₴	☒	☐
TM "INKER" Luna white, 170 мл	Чашка	158,00 ₴	☒	☐
TM "INKER" Luna red, 70 мл	Чашка	216,00 ₴	☒	☐
TM "INKER" Luna dark grey, 70 мл	Чашка	216,00 ₴	☒	☐

Рисунок 3.15 – Сторінка «Керування продуктами»

На сторінці створення нового продукту, адміністратор повинен заповнити всі поля, а саме Назва, Категорія, Опис, Ціна та Фотокартка (рис. 3.16). У разі не заповнення одного з полів, або завантаження фотокартки великого розміру, або формату який не підтримується, йому буде сповіщено про це, як це зображено на рисунку 3.17.

Рисунок 3.16 – Сторінка додавання нового продукту

Рисунок 3.17 – Не виконання параметрів валідації

У випадку, якщо треба щось замінити у вже існуючому продукті, адміністратор може скористатися сторінкою «Редагування», де внесе відповідні зміни і вони будуть збережені (рис. 3.18). У випадку, якщо товар вже не актуальний, або був доданий випадково, адміністратор може скористуватися сторінкою «Видалення», як це зображено на рисунку 3.19.

Рисунок 3.18 – Сторінка «Редагування»

Видалення продукту

Назва

ТМ "Chehovych" Brazil - Mogiana, 250 r

Категорія

Кава в зернах

Опис

Сенсорний профіль	Шоколад, горіхи, цитрус	Обробка	
Natural	Perion	Mogiana	Плантація/станція
Various	Висота	850-1250 м.н.р.м	Різновид
Catuai	Bara	250	Caturra,

Ціна

230

Url картинки

https://localhost:7000/ProductImages/1.jpg

Повернутися до списку

Видалити

Рисунок 3.19 – Сторінка «Видалення»

У права адміністратора також входить можливість роботи з купонами, а саме їх перегляд, додавання та видалення (рис. 3.20) Також це можна робити, використовуючи панель платіжної системи Stripe, якою виконуються платіжні операції на цьому сайті (рис. 3.21).

Список купонів

Створити новий купон

Код купону	Знижка	Мінімальна сума	
10OFF	10,00 ₴	30,00 ₴	
20OFF	20,00 ₴	50,00 ₴	

Рисунок 3.20 – Сторінка «Купони»

Product catalog

All products	Coupons	Shipping rates	Tax rates	Pricing tables	Meters
COUPON			TERMS		
15OFF			₹15.00 off once		
20OFF			₹20.00 off once		
10OFF			₹10.00 off once		
3 results					

Рисунок 3.21 – Сторінка «Купони» в платіжній системі Stripe

При створенні нового купону, адміністратор повинен ввести інформацію про нього, а саме Код купону, його Знижку та Мінімальну суму при якій цей купон буде застосовуватись (рис. 3.22).

Створити купон

Код купону

15OFF

Знижка

15

Мінімальна сума

70

Повернутися до списку

Створити

Рисунок 3.22 – Статистика активних користувачів (без анонімних)

Також адміністратор може видаляти купони в разі їх не потрібності (рис. 3.23).

Видалити купон

Код купону

15OFF

Знижка

15

Мінімальна сума

70

Повернутися до списку

Видалити

Рисунок 3.23 – Статистика активних користувачів (без анонімних)

При роботі із замовленнями, адміністратор може переглядати замовлення які були здійснені користувачами на сайті, сортувати за статусом та модерувати ними (рис. 3.24).

Список замовлень

Підтверджено

Готовий до відправлення

Скасовано

Всі

10 entries per page

Search:

ID	Email	Ім'я	Номер телефону	Статус	Сума	
1012	nikitatchuk6@gmail.com	Микита Ткачук	+380 66 155 144 5	Pending	1170.00	☒
1011	nikitatchuk6@gmail.com	Микита Ткачук	+380 66 155 144 5	Approved	1170.00	☒
1010	nikitatchuk6@gmail.com	Микита Ткачук	+380 66 155 144 5	Pending	1170.00	☒
1009	zheka@shytov.com	Mykyta Tkachuk	0991112223	Approved	1180.00	☒
1008	zheka@shytov.com	Zhenya Shytov	0991112223	Approved	63.99	☒
1007	olenko@mail.com	Oleg Voitenko	123897598	Completed	63.99	☒
1006	zheka@shytov.com	Zhenya Shytov	0991112223	Completed	127.98	☒
1005	nikitatchuk6@gmail.com	Mykyta Tkachuk	0661551445	Cancelled	48.99	☒
1004	olenko@mail.com	Oleg Voitenko	123897598	Cancelled	69.98	☒
1003	nikitatchuk6@gmail.com	Mykyta Tkachuk	0661551445	Cancelled	26.98	☒

Showing 1 to 10 of 24 entries

1

2

3

Рисунок 3.24 – Статистика активних користувачів (без анонімних)

		Ткачук М.А			ІПЗ.КР.Б-121-24-ІПЗ	Арк.
		Мінтій І.С				
Змн.	Арк.	№ докум.	Підпис	Дата		62

Для роботи із замовленням користувача який успішно здійснив оплату, адміністратор може відсортувати їх за фільтром «Підтверджено» та проводити операції по зміні статусу, або його скасуванню як це показано на рисунках 3.25 – 3.26. Також може побачити ID операції та ID сесії та за необхідності перевірити у панелі адміністратора платіжної системи Stripe, як це зображено на рисунку 3.27.

Список замовлень

Підтверджено

Готовий до відправлення

Скасовано

Всі

10 entries per page

Search:

ID	Email	Ім'я	Номер телефону	Статус	Сума	
1011	nikitakachuk6@gmail.com	Микита Ткачук	+380 66 155 144 5	Approved	1170.00	
1009	zheka@shytov.com	Mykyta Tkachuk	0991112223	Approved	1180.00	
1008	zheka@shytov.com	Zhenya Shytov	0991112223	Approved	63.99	

Showing 1 to 3 of 3 entries

«

«

1

»

»

Рисунок 3.25 – Фільтрація замовлень по фільтру «Підтверджено»

Інформація про замовлення		Повернутися до списку	
Ім'я	Микита Ткачук	Деталі замовлення	
Номер телефону	+380 66 155 144 5	Статус замовлення - Approved	
Email	nikitakachuk6@gmail.com	TM "Lavazza" Oro Mountain Grown 100% Арабіка, 250 г	
Дата замовлення	04-Jun-24	Price : 234,00 €	
Session ID	cs_test_b1VsJsgttPqXHZOs3Kgn6HKj5lKnaDyPGoHsy5MRxL	Quantity : 1	
Payment Intent ID	pi_3PO3STIO7QFz6AR417PGLLK3	TM "Chehovych" Brazil - Mogiana, 250 г	
Статус замовлення	Approved	Price : 230,00 €	
		Quantity : 3	
		TM "Movenpick" Der Himmlische 100% Арабіка	
		Price : 266,00 €	
		Quantity : 1	
		Кінцева ціна	
		1170,00 €	
		Готовий до відправлення	
		Скасувати замовлення	

Рисунок 3.26 – Сторінка перегляду замовлення

Payments					
All payments					Create payment
All	Succeeded	Refunded	Uncaptured	Failed	
12	8	4	0	0	
Date and time Amount Currency Status Payment method More filters Export Edit columns					
	Amount	Payment method	Description	Customer	Date
	81,170.00 UAH	Succeeded	pi_3PO3STIO7QFz6AR417PGLLK3	nikitakachuk6@gmail.com	Jun 4, 10:55 PM
	81,180.00 UAH	Succeeded	pi_3PMz6F107QFz6AR48BVGW2k	nikitakachuk6@gmail.com	Jun 4, 6:17 PM
	863.99 UAH	Succeeded	pi_3PMYX107QFz6AR41QHRBC1	nikitakachuk6@gmail.com	Jun 3, 1:54 PM
	863.99 UAH	Succeeded	pi_3PMXKL107QFz6AR48v8PQfj	nikitakachuk6@gmail.com	Jun 3, 12:38 PM
	8127.98 UAH	Succeeded	pi_3PMXV107QFz6AR48C2kuBN	nikitakachuk6@gmail.com	Jun 2, 10:56 PM
	848.99 UAH	Refunded	pi_3PMz2a107QFz6AR41F1b23CY	nikitakachuk6@gmail.com	Jun 2, 1:53 PM
	863.98 UAH	Refunded	pi_3PM8pU107QFz6AR482EhgvG	nikitakachuk6@gmail.com	Jun 2, 1:40 PM
	826.98 UAH	Refunded	pi_3PMqR107QFz6AR48gu1GHN	nikitakachuk6@gmail.com	Jun 1, 2:21 PM
	826.98 UAH	Refunded	pi_3PMow107QFz6AR48aUCtXc	nikitakachuk6@gmail.com	Jun 1, 1:13 PM
	826.98 UAH	Refunded	pi_3PMWk107QFz6AR48B9FLu8	nikitakachuk6@gmail.com	May 31, 5:41 PM
	822.97 UAH	Succeeded	pi_3PMWY107QFz6AR41bg12Xg2	nikitakachuk6@gmail.com	May 31, 5:35 PM
	8224.89 UAH	Succeeded	pi_3PMzF107QFz6AR41PUs4Vz	nikitakachuk6@gmail.com	May 31, 5:18 PM
12 results					

Рисунок 3.27 – Перегляд статусу оплати у платіжній системі Stripe

Використовуючи портал Azure, адміністратор може переглядати Topic, Queue та Subscriptions, змінювати їх, додавати їх при необхідності для логування дій користувача на сайті, як це зображено на рисунку 3.28-3.29

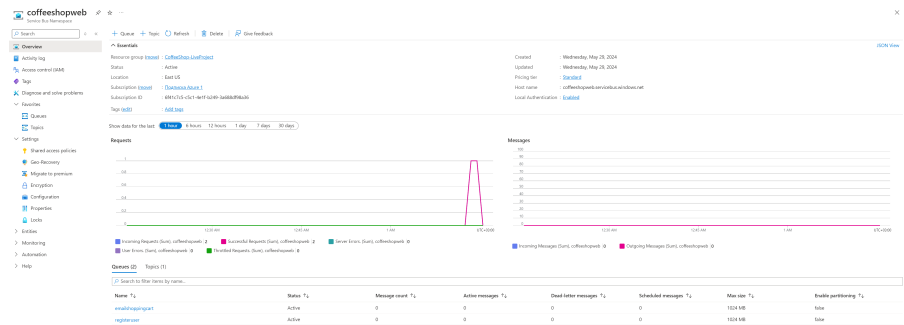


Рисунок 3.28 – Сторінка Azure порталу

Id	Email	Message	EmailSent
1	adminMykyta@admin.com	 Cart Email Request Total 129.98000000000002 Pan...	2024-05-29 17:50:53.6458030
2	nikitkachuk6@gmail.com	User Registration Successful. Email : test@test.com	2024-05-29 18:18:22.3631194
3	nikitkachuk6@gmail.com	User Registration Successful. Email : sonya@gmail.com	2024-05-31 17:38:49.6328469
4	1003 nikitkachuk6@gmail.com	New Order Placed. Order ID : 1002	2024-06-01 14:51:52.2278973
5	1004 nikitkachuk6@gmail.com	New Order Placed. Order ID : 1003	2024-06-01 14:51:52.8150593
6	1005 nikitkachuk6@gmail.com	New Order Placed. Order ID : 1004	2024-06-02 13:40:32.8263011
7	1006 nikitkachuk6@gmail.com	New Order Placed. Order ID : 1005	2024-06-02 13:53:37.1275921
8	1007 nikitkachuk6@gmail.com	User Registration Successful. Email : zheka@shytov.com	2024-06-02 22:54:41.4173244
9	1008 nikitkachuk6@gmail.com	New Order Placed. Order ID : 1006	2024-06-02 22:56:22.8427111
10	adminMykyta@admin.com	 Cart Email Request Total 48.99 Paneer Tikka x 1 ...	2024-06-02 22:58:15.0639013
11	1010 adminMykyta@admin.com	 Cart Email Request Total 63.990000000000001 Pane...	2024-06-03 12:31:30.0056386
12	1011 nikitkachuk6@gmail.com	New Order Placed. Order ID : 1007	2024-06-03 12:38:50.6744632
13	1012 nikitkachuk6@gmail.com	User Registration Successful. Email : oleg@oleg.com	2024-06-03 13:02:37.5045549
14	1013 nikitkachuk6@gmail.com	New Order Placed. Order ID : 1008	2024-06-03 13:54:49.6475712
15	1014 nikitkachuk6@gmail.com	New Order Placed. Order ID : 1009	2024-06-04 18:17:23.4098274
16	1015 nikitkachuk6@gmail.com	User Registration Successful. Email : newCustomer@gmail.com	2024-06-04 22:54:28.1515524
17	1016 nikitkachuk6@gmail.com	New Order Placed. Order ID : 1011	2024-06-04 22:55:57.3694347

Рисунок 3.29 – Логування дій користувача у базі даних MSSQL

3.4 Тестування платформи

3.4.1 Мета тестування

Метою тестування є наступне:

- Перевірка правильності роботи програмного забезпечення відповідно до функціональних вимог;
- Перевірка збереження даних;
- Перевірка сумісності застосунку з різними веб-браузерами;
- Знаходження проблем, помилок і недоліків з метою їх усунення;
- Перевірка зручності графічного інтерфейсу.

3.4.2 Методи тестування

Для перевірки програмного забезпечення застосовуються різні методи:

1. **Статичне тестування** – метод, що включає перевірку вихідного коду програми та супровідної документації без виконання коду. Аналізується відповідність стандартам програмування, правильність синтаксису та семантики.
2. **Динамічне тестування** – метод, який виконується під час роботи програми. Програмне забезпечення тестується на основі заздалегідь визначеного набору тестів. Під час виконання цих тестів збираються дані про виявлені проблеми та помилки для подальшого аналізу.
3. **Функціональне тестування** – перевірка того, чи відповідає фактична поведінка програмного забезпечення очікуванням та специфікаціям, зазначеним у технічному завданні.
4. **Системне тестування** – комплексне тестування всього програмного забезпечення в цілому, що дозволяє оцінити його роботу в інтегрованому середовищі.
5. **Мануальне тестування** – метод, що виконується вручну без використання автоматизованих інструментів. Тестувальник самостійно розробляє і виконує тест-кейси для виявлення помилок, які можуть бути пропущені автоматизованими тестами.
6. **Тестування «білої скриньки»** – метод, що фокусується на внутрішній структурі програми. Перевіряється правильність реалізації окремих компонентів програми та їх взаємодії.

3.4.3 Засоби та порядок тестування

Тестування виконується як мануально, так і автоматизовано для виявлення помилок та недоліків у функціональній частині програмного забезпечення та зручності користування. Для перевірки працездатності та відмовостійкості застосунку необхідно провести такі види тестування:

		Ткачук М.А			ІПЗ.КР.Б-121-24-ІПЗ	Арк.
		Мінтій І.С				
Змн.	Арк.	№ докум.	Підпис	Дата		65

1. **Динамічне тестування на відповідність функціональним вимогам** – перевірка того, що всі функціональні вимоги, зазначені в технічному завданні, виконуються коректно.

2. **Тестування на різних веб-браузерах** – перевірка сумісності веб-застосунку з різними браузерами, такими як Google Chrome, Mozilla Firefox, Microsoft Edge, щоб забезпечити коректну роботу на кожному з них.

3. **Тестування на різних пристроях** – перевірка роботи застосунку на різних пристроях, включаючи настільні комп'ютери, ноутбуки, планшети та мобільні телефони, для забезпечення адаптивності та коректного відображення на різних екранах.

4. **Тестування повідомлень про помилки** – перевірка коректного виведення повідомлень про помилки у випадку їх виникнення, щоб користувачі могли зрозуміти причину проблеми.

5. **Тестування працездатності програми при відсутності з'єднання з мережею** – перевірка поведінки застосунку в умовах відсутності інтернет-з'єднання, де це можливо, щоб переконатися у його стабільності.

6. **Тестування інтерфейсу користувача** – перевірка зручності та інтуїтивності використання інтерфейсу, щоб забезпечити позитивний досвід користувача.

7. **Тестування зручності використання** – оцінка легкості використання застосунку, щоб переконатися, що він є інтуїтивно зрозумілим і простим у використанні для кінцевих користувачів.

3.4.4 Мануальне тестування

Мануальне тестування – це процес перевірки програмного продукту вручну, без використання автоматизованих інструментів або скриптів. Програмне забезпечення пройшло ручне тестування, і подробиці цих перевірок можна знайти у відповідних таблицях.

		Ткачук М.А			ІПЗ.КР.Б-121-24-ІПЗ	Арк.
		Мінтій І.С				
Змн.	Арк.	№ докум.	Підпис	Дата		66

Таблиця 3.1 – Тест 1

Тест «Логін користувача за допомогою пошти»	
Модуль	Аунтетифікація
Початковий стан системи	Користувач відкрив веб-застосунок
Вхідні данні	Електронна пошта та пароль
Опис проведення тесту	Користувач вводить свою електронну пошту та пароль і натискає кнопку «Увійти»
Очікуваний результат	Користувач перенаправляється на головну сторінку
Фактичний результат	Користувач перенаправляється на головну сторінку

Таблиця 3.2 – Тест 2

Тест «Створення нового продукту»	
Модуль	Управління продуктами
Початковий стан системи	Аутентифікований користувач
Вхідні данні	Назва, опис, ціна, категорія продукту
Опис проведення тесту	Користувач вводить необхідні дані про продукт і натискає кнопку «Додати новий продукт»
Очікуваний результат	Продукт додається до списку продуктів
Фактичний результат	Продукт додається до списку продуктів

Таблиця 3.3 – Тест 3

Тест «Оновлення інформації про продукт»	
Модуль	Управління продуктами
Початковий стан системи	Аутентифікований користувач
Вхідні данні	Назва, опис, ціна, категорія продукту
Опис проведення тесту	Користувач редагує дані про продукт і натискає кнопку «Оновити»
Очікуваний результат	Зміни зберігаються
Фактичний результат	Зміни зберігаються

Таблиця 3.4 – Тест 4

Тест «Видалення продукту»	
Модуль	Управління продуктами
Початковий стан системи	Аутентифікований користувач
Вхідні данні	Ідентифікатор продукту
Опис проведення тесту	Користувач вибирає продукт для видалення і натискає кнопку «Видалити»
Очікуваний результат	Продукт видаляється зі списку
Фактичний результат	Продукт видаляється зі списку

Таблиця 3.5 – Тест 5

Тест «Перегляд статистики продажів»	
Модуль	Статистика
Початковий стан системи	Аутентифікований користувач
Вхідні данні	Дані про продажі
Опис проведення тесту	Користувач переходить на сторінку статистики
Очікуваний результат	Відображається статистика продажів у вигляді графіків
Фактичний результат	Відображається статистика продажів у вигляді графіків

Таблиця 3.6 – Тест 6

Тест «Зміна пароля користувача»	
Модуль	Профіль користувача
Початковий стан системи	Аутентифікований користувач
Вхідні данні	Старий пароль, новий пароль
Опис проведення тесту	Користувач вводить старий і новий паролі та натискає кнопку «Змінити пароль»
Очікуваний результат	Пароль успішно змінено
Фактичний результат	Пароль успішно змінено

Таблиця 3.7 – Тест 7

Тест «Зміна теми оформлення додатку»	
Модуль	Налаштування
Початковий стан системи	Аутентифікований користувач на сторінці налаштувань
Вхідні данні	Вхідних даних немає
Опис проведення тесту	Користувач натискає на кнопку перемикання теми
Очікуваний результат	Тема додатку змінюється на протилежну
Фактичний результат	Тема додатку змінюється на протилежну

Таблиця 3.8 – Тест 8

Тест «Генерація звіту»	
Модуль	Звіти
Початковий стан системи	Аутентифікований користувач
Вхідні данні	Період часу, категорія продуктів
Опис проведення тесту	Користувач обирає період та категорію, натискає кнопку «Генерувати звіт»
Очікуваний результат	Відображається звіт за обраний період та категорію
Фактичний результат	Відображається звіт за обраний період та категорію

Таблиця 3.9 – Тест 9

Тест «Додавання нового користувача»	
Модуль	Управління користувачами
Початковий стан системи	Аутентифікований адміністратор
Вхідні данні	Ім'я, прізвище, email, пароль
Опис проведення тесту	Адміністратор вводить дані нового користувача і натискає кнопку «Додати користувача»
Очікуваний результат	Користувача додано до системи
Фактичний результат	Користувача додано до системи

Таблиця 3.10 – Тест 10

Тест «Видалення користувача»	
Модуль	Управління користувачами
Початковий стан системи	Аутентифікований адміністратор
Вхідні данні	Ідентифікатор користувача
Опис проведення тесту	Адміністратор вибирає користувача для видалення і натискає кнопку «Видалити»
Очікуваний результат	Користувача видалено з системи
Фактичний результат	Користувача видалено з системи

Таблиця 3.11 – Тест 11

Тест «Редагування інформації про користувача»	
Модуль	Управління користувачами
Початковий стан системи	Аутентифікований адміністратор
Вхідні данні	Ім'я, прізвище, email
Опис проведення тесту	Адміністратор редагує дані про користувача і натискає кнопку «Зберегти»
Очікуваний результат	Зміни збережено
Фактичний результат	Зміни збережено

Таблиця 3.12 – Тест 12

Тест «Відновлення пароля користувача»	
Модуль	Управління користувачами
Початковий стан системи	Аутентифікований адміністратор
Вхідні данні	Email користувача
Опис проведення тесту	Адміністратор вводить email користувача і натискає кнопку «Відновити пароль»
Очікуваний результат	Лист з інструкціями по відновленню пароля відправлено
Фактичний результат	Лист з інструкціями по відновленню пароля відправлено

Висновки до третього розділу.

У даному розділі була наведена детальна схема тестування платформи. Описані різні методи тестування, такі як статичне, динамічне, функціональне, системне, мануальне та тестування "білої скриньки". Кожен метод був розглянутий з точки зору його ролі в загальному процесі забезпечення якості програмного забезпечення.

Далі було надано опис засобів та порядку тестування, які включають динамічне тестування на відповідність функціональним вимогам, тестування на різних веб-браузерах, тестування на різних пристроях, перевірка повідомлень про помилки, працездатність програми при відсутності мережі, а також тестування інтерфейсу та зручності використання.

Особлива увага була приділена мануальному тестуванню, яке включало детальний опис конкретних тестів, що були виконані вручну. Було представлено 12 тестів, кожен з яких мав свою мету, вхідні дані, очікуваний та фактичний результати. Це включало такі тести, як логін користувача, створення, оновлення та видалення продуктів, перегляд статистики продажів, зміна пароля, зміна теми оформлення, генерація звітів, а також управління користувачами.

Мануальне тестування дозволило виявити проблеми, які можуть бути пропущені автоматизованими тестами, забезпечити функціональність, зручність та відповідність інтуїтивним вимогам користувачів. Це також дозволило отримати об'єктивну оцінку якості розробленого програмного продукту, виявити потенційні проблеми та недоліки, а також покращити ефективність процесу розробки та підтримки програмного забезпечення.

Процеси тестування та аналізу якості, описані в цьому розділі, забезпечували високу якість платформи та її надійність в експлуатації. Виконання цих процесів гарантувало, що платформа відповідає всім встановленим вимогам та забезпечує позитивний користувацький досвід.

		Ткачук М.А			ІПЗ.КР.Б-121-24-ІПЗ	Арк.
		Мінтій І.С				
Змн.	Арк.	№ докум.	Підпис	Дата		71

ВИСНОВКИ

У ході виконання кваліфікаційної роботи бакалавра було спроектовано та реалізовано інтернет-магазин з використанням мікросервісної архітектури. Проведено детальний аналіз предметної області та визначено ключові вимоги до інтернет-магазину. На основі цього аналізу було розроблено архітектуру, підходи та основні технології для створення веб-застосунку.

Процес проектування включав аналіз варіантів використання програмного продукту та моделювання взаємодії між його компонентами. Спроектовано базу даних для зберігання інформації про користувачів, продукти, замовлення та категорії.

Було описано алгоритм роботи з додатком та послідовність його використання. Розглянуто основний функціонал: реєстрація та авторизація користувачів, додавання продуктів до кошика, оформлення замовлень, застосування купонів та обробка платежів.

Представлено схему розгортання інтернет-магазину, описано взаємодію структурних елементів системи та інтерфейс для різних типів користувачів. Описано роботу адміністративної панелі для управління замовленнями та зміни їх статусів.

Архітектура платформи побудована на базі мікросервісів, що забезпечує масштабованість та гнучкість системи. Використання сучасних технологій гарантує надійність та безпеку інтернет-магазину.

Додаток має інтуїтивно зрозумілий інтерфейс, який дозволяє користувачам легко переглядати продукти, додавати їх до кошика, оформляти замовлення та використовувати купони. Користувачі можуть реєструватися, входити до системи, відстежувати свої замовлення та отримувати сповіщення про їхні статуси.

Інтернет-магазин з використанням мікросервісів має значний потенціал для подальшого розвитку, включаючи розширення функціональності, покращення інтерфейсу користувача та інтеграцію з іншими сервісами та платформами.

		Ткачук М.А			ІПЗ.КР.Б-121-24-ІПЗ	Арк.
		Мінтій І.С				
Змн.	Арк.	№ докум.	Підпис	Дата		72

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Microsoft .NET [Електронний ресурс] – Режим доступу до ресурсу: <https://dotnet.microsoft.com/>
2. Entity Framework Core [Електронний ресурс] – Режим доступу до ресурсу: <https://docs.microsoft.com/en-us/ef/core/>
3. ASP.NET Core Documentation [Електронний ресурс] – Режим доступу до ресурсу: <https://docs.microsoft.com/en-us/aspnet/core/>
4. Azure Service Bus [Електронний ресурс] – Режим доступу до ресурсу: <https://azure.microsoft.com/en-us/services/service-bus/>
5. Stripe API Documentation [Електронний ресурс] – Режим доступу до ресурсу: <https://stripe.com/docs/api>
6. Ocelot API Gateway [Електронний ресурс] – Режим доступу до ресурсу: <https://ocelot.readthedocs.io/>
7. JWT.io – JSON Web Tokens [Електронний ресурс] – Режим доступу до ресурсу: <https://jwt.io/>
8. Microsoft SQL Server [Електронний ресурс] – Режим доступу до ресурсу: <https://www.microsoft.com/en-us/sql-server/sql-server-downloads>
9. DataTables | Javascript table library [Електронний ресурс] – Режим доступу до ресурсу: <https://datatables.net/>
10. Microsoft Azure [Електронний ресурс] – Режим доступу до ресурсу: <https://portal.azure.com/>
11. Bootswatch: Free themes for Bootstrap [Електронний ресурс] – Режим доступу до ресурсу: <https://bootswatch.com/>
12. Toastr by CodeSeven [Електронний ресурс] – Режим доступу до ресурсу: <https://codeseven.github.io/toastr/>
13. DevOps Documentation [Електронний ресурс] – Режим доступу до ресурсу: <https://azure.microsoft.com/en-us/services/devops/>
14. Розробка на ASP.NET Core (Українська) [Електронний ресурс] – Режим доступу до ресурсу: <https://docs.microsoft.com/uk-ua/aspnet/core/>

15. Dev.ua – Платформа для українських розробників [Електронний ресурс]
– Режим доступу до ресурсу: <https://dev.ua/>
16. DOU – Спільнота українських IT-фахівців [Електронний ресурс] –
Режим доступу до ресурсу: <https://dou.ua/>
17. Документація по .NET Core (Українська) [Електронний ресурс] – Режим
доступу до ресурсу: <https://docs.microsoft.com/uk-ua/dotnet/core/>
18. Інтернет-магазин кави в Україні Екава [Електронний ресурс] – Режим
доступу до ресурсу: <https://ekava.com.ua/ua>
19. Coffeum — інтернет-магазин [Електронний ресурс] – Режим доступу до
ресурсу: <https://coffeum.ua/>
20. Інтернет-магазин кави, чаю та кавомашин - coffeeok [Електронний
ресурс] – Режим доступу до ресурсу: <https://coffeeok.com.ua/ua/>
21. Інтернет-магазин кави, чаю та кавомашин Хкава [Електронний ресурс]
– Режим доступу до ресурсу: <https://xkava.com.ua/>
22. Інтернет-магазин coffe.ua – все для кави [Електронний ресурс] – Режим
доступу до ресурсу: <https://coffe.ua/>
23. База знань - QALight [Електронний ресурс] – Режим доступу до ресурсу:
<https://qalight.ua/baza-znan/>
24. Udemу [Електронний ресурс] – Режим доступу до ресурсу:
<https://www.udemy.com/>
25. .NET Archives - Learn Programming with Real Apps [Електронний ресурс]
– Режим доступу до ресурсу:
<https://learningprogramming.net/category/net/>
26. C# cheat sheet [Електронний ресурс] – Режим доступу до ресурсу:
<https://www.programming-idioms.org/cheatsheet/Csharp>
27. JS Comparison Table [Електронний ресурс] – Режим доступу до ресурсу:
<https://dorey.github.io/JavaScript-Equality-Table/>
28. W3Schools Online Web Tutorials [Електронний ресурс] – Режим доступу
до ресурсу: <https://www.w3schools.com>

		Ткачук М.А			ІПЗ.КР.Б-121-24-ІПЗ	Арк.
		Мінтій І.С				
Змн.	Арк.	№ докум.	Підпис	Дата		74

ДОДАТКИ

		Ткачук М.А			ІПЗ.КР.Б-121-24-ІПЗ	Арк.
		Мінтій І.С				
Змн.	Арк.	№ докум.	Підпис	Дата		75

CoffeeShop.Web

```
public class AuthController : Controller
{
    private readonly IAuthService _authService;
    private readonly ITokenProvider _tokenProvider;

    public AuthController(IAuthService authService, ITokenProvider tokenProvider)
    {
        _authService = authService;
        _tokenProvider = tokenProvider;
    }

    [HttpGet]
    public IActionResult Login()
    {
        LoginRequestDto loginRequestDto = new();
        return View(loginRequestDto);
    }

    [HttpPost]
    public async Task<IActionResult> Login(LoginRequestDto model)
    {
        ResponseDto response = await _authService.LoginAsync(model);

        if (response != null && response.IsSuccess)
        {
            LoginResponseDto loginResponseDto = JsonConvert.DeserializeObject<LoginResponseDto>(Convert.ToString(response.Result));

            await SignInUser(loginResponseDto);
            _tokenProvider.SetToken(loginResponseDto.Token);

            return RedirectToAction("Index", "Home");
        }
        else
        {
            TempData["error"] = response.Message;
            return View(model);
        }
    }

    [HttpGet]
    public IActionResult Register()
    {
        var roleList = new List<SelectListItem>()
        {
            new() {Text= SD.RoleAdmin, Value= SD.RoleAdmin},
            new() {Text= SD.RoleCustomer, Value= SD.RoleCustomer}
        };

        ViewBag.RoleList = roleList;
        return View();
    }

    [HttpPost]
    public async Task<IActionResult> Register(RegistrationRequestDto model)
    {
        ResponseDto result = await _authService.RegisterAsync(model);
        ResponseDto assignRole;

        if (result != null && result.IsSuccess)
        {
            if (string.IsNullOrEmpty(model.Role))
            {
                model.Role = SD.RoleCustomer;
            }
            assignRole = await _authService.AssignRoleAsync(model);
            if (assignRole != null && assignRole.IsSuccess) {
                TempData["success"] = "Реєстрація успішна";
                return RedirectToAction(nameof(Login));
            }
        }
    }
}
```

		Ткачук М.А			ІПЗ.КР.Б-121-24-ІПЗ	Арк.
		Мінтій І.С				
Змн.	Арк.	№ докум.	Підпис	Дата		76

```

    }
}
else
{
    TempData["error"] = result.Message;

}

var roleList = new List<SelectListItem>()
{
    new() {Text= SD.RoleAdmin, Value= SD.RoleAdmin},
    new() {Text= SD.RoleCustomer, Value= SD.RoleCustomer}
};

ViewBag.RoleList = roleList;
return View(model);
}

public async Task<IActionResult> Logout()
{
    await HttpContext.SignOutAsync();
    _tokenProvider.ClearToken();
    return RedirectToAction("Index", "Home");
}

private async Task SignInUser(LoginResponseDto login)
{
    var handler = new JwtSecurityTokenHandler();

    var jwt = handler.ReadJwtToken(login.Token);

    var identity = new ClaimsIdentity(CookieAuthenticationDefaults.AuthenticationScheme);
    identity.AddClaim(new Claim(JwtRegisteredClaimNames.Email,
        jwt.Claims.FirstOrDefault(u => u.Type == JwtRegisteredClaimNames.Email).Value));

    identity.AddClaim(new Claim(JwtRegisteredClaimNames.Sub,
        jwt.Claims.FirstOrDefault(u => u.Type == JwtRegisteredClaimNames.Sub).Value));

    identity.AddClaim(new Claim(JwtRegisteredClaimNames.Name,
        jwt.Claims.FirstOrDefault(u => u.Type == JwtRegisteredClaimNames.Name).Value));

    identity.AddClaim(new Claim(ClaimTypes.Name,
        jwt.Claims.FirstOrDefault(u => u.Type == JwtRegisteredClaimNames.Email).Value));
    identity.AddClaim(new Claim(ClaimTypes.Role,
        jwt.Claims.FirstOrDefault(u => u.Type == "role").Value));

    var principal = new ClaimsPrincipal(identity);
    await HttpContext.SignInAsync(CookieAuthenticationDefaults.AuthenticationScheme, principal);
}
}

public class CartController : Controller
{
    private readonly ICartService _cartService;
    private readonly IOrderService _orderService;
    public CartController(ICartService cartService, IOrderService orderService)
    {
        _cartService = cartService;
        _orderService = orderService;
    }

    [Authorize]
    public async Task<IActionResult> CartIndex()
    {
        return View(await LoadCartDtoBasedOnLoggedInUser());
    }

    [Authorize]
    public async Task<IActionResult> Checkout()
    {
        return View(await LoadCartDtoBasedOnLoggedInUser());
    }
}

```

		Ткачук М.А			ІПЗ.КР.Б-121-24-ІІЗ	Арк.
		Мінтій І.С				
Змн.	Арк.	№ докум.	Підпис	Дата		77

```

    }

    [HttpPost]
    [ActionName("Checkout")]
    public async Task<IActionResult> Checkout(CartDto cartDto)
    {

        CartDto cart = await LoadCartDtoBasedOnLoggedInUser();
        cart.CartHeader.Phone = cartDto.CartHeader.Phone;
        cart.CartHeader.Email = cartDto.CartHeader.Email;
        cart.CartHeader.Name = cartDto.CartHeader.Name;

        var response = await _orderService.CreateOrder(cart);
        OrderHeaderDto orderHeaderDto = JsonConvert.DeserializeObject<OrderHeaderDto>(Convert.ToString(response.Result));

        if (response != null && response.IsSuccess)
        {
            // get stripe session and redirect to stripe to place order
            var domain = Request.Scheme + "://" + Request.Host.Value + "/";

            StripeRequestDto stripeRequestDto = new()
            {
                ApprovedUrl = domain + "cart/Confirmation?orderId=" + orderHeaderDto.OrderHeaderId,
                CancelUrl = domain + "cart/checkout",
                OrderHeader = orderHeaderDto
            };

            var stripeResponse = await _orderService.CreateStripeSession(stripeRequestDto);
            StripeRequestDto stripeResponseResult = JsonConvert.DeserializeObject<StripeRequestDto>(Convert.ToString(stripeResponse.Result));
            Response.Headers.Add("Location", stripeResponseResult.StripeSessionUrl);

            return new StatusCodeResult(303);

        }
        return View();
    }

    [Authorize]
    public async Task<IActionResult> Confirmation(int orderId)
    {

        ResponseDto? response = await _orderService.ValidateStripeSession(orderId);
        if (response != null & response.IsSuccess)
        {
            OrderHeaderDto orderHeader = JsonConvert.DeserializeObject<OrderHeaderDto>(Convert.ToString(response.Result));
            if (orderHeader.Status == SD.Status_Approved)
            {
                return View(orderId);
            }
            //return RedirectToAction(nameof(CartIndex));
        }
        //redirect to error page
        return View(orderId);
    }

    public async Task<IActionResult> Remove(int cartDetailsId)
    {
        var userId = User.Claims.Where(u => u.Type == JwtRegisteredClaimNames.Sub)?.FirstOrDefault()?.Value;
        ResponseDto? response = await _cartService.RemoveFromCartAsync(cartDetailsId);
        if (response != null & response.IsSuccess)
        {
            TempData["success"] = "Кошик успішно оновлено";
            return RedirectToAction(nameof(CartIndex));
        }
        return View();
    }

    [HttpPost]
    public async Task<IActionResult> ApplyCoupon(CartDto cartDto)

```

		Ткачук М.А			ІІЗ.КР.Б-121-24-ІІЗ	Арк.
		Мінтій І.С				78
Змн.	Арк.	№ докум.	Підпис	Дата		

```

{
    ResponseDto? response = await _cartService.ApplyCouponAsync(cartDto);
    if (response != null & response.IsSuccess)
    {
        TempData["success"] = "Кошик успішно оновлено";
        return RedirectToAction(nameof(CartIndex));
    }
    return View();
}

[HttpPost]
public async Task<IActionResult> EmailCart(CartDto cartDto)
{
    CartDto cart = await LoadCartDtoBasedOnLoggedInUser();
    cart.CartHeader.Email = User.Claims.Where(u => u.Type == JwtRegisteredClaimNames.Email)?.FirstOrDefault()?.Value;
    ResponseDto? response = await _cartService.EmailCart(cart);
    if (response != null & response.IsSuccess)
    {
        TempData["success"] = "Електронна пошта буде оброблена та надіслана найближчим часом.";
        return RedirectToAction(nameof(CartIndex));
    }
    return View();
}

[HttpPost]
public async Task<IActionResult> RemoveCoupon(CartDto cartDto)
{
    cartDto.CartHeader.CouponCode = "";
    ResponseDto? response = await _cartService.ApplyCouponAsync(cartDto);
    if (response != null & response.IsSuccess)
    {
        TempData["success"] = "Кошик успішно оновлено";
        return RedirectToAction(nameof(CartIndex));
    }
    return View();
}

private async Task<CartDto> LoadCartDtoBasedOnLoggedInUser()
{
    var userId = User.Claims.Where(u => u.Type == JwtRegisteredClaimNames.Sub)?.FirstOrDefault()?.Value;
    ResponseDto? response = await _cartService.GetCartByUserIdAsync(userId);
    if (response != null & response.IsSuccess)
    {
        CartDto cartDto = JsonConvert.DeserializeObject<CartDto>(Convert.ToString(response.Result));
        return cartDto;
    }
    return new CartDto();
}

}

public class CouponController : Controller
{
    private readonly ICouponService _couponService;

    public CouponController(ICouponService couponService)
    {
        _couponService = couponService;
    }

    public async Task<IActionResult> CouponIndex()
    {
        List<CouponDto>? list = new();

        ResponseDto? response = await _couponService.GetAllCouponsAsync();

        if (response != null && response.IsSuccess)
        {
            list = JsonConvert.DeserializeObject<List<CouponDto>>(Convert.ToString(response.Result));

```

		Ткачук М.А			ІПЗ.КР.Б-121-24-ІПЗ	Арк.
		Мінтій І.С				
Змн.	Арк.	№ докум.	Підпис	Дата		79

```

    }
    else
    {
        TempData["error"] = response?.Message;
    }

    return View(list);
}

public async Task<IActionResult> CouponCreate()
{
    return View();
}

[HttpPost]
public async Task<IActionResult> CouponCreate(CouponDto couponDto)
{
    if (ModelState.IsValid)
    {
        ResponseDto? response = await _couponService.CreateCouponAsync(couponDto);

        if (response != null && response.IsSuccess)
        {
            TempData["success"] = "Купон успішно створено";

            return RedirectToAction(nameof(CouponIndex));
        }
        else
        {
            TempData["error"] = response?.Message;
        }
    }

    return View(couponDto);
}

public async Task<IActionResult> CouponDelete(int couponId)
{
    ResponseDto? response = await _couponService.GetCouponByIdAsync(couponId);

    if (response != null && response.IsSuccess)
    {
        CouponDto? model = JsonConvert.DeserializeObject<CouponDto>(Convert.ToString(response.Result));
        return View(model);
    }
    else
    {
        TempData["error"] = response?.Message;
    }
    return NotFound();
}

[HttpPost]
public async Task<IActionResult> CouponDelete(CouponDto model)
{
    ResponseDto? response = await _couponService.DeleteCouponAsync(model.CouponId);

    if (response != null && response.IsSuccess)
    {
        TempData["success"] = "Купон успішно видалено";

        return RedirectToAction(nameof(CouponIndex));
    }
    else
    {
        TempData["error"] = response?.Message;
    }
}

```

		Ткачук М.А			ІІЗ.КР.Б-121-24-ІІЗ	Арк.
		Мінтій І.С				
Змн.	Арк.	№ докум.	Підпис	Дата		80

```

        return View(model);
    }
}
public class HomeController : Controller
{
    private readonly IProductService _productService;
    private readonly ICartService _cartService;
    public HomeController(IProductService productService, ICartService cartService)
    {
        _productService = productService;
        _cartService = cartService;
    }

    public async Task<IActionResult> Index()
    {
        List<ProductDto>? list = new();

        ResponseDto? response = await _productService.GetAllProductsAsync();

        if (response != null && response.IsSuccess)
        {
            list = JsonConvert.DeserializeObject<List<ProductDto>>(Convert.ToString(response.Result));
        }
        else
        {
            TempData["error"] = response?.Message;
        }

        return View(list);
    }

    [Authorize]
    public async Task<IActionResult> ProductDetails(int productId)
    {
        ProductDto model = new();

        ResponseDto? response = await _productService.GetProductByIdAsync(productId);

        if (response != null && response.IsSuccess)
        {
            model = JsonConvert.DeserializeObject<ProductDto>(Convert.ToString(response.Result));
        }
        else
        {
            TempData["error"] = response?.Message;
        }

        return View(model);
    }

    [Authorize]
    [HttpPost]
    [ActionName("ProductDetails")]
    public async Task<IActionResult> ProductDetails(ProductDto productDto)
    {
        CartDto cartDto = new CartDto()
        {
            CartHeader = new CartHeaderDto
            {
                UserId = User.Claims.Where(u => u.Type == JwtClaimTypes.Subject)?.FirstOrDefault()?.Value
            }
        };

        CartDetailsDto cartDetails = new CartDetailsDto()
        {
            Count = productDto.Count,
            ProductId = productDto.ProductId,
        };
    }
}

```

		Ткачук М.А			ІІЗ.КР.Б-121-24-ІІЗ	Арк.
		Мінтій І.С				81
Змн.	Арк.	№ докум.	Підпис	Дата		

```

        List<CartDetailsDto> cartDetailsDtos = new() { cartDetails };
        cartDto.CartDetails = cartDetailsDtos;

        ResponseDto? response = await _cartService.UpsertCartAsync(cartDto);

        if (response != null && response.IsSuccess)
        {
            TempData["success"] = "Товар додано до кошика";
            return RedirectToAction(nameof(Index));
        }
        else
        {
            TempData["error"] = response?.Message;
        }

        return View(cartDto);
    }

    public IActionResult Privacy()
    {
        return View();
    }

    [ResponseCache(Duration = 0, Location = ResponseCacheLocation.None, NoStore = true)]
    public IActionResult Error()
    {
        return View(new ErrorViewModel { RequestId = Activity.Current?.Id ?? HttpContext.TraceIdentifier });
    }
}

public class OrderController : Controller
{
    private readonly IOrderService _orderService;

    public OrderController(IOrderService orderService)
    {
        _orderService = orderService;
    }

    [Authorize]
    public IActionResult OrderIndex()
    {
        return View();
    }

    public async Task<ActionResult> OrderDetail(int orderId)
    {
        OrderHeaderDto orderHeaderDto = new OrderHeaderDto();
        string userId = User.Claims.Where(u => u.Type == JwtRegisteredClaimNames.Sub)?.FirstOrDefault()?.Value;

        var response = await _orderService.GetOrder(orderId);

        if (response != null && response.IsSuccess)
        {
            orderHeaderDto = JsonConvert.DeserializeObject<OrderHeaderDto>(Convert.ToString(response.Result));
        }
        if(!User.IsInRole(SD.RoleAdmin) && userId != orderHeaderDto.UserId)
        {
            return NotFound();
        }
        return View(orderHeaderDto);
    }

    [HttpPost("OrderReadyForPickup")]
    public async Task<ActionResult> OrderReadyForPickup (int orderId)
    {
        var response = await _orderService.UpdateOrderStatus(orderId, SD.Status_ReadyForPickup);

        if (response != null && response.IsSuccess)
    }

```

		Ткачук М.А			ІІЗ.КР.Б-121-24-ІІЗ	Арк.
		Мінтій І.С				
Змн.	Арк.	№ докум.	Підпис	Дата		82

```

    {
        TempData["success"] = "Статус успішно оновлено";
        return RedirectToAction(nameof(OrderDetail), new { orderId = orderId });
    }
    return View();
}

[HttpPost("CompleteOrder")]
public async Task<IActionResult> CompleteOrder(int orderId)
{
    var response = await _orderService.UpdateOrderStatus(orderId, SD.Status_Compteted);

    if (response != null && response.IsSuccess)
    {
        TempData["success"] = "Статус успішно оновлено";
        return RedirectToAction(nameof(OrderDetail), new { orderId = orderId });
    }
    return View();
}

[HttpPost("CancelOrder")]
public async Task<IActionResult> CancelOrder(int orderId)
{
    var response = await _orderService.UpdateOrderStatus(orderId, SD.Status_Cancelled);

    if (response != null && response.IsSuccess)
    {
        TempData["success"] = "Статус успішно оновлено";
        return RedirectToAction(nameof(OrderDetail), new { orderId = orderId });
    }
    return View();
}

[HttpGet]
public IActionResult GetAll(string status)
{
    IEnumerable<OrderHeaderDto> list;
    string userId = "";
    if (!User.IsInRole(SD.RoleAdmin))
    {
        userId = User.Claims.Where(u => u.Type == JwtRegisteredClaimNames.Sub)?.FirstOrDefault()?.Value;
    }
    ResponseDto response = _orderService.GetAllOrder(userId).GetAwaiter().GetResult();

    if(response != null && response.IsSuccess)
    {
        list = JsonConvert.DeserializeObject<List<OrderHeaderDto>>(Convert.ToString(response.Result));
        switch (status)
        {
            case "approved":
                list = list.Where(u=>u.Status == SD.Status_Approved);
                break;
            case "readyforpickup":
                list = list.Where(u => u.Status == SD.Status_ReadyForPickup);
                break;
            case "cancelled":
                list = list.Where(u => u.Status == SD.Status_Cancelled);
                break;
            default:
                break;
        }
    }
    else
    {
        list = new List<OrderHeaderDto>();
    }
    return Json(new {data = list});
}

```

		Ткачук М.А			ІПЗ.КР.Б-121-24-ІПЗ	Арк.
		Мінтій І.С				
Змн.	Арк.	№ докум.	Підпис	Дата		83

```

    }
}
public class ProductController : Controller
{
    private readonly IProductService _productService;

    public ProductController(IProductService productService)
    {
        _productService = productService;
    }

    public async Task<IActionResult> ProductIndex()
    {
        List<ProductDto>? list = new();

        ResponseDto? response = await _productService.GetAllProductsAsync();

        if (response != null && response.IsSuccess)
        {
            list = JsonConvert.DeserializeObject<List<ProductDto>>(Convert.ToString(response.Result));
        }
        else
        {
            TempData["error"] = response?.Message;
        }

        return View(list);
    }

    public async Task<IActionResult> ProductCreate()
    {
        return View();
    }

    [HttpPost]
    public async Task<IActionResult> ProductCreate(ProductDto productDto)
    {
        if (ModelState.IsValid)
        {
            ResponseDto? response = await _productService.CreateProductAsync(productDto);

            if (response != null && response.IsSuccess)
            {
                TempData["success"] = "Продукт створено";

                return RedirectToAction(nameof(ProductIndex));
            }
            else
            {
                TempData["error"] = response?.Message;
            }
        }

        return View(productDto);
    }

    public async Task<IActionResult> ProductDelete(int productId)
    {
        ResponseDto? response = await _productService.GetProductByIdAsync(productId);

        if (response != null && response.IsSuccess)
        {
            ProductDto? model = JsonConvert.DeserializeObject<ProductDto>(Convert.ToString(response.Result));
            return View(model);
        }
        else
        {
            TempData["error"] = response?.Message;
        }
    }
}

```

		Ткачук М.А			ІІЗ.КР.Б-121-24-ІІЗ	Арк.
		Мінтій І.С				
Змн.	Арк.	№ докум.	Підпис	Дата		84

```

    }
    return NotFound();
}

[HttpPost]
public async Task<IActionResult> ProductDelete(ProductDto model)
{
    ResponseDto? response = await _productService.DeleteProductAsync(model.ProductId);

    if (response != null && response.IsSuccess)
    {
        TempData["success"] = "Продукт видалено";

        return RedirectToAction(nameof(ProductIndex));
    }
    else
    {
        TempData["error"] = response?.Message;
    }
    return View(model);
}

public async Task<IActionResult> ProductEdit(int productId)
{
    ResponseDto? response = await _productService.GetProductByIdAsync(productId);

    if (response != null && response.IsSuccess)
    {
        ProductDto? model = JsonConvert.DeserializeObject<ProductDto>(Convert.ToString(response.Result));
        return View(model);
    }
    else
    {
        TempData["error"] = response?.Message;
    }
    return NotFound();
}

[HttpPost]
public async Task<IActionResult> ProductEdit(ProductDto model)
{
    if (ModelState.IsValid)
    {
        ResponseDto? response = await _productService.UpdateProductAsync(model);

        if (response != null && response.IsSuccess)
        {
            TempData["success"] = "Продукт успішно оновлено";

            return RedirectToAction(nameof(ProductIndex));
        }
        else
        {
            TempData["error"] = response?.Message;
        }
    }
    return View(model);
}
}

public class AuthService : IAuthService
{
    private readonly IBaseService _baseService;

    public AuthService(IBaseService baseService)
    {
        _baseService = baseService;
    }
}

```

		Ткачук М.А			ІПЗ.КР.Б-121-24-ІПЗ	Арк.
		Мінтій І.С				
Змн.	Арк.	№ докум.	Підпис	Дата		85

```

public async Task<ResponseDto> AssignRoleAsync(RegistrationRequestDto registrationRequestDto)
{
    return await _baseService.SendAsync(new RequestDto()
    {
        ApiType = SD.ApiType.POST,
        Data = registrationRequestDto,
        Url = SD.AuthAPIBase + "/api/auth/AssignRole"
    });
}

public async Task<ResponseDto> LoginAsync(LoginRequestDto loginRequestDto)
{
    return await _baseService.SendAsync(new RequestDto()
    {
        ApiType = SD.ApiType.POST,
        Data = loginRequestDto,
        Url = SD.AuthAPIBase + "/api/auth/login"
    }, withBearer: false);
}

public async Task<ResponseDto> RegisterAsync(RegistrationRequestDto registrationRequestDto)
{
    return await _baseService.SendAsync(new RequestDto()
    {
        ApiType = SD.ApiType.POST,
        Data = registrationRequestDto,
        Url = SD.AuthAPIBase + "/api/auth/register"
    }, withBearer: false);
}
}

public class BaseService : IBaseService
{
    private readonly IHttpClientFactory _httpClientFactory;
    private readonly ITokenProvider _tokenProvider;
    public BaseService(IHttpClientFactory httpClientFactory, ITokenProvider tokenProvider)
    {
        _httpClientFactory = httpClientFactory;
        _tokenProvider = tokenProvider;
    }

    public async Task<ResponseDto> SendAsync(RequestDto requestDto, bool withBearer = true)
    {
        try
        {
            HttpClient client = _httpClientFactory.CreateClient("CoffeeShopAPI");
            HttpRequestMessage message = new();

            if(requestDto.ContentType == ContentType.MultipartFormData)
            {
                message.Headers.Add("Accept", "*/*");
            }
            else
            {
                message.Headers.Add("Accept", "application/json");
            }

            //token
            if (withBearer)
            {
                var token = _tokenProvider.GetToken();
                message.Headers.Add("Authorization", $"Bearer {token}");
            }

            message.RequestUri = new Uri(requestDto.Url); //Get request

            if(requestDto.ContentType == ContentType.MultipartFormData)
            {
                var content = new MultipartFormDataContent();
            }
        }
    }
}

```

		Ткачук М.А			ІІЗ.КР.Б-121-24-ІІЗ	Арк.
		Мінтій І.С				
Змн.	Арк.	№ докум.	Підпис	Дата		86

```

foreach(var prop in requestDto.Data.GetType().GetProperties())
{
    var value = prop.GetValue(requestDto.Data);
    if(value is FormFile)
    {
        var file = (FormFile)value;
        if(file != null)
        {
            content.Add(new StreamContent(file.OpenReadStream()), prop.Name, file.FileName);
        }
    }
    else
    {
        content.Add(new StringContent(value == null ? "" : value.ToString()), prop.Name);
    }
}
message.Content = content;
}
else
{
    if (requestDto.Url != null)
    {
        message.Content = new StringContent(JsonConvert.SerializeObject(requestDto.Data), Encoding.UTF8, "application/json");
    }
}

//serialize

}

HttpResponseMessage? apiResponse = null;

switch (requestDto.ApiType)
{
    case ApiType.POST:
        message.Method = HttpMethod.Post;
        break;
    case ApiType.DELETE:
        message.Method = HttpMethod.Delete;
        break;
    case ApiType.PUT:
        message.Method = HttpMethod.Put;
        break;
    default:
        message.Method = HttpMethod.Get;
        break;
}

apiResponse = await client.SendAsync(message);

switch (apiResponse.StatusCode)
{
    case HttpStatusCode.NotFound:
        return new() { IsSuccess = false, Message = "Not Found" };
    case HttpStatusCode.Forbidden:
        return new() { IsSuccess = false, Message = "Access Denied" };
    case HttpStatusCode.Unauthorized:
        return new() { IsSuccess = false, Message = "Unauthorized" };
    case HttpStatusCode.InternalServerError:
        return new() { IsSuccess = false, Message = "Internal Server Error" };
    default:
        var apiContent = await apiResponse.Content.ReadAsStringAsync();
        var apiResponseDto = JsonConvert.DeserializeObject<ResponseDto>(apiContent);
        return apiResponseDto;
}

}

catch (Exception ex)
{
    var dto = new ResponseDto
    {

```

		Ткачук М.А			ІІЗ.КР.Б-121-24-ІІЗ	Арк.
		Мінтій І.С				
Змн.	Арк.	№ докум.	Підпис	Дата		87

```

        Message = ex.Message.ToString(),
        IsSuccess = false
    };
    return dto;
}

}

}

public class CartService : ICartService
{
    private readonly IBaseService _baseService;

    public CartService(IBaseService baseService)
    {
        _baseService = baseService;
    }

    public async Task<ResponseDto> ApplyCouponAsync(CartDto cartDto)
    {
        return await _baseService.SendAsync(new RequestDto()
        {
            ApiType = SD.ApiType.POST,
            Data = cartDto,
            Url = SD.ShoppingCartAPIBase + "/api/cart/ApplyCoupon"
        });
    }

    public async Task<ResponseDto> EmailCart(CartDto cartDto)
    {
        return await _baseService.SendAsync(new RequestDto()
        {
            ApiType = SD.ApiType.POST,
            Data = cartDto,
            Url = SD.ShoppingCartAPIBase + "/api/cart/EmailCartRequest"
        });
    }

    public async Task<ResponseDto> GetCartByUserIdAsync(string userId)
    {
        return await _baseService.SendAsync(new RequestDto()
        {
            ApiType = SD.ApiType.GET,
            Url = SD.ShoppingCartAPIBase + "/api/cart/GetCart/" + userId
        });
    }

    public async Task<ResponseDto> RemoveFromCartAsync(int cartDetailsId)
    {
        return await _baseService.SendAsync(new RequestDto()
        {
            ApiType = SD.ApiType.POST,
            Data = cartDetailsId,
            Url = SD.ShoppingCartAPIBase + "/api/cart/RemoveCart"
        });
    }

    public async Task<ResponseDto> UpsertCartAsync(CartDto cartDto)
    {
        return await _baseService.SendAsync(new RequestDto()
        {
            ApiType = SD.ApiType.POST,
            Data = cartDto,
            Url = SD.ShoppingCartAPIBase + "/api/cart/CartUpsert"
        });
    }
}

```

		Ткачук М.А			ІІЗ.КР.Б-121-24-ІІЗ	Арк.
		Мінтій І.С				88
Змн.	Арк.	№ докум.	Підпис	Дата		

```

}
public class CouponService : ICouponService
{
    private readonly IBaseService _baseService;

    public CouponService(IBaseService baseService)
    {
        _baseService = baseService;
    }

    public async Task<ResponseDto>> CreateCouponAsync(CouponDto couponDto)
    {
        return await _baseService.SendAsync(new RequestDto()
        {
            ApiType = SD.ApiType.POST,
            Data = couponDto,
            Url = SD.CouponAPIBase + "/api/coupon"
        });
    }

    public async Task<ResponseDto> DeleteCouponAsync(int id)
    {
        return await _baseService.SendAsync(new RequestDto()
        {
            ApiType = SD.ApiType.DELETE,
            Url = SD.CouponAPIBase + "/api/coupon/" + id
        });
    }

    public async Task<ResponseDto>> GetAllCouponsAsync()
    {
        return await _baseService.SendAsync(new RequestDto()
        {
            ApiType = SD.ApiType.GET,
            Url = SD.CouponAPIBase + "/api/coupon"
        });
    }

    public async Task<ResponseDto>> GetCouponAsync(string couponCode)
    {
        return await _baseService.SendAsync(new RequestDto()
        {
            ApiType = SD.ApiType.GET,
            Url = SD.CouponAPIBase + "/api/coupon/GetByCode/" + couponCode
        });
    }

    public async Task<ResponseDto>> GetCouponByIdAsync(int id)
    {
        return await _baseService.SendAsync(new RequestDto()
        {
            ApiType = SD.ApiType.GET,
            Url = SD.CouponAPIBase + "/api/coupon/" + id
        });
    }

    public async Task<ResponseDto>> UpdateCouponAsync(CouponDto couponDto)
    {
        return await _baseService.SendAsync(new RequestDto()
        {
            ApiType = SD.ApiType.PUT,
            Data = couponDto,
            Url = SD.CouponAPIBase + "/api/coupon"
        });
    }
}

public class OrderService : IOrderService
{
    private readonly IBaseService _baseService;

```

		Ткачук М.А			ІІЗ.КР.Б-121-24-ІІЗ	Арк.
		Мінтій І.С				
Змн.	Арк.	№ докум.	Підпис	Дата		89

```

public OrderService(IBaseService baseService)
{
    _baseService = baseService;
}

public async Task<ResponseDto> GetAllOrder(string? userId)
{
    return await _baseService.SendAsync(new RequestDto()
    {
        ApiType = SD.ApiType.GET,
        Url = SD.OrderAPIBase + "/api/order/GetOrders/" + userId
    });
}

public async Task<ResponseDto> GetOrder(int orderId)
{
    return await _baseService.SendAsync(new RequestDto()
    {
        ApiType = SD.ApiType.GET,
        Url = SD.OrderAPIBase + "/api/order/GetOrder/" + orderId
    });
}

public async Task<ResponseDto> CreateOrder(CartDto cartDto)
{
    return await _baseService.SendAsync(new RequestDto()
    {
        ApiType = SD.ApiType.POST,
        Data = cartDto,
        Url = SD.OrderAPIBase + "/api/order/CreateOrder"
    });
}

public async Task<ResponseDto> CreateStripeSession(StripeRequestDto stripeRequestDto)
{
    return await _baseService.SendAsync(new RequestDto()
    {
        ApiType = SD.ApiType.POST,
        Data = stripeRequestDto,
        Url = SD.OrderAPIBase + "/api/order/CreateStripeSession"
    });
}

public async Task<ResponseDto> UpdateOrderStatus(int orderId, string newStatus)
{
    return await _baseService.SendAsync(new RequestDto()
    {
        ApiType = SD.ApiType.POST,
        Data = newStatus,
        Url = SD.OrderAPIBase + "/api/order/UpdateOrderStatus/" + orderId
    });
}

public async Task<ResponseDto> ValidateStripeSession(int orderHeaderId)
{
    return await _baseService.SendAsync(new RequestDto()
    {
        ApiType = SD.ApiType.POST,
        Data = orderHeaderId,
        Url = SD.OrderAPIBase + "/api/order/ValidateStripeSession"
    });
}
}

public class ProductService : IProductService
{
    private readonly IBaseService _baseService;

```

		Ткачук М.А			ІПЗ.КР.Б-121-24-ІІЗ	Арк.
		Мінтій І.С				90
Змн.	Арк.	№ докум.	Підпис	Дата		

```

public ProductService(IBaseService baseService)
{
    _baseService = baseService;
}

public async Task<ResponseDto> CreateProductAsync(ProductDto productDto)
{
    return await _baseService.SendAsync(new RequestDto()
    {
        ApiType = SD.ApiType.POST,
        Data = productDto,
        Url = SD.ProductAPIBase + "/api/product",
        ContentType = SD.ContentType.MultipartFormData
    });
}

public async Task<ResponseDto> DeleteProductAsync(int id)
{
    return await _baseService.SendAsync(new RequestDto()
    {
        ApiType = SD.ApiType.DELETE,
        Url = SD.ProductAPIBase + "/api/product/" + id
    });
}

public async Task<ResponseDto> GetAllProductsAsync()
{
    return await _baseService.SendAsync(new RequestDto()
    {
        ApiType = SD.ApiType.GET,
        Url = SD.ProductAPIBase + "/api/product"
    });
}

public async Task<ResponseDto> GetProductByIdAsync(int id)
{
    return await _baseService.SendAsync(new RequestDto()
    {
        ApiType = SD.ApiType.GET,
        Url = SD.ProductAPIBase + "/api/product/" + id
    });
}

public async Task<ResponseDto> UpdateProductAsync(ProductDto productDto)
{
    return await _baseService.SendAsync(new RequestDto()
    {
        ApiType = SD.ApiType.PUT,
        Data = productDto,
        Url = SD.ProductAPIBase + "/api/product",
        ContentType = SD.ContentType.MultipartFormData
    });
}
}

public class TokenProvider : ITokenProvider
{
    private readonly IHttpContextAccessor _contextAccessor;

    public TokenProvider(IHttpContextAccessor contextAccessor)
    {
        _contextAccessor = contextAccessor;
    }

    public void ClearToken()
    {
        _contextAccessor.HttpContext?.Response.Cookies.Delete(SD.TokenCookie);
    }
}

```

		Ткачук М.А			ІІЗ.КР.Б-121-24-ІІЗ	Арк.
		Мінтій І.С				
Змн.	Арк.	№ докум.	Підпис	Дата		91

```

public string? GetToken()
{
    string? token = null;

    bool? hasToken = _contextAccessor.HttpContext?.Request.Cookies.TryGetValue(SD.TokenCookie, out token);

    return hasToken is true ? token : null;
}

public void SetToken(string token)
{
    _contextAccessor.HttpContext?.Response.Cookies.Append(SD.TokenCookie, token);
}
}

public class AllowedExtensionsAttribute : ValidationAttribute
{
    private readonly string[] _extensions;
    public AllowedExtensionsAttribute(string[] extensions)
    {
        _extensions = extensions;
    }
    protected override ValidationResult? IsValid(object? value, ValidationContext validationContext)
    {
        var file = value as IFormFile;

        if (file != null)
        {
            var extension = Path.GetExtension(file.FileName);
            if(!_extensions.Contains(extension.ToLower()))
            {
                return new ValidationResult("Цей формат фото не дозволений!");
            }
        }

        return ValidationResult.Success;
    }
}

public class MaxFileSizeAttribute : ValidationAttribute
{
    private readonly int _maxFileSize;
    public MaxFileSizeAttribute(int maxFileSize)
    {
        _maxFileSize = maxFileSize;
    }
    protected override ValidationResult? IsValid(object? value, ValidationContext validationContext)
    {
        var file = value as IFormFile;

        if (file != null)
        {
            if (file.Length > (_maxFileSize * 1024 * 1024))
            {
                return new ValidationResult($"Максимальний розмір файлу {_maxFileSize} MB!");
            }
        }

        return ValidationResult.Success;
    }
}

public class SD
{
    public static string CouponAPIBase { get; set; }
    public static string ProductAPIBase { get; set; }
    public static string AuthAPIBase { get; set; }
    public static string ShoppingCartAPIBase { get; set; }
    public static string OrderAPIBase { get; set; }

    public const string RoleAdmin = "ADMIN";
    public const string RoleCustomer = "CUSTOMER";
}

```

		Ткачук М.А			ІПЗ.КР.Б-121-24-ІПЗ	Арк.
		Мінтій І.С				
Змн.	Арк.	№ докум.	Підпис	Дата		92

```

public const string TokenCookie = "JWTToken";
public enum ApiType
{
    GET,
    POST,
    PUT,
    DELETE
}

public const string Status_Pending = "Pending";
public const string Status_Approved = "Approved";
public const string Status_ReadyForPickup = "ReadyForPickup";
public const string Status_Compteted = "Completed";
public const string Status_Refunded = "Refunded";
public const string Status_Cancelled = "Cancelled";
//public const string Status_Pending = "В очікуванні";
//public const string Status_Approved = "Підтверджено";
//public const string Status_ReadyForPickup = "Готовий до отримання";
//public const string Status_Compteted = "Виконано";
//public const string Status_Refunded = "Повернено";
//public const string Status_Cancelled = "Скасовано";

public enum ContentType
{
    Json,
    MultipartFormData,
}
}

var builder = WebApplication.CreateBuilder(args);

// Add services to the container.
builder.Services.AddControllersWithViews();
builder.Services.AddHttpContextAccessor();
builder.Services.AddHttpClient();

builder.Services.AddHttpClient<IProductService, ProductService>();
builder.Services.AddHttpClient<ICouponService, CouponService>();
builder.Services.AddHttpClient<ICartService, CartService>();
builder.Services.AddHttpClient<IAuthService, AuthService>();
builder.Services.AddHttpClient<IOrderService, OrderService>();

SD.CouponAPIBase = builder.Configuration["ServiceUrls:CouponAPI"];
SD.OrderAPIBase = builder.Configuration["ServiceUrls:OrderAPI"];
SD.ShoppingCartAPIBase = builder.Configuration["ServiceUrls:ShoppingCartAPI"];
SD.AuthAPIBase = builder.Configuration["ServiceUrls:AuthAPI"];
SD.ProductAPIBase = builder.Configuration["ServiceUrls:ProductAPI"];

builder.Services.AddScoped<ITokenProvider, TokenProvider>();
builder.Services.AddScoped<IBaseService, BaseService>();
builder.Services.AddScoped<IOrderService, OrderService>();
builder.Services.AddScoped<IProductService, ProductService>();
builder.Services.AddScoped<ICartService, CartService>();
builder.Services.AddScoped<IAuthService, AuthService>();
builder.Services.AddScoped<ICouponService, CouponService>();

builder.Services.AddAuthentication(CookieAuthenticationDefaults.AuthenticationScheme)
    .AddCookie(options =>
    {
        options.ExpireTimeSpan = TimeSpan.FromHours(10);
        options.LoginPath = "/Auth/Login";
        options.AccessDeniedPath = "/Auth/AccessDenied";
    });

var app = builder.Build();

// Configure the HTTP request pipeline.

```

		Ткачук М.А			ІІЗ.КР.Б-121-24-ІІЗ	Арк.
		Мінтій І.С				
Змн.	Арк.	№ докум.	Підпис	Дата		93

```
if (!app.Environment.IsDevelopment())
{
    app.UseExceptionHandler("/Home/Error");
    // The default HSTS value is 30 days. You may want to change this for production scenarios, see https://aka.ms/aspnetcore-hsts.
    app.UseHsts();
}

app.UseHttpsRedirection();
app.UseStaticFiles();

app.UseRouting();
app.UseAuthentication();
app.UseAuthorization();
app.UseAuthorization();

app.MapControllerRoute(
    name: "default",
    pattern: "{controller=Home}/{action=Index}/{id?}");

app.Run();
```

CoffeeShop.GatewaySolution

```
public static class WebApplicationsBuilderExtensions
{
    public static WebApplicationBuilder AddAppAuthetication(this WebApplicationBuilder builder)
    {
        var settingsSection = builder.Configuration.GetSection("ApiSettings");

        var secret = settingsSection.GetValue<string>("Secret");
        var issuer = settingsSection.GetValue<string>("Issuer");
        var audience = settingsSection.GetValue<string>("Audience");
        var key = Encoding.ASCII.GetBytes(secret);
        builder.Services.AddAuthentication(x =>
        {
            x.DefaultAuthenticateScheme = JwtBearerDefaults.AuthenticationScheme;
            x.DefaultChallengeScheme = JwtBearerDefaults.AuthenticationScheme;
        }).AddJwtBearer(x =>
        {
            x.TokenValidationParameters = new TokenValidationParameters
            {
                ValidateIssuerSigningKey = true,
                IssuerSigningKey = new SymmetricSecurityKey(key),
                ValidateIssuer = true,
                ValidIssuer = issuer,
                ValidAudience = audience,
                ValidateAudience = true
            };
        });
        return builder;
    }
}

{
    "Routes": [
        //ProductAPI
        {
            "DownstreamPathTemplate": "/api/product",
            "DownstreamScheme": "https",
            "DownstreamHostAndPorts": [
                {
                    "Host": "localhost",
                    "Port": 7000
                }
            ],
            "UpstreamPathTemplate": "/api/product",
            "UpstreamHttpMethod": [ "Get" ]
        },
        {

```

```

"DownstreamPathTemplate": "/api/product/{id}",
"DownstreamScheme": "https",
"DownstreamHostAndPorts": [
  {
    "Host": "localhost",
    "Port": 7000
  }
],
"UpstreamPathTemplate": "/api/product/{id}",
"UpstreamHttpMethod": [ "Get", "Delete" ],
"AuthenticationOptions": {
  "AuthenticationProviderKey": "Bearer"
}
},
{
  "DownstreamPathTemplate": "/api/product",
  "DownstreamScheme": "https",
  "DownstreamHostAndPorts": [
    {
      "Host": "localhost",
      "Port": 7000
    }
  ],
  "UpstreamPathTemplate": "/api/product/",
  "UpstreamHttpMethod": [ "POST", "PUT" ],
  "AuthenticationOptions": {
    "AuthenticationProviderKey": "Bearer"
  }
},
//CouponAPI
{
  "DownstreamPathTemplate": "/api/coupon",
  "DownstreamScheme": "https",
  "DownstreamHostAndPorts": [
    {
      "Host": "localhost",
      "Port": 7001
    }
  ],
  "UpstreamPathTemplate": "/api/coupon",
  "UpstreamHttpMethod": [ "Get" ]
},
{
  "DownstreamPathTemplate": "/api/coupon/GetByCode/{code}",
  "DownstreamScheme": "https",
  "DownstreamHostAndPorts": [
    {
      "Host": "localhost",
      "Port": 7001
    }
  ],
  "UpstreamPathTemplate": "/api/coupon/GetByCode/{code}",
  "UpstreamHttpMethod": [ "Get" ]
},
{
  "DownstreamPathTemplate": "/api/coupon/{id}",
  "DownstreamScheme": "https",
  "DownstreamHostAndPorts": [
    {
      "Host": "localhost",
      "Port": 7001
    }
  ],
  "UpstreamPathTemplate": "/api/coupon/{id}",
  "UpstreamHttpMethod": [ "Get", "Delete" ],
  "AuthenticationOptions": {
    "AuthenticationProviderKey": "Bearer"
  }
},

```

		Ткачук М.А			ІІЗ.КР.Б-121-24-ІІЗ	Арк.
		Мінтій І.С				
Змн.	Арк.	№ докум.	Підпис	Дата		95

```

{
  "DownstreamPathTemplate": "/api/coupon",
  "DownstreamScheme": "https",
  "DownstreamHostAndPorts": [
    {
      "Host": "localhost",
      "Port": 7001
    }
  ],
  "UpstreamPathTemplate": "/api/coupon/",
  "UpstreamHttpMethod": [ "POST", "PUT" ],
  "AuthenticationOptions": {
    "AuthenticationProviderKey": "Bearer"
  }
},

//ShoppingCartAPI
{
  "DownstreamPathTemplate": "/api/cart/GetCart/{userId}",
  "DownstreamScheme": "https",
  "DownstreamHostAndPorts": [
    {
      "Host": "localhost",
      "Port": 7003
    }
  ],
  "UpstreamPathTemplate": "/api/cart/GetCart/{userId}",
  "UpstreamHttpMethod": [ "Get" ]
},
{
  "DownstreamPathTemplate": "/api/cart/RemoveCart",
  "DownstreamScheme": "https",
  "DownstreamHostAndPorts": [
    {
      "Host": "localhost",
      "Port": 7003
    }
  ],
  "UpstreamPathTemplate": "/api/cart/RemoveCart",
  "UpstreamHttpMethod": [ "POST" ],
  "AuthenticationOptions": {
    "AuthenticationProviderKey": "Bearer"
  }
},
{
  "DownstreamPathTemplate": "/api/cart/CartUpsert",
  "DownstreamScheme": "https",
  "DownstreamHostAndPorts": [
    {
      "Host": "localhost",
      "Port": 7003
    }
  ],
  "UpstreamPathTemplate": "/api/cart/CartUpsert",
  "UpstreamHttpMethod": [ "POST" ],
  "AuthenticationOptions": {
    "AuthenticationProviderKey": "Bearer"
  }
},
{
  "DownstreamPathTemplate": "/api/cart/EmailCartRequest",
  "DownstreamScheme": "https",
  "DownstreamHostAndPorts": [
    {
      "Host": "localhost",
      "Port": 7003
    }
  ],
  "UpstreamPathTemplate": "/api/cart/EmailCartRequest",

```

		Ткачук М.А			ІІЗ.КР.Б-121-24-ІІЗ	Арк.
		Мінтій І.С				
Змн.	Арк.	№ докум.	Підпис	Дата		96

```

    "UpstreamHttpMethod": [ "POST" ],
    "AuthenticationOptions": {
        "AuthenticationProviderKey": "Bearer"
    }
},
{
    "DownstreamPathTemplate": "/api/cart/ApplyCoupon",
    "DownstreamScheme": "https",
    "DownstreamHostAndPorts": [
        {
            "Host": "localhost",
            "Port": 7003
        }
    ],
    "UpstreamPathTemplate": "/api/cart/ApplyCoupon",
    "UpstreamHttpMethod": [ "POST" ],
    "AuthenticationOptions": {
        "AuthenticationProviderKey": "Bearer"
    }
},

//OrderAPI
{
    "DownstreamPathTemplate": "/api/order/GetOrders?userId={id}",
    "DownstreamScheme": "https",
    "DownstreamHostAndPorts": [
        {
            "Host": "localhost",
            "Port": 7004
        }
    ],
    "UpstreamPathTemplate": "/api/order/GetOrders/{id}",
    "UpstreamHttpMethod": [ "Get" ]
},
{
    "DownstreamPathTemplate": "/api/order/GetOrders?userId=''",
    "DownstreamScheme": "https",
    "DownstreamHostAndPorts": [
        {
            "Host": "localhost",
            "Port": 7004
        }
    ],
    "UpstreamPathTemplate": "/api/order/GetOrders/",
    "UpstreamHttpMethod": [ "Get" ]
},
{
    "DownstreamPathTemplate": "/api/order/GetOrder/{id}",
    "DownstreamScheme": "https",
    "DownstreamHostAndPorts": [
        {
            "Host": "localhost",
            "Port": 7004
        }
    ],
    "UpstreamPathTemplate": "/api/order/GetOrder/{id}",
    "UpstreamHttpMethod": [ "Get" ]
},
{
    "DownstreamPathTemplate": "/api/order/UpdateOrderStatus/{id}",
    "DownstreamScheme": "https",
    "DownstreamHostAndPorts": [
        {
            "Host": "localhost",
            "Port": 7004
        }
    ],
    "UpstreamPathTemplate": "/api/order/UpdateOrderStatus/{id}",
    "UpstreamHttpMethod": [ "POST" ],

```

		Ткачук М.А			ІІЗ.КР.Б-121-24-ІІЗ	Арк.
		Мінтій І.С				
Змн.	Арк.	№ докум.	Підпис	Дата		97

```

    "AuthenticationOptions": {
      "AuthenticationProviderKey": "Bearer"
    }
  },
  {
    "DownstreamPathTemplate": "/api/order/ValidateStripeSession",
    "DownstreamScheme": "https",
    "DownstreamHostAndPorts": [
      {
        "Host": "localhost",
        "Port": 7004
      }
    ],
    "UpstreamPathTemplate": "/api/order/ValidateStripeSession",
    "UpstreamHttpMethod": [ "POST" ],
    "AuthenticationOptions": {
      "AuthenticationProviderKey": "Bearer"
    }
  },
  {
    "DownstreamPathTemplate": "/api/order/CreateStripeSession",
    "DownstreamScheme": "https",
    "DownstreamHostAndPorts": [
      {
        "Host": "localhost",
        "Port": 7004
      }
    ],
    "UpstreamPathTemplate": "/api/order/CreateStripeSession",
    "UpstreamHttpMethod": [ "POST" ],
    "AuthenticationOptions": {
      "AuthenticationProviderKey": "Bearer"
    }
  },
  {
    "DownstreamPathTemplate": "/api/order/CreateOrder",
    "DownstreamScheme": "https",
    "DownstreamHostAndPorts": [
      {
        "Host": "localhost",
        "Port": 7004
      }
    ],
    "UpstreamPathTemplate": "/api/order/CreateOrder",
    "UpstreamHttpMethod": [ "POST" ],
    "AuthenticationOptions": {
      "AuthenticationProviderKey": "Bearer"
    }
  }
],
"GlobalConfiguration": {
  "BaseUrl": "https://localhost:7777"
}
}

var builder = WebApplication.CreateBuilder(args);
builder.AddAppAuthetication();
builder.Configuration.AddJsonFile("ocelot.json", optional: false, reloadOnChange: true);
builder.Services.AddOcelot(builder.Configuration);

var app = builder.Build();

app.MapGet("/", () => "Hello World!");

app.UseOcelot().GetAwaiter().GetResult();
app.Run();

```

		Ткачук М.А			ІІЗ.КР.Б-121-24-ІІЗ	Арк.
		Мінтій І.С				
Змн.	Арк.	№ докум.	Підпис	Дата		98