

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:

«Розробка книжкового інтернет-магазину»

Виконав студент групи ЗПЗ-19-1
КОРЕНІВСЬКИЙ Матвій Олександрович

Керівник роботи:
МОРОЗОВ Андрій Васильович

Рецензент:
ЛОКТИКОВА Тамара Миколаївна

Житомир – 2023

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

«ЗАТВЕРДЖУЮ»

В.о. зав. кафедри інженерії
програмного забезпечення
Андрій МОРОЗОВ
«14» лютого 2023 р.

ЗАВДАННЯ
на кваліфікаційну роботу

Здобувач вищої освіти: **КОРЕНІВСЬКИЙ Матвій Олександрович**

Керівник роботи: **МОРОЗОВ Андрій Васильович**

Тема роботи: **«Розробка книжкового інтернет-магазину»**,
затверджена Наказом закладу вищої освіти від «14» лютого 2023 р., №71с

Вихідні дані для роботи: Методи та засоби розробки веб-сайту за визначеними предметними областями. Технології, структури, моделі, засоби та методи розробки взаємодії веб-сайтів та модулів керування сайту.

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

Розділ	Консультант	Завдання видав	Завдання прийняв
1	Морозов А.В.	16.02.2023р.	16.02.2023р.
2	Морозов А.В.	14.03.2023р.	14.03.2023р.
3	Морозов А.В.	05.05.2023р.	05.05.2023р.

Календарний план

№ з/п	Назва етапів випускної кваліфікаційної роботи	Термін виконання етапів роботи	Примітка
1	Аналіз вимог до програмного продукту	16 лютого 2023 – 26 лютого 2023	Виконано
2	Пошук, огляд та аналіз аналогічних веб-сайтів	27 лютого 2023 – 2 березня 2023	Виконано
3	Вивчення матеріалу по конкретній тематиці	3 березня 2023 – 5 березня 2023	Виконано
4	Планування структури програми	6 березня 2023 – 12 березня 2023	Виконано
5	Проектування та реалізація архітектури веб-системи книжкового інтернет-магазину	13 березня 2023 – 22 березня 2023	Виконано
6	Написання програмного коду	23 березня 2023 – 30 квітня 2023	Виконано
7	Тестування роботи програми	1 травня 2023 – 6 травня 2023	Виконано
8	Аналіз отриманих результатів тестування, відлагодження програми	7 червня 2023 – 8 травня 2023	Виконано
9	Оформлення пояснювальної записки	9 травня 2023 – 19 травня 2023	Виконано
10	Попередній захист	18 травня 2023	Виконано

РЕФЕРАТ

Випускна робота бакалавра складається з програмного комплексу веб-сайту книжкового інтернет-магазину та пояснювальної записки. Пояснювальна записка до випускної роботи містить текстову частину, викладену на 77 сторінках друкованого тексту, має 17 сторінок додатків. Список використаних джерел містить 30 найменувань і займає 3 сторінки. В роботі наведено 37 рисунків та 11 таблиць. Загальний обсяг роботи – 94 сторінки.

Метою роботи є розробка книжкового інтернет-магазину засобами HTML, CSS, JavaScript, а також розробка системи адміністрування за допомогою мови програмування C# із використанням ASP.NET та бази даних MS SQL.

У роботі були встановлені потреби та вимоги користувача, а також обґрунтовано створення концептуальної моделі. Для досягнення поставленої мети було проведено аналіз аналогічних програмних продуктів та вибрано необхідні інструменти для реалізації проєкту. Додатково, було розроблено структуру бази даних, визначено алгоритм роботи системи, обрано відповідну архітектуру та розроблено об'єктно-орієнтовану структуру системи.

КЛЮЧОВІ СЛОВА: КНИЖКИ, БАЗА ДАНИХ, ІНТЕРНЕТ, МАГАЗИН, САЙТ.

					ІПЗ.КР.Б-121-23-ПЗ			
Зм.	Арк.	№ докум.	Підпис	Дата	Розробка книжкового інтернет-магазину Пояснювальна записка	Літ.	Арк.	Аркушів
Розроб.		М. Коренівський					4	94
Керівник		А. Морозов						
Н. контр.		О. Тичина				Житомирська політехніка, група ЗІПЗ-19-1		
Зав. каф.		А. Морозов						

ABSTRACT

The bachelor's thesis consists of a software package containing the website of an online bookstore and an explanatory note. The explanatory note to the bachelor's thesis contains the text part, set out on 77 pages of printed text, has 17 pages of appendices. The list of used sources contains 30 names and occupies 3 pages. The paper presents 37 illustrations and 11 tables. The total volume of the work is 94 pages.

The aim of the work is to develop online bookstore using HTML, CSS, JavaScript, as well as to develop an administration system using the C# programming language and ASP.NET and the MS SQL database.

In the work, the needs and requirements of the user were established, as well as the creation of a conceptual model was substantiated. To achieve the aim, an analysis of similar software products was conducted and the necessary tools were selected for the implementation of the project. In addition, the structure of the database was developed, the algorithm of the system was determined, the appropriate architecture was chosen, and the object-oriented structure of the system was developed.

KEYWORDS: BOOKS, DATABASE, INTERNET, SHOP, WEBSITE.

		Коренівський М.О.			ІПЗ.КР.Б-121-23-ПЗ	Арк.
		Морозов А.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		4

ЗМІСТ

ВСТУП	7
РОЗДІЛ 1. АНАЛІЗ НАПРЯМКІВ ВИКОРИСТАННЯ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ ДЛЯ РЕАЛІЗАЦІЇ КНИЖКОВОГО ІНТЕРНЕТ-МАГАЗИНУ	10
1.1 ОПИС ПРОЄКТУ ТА ПОСТАНОВКА ЗАДАЧІ	10
1.2 АНАЛІЗ АНАЛОГІВ ПРОГРАМНОГО ПРОДУКТУ.....	12
1.3 ВИБІР АРХІТЕКТУРИ ВЕБ-ОРІЄНТОВАНОЇ СИСТЕМИ КНИЖКОВОГО ІНТЕРНЕТ- МАГАЗИНУ	14
1.4 ОБГРУНТУВАННЯ ВИБОРУ ЗАСОБІВ РЕАЛІЗАЦІЇ ТА ВИМОГИ ДО АПАРАТНОГО ЗАБЕЗПЕЧЕННЯ	17
Висновки до першого розділу	19
РОЗДІЛ 2. ПРОЕКТУВАННЯ ВЕБ-ОРІЄНТОВАНОЇ СИСТЕМИ КНИЖКОВОГО ІНТЕРНЕТ-МАГАЗИНУ	21
2.1 ВИЗНАЧЕННЯ ВАРІАНТІВ ВИКОРИСТАННЯ ТА ОБ'ЄКТНО-ОРІЄНТОВАНОЇ СТРУКТУРИ СИСТЕМИ	21
2.2 РОЗРОБКА СТРУКТУРИ БАЗИ ДАНИХ СИСТЕМИ.....	26
2.3 ПРОЄКТУВАННЯ ТА РЕАЛІЗАЦІЯ АЛГОРИТМІВ РОБОТИ СИСТЕМИ	34
2.4 РЕАЛІЗАЦІЯ ДОДАТКУ ВЕБ-ОРІЄНТОВАНОЇ СИСТЕМИ КНИЖКОВОГО ІНТЕРНЕТ- МАГАЗИНУ	41
Висновки до другого розділу	52
РОЗДІЛ 3: ІНТЕРФЕЙС ТА ПОРЯДОК РОБОТИ З ВЕБ-ОРІЄНТОВАНОЮ СИСТЕМОЮ КНИЖКОВОГО ІНТЕРНЕТ-МАГАЗИНУ	53
3.1 ПОРЯДОК ВСТАНОВЛЕННЯ ТА НАЛАШТУВАННЯ ПАРАМЕТРІВ СИСТЕМИ	53
3.2 СТРУКТУРА ІНТЕРФЕЙСУ ТА ПОРЯДОК РОБОТИ З ВЕБ-ОРІЄНТОВАНОЮ СИСТЕМОЮ КНИЖКОВОГО ІНТЕРНЕТ-МАГАЗИНУ	54
3.3 ТЕСТУВАННЯ РОБОТИ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ.....	67
Висновки до третього розділу.....	72
ВИСНОВКИ.....	73
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ	75
ДОДАТКИ.....	78

		Коренівський М.О.			ІПЗ.КР.Б-121-23-ПЗ	Арк.
		Морозов А.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		5

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

БД – База даних

MVC – Модель–вигляд–контролер (або Модель–представлення–контролер, Model-view-controller, MVC) — архітектурний шаблон, який використовується під час проєктування та розробки програмного забезпечення.

ПЗ – Програмне забезпечення

CMS – Система керування вмістом (англ. Content Management System, CMS) — програмне забезпечення для організації веб-сайтів чи інших інформаційних ресурсів в Інтернеті чи окремих комп'ютерних мережах.

API – прикладний програмний інтерфейс

ORM – Об'єктно-реляційна проєкція(Object-relational mapping), технологія програмування, яка зв'язує бази даних з концепціями об'єктно-орієнтованих мов програмування

		Коренівський М.О.			ІПЗ.КР.Б-121-23-ПЗ	Арк.
		Морозов А.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		6

ВСТУП

Актуальність теми. Використання веб-технологій набуває все більшого значення у різних аспектах нашого життя. Веб-сайт є важливим засобом спілкування між підприємством та його спільнотою - партнерами, постачальниками та клієнтами.

Тому розробка веб-сайту стає одним з основних завдань для підприємців, зокрема в онлайн-середовищі. Все більше компаній використовують Інтернет для обміну інформацією та оптимізації взаємодії з клієнтами. Створення веб-сайту є найпоширенішим способом присутності в мережі Інтернет.

Структурована інформація, яку містить інтернет-магазин, надає користувачам повне уявлення про діяльність та характеристики підприємства. Користувачі мають можливість отримати багато корисних веб-адрес з рекламних банерів сайту та комерційних видань, де вони можуть дізнатися ціни та замовити товари.

Такий спосіб комунікації з потенційними клієнтами є ефективнішим, ніж звичайні засоби зв'язку, такі як телефон, пошта чи факс. За допомогою інтерфейсу та навігації клієнти мають можливість самостійно вибирати інформацію, яка їх цікавить. Інтернет-магазин виступає основним інструментом забезпечення комерційної діяльності підприємства, а також як доповнення до рекламної кампанії, забезпечуючи інтерактивну роботу з даними підприємства та можливість зворотного зв'язку.

Організація каталогу товарів, реалізація доступу до особистого кабінету з можливістю перегляду історії замовлень, широке використання сучасних технологій для взаємодії з клієнтами максимально підвищують продуктивність всіх процесів функціонування інтернет-магазину. Автоматизація роботи з даними за допомогою сучасного програмного забезпечення підвищує рівень надання комерційних послуг.

Метою роботи є розробка книжкового інтернет-магазину засобами HTML, CSS, JavaScript, а також розробка системи адміністрування за

		Коренівський М.О.			ІПЗ.КР.Б-121-23-ПЗ	Арк.
		Морозов А.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		7

допомогою мови програмування C# із використанням .NET Framework та бази даних MS SQL. Сервіс має забезпечувати функціонал для повної і безперебійної роботи порталу та взаємодіяти як з комп'ютерними платформами, так і мобільними. Також метою є поглиблення теоретичних та практичних знань та формуванні навичок, необхідних для розробки веб-системи сайту.

Для досягнення поставленої цілі необхідно виконати наступні завдання:

- провести аналіз теоретичних та функціональних потреб проєктування та реалізації інтернет-магазину;
- проаналізувати існуючі веб-сайти, що належать до обраної тематики, та визначити їх унікальність;
- визначити технології, які будуть використовуватись для реалізації веб-сервісу;
- визначити інформаційні потреби предметної області дослідження;
- обрати оптимальну архітектуру програмного комплексу для його функціонування в сучасних умовах;
- розробити опис методів взаємодії з веб-системою;
- спроектувати базу даних для збереження інформації про сервіс та користувачів;
- визначити об'єктну структуру веб-сервісу книжкового інтернет-магазину;
- розробити адаптивний інтерфейс веб-сайту, що буде пристосований до різних пристроїв;
- розробити математичну та алгоритмічну модель функціонування системи на основі бази даних;
- написати серверну частину програми;
- реалізувати CMS.

Об'єктом дослідження є методи та засоби розробки веб-сайту за визначеними предметними областями.

		Коренівський М.О.			ІПЗ.КР.Б-121-23-ПЗ	Арк.
		Морозов А.В.				8
Змн.	Арк.	№ докум.	Підпис	Дата		

Предметом дослідження є технології, структури, моделі, засоби та методи розробки взаємодії веб-сайтів та модулів керування сайту.

В роботі було використано методи об'єктно орієнтованого проєктування, веб-програмування та дизайну інтерфейсів користувача.

Розроблена програмна система може бути використана як комерційна онлайн-платформа для підприємств, основним напрямом роботи яких є продаж книжок.

Випускна робота складається з вступу, 3-х розділів, висновків, списку використаних джерел із 30 найменувань (3 стор.), 1 додатку (116 стор.), 11 таблиць та 59 ілюстрацій по тексту, з них 2 оформлені на окремій сторінці. Повний обсяг роботи становить 220 сторінок, з них 104 сторінки основного тексту.

		Коренівський М.О.			ІПЗ.КР.Б-121-23-ПЗ	Арк.
		Морозов А.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		9

РОЗДІЛ 1. АНАЛІЗ НАПРЯМКІВ ВИКОРИСТАННЯ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ ДЛЯ РЕАЛІЗАЦІЇ КНИЖКОВОГО ІНТЕРНЕТ-МАГАЗИНУ

1.1 Опис проєкту та постановка задачі

На сьогоднішній день майже у кожній комерційній організації є свій власний інтернет-магазин. У сучасному інформаційному середовищі це стало необхідним чинником для успішного функціонування бізнесу, оскільки електронна комерція є невід’ємним чинником сучасного світу. Інтернет-магазин дає змогу користувачам швидко та зручно здійснювати покупки онлайн, що в свою чергу суттєво збільшує обсяги продажів у порівнянні з іншими видами ведення комерційної діяльності.

Автоматизація бізнес-процесів - це впровадження технологій та програмних рішень для заміни або покращення ручних операцій, які виконуються в рамках певних бізнес-процесів. Це означає перенесення частини або повного потоку робочих дій та задач на автоматизовану систему, що дозволяє зберігати час, енергію та ресурси.

Аналізуючи процес автоматизації бізнесу варто зазначити наступні позитивні аспекти:

- підвищення продуктивності: автоматизація збільшує швидкість та ефективність виконання завдань;
- зниження помилок: автоматизація усуває людські помилки та покращує точність роботи;
- економія часу та ресурсів: автоматизація зменшує витрати часу та ресурсів на рутинні процеси;
- покращення якості та точності: Автоматизація сприяє виявленню та усуненню помилок, покращуючи якість виконання завдань;
- збільшення ефективності управління: автоматизація надає швидкий доступ до даних та допомагає в прийнятті обґрунтованих управлінських рішень.

		Коренівський М.О.			ІПЗ.КР.Б-121-23-ПЗ	Арк.
		Морозов А.В.				10
Змн.	Арк.	№ докум.	Підпис	Дата		

В результаті спостережень можна зробити висновок, що автоматизація процесів загалом сприяє суттєвому підвищенню якості надання послуг. Враховуючи предметну область, для оптимізації бізнес-процесів та забезпечення зростання якості надання послуг інтернет комерції, необхідно використання інформаційної системи, яка включає авторизацію користувача, перегляд каталогу товарів, додавання товарів в кошик з можливістю подальшої онлайн-оплати, оформлення замовлення та перегляд історії попередніх замовлень.

Завданням проекту є розробка системи веб-сайту на обрану тематику, який складається з таких вимог:

Назва системи – Веб-орієнтована система книжкового інтернет-магазину «yoBooks».

Призначення системи – система впроваджується в структурі інтернет-магазину для здійснення покупок через глобальну мережу Інтернет.

Мета. Впровадження веб-орієнтованої системи книжкового інтернет-магазину дозволить забезпечити максимальний доступ до послуг наданих інтернет-магазином, зокрема перегляд каталогу товарів та оформлення замовлення. Також адміністрації підприємства буде надано зручний інструмент для роботи з даними інтернет-магазину, можливість пошуку інформації в системі за допомогою механізмів фільтрації та сортування, отримання інформації про кількість товару на складі в режимі реального часу.

Вхідними даними є дані про адміністраторів, клієнтів, категорії книжок, книжки та замовлення. Організація вхідних та вихідних даних повинна відповідати інформаційній структурі виконуваних операцій. Введення оперативних даних повинно виконуватися з використанням діалогових екранних форм, побудованих на основі візуальних компонентів. Введення даних виконується в режимі online.

Технічні вимоги, апаратне та системне забезпечення:

- ПК на базі процесорів Intel, AMD у стандартній комплектації;

		Коренівський М.О.			ІПЗ.КР.Б-121-23-ПЗ	Арк.
		Морозов А.В.				11
Змн.	Арк.	№ докум.	Підпис	Дата		

- засоби для під'єднання до локальної мережі. засоби для під'єднання до локальної мережі;
- оперативна пам'ять - 128Мбайт;
- об'єм дискової пам'яті - 20Гбайт.

Функціональні характеристики системи:

- можливість дистанційної роботи з робочих станцій локальної та глобальної мережі підприємства;
- інтерфейс користувача, що не залежить від операційної системи;
- постійний доступ користувачів до БД;
- оптимальне збереження даних (за обсягом та структурою);
- автентифікацію користувачів та захист інформації від несанкціонованого доступу;
- надійне збереження даних;
- можливість модернізації системи через зміну функціональних потреб користувача або модернізацію обладнання

1.2 Аналіз аналогів програмного продукту

Для аналізування існуючих веб-сайтів за тематикою проєкту було обрано 3 приклади книжкових інтернет-магазинів: «Знання» - znannia.com.ua, «Дім книги» - dk-books.com та «Наш формат» - nashformat.ua, що зображені на рисунках 1.1 - 1.3 відповідно.



Рис. 1.1 – Веб-сайт znannia.com.ua

		Коренівський М.О.			ІПЗ.КР.Б-121-23-ПЗ	Арк.
		Морозов А.В.				12
Змн.	Арк.	№ докум.	Підпис	Дата		

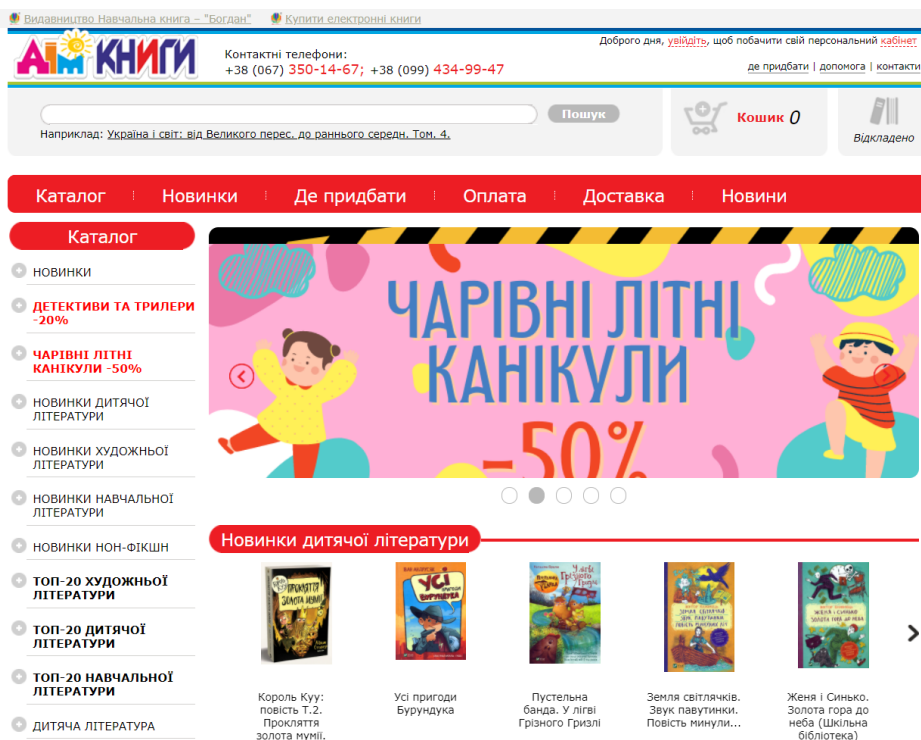


Рис. 1.2 – Веб-сайт dk-books.com

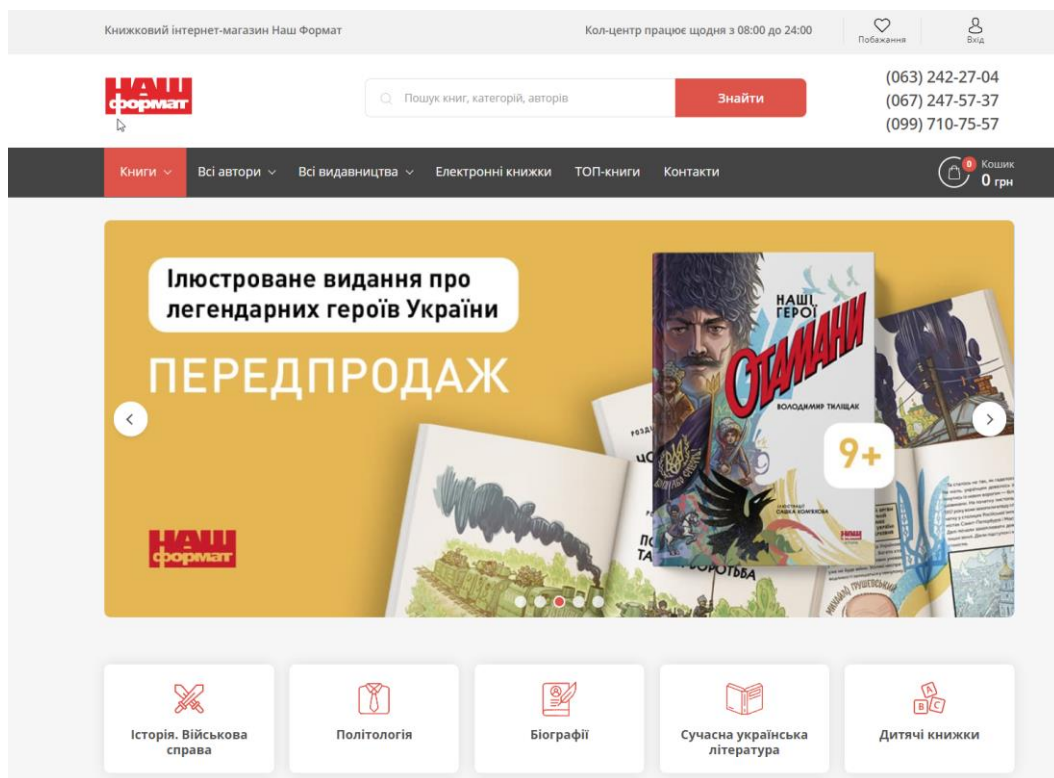


Рис. 1.3 – Веб-сайт nashformat.ua

У таблиці 1.1. приведено узагальнену характеристику та порівняння всіх аналізованих веб-сайтів.

		Коренівський М.О.			ІПЗ.КР.Б-121-23-ПЗ	Арк.
		Морозов А.В.				13
Змн.	Арк.	№ докум.	Підпис	Дата		

Порівняльна таблиця

Назва	Зручність інтерфейсу	Наявність авторизації	Відсутність реклами	Зручний пошук інформації	Адаптивний інтерфейс
dk-books.com	+	+	-	-	-
znannia.com.ua	-	+	+	-	-
nashformat.ua	-	+	+	-	-
yoBooks	+	+	+	+	+

Проаналізувавши аналогічні програмні продукти, які були наведені вище, було встановлено особливості власного інтернет-магазину:

- додаток матиме інтуїтивно зрозумілий і зручний інтерфейс для зручного використання його користувачами;
- веб-сайт буде адаптивним, що дозволить йому коректно відображатися на різних розмірах екранів та на різних пристроях;
- користувачам надаватиметься можливість авторизації, покупок, зміни облікового запису та перегляду історії замовлень. Також будуть доступні зручні фільтри, що допоможуть швидко знайти потрібні товари;
- веб-сайт буде містити достатню кількість інформації для задоволення потреб користувачів.

1.3 Вибір архітектури веб-орієнтованої системи книжкового інтернет-магазину

Щоб інформаційна система функціонувала ефективно та довго, вона повинна задовольняти усі вимоги, мати високу якість та бути простою у використанні для звичайних користувачів.

Архітектура програмного забезпечення впливає на всі ці параметри, оскільки вона визначає організацію системи, її компоненти та взаємозв'язки.

Правильно спроектована архітектура спрощує розробку, впровадження,

		Коренівський М.О.			ІПЗ.КР.Б-121-23-ПЗ	Арк.
		Морозов А.В.				14
Змн.	Арк.	№ докум.	Підпис	Дата		

користування та модифікацію системи. Клієнт-серверна архітектура є доцільним варіантом для створення подібних систем.

Патерн проєктування є загальним рішенням для конкретної проблеми в дизайні архітектури. Він представляє загальну концепцію або приклад рішення, яке може бути адаптоване під потреби конкретної програми.

Використання перевірених патернів при проєктуванні архітектури інформаційної системи сприяє полегшенню подальшого розвитку системи. Завдяки можливості повторного використання коду, яку надають патерни проєктування, розширення системи або внесення змін до неї відбуваються швидко та з мінімальними витратами.

Розглянемо патерни проєктування, які можуть бути використані для створення системи книжкового інтернет-магазину.

Клієнтська частина інформаційної системи відповідає за обробку дій користувача. Після введення користувачем інформації, вона формує та надсилає запит до серверної частини, яка оброблює цей запит та повертає відповідь. Таким чином, клієнтська частина взаємодіє з користувачем, відображаючи необхідну інформацію в зручному та зрозумілому форматі. Вона забезпечує зручний інтерфейс для взаємодії з системою.

Для досягнення поставленої мети було вибрано використання архітектурного шаблону MVC (Model-View-Controller). Цей шаблон використовується при проєктуванні та розробці програмного забезпечення, включаючи веб-сайти.

В шаблоні MVC програма розділяється на три взаємопов'язані компоненти з розподіленими функціями:

- Модель (Model) відповідає за зберігання даних, їх логіку та структуру.
- Представлення (View) відображає ці дані користувачу, тобто створює користувацький інтерфейс програми.

		Коренівський М.О.			ІПЗ.КР.Б-121-23-ПЗ	Арк.
		Морозов А.В.				15
Змн.	Арк.	№ докум.	Підпис	Дата		

- Контролер (Controller) керує компонентами, обробляє запити від користувача та відповідає на них. Основна логіка роботи з даними, а також поведінка системи реалізується в контролері.

Таким чином, шаблон MVC дозволяє ефективно організувати логіку та взаємодію компонентів в програмі, спрощує розробку та розширення системи, а також полегшує управління даними та їх відображення для користувача[4].

На рисунку 1.4 наведена діаграма взаємодії між компонентами шаблону та додатковими компонентами. У загальному випадку, коли програма отримує запит, система маршрутизації вибирає відповідний контролер для обробки запиту. Контролер здійснює обробку запиту, взаємодіючи з даними через моделі та використовуючи представлення для генерації відповіді. Результат обробки контролера надсилається клієнту у відповідь, часто у вигляді HTML-сторінки, яку користувач бачить у своєму веб-браузері.

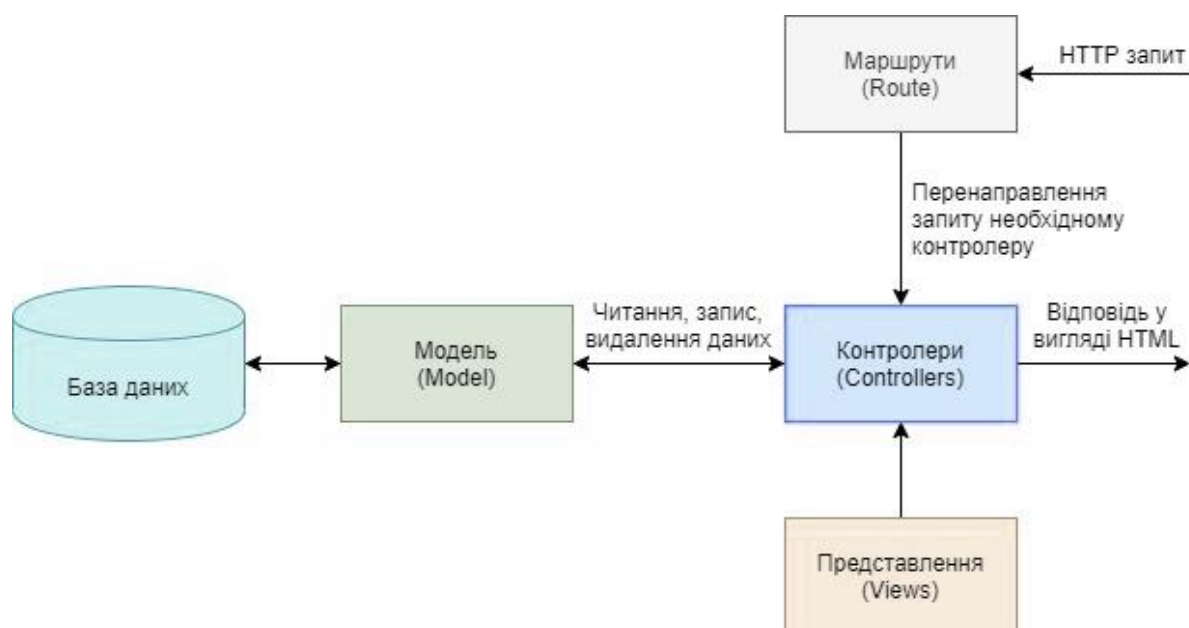


Рис. 1.4 – Діаграма взаємодії між компонентами шаблону

Використання патерну MVC є доцільним для розробки веб-сайту з кількома причинами:

1. Розділення відповідальностей: MVC дозволяє чітко розділити логіку додатку на три компоненти: модель (Model), представлення (View) і контролер (Controller). Це сприяє кращій організації коду і підтримці чистоти архітектури[14].

2. Легка зміна внутрішнього вигляду: Представлення відповідає за відображення даних користувачу. Завдяки відокремленню представлення від решти системи, зміни в дизайні або вигляді веб-сайту можуть бути внесені без впливу на логіку та функціональність.
3. Повторне використання коду: Модель і контролер можуть бути повторно використані в різних частинах веб-сайту або навіть в інших проєктах. Це сприяє ефективності розробки і зменшенню часу, необхідного для створення нових функціональних частин.
4. Легка тестованість: Оскільки логіка додатку розділена на компоненти, кожен з них може бути легко тестований окремо. Це дозволяє виявляти і усувати помилки на ранніх етапах розробки.
5. Інші переваги при розробці: MVC сприяє розподілу роботи між розробниками, забезпечуючи їм можливість працювати паралельно над різними компонентами системи. Він також полегшує підтримку і розширення веб-сайту, оскільки зміни в одному компоненті мають мінімальний вплив на інші.

1.4 Обґрунтування вибору засобів реалізації та вимоги до апаратного забезпечення

Після ретельного вивчення задачі та аналізу подібних програмних продуктів можна перейти до наступного етапу - вибору засобів реалізації та формулювання вимог до програмного забезпечення.

У якості мови програмування для розробки серверної частини було обрано C# та платформу .NET Framework. Такий вибір має кілька переваг. По-перше, C# є мовою програмування з високим рівнем безпеки та стабільності, що дозволяє побудувати надійне програмне забезпечення. Друге, .NET Framework надає широкий набір бібліотек та фреймворків, що спрощують розробку, забезпечують швидкість розробки та підтримку широкого функціоналу[23].

		Коренівський М.О.			ІПЗ.КР.Б-121-23-ПЗ	Арк.
		Морозов А.В.				17
Змн.	Арк.	№ докум.	Підпис	Дата		

Крім того, екосистема .NET має велику спільноту розробників та підтримку з боку Microsoft, що забезпечує доступ до документації, інструментів та ресурсів.

Клієнтська частина розроблятиметься за допомогою HTML, CSS, JavaScript та бібліотеки jQuery.

HTML є стандартною мовою розмітки для веб-сторінок, що забезпечує структуроване відображення контенту на сайті. Використання HTML дозволяє легко визначити заголовки, абзаци, списки, таблиці та інші елементи сторінки.

CSS (Cascading Style Sheets) дозволяє задавати стиль та зовнішній вигляд елементів веб-сторінки. Завдяки CSS можна контролювати кольори, розміри, шрифти, межі та інші візуальні аспекти сайту. Це дозволяє створювати привабливий та цілісний дизайн для всіх сторінок сайту.

JavaScript є мовою програмування, яка надає можливості для взаємодії з користувачем та динамічної зміни вмісту сторінки. Використання JavaScript дозволяє створювати інтерактивні елементи, які реагують на дії користувача, такі як кліки, наведення курсора, заповнення форм та інші події.

Бібліотека jQuery є потужним інструментом, який спрощує роботу з JavaScript. Вона надає готові функції та методи, які дозволяють швидко вибирати елементи, змінювати їх властивості, обробляти події та здійснювати асинхронні запити до сервера[8].

Враховуючи той факт, що розробка клієнтської частини сайту відбувається на платформі .NET, варто зазначити сильні сторони Razor – технології для динамічного відображення контенту на серверній стороні. Razor пропонує простий та зрозумілий синтаксис, що полегшує розробку веб-сторінок. Ця технологія інтегрується з мовою програмування C# та дозволяє використовувати код в HTML-розмтці. Технологія підтримує модульність та повторне використання фрагментів розмітки[29].

Для розробки математичної та алгоритмічної моделі функціонування системи, яка базується на базі даних, а також для проектування самої бази даних відповідно до визначеної предметної області, важливо вибрати

		Коренівський М.О.			ІПЗ.КР.Б-121-23-ПЗ	Арк.
		Морозов А.В.				18
Змн.	Арк.	№ докум.	Підпис	Дата		

відповідну систему управління базами даних. З метою досягнення цієї мети, було обрано MS SQL як СУБД для створення бази даних. Обрана система має свої особливості та переваги, які сприятимуть ефективній та надійній роботі з базою даних, а також дозволять використовувати потрібні функціональні можливості для розробки та оптимізації системи.

При розробці веб-додатків часто використовуються фреймворки, які спрощують процес розробки та забезпечують широкий функціонал для ефективної побудови додатків. Одним з популярних виборів для розробки веб-додатків є .NET Framework, який має кілька вагомих переваг.

Масштабованість: .NET Framework дозволяє легко розширювати веб-додатки, додаючи новий функціонал або вдосконалювати існуючий.

Безпека: .NET Framework має вбудовані механізми безпеки, які захищають веб-додатки від різних атак.

Крім того, для розробки веб-додатків на базі .NET Framework зазвичай використовується фреймворк ASP.NET. ASP.NET використовує компіляцію на етапі розгортання та кешування результатів виконання, що забезпечує швидкий відгук веб-сайту та ефективну обробку запитів. За допомогою ASP.NET можна розробляти розширювані веб-додатки, які можна легко модифікувати та розширювати у майбутньому.

Зважаючи на вищевказані переваги, використання .NET Framework та ASP.NET для розробки веб-додатків, зокрема з використанням мови C#, дозволяє створювати потужні та безпечні додатки з високою продуктивністю та можливістю простого та ефективного масштабування.

Висновки до першого розділу

У першому підрозділі роботи поставлено задачу дипломного проєкту а також детально описано його актуальність. Також було вирішено, в якому середовищі буде розроблятися веб-сайт і які засоби для цього використовуватимуться.

		Коренівський М.О.			ІПЗ.КР.Б-121-23-ПЗ	Арк.
		Морозов А.В.				19
Змн.	Арк.	№ докум.	Підпис	Дата		

У другому підрозділі були проаналізовані аналогічні книжкові інтернет-магазини, що відповідають тематиці проєкту. Після аналізу були визначені особливості та функції, які будуть реалізовані в рамках проєкту.

У третьому підрозділі було описано архітектуру веб-орієнтованої системи книжкового інтернет-магазину, а також обраний патерн проєктування MVC.

У четвертому підрозділі було сформульовано технічне завдання для веб-додатку, в якому були визначені вимоги до функціоналу та інтерфейсу. Також були розглянуті мінімальні системні вимоги для ефективної роботи додатку.

		Коренівський М.О.			ІПЗ.КР.Б-121-23-ПЗ	Арк.
		Морозов А.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		20

РОЗДІЛ 2. ПРОЕКТУВАННЯ ВЕБ-ОРІЄНТОВАНОЇ СИСТЕМИ КНИЖКОВОГО ІНТЕРНЕТ-МАГАЗИНУ

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

Для забезпечення неперервного функціонування системи та відповідності вимогам стандартів та специфікацій, були визначені високорівневі вимоги до веб-орієнтованої системи книжкового інтернет-магазину. Були описані умови, обов'язкові для виконання користувачем для роботи з системою, а також основні бізнес-переваги даної системи. Крім того, були наведені можливості системи, обмеження та показники якості.

Бізнес-вимоги

- Основні цілі: проєкт створюється з призначенням для збору, обробки та відображення даних книжкового інтернет-магазину, а саме інформації про категорії товарів, товари, користувачів та їх замовлення. Передбачено використання механізму пошуку за набором критеріїв будь-яких даних системи. Призначенням книжкового інтернет-магазину є надання послуг в області електронної комерції, можливість керування інформацією про товари, оброблювати замовлення клієнтів.
- Представлення проєкту: проєкт буде реалізовано у вигляді сайту, в якому міститься структурована інформація за даною тематичною областю.
- Розрахунок нормативного часу для обробки замовлень: кожне замовлення виконується окремо та відразу після підтвердження оплати.

Вимоги користувачів

- Зовнішні користувачі
 - Автентифікація та авторизація користувача.
 - Реєстрація нового користувача.
 - Зміна даних облікового запису користувача.

		Коренівський М.О.			ІПЗ.КР.Б-121-23-ПЗ	Арк.
		Морозов А.В.				21
Змн.	Арк.	№ докум.	Підпис	Дата		

- Перегляд каталогу товарів.
- Можливість пошуку, фільтрації та сортування товарів за набором різноманітних критеріїв.
- Додавання товарів у кошик.
- Перегляд списку товарів у кошику.
- Оформлення замовлення із зазначенням інформації про отримувача.
- Онлайн-оплата товарів за допомогою платіжної системи LiqPay.
- Внутрішній користувач-адміністратор
 - Створення, редагування та видалення записів у БД.

Нефункціональні вимоги то програмного забезпечення:

- Сприйняття
 - Час для опанування веб-системи звичайним користувачем дорівнює 5-10 хвилин.
 - Час відповіді системи не повинен перевищувати 5 секунд, у випадку високої завантаженості – 10 секунд
 - Інтерфейс представлення має бути зручним та інтуїтивно зрозумілим, не вимагаючи додаткової підготовки від користувача.
- Надійність
 - Система повинна мати високу доступність, час обслуговування системи не повинен перевищувати 10% від загального часу роботи.
 - Середній час безперервної роботи – цілодобово.
 - Норма помилок та дефектів, що є максимально допустимою в роботі системи, дорівнює 1 помилці на 1000 запитів користувача.
- Продуктивність
 - Підтримка мінімум 100 одночасно працюючих зі спільною базою даних користувачів.
- Можливість експлуатації
 - Масштабування – система має бути адаптивною до збільшення продуктивності без негативних наслідків для роботи системи.

Системні вимоги

		Коренівський М.О.			ІПЗ.КР.Б-121-23-ПЗ	Арк.
		Морозов А.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		22

- Вимоги до середовища виконання:

Система повинна задовольняти вимогам на комп'ютері, що знаходиться в наступній мінімальній комплектації:

- 128Мб пам'яті
- Процесор з тактовою частотою 600MHz.
- Операційна система Windows XP та вище або MacOS
- Браузери (Opera, Firefox, Chrome, Safari).
- Загальна освіта з інформатики для користувачів.

- Вимоги до СУБД та доступу до даних:

- Представлення СУБД реляційного доступу у ядрі системи.

Визначивши вимоги, можливості та обмеження системи можна створити діаграму варіантів використання, яка зображена на рисунку 2.1.

Діаграма варіантів використання(діаграма прецедентів) - це засіб, що відображає взаємозв'язки між акторами та прецедентами в системі. Головна мета цієї діаграми полягає в тому, щоб замовник, кінцевий користувач та розробник могли спільно обговорювати поведінку системи та її функціональність[5].

На рисунку 2.1 визначено ролі користувачів та їх можливості в системі. Враховуючи мету розробки, система повинна містити наступні типи користувачів:

- неавтентифікований користувач;
- авторизований користувач;
- адміністратор.

Неавтентифікований користувач має можливість створення облікового запису шляхом реєстрації, а також може увійти в свій обліковий запис.

Авторизований користувач має можливість перегляду каталогу, додавання товарів в кошик та оформлення замовлення.

Адміністратор має доступ до панелі керування інтернет-магазину та може редагувати, додавати або видаляти дані про категорії товарів, товари та облікові записи користувачів. Також адміністратор може змінювати статус замовлення в залежності від стадії його оброблення.

		Коренівський М.О.			ІПЗ.КР.Б-121-23-ПЗ	Арк.
		Морозов А.В.				23
Змн.	Арк.	№ докум.	Підпис	Дата		

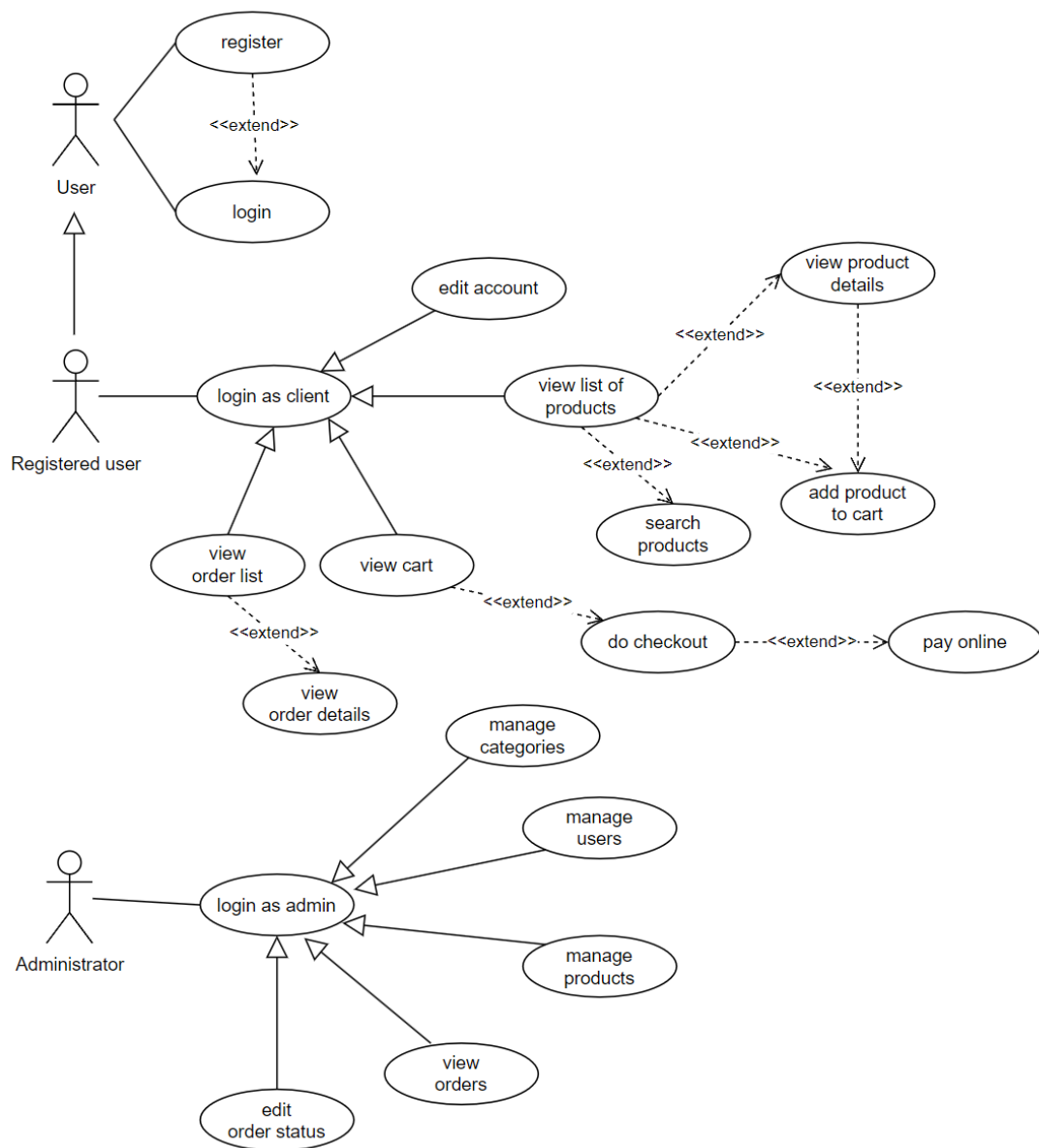


Рис. 2.1 – Діаграма варіантів використання

Беручи до уваги загальний алгоритм роботи та станів веб-орієнтованої системи, описаний в попередніх підрозділах, можна перейти до опису об'єктно-орієнтованої моделі програмного комплексу.

Веб-сайт складатиметься з двох частин – адміністративної та користувацької. Адміністративна частина надаватиме можливість додавання, перегляду, редагування та видалення даних, що містяться в системі. Користувацька частина забезпечуватиме надання послуг в сфері електронної комерції.

Результатом визначення архітектури у попередніх розділах та необхідної інформації для проєктування схем є побудована діаграма класів модуля роботи з даним, яку зображено на рисунку 2.2.

Діаграма класів є важливим інструментом в об'єктно-орієнтованому програмуванні, який використовується для візуального представлення класів, їх взаємозв'язків та структури системи. Вона надає графічне зображення класів, їх атрибутів (змінних) і методів (функцій), а також відносин між класами, таких як наслідування, асоціація, агрегація та композиція[5].

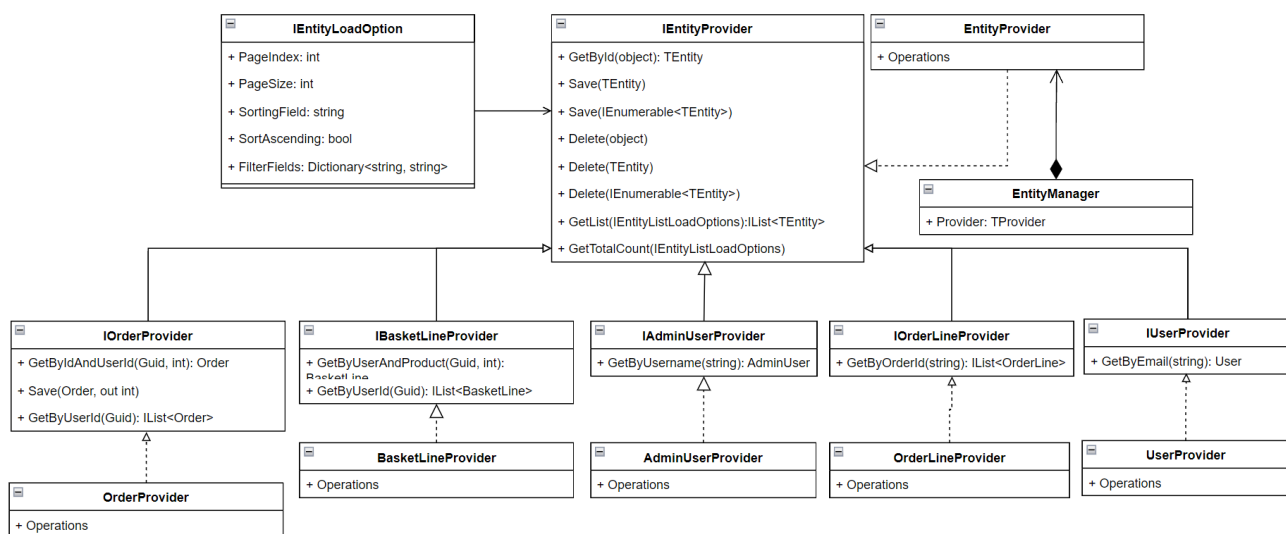


Рис. 2.2 – Діаграма класів модуля роботи з даними книжкового інтернет-магазину

Внаслідок проєктування діаграми до веб-орієнтованої системи книжкового інтернет-магазину було включено такі класи та інтерфейси:

- IEntityProvider: загальний інтерфейс для провайдерів, визначає базові методи для роботи з даними, такі як отримання списку записів, видалення, збереження, отримання загальної кількості записів.

- EntityProvider: абстрактний клас, який реалізує інтерфейс IEntityProvider та використовує Generic-підхід – такий клас може взаємодіяти з об'єктами різних класів в одних і тих самих методах. Таким чином забезпечується універсальність класу-провайдера та можливість роботи з різноманітними структурами даних.

- EntityLoadOptions: клас, що визначає список критеріїв для запиту даних, таких як параметри сортування та фільтрації, кількість записів для запиту та сторінка запиту.

- Нащадки класу EntityProvider: визначають додаткові методи для роботи з даними для об'єктів конкретного типу. Наприклад, в класі OrderLineProvider визначено метод для отримання всіх товарів в замовленні за ідентифікатором замовлення.

- EntityManager: клас, який об'єднує роботу з даними через конкретний провайдер та надає інтерфейс для зовнішнього доступу до даних.

2.2 Розробка структури бази даних системи

Маючи загальний алгоритм веб-додатку, що отриманий з попередніх підрозділів, можна перейти до опису структури бази даних та логічну ERD діаграму.

Для реалізації поставленої мети ми скористаємося методом для проєктування БД. Цим підходом до проєктування є метод сутність - зв'язок.

ER-модель, також відома як модель сутність-зв'язок, є моделлю даних, яка служить для опису концептуальних схем предметної області. Ця модель використовується під час високорівневого проєктування баз даних та дозволяє ідентифікувати ключові сутності та встановлювати зв'язки між ними[6].

ER-модель є потужним інструментом для візуалізації структури даних та взаємозв'язків в предметній області. Вона дозволяє відображати основні сутності, такі як об'єкти, людей, місця, події, а також відношення та зв'язки між ними.

Модель сутність-зв'язок визначає наступні етапи проєктування БД:

1. визначення сутностей;
2. визначення зв'язків;
3. визначення атрибутів;
4. визначення ключів сутностей;
5. визначення ступеня зв'язку;

		Коренівський М.О.			ІПЗ.КР.Б-121-23-ПЗ	Арк.
		Морозов А.В.				26
Змн.	Арк.	№ докум.	Підпис	Дата		

6. визначення класу приналежності.

На рисунку 2.3 зображено діаграму «сутність - зв'язок» на логічному рівні системи.

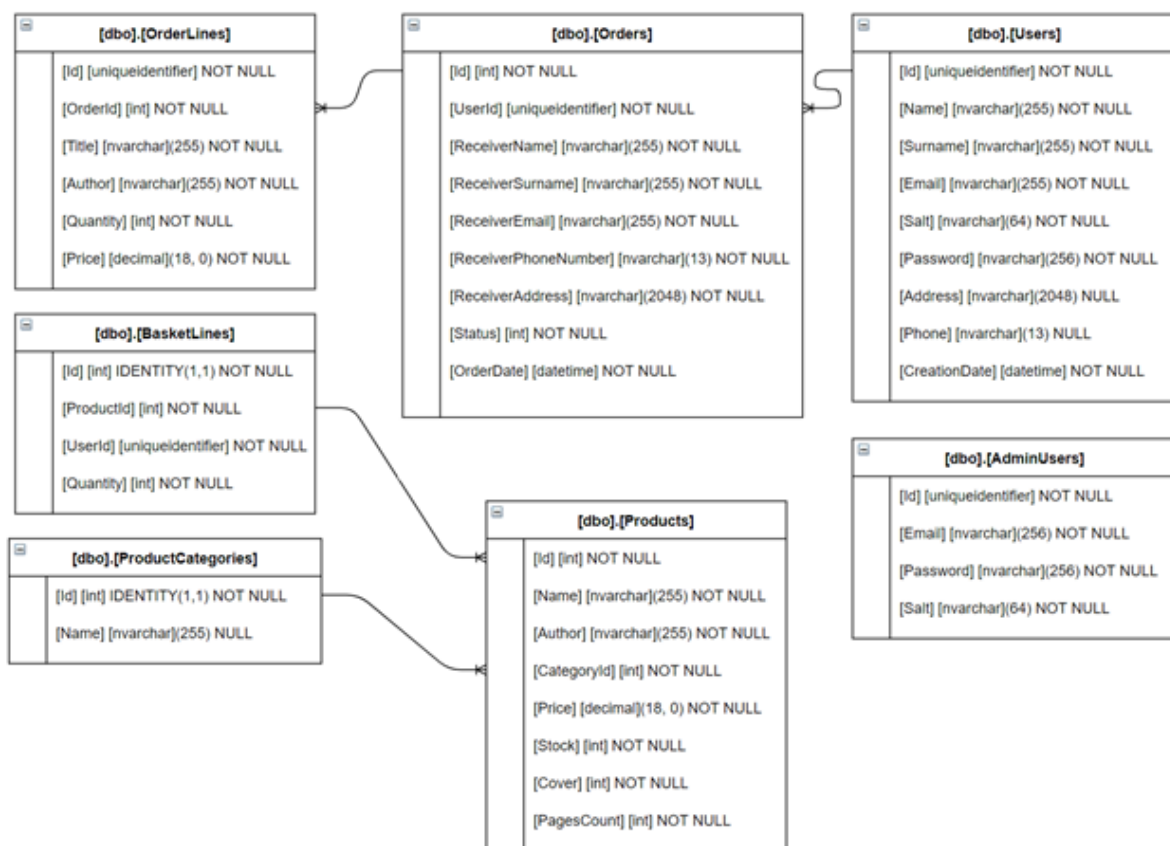


Рис. 2.3 – Діаграма «сутність-зв'язок» (логічний рівень) системи

В результаті проектування бази даних книжкового інтернет-магазину до неї було включено такі таблиці:

- AdminUsers
- Users
- ProductCategories
- Products
- BasketLines
- Orders
- OrderLines

Для збереження даних про адміністраторів призначена таблиця «AdminUsers». В дану таблицю записується вся необхідна інформація про

користувачів-адміністраторів. До неї належать відомості про логін, пароль та «сіль». Структуру таблиці наведено нижче:

Таблиця 2.1

Структура таблиці «AdminUsers»

Назва	Тип даних	ПК	РК	Опис поля
Id	UNIQUEIDENTIFIER	+	-	Ідентифікатор
Email	NVARCHAR(256)	-	-	Email-адреса (максимальна кількість символів 256)
Password	NVARCHAR(256)	-	-	Пароль(максимальна кількість символів 256)
Salt	NVARCHAR(64)	-	-	«Сіль» до паролю (максимальна кількість символів 256)

Для збереження даних про користувачів магазину призначена таблиця «Users». В дану таблицю записується вся необхідна інформація про користувачів. До неї належать відомості про логін, пароль, «сіль», прізвище, ім'я, адресу, номер телефону, дату реєстрації. Структуру таблиці наведено нижче:

Структура таблиці «Users»

Назва	Тип даних	ПК	РК	Опис поля
Id	UNIQUEIDENTIFIER	+	-	Ідентифікатор
Email	NVARCHAR(256)	-	-	Email-адреса (максимальна кількість символів 256)
Password	NVARCHAR(256)	-	-	Пароль(максимальна кількість символів 256)
Salt	NVARCHAR(64)	-	-	«Сіль» до паролю (максимальна кількість символів 256)
Name	NVARCHAR(255)	-	-	Ім'я (максимальна кількість символів 255)
Surname	NVARCHAR(255)	-	-	Прізвище (максимальна кількість символів 255)
Address	NVARCHAR(2048)	-	-	Адреса (максимальна кількість символів 2048)
Phone	NVARCHAR(13)	-	-	Номер телефону (максимальна кількість символів 13)
CreationDate	DATETIME	-	-	Дата створення облікового запису

Для збереження даних про користувачів категорії товарів призначена таблиця «ProductCategories». В дану таблицю записується вся необхідна інформація про категорії товарів, а саме їх пари ідентифікатор-назва. Структуру таблиці наведено нижче:

		Коренівський М.О.			ІПЗ.КР.Б-121-23-ПЗ	Арк.
		Морозов А.В.				29
Змн.	Арк.	№ докум.	Підпис	Дата		

Структура таблиці «ProductCategories»

Назва	Тип даних	ПК	РК	Опис поля
Id	INT	+	-	Ідентифікатор
Name	NVARCHAR(255)	-	-	Назва (максимальна кількість символів 255)

Для збереження даних про товари призначена таблиця «Products». В дану таблицю записується вся необхідна інформація про товари магазину. До неї належать відомості про назву, автора, категорію, ціну, кількість на складі, обгортку, кількість сторінок.

Структуру таблиці наведено нижче:

Структура таблиці «Products»

Назва	Тип даних	ПК	РК	Опис поля
Id	INT	+	-	Ідентифікатор
Name	NVARCHAR(255)	-	-	Назва (максимальна кількість символів 255)
Author	NVARCHAR(255)	-	-	Автор (максимальна кількість символів 255)
CategoryId	INT	-	+	Код категорії (зв'язок з таблицею «ProductCategories»)
Price	DECIMAL	-	-	Ціна
Stock	INT	-	-	Кількість на складі
Cover	INT	-	-	Код типу обкладинки
PagesCount	INT	-	-	Кількість сторінок

Для збереження даних про кошики користувачів магазину призначена таблиця «BasketLines». В дану таблицю записується вся необхідна інформація про товари у кошиках користувачів інтернет-магазину. До неї належать

відомості про код товару, код користувача та кількість одиниць товару у кошику.
Структуру таблиці наведено нижче:

Таблиця 2.5

Структура таблиці «BasketLines»

Назва	Тип даних	ПК	РК	Опис поля
Id	INT	+	-	Ідентифікатор
ProductId	INT	-	+	Код товару (зв'язок з таблицею «Products»)
UserId	UNIQUEIDENTIFIER	-	+	Код користувача (зв'язок з таблицею «Users»)
Quantity	INT	-	-	Кількість товару в кошику

Для збереження даних про замовлення призначена таблиця «Orders». В дану таблицю записується вся необхідна інформація про замовлення в інтернет-магазині. До неї належать відомості про код користувача, ім'я одержувача, прізвище одержувача, email-адресу одержувача, номер телефону одержувача, адресу одержувача, статус замовлення, дату замовлення. Структуру таблиці наведено нижче:

Таблиця 2.6

Структура таблиці «Orders»

Назва	Тип даних	ПК	РК	Опис поля
Id	INT	+	-	Ідентифікатор
UserId	UNIQUEIDENTIFIER	-	-	Код отримувача (зв'язок з таблицею «Users»)
ReceiverName	NVARCHAR(256)	-	-	Ім'я отримувача (максимальна кількість символів 256)

Продовження таблиці 2.6

ReceiverSurname	NVARCHAR(255)	-	-	Прізвище отримувача (максимальна кількість символів 255)
ReceiverEmail	NVARCHAR(255)	-	-	Е-mail адреса отримувача (максимальна кількість символів 255)
ReceiverPhoneNumber	NVARCHAR(13)	-	-	Номер телефону отримувача (максимальна кількість символів 13)
ReceiverAddress	NVARCHAR(2048)	-	-	Адреса отримувача (максимальна кількість символів 2048)
Status	INT	-	-	Код статусу замовлення
OrderDate	DATETIME	-	-	Дата замовлення

Для збереження даних про товари в замовленнях призначена таблиця «OrderLines». В дану таблицю записується вся необхідна інформація про товари, які містяться в замовленнях. До неї належать відомості про код замовлення, назву товару, автора, кількість та ціну. Структуру таблиці наведено нижче:

		Коренівський М.О.			ІПЗ.КР.Б-121-23-ПЗ	Арк.
		Морозов А.В.				32
Змн.	Арк.	№ докум.	Підпис	Дата		

Структура таблиці «OrderLines»

Назва	Тип даних	ПК	РК	Опис поля
Id	UNIQUEIDENTIFIER	+	-	Ідентифікатор
OrderId	INT	-	+	Код замовлення(зв'язок з таблицею «Orders»)
Title	NVARCHAR(256)	-	-	Назва товару(максимальна кількість символів 256)
Author	NVARCHAR(256)	-	-	Автор (максимальна кількість символів 256)
Quantity	INT	-	-	Кількість товару
Price	DECIMAL	-	-	Ціна одиниці товару

В залежності від ролі користувача, додаток буде виконувати різні функції. Тому необхідно визначити групи користувачів і надати їм відповідні права доступу для виконання різних дій. Розглянемо категорії користувачів, які можуть використовувати базу даних книжкового інтернет-магазину: гості, авторизовані користувачі та адміністратори.

Для організації роботи книжкового інтернет-магазину адміністратори мають найвищий рівень доступу до даних. Вони мають можливість переглядати інформацію про категорії товарів, товари, замовлення та облікові записи клієнтів. Окрім перегляду, адміністратори можуть змінювати дані, видаляти записи та додавати нові. З цією метою рекомендується створити окрему роль під назвою "Адміністратор", яка буде мати доступ до відповідних даних. Детальний перелік об'єктів бази даних, до яких роль "Адміністратор" має доступ, наведено в таблиці 2.8. У цій таблиці представлено можливі дії, які може виконувати адміністратори: 0 - немає доступу, 1 - перегляд даних, 2 - редагування, 3 - видалення, 4 - створення, 5 - повний доступ.

		Коренівський М.О.			ІПЗ.КР.Б-121-23-ПЗ	Арк.
		Морозов А.В.				33
Змн.	Арк.	№ докум.	Підпис	Дата		

Таблиця 2.8

Матриця доступу для ролі «Адміністратор»

	Користувачі	Категорії товарів	Товари	Замовлення	Товари в кошику	Товари замовлення
Адміністратор	5	5	5	2	0	0

Наступною роллю є авторизований користувач. Він має можливості переглядати дані про товари та їх категорії, а також він має розширені можливості щодо замовлень та товарів у кошику. Права доступу для авторизованого користувача до об'єктів бази даних наведено у таблиці 2.9.

Таблиця 2.9

Матриця доступу для ролі «Авторизований користувач»

	Користувачі	Категорії товарів	Товари	Замовлення	Товари в кошику	Товари замовлення
Авторизований користувач	0	1	1	4	5	4

Останній тип користувачів у системі називається «Гість». Він представляє собою користувача, який вперше потрапив на сайт та не є зареєстрованим. Тому для такої ролі було визначено заборонити доступ до будь-яких даних. Зважаючи на це, матриця доступу для ролі «Гість» буде мати вигляд, як показано в таблиці 2.10.

Таблиця 2.10

Матриця доступу для ролі «Гість»

	Користувачі	Категорії товарів	Товари	Замовлення	Товари в кошику	Товари замовлення
Гість	0	0	0	0	0	0

2.3 Проєктування та реалізація алгоритмів роботи системи

Під час проєктування системи було встановлено, виконано і перевірено обов'язкові характеристики її структури. У попередньому розділі, шляхом аналізу вимог, були визначені цілі та завдання, які система має виконувати, а

		Коренівський М.О.			ІПЗ.КР.Б-121-23-ПЗ	Арк.
		Морозов А.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		34

також обрані інструменти для її реалізації. На даному етапі, за допомогою діаграми діяльності, необхідно описати послідовність дій для кожного прецеденту та розробити алгоритм роботи системи.

Після переходу на сайт, користувачі інтернет-магазину матимуть доступ до наступних функцій:

- реєстрація на сайті;
- авторизація на сайті;
- перегляд каталогу товарів;
- пошук, фільтрування та сортування товарів;
- перегляд детальної інформації про товар;
- додавання товарів в кошик;
- перегляд товарів в кошику;
- оформлення замовлення;
- проведення онлайн-оплати замовлення;
- перегляд історії замовлень;
- перегляд детальної інформації про замовлення.

В свою чергу, адміністратор книжкового інтернет-магазину матиме змогу виконувати наступні дії:

- перегляд, додавання, редагування та видалення категорій; товарів, товарів та облікових записів користувачів;
- перегляд та зміна статусу замовлень;
- пошук, фільтрування та сортування даних книжкового інтернет-магазину.

Виконання вищевказаних функцій має відповідати всім вимогам функціонування системи. Для цілісного зображення роботи системи та її функціональних можливостей доречно використати діаграму активності та станів, яку зображено на рисунку 2.4.

Діаграма активності - це тип діаграми, який використовується для моделювання послідовності дій або процесів у системі. Вона демонструє потік управління та взаємодії між різними елементами системи, такими як дії,

		Коренівський М.О.			ІПЗ.КР.Б-121-23-ПЗ	Арк.
		Морозов А.В.				35
Змн.	Арк.	№ докум.	Підпис	Дата		

прийняття рішень, розділові вузли, умовні вирази та інші компоненти. Діаграма активності дозволяє візуалізувати робочі процеси, бізнес-логіку або поведінку системи з різних перспектив. Вона може бути використана для моделювання процесу роботи програми, взаємодії між користувачем і системою, або для аналізу послідовності подій у складному алгоритмі[5].

Відповідно до діаграми активності, наведеної на рисунку 2.4, для входу в систему користувачу спочатку необхідно пройти автентифікацію та авторизацію. Відповідно до ролі користувача йому буде надано доступ до адміністративної панелі або до клієнтської частини інтернет-магазину. Після введення логіну та пароля буде проведена валідація цих даних, якщо дані невірні, доступ до системи надано не буде. Якщо авторизація пройшла успішно, користувач зможе виконувати будь-які дії передбачені для його ролі.

Якщо користувач знаходиться в ролі адміністратора, він зможе керувати даними системи книжкового інтернет-магазину, наприклад переглянути нові замовлення та змінити статус в замовленні, які було відправлено клієнту чи скасовано.

В ролі клієнта користувач матиме змогу переглянути каталог та придбати обрані товари, а також подивитись історію власних замовлень та детальну інформацію про них.

		Коренівський М.О.			ІПЗ.КР.Б-121-23-ПЗ	Арк.
		Морозов А.В.				36
Змн.	Арк.	№ докум.	Підпис	Дата		

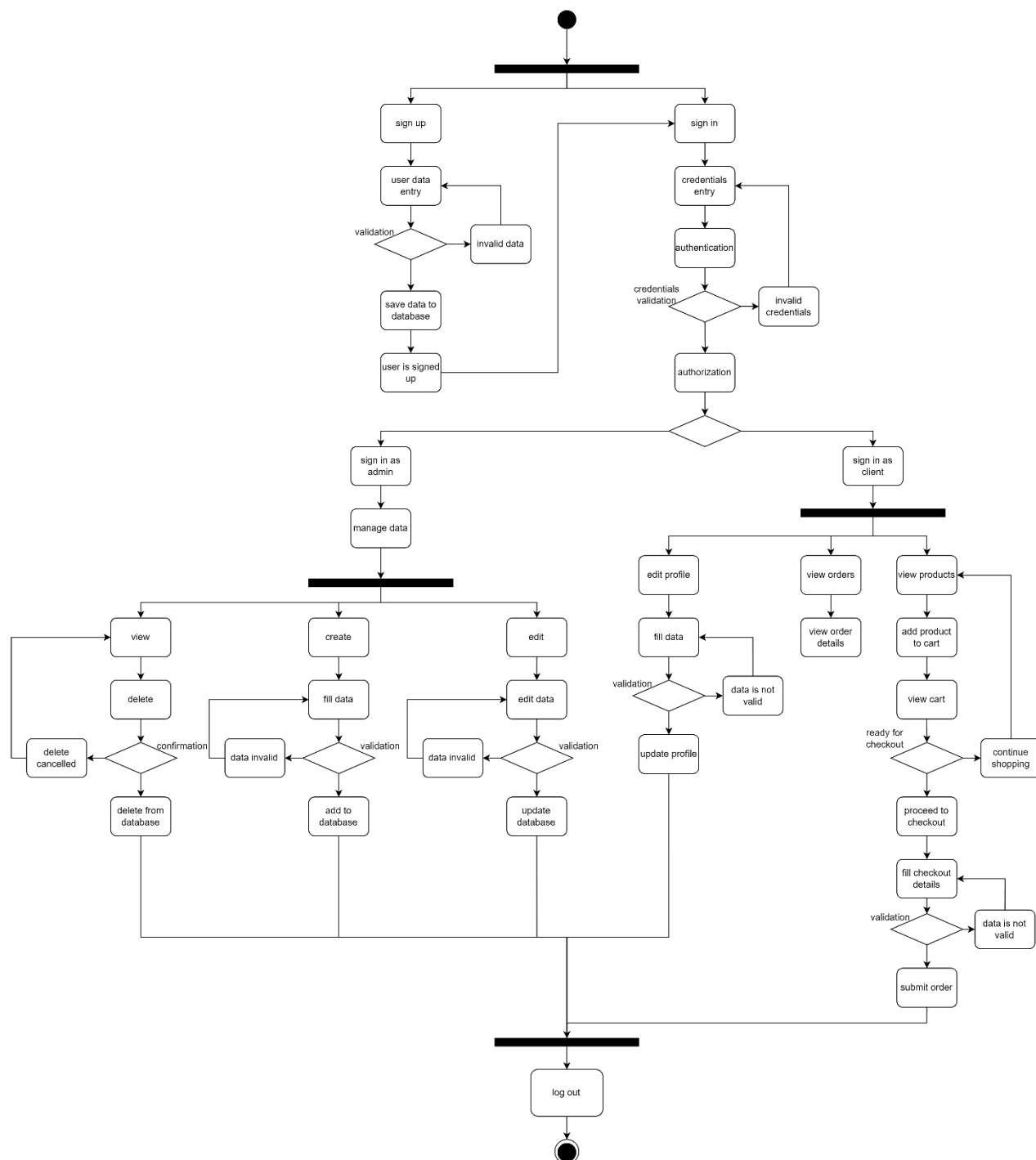


Рис. 2.4 – Діаграма діяльності веб-орієнтованої системи
книжкового інтернет-магазину

Для опису певної діяльності та її складових частин використовується діаграма активності. У веб-додатку діаграма активності використовується для моделювання та опису послідовності дій, які відбуваються в процесі взаємодії користувача з додатком. Діаграма послідовності використовує об'єкти як головні елементи, які представляють логічні сутності в системі і взаємодіють між собою

шляхом обміну повідомленнями. Об'єкти можуть включати не лише компоненти системи, але і акторів, які взаємодіють з системою ззовні.

На діаграмі відображається послідовність виконання повідомлень між об'єктами та акторами в рамках певного сценарію або взаємодії. Кожне повідомлення зображується у вигляді стрілки, що вказує на напрям передачі інформації. Ця діаграма допомагає візуалізувати взаємодію між елементами системи та розуміти послідовність виконання операцій та обміну даними. Діаграму активності зображено на рисунку 2.5.

Послідовний опис процесів, зображених на діаграмі активності:

- адміністратор авторизується в системі;
- адміністратор додає товар;
- адміністратор редагує товар;
- адміністратор видаляє товар;
- не зареєстрований користувач створює обліковий запис;
- зареєстрований користувач авторизується в системі;
- зареєстрований користувач додає товар в кошик;
- зареєстрований користувач переходить на сторінку оформлення замовлення;
- зареєстрований користувач вводить інформацію щодо отримувача замовлення;
- зареєстрований користувач проводить оплату замовлення;
- зареєстрований користувач переглядає історію замовлень;
- адміністратор змінює статус замовлення;
- адміністратор виходить з облікового запису;
- зареєстрований користувач виходить з облікового запису.

		Коренівський М.О.			ІПЗ.КР.Б-121-23-ПЗ	Арк.
		Морозов А.В.				38
Змн.	Арк.	№ докум.	Підпис	Дата		

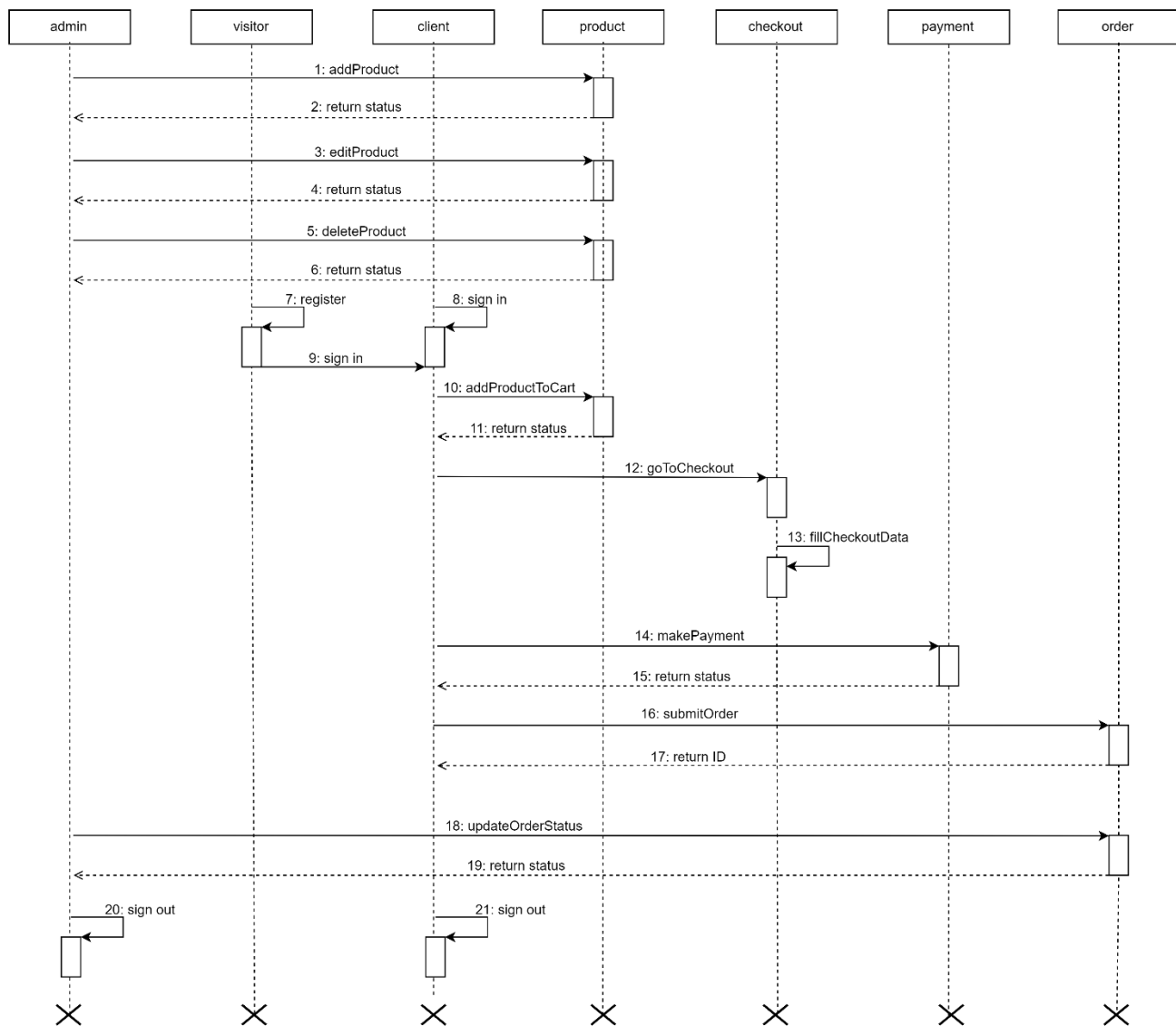


Рис. 2.5 – Діаграма активності користувачів книжкового інтернет-магазину

Раніше представлені діаграми відображають як концептуальні, так і логічні аспекти побудови моделі системи. Логічне представлення відрізняється тим, що воно оперує абстрактними поняттями, які не мають фізичного втілення. Елементи логічного представлення, такі як класи, асоціації, стани та повідомлення, не існують фізично, але відображають статичну структуру системи або динамічні аспекти її поведінки[5].

Для реалізації конкретної фізичної системи необхідно перетворити всі елементи логічного представлення на конкретні матеріальні сутності. Для опису таких реальних сутностей використовується фізичне представлення моделі. Фізичне представлення включає пов'язані фізичні сутності, такі як програмне та апаратне забезпечення, а також персонал, що виконує специфічні завдання.

Фізична система є реально існуючим прототипом моделі системи. Сукупність моделей логічного та фізичного представлень, узгоджені між собою, об'єднуються в повний проєкт програмної системи. Для фізичного представлення моделей систем в мові UML використовуються діаграми реалізації, що містять дві окремі канонічні діаграми: діаграму компонентів та діаграму розгортання.

Діаграма компонентів відображає фізичне представлення системи і відрізняється від інших діаграм тим, що детально описує структуру та взаємозв'язки програмних компонентів системи. Вона дозволяє встановити залежності між компонентами, які можуть бути представлені вихідним, бінарним або виконуваним кодом. В багатьох середовищах розробки модуль або компонент відповідає окремому файлу. Головними графічними елементами діаграми компонентів є самі компоненти, інтерфейси та залежності між ними.

Після побудови діаграми компонентів отримується загальне представлення архітектури системи, де видно, як компоненти взаємодіють між собою. На рисунку 2.6 зображено діаграму компонентів системи книжкового інтернет-магазину.

Систему спроектовано у відповідності до архітектурного патерну MVC (Model-View-Controller). Серверна частина містить в собі моделі, що відповідають за дані системи, та контролери, які керують поведінкою системи. В свою чергу, клієнтська частина складається з представлень, що відповідають за відображення даних системи користувачеві. Запити з клієнтської частини надсилаються на сервер через RESTful сервіс, де потім вбудований механізм маршрутизації ASP.NET RouteHandler направляє запити до відповідних контролерів на обробку.

Запити серверу до модуля БД опрацьовуються конектором IdatabaseAccessSession, що інкапсулює логіку роботи із SQL-запитами.

		Коренівський М.О.			ІПЗ.КР.Б-121-23-ПЗ	Арк.
		Морозов А.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		40

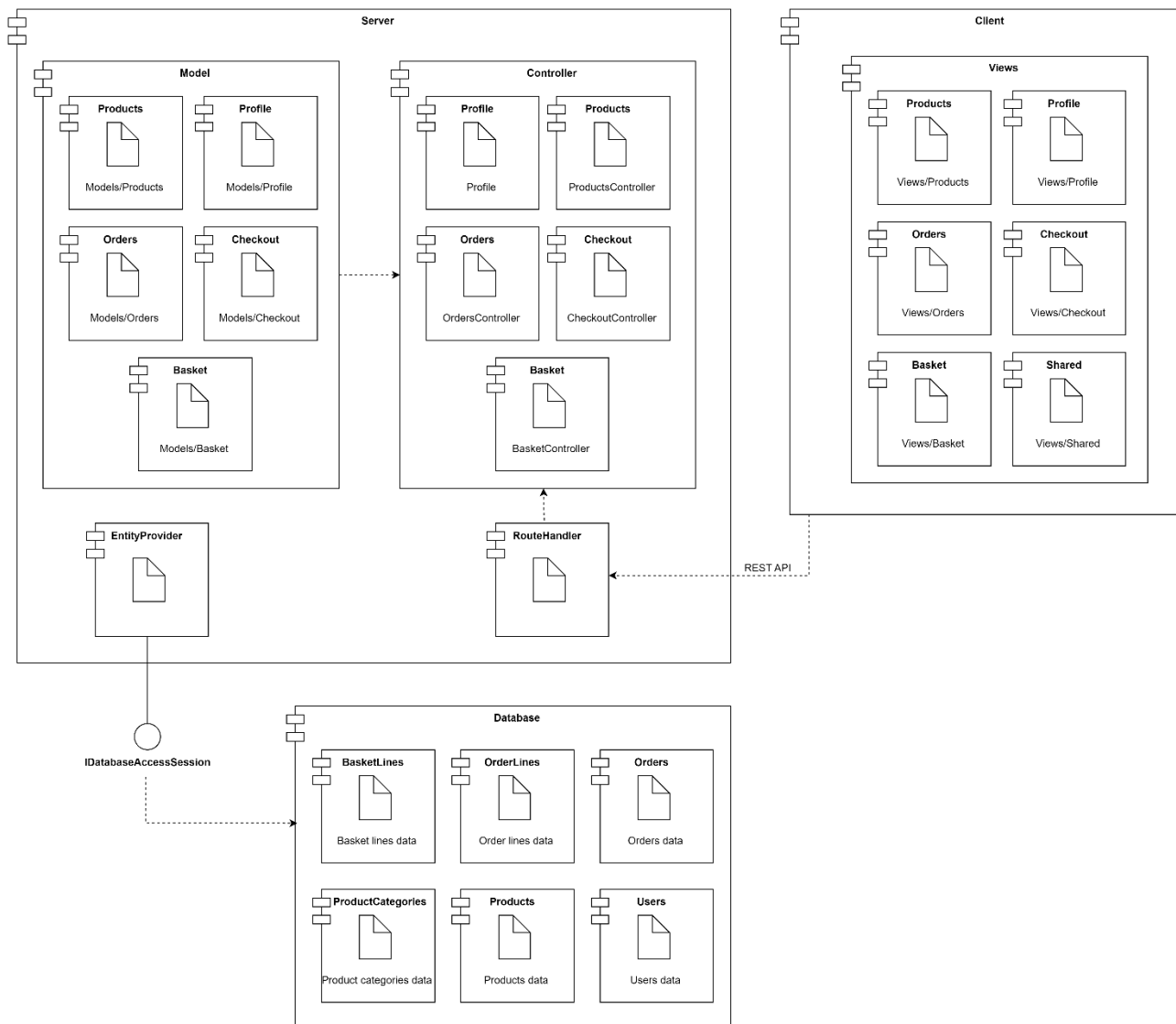


Рис. 2.6 – Діаграма взаємодії компонентів системи

2.4 Реалізація додатку веб-орієнтованої системи книжкового інтернет-магазину

Алгоритм роботи веб-сайту, який було спроектовано в попередніх кроках, надав багато вказівок та дій, які потрібно виконати та реалізувати у веб-додатку. Для забезпечення чітко окресленої структури та компактності, основний функціонал та методи програми було розбито на окремі підпроекти. Першим кроком є проектування структури файлів. На рисунку 2.7 представлено загальну структуру проекту.

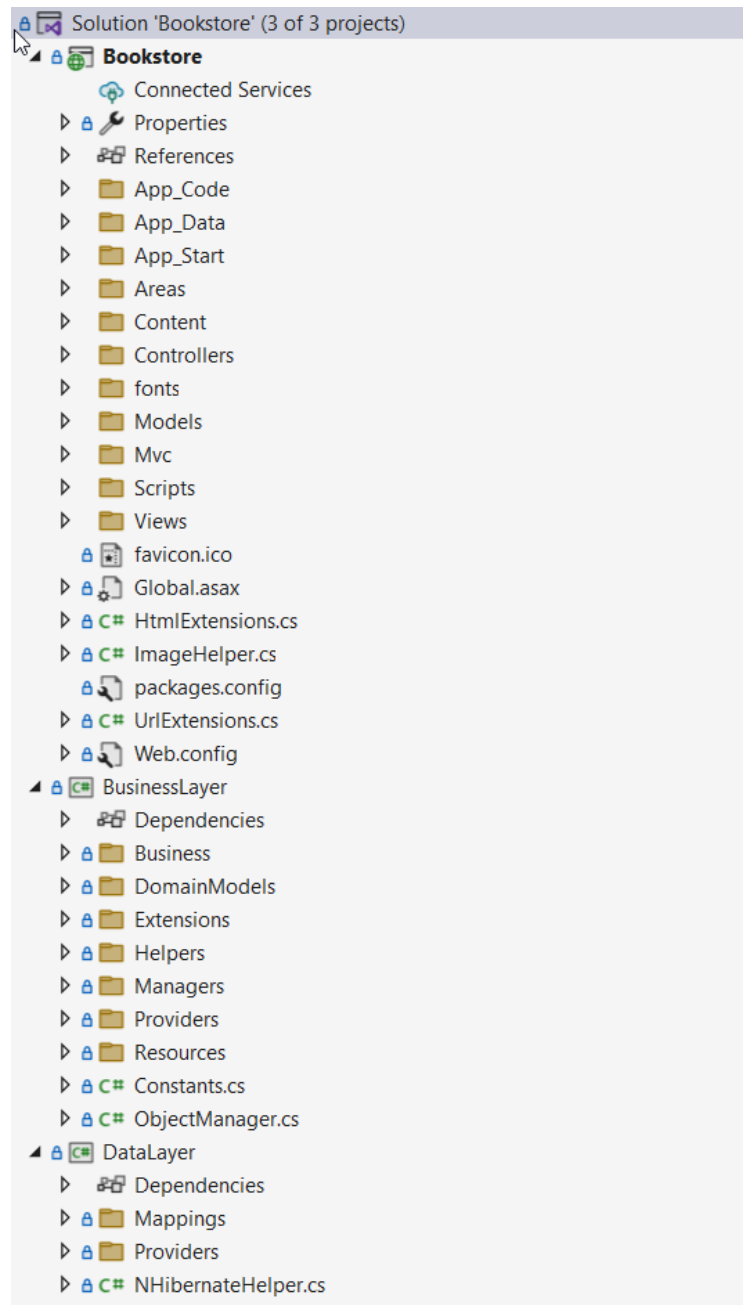


Рис. 2.7 – Структура проєкту

Папки та файли зображені на рисунку 2.7. мають таке призначення:

- Проєкт Bookstore – MVC проєкт.
 - Controllers – папка, що містить контролери;
 - Content – папка, що містить зображення для додатку;
 - Models – папка, що містить класи моделей;
 - Views – папка, що містить представлення;
 - Scripts – папка, що містить JavaScript скрипти.

- Проєкт BusinessLayer – бібліотека класів, що містить основний функціонал додатку, реалізацію логіки роботи інтернет-магазину.
- Проєкт DataLayer – бібліотека класів, що містить функціонал роботи із даними, формування та обробку SQL запитів до БД.

Для спрощення та полегшення взаємодії між об'єктно-орієнтованою моделлю програмного забезпечення і реляційною базою даних було використано ORM-рішення NHibernate. За допомогою Nhibernate можна інкапсулювати процес підключення до БД та надсилання SQL-запитів в окремому модулі та описати функціонал за допомогою API для мови програмування С# – це дає змогу уникнути написання прямих SQL запитів[24].

Нижче наведено фрагмент коду створення сесії використання БД за допомогою ORM-рішення Nhibernate.

```
public class NHibernateHelper
{
    private static ISessionFactory sessionFactory;

    static NHibernateHelper()
    {
        var configuration = new Configuration();
        configuration.Configure();
        sessionFactory = configuration.BuildSessionFactory();
    }

    public static ISession OpenSession()
    {
        var session = sessionFactory.OpenSession();
        return session;
    }
}
```

Для коректної роботи із БД за посередництва ORM-рішення Nhibernate необхідно описати відповідність між таблицями в БД та класами С#, які представлятимуть записи з таблиць БД. Для досягнення цієї мети відповідності записуються у спеціальний файл «mapping» у форматі XML[20]. Нижче наведено приклад проекції таблиці AdminUsers на клас AdminUser.

```
<class table="AdminUsers" name="AdminUser" lazy="false">
  <id name="Id" type="Guid" column="Id">
    <generator class="guid"/>
  </id>
  <property name="Email" type="string"/>
  <property name="Password" type="string"/>
  <property name="Salt" type="string"/>
</class>
```

		Коренівський М.О.			ІПЗ.КР.Б-121-23-ПЗ	Арк.
		Морозов А.В.				43
Змн.	Арк.	№ докум.	Підпис	Дата		

Порядок, визначення та реалізацію прав доступу забезпечує інструмент RBAC, що є механізмом керування доступом на основі ролей.

Основні концепції RBAC:

- Роль є набором дозволів (permissions)
- Одна роль може бути призначена одному чи декільком користувачам
- Ролі та дозволи можуть формувати ієрархію:
 - Роль може містити інші ролі та дозволи
 - Дозволи можуть мати інші дозволи
 - Ролі та дозволи можуть зв'язуватись з правилами, що накладають додаткові обмеження на них

Для реалізації механізму RBAC було описано провайдер ролей, код якого зображено нижче:

```
public class AppRoleProvider : RoleProvider
{
    public override string[] GetRolesForUser(string username)
    {
        var roles = new string[] { };
        var user = WebsiteManager.Users.GetByEmail(username);
        if (user != null)
            roles = new string[] { Constants.Roles.User };
        var adminUser = WebsiteManager.AdminUser.GetByUsername(username);
        if (adminUser != null)
            roles = new string[] { Constants.Roles.Admin };

        return roles;
    }

    public override bool IsUserInRole(string username, string roleName)
    {
        if (roleName.Equals(Constants.Roles.User))
        {
            var user = WebsiteManager.Users.GetByEmail(username);
            return user != null;
        }
        else if (roleName.Equals(Constants.Roles.Admin))
        {
            var user = WebsiteManager.AdminUser.GetByUsername(username);
            return user != null;
        }

        return false;
    }
}
```

		Коренівський М.О.			ІПЗ.КР.Б-121-23-ПЗ	Арк.
		Морозов А.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		44

Щоб забезпечити єдиний підхід до роботи з БД, а також підвищити універсальність певних модулів програмної системи, було описано спільний інтерфейс для всіх типів провайдерів, що працюватимуть з БД:

```
public interface IEntityProvider<TEntity>
{
    TEntity GetById(object id);

    void Save(TEntity entity);

    void Save(IEnumerable<TEntity> entities);

    void Delete(object id);

    void Delete(TEntity entity);

    void Delete(IEnumerable<TEntity> entities);

    IList<TEntity> GetList(IEntityListLoadOptions loadOptions);

    int GetTotalCount(IEntityListLoadOptions loadOptions);
}
```

Generic-типи в мові програмування C# дозволяють класам та методам працювати одночасно з різними типами даних, не змінюючи при цьому функціональний підхід та не дублюючи програмний код[2]. В прикладі коду вище таким типом оголошено «TEntity» – тип сутності з БД.

Також Generic-типи було використано для проектування класу параметрів фільтрування та сортування. Оскільки система книжкового інтернет-магазину працює з різними типами об'єктів, виникає потреба в спільному підході до їх фільтрування та сортування під час відображенні на сторінці. Клас EntityListLoadOptions містить в собі параметри фільтрування та сортування, а також параметри, що забезпечують отримання даних порціями – індекс поточної сторінки та кількість записів на одній сторінці:

```
public class EntityListLoadOptions : IEntityListLoadOptions
{
    public int PageIndex { get; set; }

    public int PageSize { get; set; }

    public string SortingField { get; set; }

    public bool SortAscending { get; set; }

    public IDictionary<string, string> FieldFilters { get; set; }

    public EntityListLoadOptions()
    {
        this.FieldFilters = new Dictionary<string, string>();
    }
}
```

		Коренівський М.О.			ІПЗ.КР.Б-121-23-ПЗ	Арк.
		Морозов А.В.				45
Змн.	Арк.	№ докум.	Підпис	Дата		

З метою зменшення залежностей між компонентами системи та полегшення тестування шляхом передачі залежностей ззовні, при проєктуванні системи було використано принцип Dependency Injection. Використання такого підходу має на меті декілька цілей:

- Зменшення залежностей: Dependency Injection допомагає зменшити залежності між компонентами системи, тому що компонент не самостійно створює свої залежності, а отримує їх ззовні[10]. Це полегшує зміни та розширення системи, оскільки зміни в одному компоненті не потребують змін у всіх його залежностях.
- Забезпечення тестованості: Dependency Injection дозволяє легко використовувати підроблені об'єкти під час тестування. Це допомагає створювати автономні тести для окремих компонентів системи без необхідності включення реальних залежностей.
- Покращення повторного використання: Завдяки Dependency Injection компоненти стають більш незалежними та зручно перевикористовуватись в різних контекстах. Це дозволяє знизити дублювання коду та прискорює розробку нових функцій.

Для реалізації підходу Dependency Injection та простоти його застосування в різних модулях системи було описано клас ObjectManager, який включав усі необхідні для цього функції:

```
public static class ObjectManager
{
    private static IUnityContainer container;
    public static void Initialize() => container = new UnityContainer();

    public static void RegisterType<I, T>() where T : I =>
        container.RegisterType<I, T>(new InjectionConstructor());

    public static void RegisterTypeSingleton<I, T>() where T : I =>
        container.RegisterType<I, T>(null, new ContainerControlledLifetimeManager(), new
        InjectionConstructor());

    public static T GetInstance<T>() => container.Resolve<T>();

    public static Type GetType<T>() => container.Resolve<T>().GetType();
}
```

Під час розробки модуля панелі адміністратора та функціоналу створення та редагування товару виникла потреба в масштабуванні завантажуваних зображень, оскільки на сторінці каталогу, сторінці товару та сторінці кошику

		Коренівський М.О.			ІПЗ.КР.Б-121-23-ПЗ	Арк.
		Морозов А.В.				46
Змн.	Арк.	№ докум.	Підпис	Дата		

необхідні зображення різних розмірів. Розглядалася можливість масштабування зображень засобами CSS, але в такому випадку швидкість завантаження веб-сторінки була б суттєво меншою, адже спочатку зображення в повному розмірі надходило з серверу і лише потім масштабувалося в браузері користувача.

Задля вирішення цієї проблеми було додано допоміжний клас, що працював із зображеннями та надав змогу змінити їх розмір при збереженні. Для покриття усіх можливих сценаріїв використання зображення було вирішено працювати з трьома розмірами. Наступний фрагмент коду відповідає саме за зміну розміру зображення для його подальшого збереження у файловій системі на сервері[16].

```
string imagesPath = $"Content/images/products/{imageFolder}";

var destRect = new Rectangle(0, 0, width, height);
var destImage = new Bitmap(width, height);

destImage.SetResolution(image.HorizontalResolution,
image.VerticalResolution);

using (var graphics = Graphics.FromImage(destImage))
{
    graphics.CompositingMode = CompositingMode.SourceCopy;
    graphics.CompositingQuality = CompositingQuality.HighQuality;
    graphics.InterpolationMode = InterpolationMode.HighQualityBicubic;
    graphics.SmoothingMode = SmoothingMode.HighQuality;
    graphics.PixelOffsetMode = PixelOffsetMode.HighQuality;

    using (var wrapMode = new ImageAttributes())
    {
        wrapMode.SetWrapMode(WrapMode.TileFlipXY);
        graphics.DrawImage(image, destRect, 0, 0, image.Width,
image.Height, GraphicsUnit.Pixel, wrapMode);
    }
}

destImage.Save(AppDomain.CurrentDomain.BaseDirectory+
$"\\{imagesPath}\\{fileName}.jpg", ImageFormat.Jpeg);
```

Засоби ASP.NET, зокрема синтаксис Razor, дозволяють швидко та ефективно будувати представлення, які в свою чергу формують користувацький інтерфейс. За допомогою анотацій даних можна визначити певні характеристики для властивостей моделі, а потім при побудові HTML-сторінки використати ці характеристики для відображення певної інформації, визначення формату полів форми чи проведення валідації даних.

		Коренівський М.О.			ІПЗ.КР.Б-121-23-ПЗ	Арк.
		Морозов А.В.				47
Змн.	Арк.	№ докум.	Підпис	Дата		

Нижче показано фрагмент класу моделі, яка використовується на сторінці створення нового товару:

```
public class ProductInputModel
{
    public int Id { get; set; }

    [Display(Name = "ItemName")]
    [Required(ErrorMessageResourceName = "ValidationError_Required")]
    public virtual string Name { get; set; }

    [Display(Name = "Stock")]
    [Required(ErrorMessageResourceName = "ValidationError_Required")]
    [Range(0, int.MaxValue, ErrorMessageResourceName =
        "ValidationError_GreaterThanZero")]
    public virtual int Stock { get; set; }
}
```

Анотації даних[28], записані в квадратних дужках над властивостями Name (назва товару) та Stock (кількість товару на складі), вказують на те, що обидва ці поля є обов'язковими для заповнення, а також, що кількість товару на складі має бути більшою або дорівнювати нулю.

Синтаксис Razor має широкий набір допоміжних функцій, що спрощують та стандартизують генерацію HTML-сторінки[29]. Наприклад, функція EditorFor дозволяє згенерувати поле у формі, яке відповідає певній властивості C# класу моделі. Функція LabelFor дозволяє згенерувати підпис (тег label) до поля у формі, а функція ValidationMessageFor – валідаційне повідомлення. Проєктування HTML-сторінки з використанням Razor виглядає наступним чином:

```
<div class="form-group">
    @Html.LabelFor(m => m.Stock)
    @Html.EditorFor(m => m.Stock, new { htmlAttributes = new { @class = "form-control" } })
    <div class="invalid-feedback">@Html.ValidationMessageFor(m => m.Stock)</div>
</div>
```

Після опрацювання вищевказаного фрагменту коду буде згенерована наступна HTML-розмітка:

```
<div class="form-group">
    <label for="Stock">В наявності*</label>
    <input class="form-control text-box single-line valid" data-val="true" data-val-range="Це поле має бути додатнім" data-val-range-max="2147483647" data-val-range-min="0" data-val-required="Це обов'язкове поле" id="Stock" name="Stock" type="number" value="58" aria-describedby="Stock-error" aria-invalid="false">
    <div class="invalid-feedback"><span class="field-validation-valid" data-valmsg-for="Stock" data-valmsg-replace="true"></span></div>
</div>
```

		Коренівський М.О.			ІПЗ.КР.Б-121-23-ПЗ	Арк.
		Морозов А.В.				48
Змн.	Арк.	№ докум.	Підпис	Дата		

В цьому фрагменті можна побачити HTML-атрибут «required», який вказує на обов'язковість заповнення поля, а також специфічні валідаційні атрибути – це є результатом використання анотацій даних при описі класу моделі.

Платформа ASP.NET також надає можливість визначити ресурси додатку в окремих файлах для їх подальшого використання. В проєкті файл ресурсів було використано для збереження локалізованих текстів. Усі дані в файлі ресурсів зберігаються в XML-форматі:

```
<data name="ValidationError_NotValidEmailFormat" xml:space="preserve">
  <value>Неправильний формат e-mail адреси</value>
</data>
<data name="ValidationError_NotValidPasswordFormat" xml:space="preserve">
  <value>Пароль має містити мінімум 6 символів, букви та цифри</value>
</data>
```

В прикладі вище наведено збереження валідаційних повідомлень в ресурс-файлі. Кожен текст має ідентифікатор (поле «name»), та власне сам текст. Такий спосіб організації текстів дає змогу створити ще один ресурс-файл з текстами на іншій мові, але зі збереженням ідентифікаторів – в результаті веб-додаток може бути локалізовано для будь-якої мови шляхом зміни лише одного файлу, адже клас для роботи з ресурсами використовує лише їх ідентифікатори:

```
public class TextManager : ITextManager
{
    protected readonly ResourceManager resourceManager;

    public TextManager()
    {
        this.resourceManager = new
        ResourceManager("BusinessLayer.Resources.Texts", Assembly.GetExecutingAssembly());
    }

    public string GetText(string key)
    {
        var text = resourceManager.GetString(key);
        if (text.IsNullOrEmpty())
            return key;

        return text;
    }
}
```

У веб-додатку книжкового інтернет-магазину було реалізовано інтеграцію зі стороннім сервісом електронних платежів LiqPay. За допомогою LiqPay користувач може оплатити замовлення онлайн будь-яким зручним засобом, список підтримуваних засобів оплати містить в собі оплату банківською

		Коренівський М.О.			ІПЗ.КР.Б-121-23-ПЗ	Арк.
		Морозов А.В.				49
Змн.	Арк.	№ докум.	Підпис	Дата		

карткою, GooglePay, ApplePay, оплату через сервіс інтернет-банкінгу Privat24, тощо.

Для інтеграції книжкового інтернет-магазину з сервісом електронних платежів LiqPay було використано JavaScript бібліотеку, яка надається порталом розробників LiqPay. Щоб створити запит на оплату, потрібно надіслати детальні дані про платіж, а також згенерований особистий підпис клієнта банківської системи[27]. Для організації даних в потрібний формат, а також для створення цілісного інтерфейсу взаємодії з LiqPay, було створено допоміжний клас, в якому описані методи для підготовки даних для запиту до платіжної системи. Реалізація класу LiqPayHelper зображена нижче.

```
public static class LiqpayHelper
{
    public static string GenerateData(decimal orderAmount)
    {
        var dataObject = new LiqpayData() { Amount = orderAmount };
        var dataJson = JsonSerializer.Serialize(dataObject);

        return Convert.ToBase64String(Encoding.UTF8.GetBytes(dataJson));
    }

    public static string GenerateSignature(string data)
    {
        string privateKey = "privateKey";

        using (SHA1Managed sha1 = new SHA1Managed())
        {
            var hash = sha1.ComputeHash(Encoding.UTF8.GetBytes(privateKey + data +
privateKey));
            return Convert.ToBase64String(hash);
        }
    }

    protected class LiqpayData
    {
        [JsonPropertyName("public_key")]
        public string PublicKey => "publicKey
        [JsonPropertyName("version")]
        public int Version => 3;
        [JsonPropertyName("amount")]
        public decimal Amount { get; set; }
        [JsonPropertyName("action")]
        public string Action => "pay";
        [JsonPropertyName("currency")]
        public string Currency => "UAH";
        [JsonPropertyName("description")]
        public string Description => "Оплата замовлення в книжковому магазині
yoBooks";
        [JsonPropertyName("order_id")]
        public string OrderId => Guid.NewGuid().ToString();
    }
}
```

		Коренівський М.О.			ІПЗ.КР.Б-121-23-ПЗ	Арк.
		Морозов А.В.				50
Змн.	Арк.	№ докум.	Підпис	Дата		

Статичний метод `GenerateData` відповідає за створення JSON-об'єкту для запиту оплати в системі `LiqPay`, шляхом серіалізації об'єкту класу `LiqpayData`. Клас `LiqpayData` описує властивості та їх назви у форматі JSON та містить основну інформацію про платіж. Статичний метод `GenerateSignature` відповідає за створення підпису користувача платіжної системи шляхом шифрування ключа користувача та даних, отриманих в результаті виконання методу `GenerateData` з їх наступною конвертацією у формат `Base64`.

Під час оформлення замовлення надсилається асинхронний запит на сервер для отримання даних `LiqPay`. Після отримання даних та підпису користувача надсилається запит до системи `LiqPay`. Після проведення платежу та отримання підтвердження від системи `LiqPay` користувача буде перенаправлено на сторінку з інформацією про успішне оформлення замовлення.

Цикл взаємодії із системою онлайн-платежів `LiqPay` наведено нижче.

```
(function () {
    var $page = $('#checkout-page');
    if ($page.length === 0)
        return;

    var $checkoutForm = $("#checkout");
    $checkoutForm.on("submit", function (event) {
        event.preventDefault();
        var url = $checkoutForm.attr('action');
        var formData = $(this).serialize();
        var paymentMethod =
$(this).find('input[name="PaymentInfo.PaymentMethod"]:checked').val();
        if (paymentMethod != "Online")
            handleOrderSubmit(url, formData);
        else {
            var liqpayStatus = null;
            var $liqpayContainer = $('#liqpay-checkout');
            var liqpayData = $liqpayContainer.data('body');
            var liqpaySignature = $liqpayContainer.data('signature');
            LiqPayCheckout.init({
                data: liqpayData,
                signature: liqpaySignature,
                embedTo: "#liqpay-checkout",
                language: "uk",
                mode: "popup"
            }).on("liqpay.callback", function (data) {
                liqpayStatus = data.status;
                handleOrderSubmit(url, formData);
            }).on("liqpay.ready", function (data) {

            }).on("liqpay.close", function (data) {
                if (liqpayStatus == "success")
                    handleOrderSubmit(url, formData);
            });
        }
    });
})();
```

		Коренівський М.О.			ІПЗ.КР.Б-121-23-ПЗ	Арк.
		Морозов А.В.				51
Змн.	Арк.	№ докум.	Підпис	Дата		

```
function handleOrderSubmit(url, data) {
    $.post(url, data, function (data) {
        window.location = data.redirectUrl;
    });
}
```

Висновки до другого розділу

На даному етапі дипломної роботи було проведено аналіз та опис вимог до розроблюваної системи, включаючи мінімальні системні вимоги. За результатами аналізу була створена діаграма варіантів використання, спроектовано загальний алгоритм та розроблено діаграму діяльності.

Також була створена об'єктно-орієнтована модель програмного комплексу, визначено комунікації та послідовність взаємодії об'єктів системи книжкового інтернет-магазину за допомогою відповідних діаграм. Додатково, була розроблена структура бази даних, описана структура файлів проєкту з класами, що реалізують основний функціонал, та наведені частини коду для кращого розуміння ідеї веб-сайту та особливостей використаних технологічних рішень.

В результаті виконаної роботи було успішно створено стабільний та функціональний веб-додаток, який повністю задовольняє всі визначені вимоги та потреби проєкту.

		Коренівський М.О.			ІПЗ.КР.Б-121-23-ПЗ	Арк.
		Морозов А.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		52

РОЗДІЛ 3: ІНТЕРФЕЙС ТА ПОРЯДОК РОБОТИ З ВЕБ-ОРІЄНТОВАНОЮ СИСТЕМОЮ КНИЖКОВОГО ІНТЕРНЕТ-МАГАЗИНУ

3.1 Порядок встановлення та налаштування параметрів системи

Веб-орієнтована система книжкового інтернет-магазину це веб-сайт, відповідно користуватися ним можна з різних пристроїв, як то телефон, планшет, чи ноутбук, так і з різних веб-браузерів. Пристрій для використання системи повинен відповідати наступним мінімальним вимогам:

- Операційна система Windows 7 і вище або MacOS 10.11 і вище.
- Android 6.0 і вище або iOS 8.0 і вище
- Процесор з тактовою частотою 1,2—3,6 ГГц.
- 512 Мб ОЗУ.
- Будь-який браузер.

Розроблений додаток не потребує спеціалізованого забезпечення, додаткових налаштувань або розгортання, відмінного від стандартних для більшості сучасних веб-сайтів на будь-яких операційних системах. Це дозволяє користувачам максимально зручно користуватися сайтом, покращуючи їх досвід.

Продовжуючи роботу з алгоритмами взаємодії компонентів системи, наступним етапом є розробка діаграми розгортання, яка описує архітектуру програмного комплексу і його розташування на фізичних або віртуальних середовищах.

Діаграма розгортання використовується для візуального зображення архітектури системи та розташування її компонентів (програмного забезпечення, апаратного забезпечення, мережевих ресурсів і т.д.) на фізичних або віртуальних середовищах. Діаграма розгортання дозволяє визначити структуру та зв'язки між компонентами системи, а також вказує на характеристики апаратного та мережевого середовища, на якому система буде розгортатися.

Діаграма розгортання включає графічне відображення процесорів, пристроїв, процесів та їх зв'язків. Вона відрізняється від логічних діаграм тим,

		Коренівський М.О.			ІПЗ.КР.Б-121-23-ПЗ	Арк.
		Морозов А.В.				53
Змн.	Арк.	№ докум.	Підпис	Дата		

що охоплює всю систему в цілому, показуючи всі особливості її реалізації. Ця діаграма є заключним етапом у процесі об'єктно-орієнтованого проєктування для конкретної програмної системи, і її розробка зазвичай є останнім кроком у створенні моделі. Результат побудованої схеми можна побачити на рисунку 3.1.

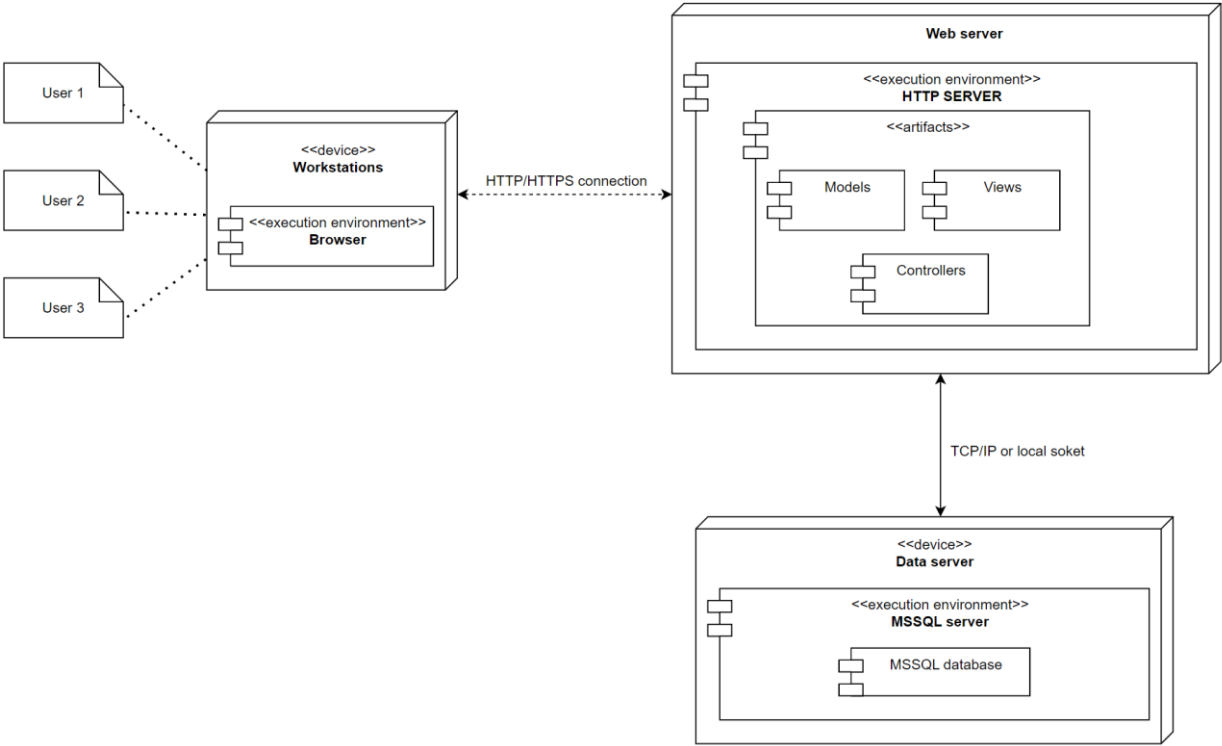


Рис. 3.1 – Діаграма розгортання архітектури програмного комплексу

3.2 Структура інтерфейсу та порядок роботи з веб-орієнтованою системою книжкового інтернет-магазину

Після відкриття веб-сайту неавторизований користувач бачить сторінку входу на веб-сайт – рисунок 3.2. Сторінка входу на сайт містить логотип інтернет-магазину, форму введення імені користувача та паролю, а також кнопки входу на сайт та реєстрації.

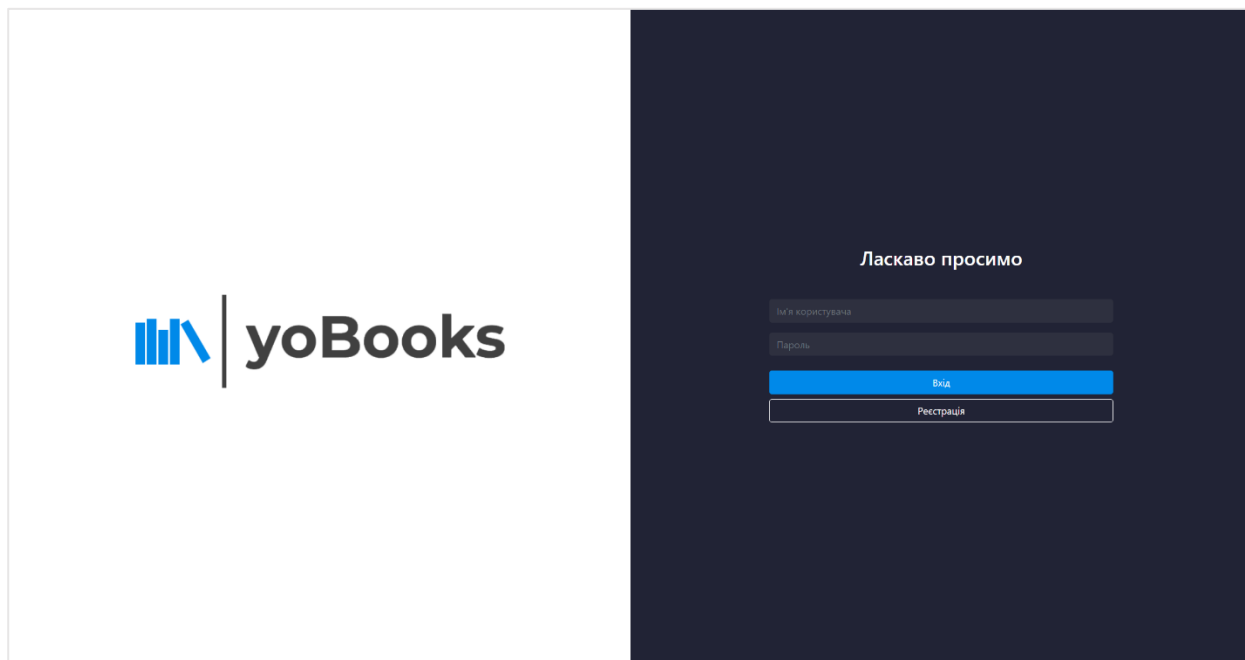


Рис. 3.2 – Сторінка входу на сайт

Якщо користувач не зареєстрований, він має можливість створити обліковий запис натиснувши на кнопку «Реєстрація». Після цього користувач побачить форму реєстрації, яку зображено на рисунку 3.3. У формі реєстрації користувач має вказати наступні дані: e-mail адресу, пароль, своє прізвище та ім'я. Також за бажанням можна вказати адресу та номер телефону, але на етапі реєстрації ці дані не є обов'язковими. Під час введення даних буде здійснена перевірка на заповнення обов'язкових полів, унікальність e-mail адреси, правильність формату мобільного телефону та e-mail адреси. Якщо валідація пройдена успішно, то буде створено новий обліковий запис, а користувача буде переведено на головну сторінку сайту.

		Коренівський М.О.			ІПЗ.КР.Б-121-23-ПЗ	Арк.
		Морозов А.В.				55
Змн.	Арк.	№ докум.	Підпис	Дата		

Реєстрація

E-mail*

Пароль*

Це обов'язкове поле

Підтвердження пароля*

Це обов'язкове поле

Ім'я*

Прізвище*

Номер телефону

Адреса

Створити Скасувати

Рис. 3.3 – Форма реєстрації

Після успішної авторизації користувач потрапляє на головну сторінку магазину, яку зображено на рисунку 3.4.

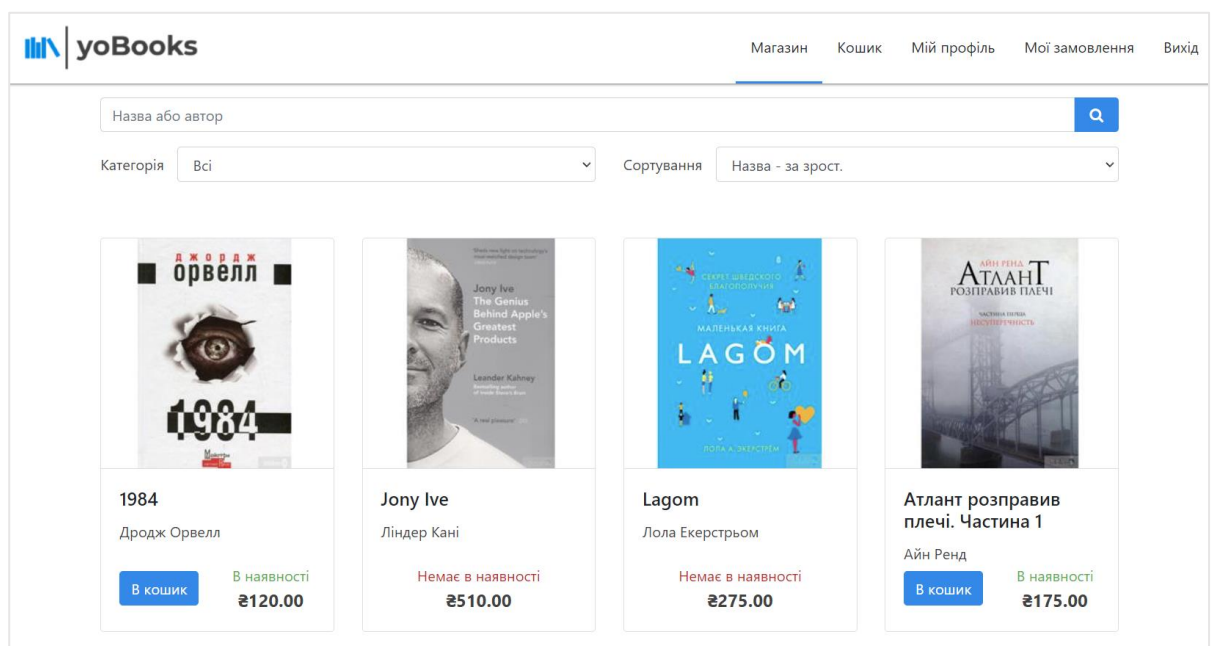


Рис. 3.4 – Головна сторінка магазину

		Коренівський М.О.			ІПЗ.КР.Б-121-23-ПЗ	Арк.
		Морозов А.В.				56
Змн.	Арк.	№ докум.	Підпис	Дата		

Зверху знаходиться меню, в якому наявні наступні посилання: «Магазин», «Кошик», «Мій профіль», «Мої замовлення». Також є посилання на вихід із магазину. Товари в списку відображаються порційно, по 16 найменувань. Внизу є кнопки для переходу між сторінками. Зверху знаходиться поле пошуку товару за автором або назвою книги, фільтрування товарів за категорією та опції сортування. На рисунку 3.5 зображено товари в категорії “Біографія” відсортовані за зростанням ціни.

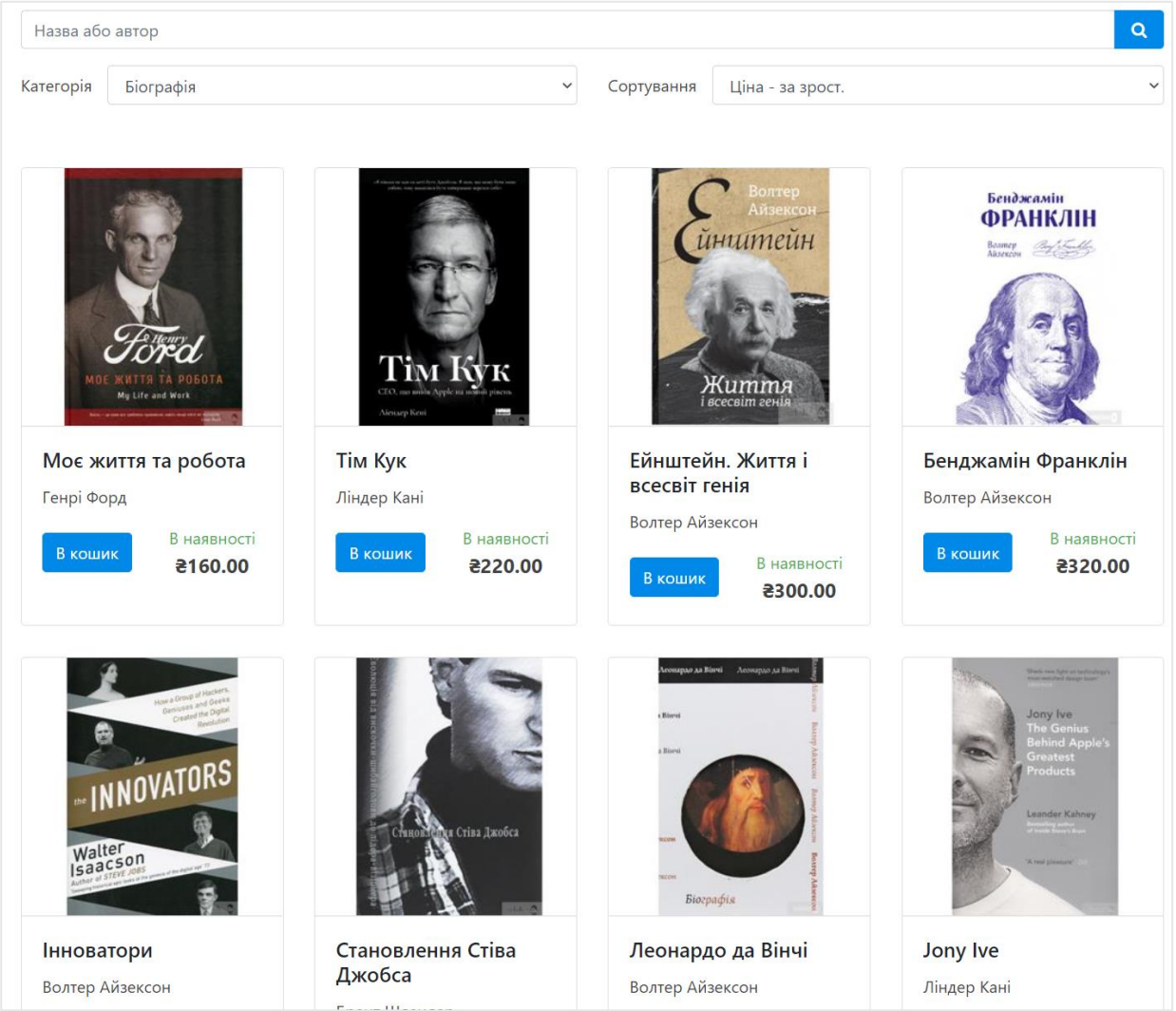


Рис. 3.5 – Результат роботи фільтрів

Кожен елемент з товаром має інформацію про його назву, автора, ціну та наявність. Якщо товар не має зображення, то його буде замінено на шаблон, це показано на рисунку 3.6.

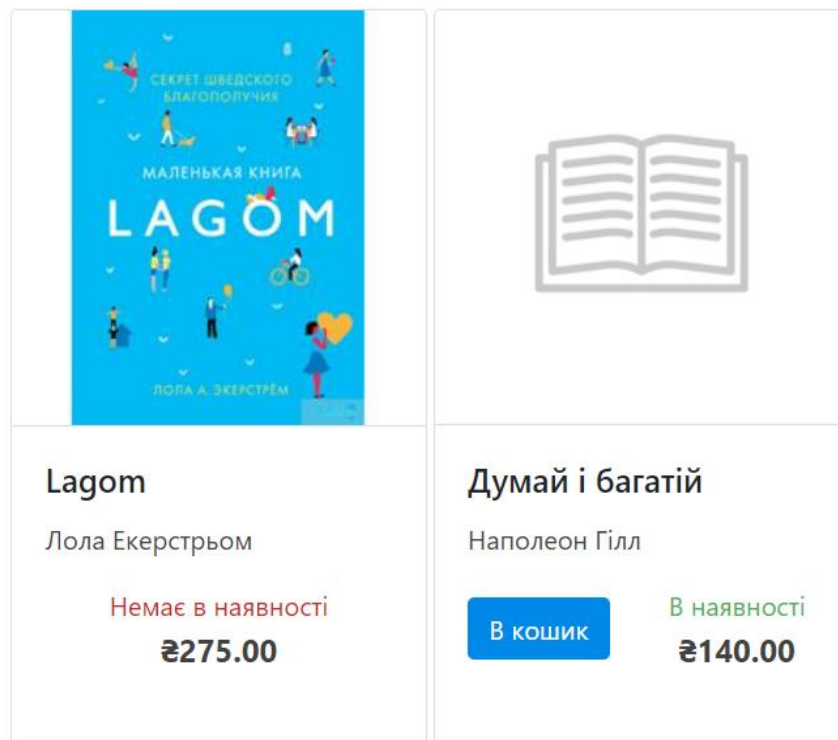


Рис. 3.6 – Блоки товарів

Натиснувши на зображення або назву товару можна перейти сторінки з його детальною інформацією – рисунок 3.7.

yoBooks

МагазинКошикМій профільМої замовленняВихід

1984

Джордж Орвелл

Категорія: Художня література

Кількість сторінок: 320

Обкладинка: Тверда

В наявності **₴120.00**

В кошик

Опис

«1984» (відомий також під назвою «Одна тисяча дев'ятсот вісімдесят четвертий») — антиутопія англійського письменника Джорджа Орвелла, написана 1948 року (у назві роману цифри 4 і 8 переставлені місцями) й опублікована 1949-го.

Роман розповідає історію Вінстона Сміта і його деградації під впливом тоталітарної держави, у якій він живе. Є найвідомішою і найцитованішою антиутопією у літературі і стоїть в одному ряду з «Прекрасним новим світом» (англ. Brave new world) Олдоса Гакслі і «451° за Фаренгейтом» Рея Бредбері.

Перекладений 62 мовами, твір, проте здійснив значний вплив і на саму англійську мову. «1984», його термінологія й автор стали специфічними термінами при обговоренні питання приватності і державної безпеки. Так, термін «орвелліанство» став описувати дії або організації, що нагадують про тоталітарне суспільство, змальоване у романі, а фраза «Старший Брат пильнує за тобою!» (англ. Big Brother is watching you!) почала позначати будь-який акт спостереження, який сприймається як насильницький.

Рис. 3.7 – Сторінка детальної інформації про товар

		Коренівський М.О.			ІПЗ.КР.Б-121-23-ПЗ	Арк.
		Морозов А.В.				58
Змн.	Арк.	№ докум.	Підпис	Дата		

При додаванні товару до кошику у верхньому правому кутку сторінки буде виведено повідомлення про те, що товар успішно додано до кошику, яке зникає з часом – рисунок 3.8.

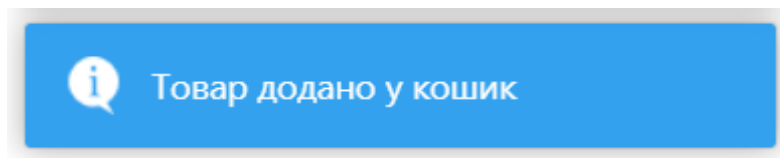


Рис. 3.8. – Повідомлення про додавання товару до кошику

Сторінка кошику має список доданих товарів, їх загальну кількість та вартість. Також передбачена можливість видалення товару з кошику шляхом натискання на кнопку із зображенням смітника. Загальний вигляд сторінки кошика зображено на рисунку 3.9. На рисунку 3.10 зображене повідомлення, що відображається при спробі видалення товару – користувач має підтвердити намір видалити товар, повідомлення унеможлиблює випадкове видалення товару.

Кошик				
Всього товарів: 11		Оформити замовлення		
Загальна вартість: ₴2660.00				
Товар	Кількість	Ціна	Вартість	
 Кобзар Тарас Шевченко	1	₴200.00	₴200.00	
 Зачарована Десна Олександр Довженко	1	₴55.00	₴55.00	
 Людина в пошуках справжнього сенсу Віктор Франкл	1	₴110.00	₴110.00	
 Історія ІКЕА Інгвар Кампрад	1	₴195.00	₴195.00	
 Ейнштейн. Життя і всесвіт генія Волтер Айзексон	7	₴300.00	₴2100.00	

Рис. 3.9 – Сторінка кошика

		Коренівський М.О.			ІПЗ.КР.Б-121-23-ПЗ	Арк.
		Морозов А.В.				59
Змн.	Арк.	№ докум.	Підпис	Дата		

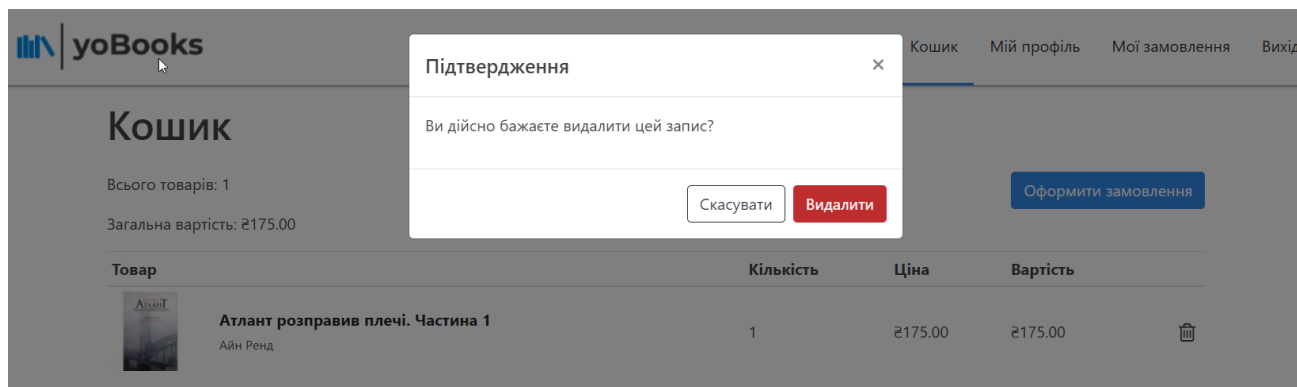


Рис. 3.10 – Повідомлення про видалення товару

Натиснувши на кнопку «Оформити замовлення», користувач переходить на сторінку оформлення замовлення – рисунок 3.11. Там необхідно ввести інформацію про одержувача замовлення, за замовчуванням поля заповнені даними поточного користувача. Також потрібно обрати спосіб оплати.

Рис. 3.11 – Сторінка оформлення замовлення

Як і на сторінці кошика, сторінка оформлення замовлення містить інформацію про товари в замовленні, їх загальну кількість та вартість. Під

		Коренівський М.О.			ІПЗ.КР.Б-121-23-ПЗ	Арк.
		Морозов А.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		60

списком товарів наявні кнопки для повернення до кошику та для завершення оформлення замовлення.

У випадку вибору оплати замовлення онлайн, при натисканні кнопки «Оформити замовлення» користувач побачить вікно для здійснення оплати через сервіс онлайн-платежів LiqPay, яке зображено на рисунку 3.12.

Тестовий режим

До сплати:175.00 UAH

Оплата замовлення в книжковому магазині уoBooks

Сплатити через Приват24

24 Pay

Pay | VISA 1118

КарткаІнший спосіб

Номер картки

Термін дії

CVV2

ММ/YY

...

?

Натискаючи на кнопку «Сплатити», ви підтверджуєте що ознайомлені з переліком інформації про послугу та приймаєте умови публічного договору

Сплатити

Рис. 3.12 – Вікно оплати замовлення

Якщо всі дані вірні, а платіж успішний, користувач завершує оформлення замовлення та переходить на сторінку підтвердження замовлення, яка зображена на рисунку 3.13.

Рис. 3.13 – Сторінка підтвердження замовлення

В розділі «Мій профіль» користувач може змінити дані облікового запису. На формі присутні усі правила валідації, що і на сторінці реєстрації. Сторінку редагування профілю зображено на рисунку 3.14.

Рис. 3.14 – Сторінка редагування профілю

		Коренівський М.О.			ІПЗ.КР.Б-121-23-ПЗ	Арк.
		Морозов А.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		62

Розділ «Мої замовлення» містить історію замовлень користувача. На цій сторінці зображений список замовлень з можливістю фільтрування та сортування. В таблиці замовлень міститься інформація щодо номера замовлення, дати оформлення, загальної кількості та вартості товарів, статусу та способу оплати. Сторінку замовлень зображено на рисунку 3.15. Перейшовши на сторінку деталей замовлення, користувач має можливість отримати детальну інформацію про замовлення, його товари, тощо – рисунок 3.16.

Замовлення

Номер

⋮

Сортування

Дата замовлення - за спад.

▼

Email адреса отримувача

Статус

▼

Дата замовлення

Фільтрувати

Номер	Одержувач	Кількість товарів	Загальна вартість	Дата замовлення	Статус	
000050	Матвій Коренівський	1	₴175.00	11.06.2023 21:51	Нове	
000049	Матвій Коренівський	1	₴175.00	11.06.2023 21:51	Нове	
000048	Матвій Коренівський	5	₴1040.00	08.06.2023 19:57	Нове	
000047	Матвій Коренівський	1	₴175.00	08.06.2023 19:54	Нове	
000046	Матвій Коренівський	1	₴320.00	08.06.2023 19:53	Нове	
000045	Матвій Коренівський	1	₴320.00	08.06.2023 19:16	В обробці	
000044	Матвій Коренівський	1	₴175.00	08.06.2023 19:15	Нове	
000043	Матвій Коренівський	1	₴175.00	08.06.2023 19:14	Нове	
000042	Матвій Коренівський	1	₴175.00	08.06.2023 19:11	Нове	
000041	Матвій Коренівський	1	₴120.00	08.06.2023 19:10	Нове	

Всього замовлень: 29

Сторінка 1 з 3

<

>

Рис. 3.15 – Сторінка історії замовлень

Замовлення #000048

Одержувач	Матвій Коренівський	Дата замовлення	08.06.2023 19:57
E-mail	matvii@gmail.com	Спосіб оплати	Оплата при отриманні
Номер телефону	+380983015572	Статус	Нове
Адреса	м. Житомир, вул. Велика Бердичівська, буд. 12, кв. 3		

Товар	Кількість	Ціна	Вартість
Бенджамін Франклін Волтер Айзексон	1	€320.00	€320.00
Атлант розправив плечі. Частина 2 Айн Ренд	1	€185.00	€185.00
Від хорошого до величного Джим Коллінз	1	€195.00	€195.00
Думай і багатій Наполеон Гілл	1	€140.00	€140.00
Атлант розправив плечі. Частина 3 Айн Ренд	1	€200.00	€200.00
Всього товарів: 5			Загальна вартість: €1040.00

Рис. 3.16 – Сторінка детальної інформації про замовлення

Користувач-адміністратор після авторизації потрапляє до панелі керування інтернет-магазину. В правій частині зображено вертикальне меню для переходу між різними секціями панелі адміністратора – рисунок 3.17.

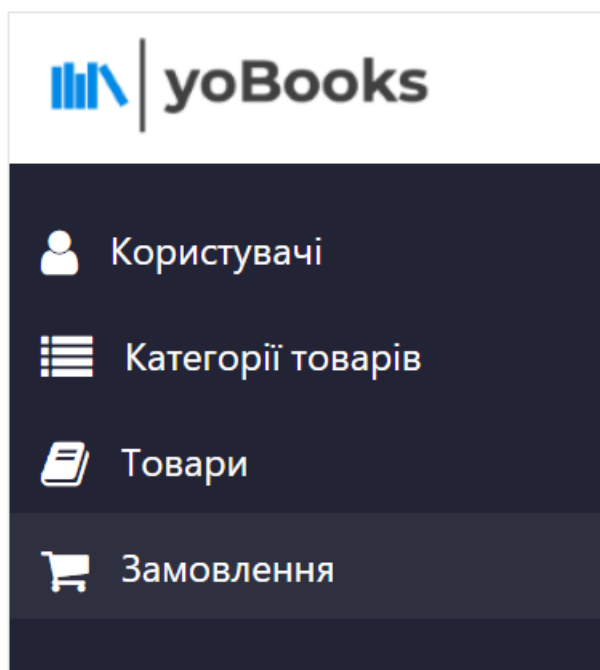
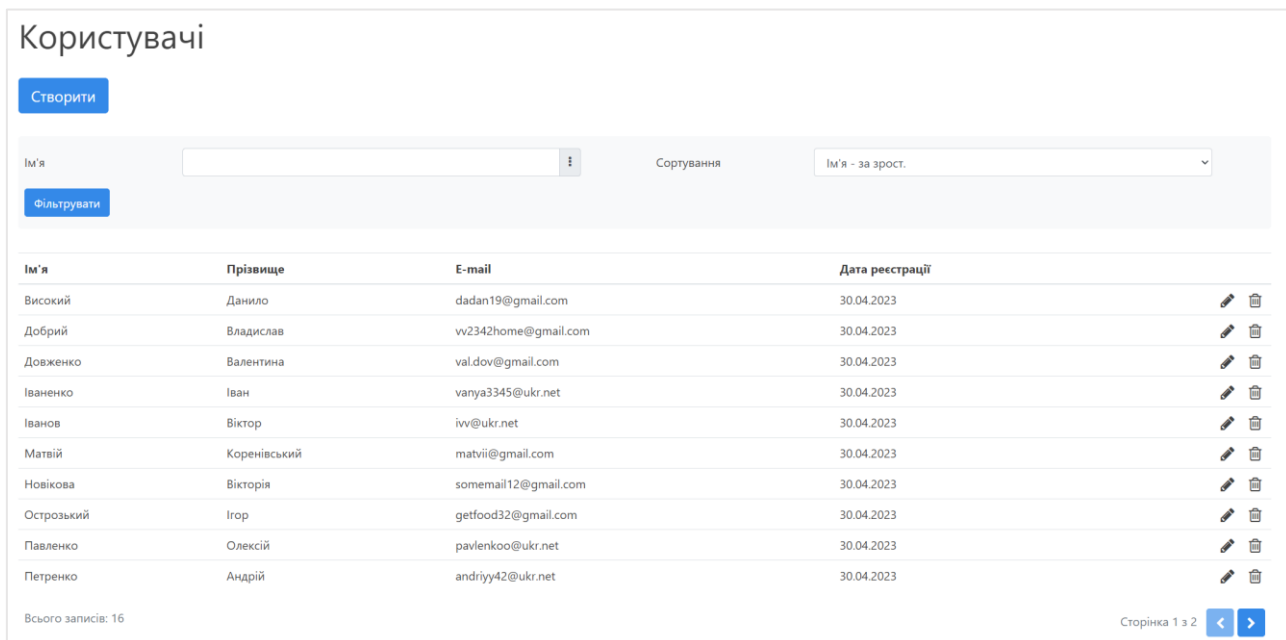


Рис. 3.17 – Меню навігації панелі адміністратора

На сторінці «Користувачі» наявний список усіх клієнтів інтернет магазину, а також блок для пошуку, фільтрування та сортування записів. Знизу списку

		Коренівський М.О.			ІПЗ.КР.Б-121-23-ПЗ	Арк.
		Морозов А.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		64

наявна інформація про загальну кількість записів, а також кнопки для переходу між сторінками списку. Кожен елемент списку містить інформацію про користувача, та блок кнопок для редагування чи видалення записів. Сторінку «Користувачі» зображено на рисунку 3.18.



Ім'я	Прізвище	E-mail	Дата реєстрації
Високий	Данило	dadan19@gmail.com	30.04.2023
Добрий	Владислав	vv2342home@gmail.com	30.04.2023
Довженко	Валентина	val.dov@gmail.com	30.04.2023
Іваненко	Іван	vanya3345@ukr.net	30.04.2023
Іванов	Віктор	ivv@ukr.net	30.04.2023
Матвій	Коренівський	matvii@gmail.com	30.04.2023
Новікова	Вікторія	somemail12@gmail.com	30.04.2023
Острозький	Ігор	getfood32@gmail.com	30.04.2023
Павленко	Олексій	pavlenkoo@ukr.net	30.04.2023
Петренко	Андрій	andriyy42@ukr.net	30.04.2023

Рис. 3.18 – Сторінка «Користувачі»

При натисканні на кнопку видалення адміністратор побачить повідомлення про підтвердження видалення запису, яке зображено на рисунку 3.19.

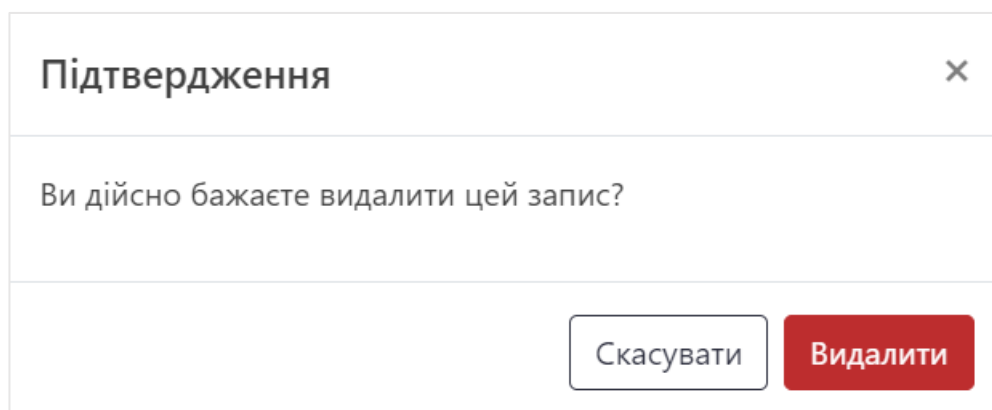
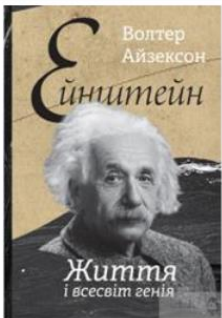


Рис. 3.19 – Повідомлення про підтвердження видалення запису

Усі сторінки зі списком записів мають однакову структуру та надають однаковий набір операцій для взаємодії із записами. Також однакову структуру мають і сторінки створення та редагування записів.

При переході на сторінку створення чи редагування запису адміністратор матиме змогу ввести необхідну інформацію. Сторінку редагування товару зображено на рисунку 3.20.

Редагувати товар



Назва*

Автор*

Категорія*

Опис

Ціна*

Обкладинка*

Кількість сторінок*

В наявності*

Змінити зображення

Зберегти

Скасувати

Видалити

Рис. 3.20 – Сторінка редагування товару

На сторінці з деталями замовлення клієнта адміністратор може змінити статус замовлення в залежності від стадії його виконання. Сторінку деталей замовлення клієнта магазину зображено на рисунку 3.21.

		Коренівський М.О.			ІПЗ.КР.Б-121-23-ПЗ	Арк.
		Морозов А.В.				66
Змн.	Арк.	№ докум.	Підпис	Дата		

Замовлення #000048

Ім'я	Матвій Коренівський	Дата замовлення	08.06.2023 19:57
E-mail	matvii@gmail.com	Спосіб оплати	Оплата при отриманні
Номер телефону	+380983015572	Статус	Нове Скасоване Нове В обробці Відправлене Виконане
Адреса	м. Житомир, вул. Велика Бердичівська, буд. 12, кв. 3		
Зберегти Скасувати			

Товар	Кількість	Ціна	Вартість
Бенджамін Франклін Волтер Айзексон	1	₴320.00	₴320.00
Атлант розправив плечі. Частина 2 Айн Ренд	1	₴185.00	₴185.00
Від хорошого до величного Джим Коллінз	1	₴195.00	₴195.00
Думай і багатій Наполеон Гілл	1	₴140.00	₴140.00
Атлант розправив плечі. Частина 3 Айн Ренд	1	₴200.00	₴200.00
Всього товарів: 10			Загальна вартість: ₴14.00

Рис. 3.21 – Сторінка деталей замовлення клієнта

3.3 Тестування роботи програмного забезпечення

Наступним етапом є тестування розробленого веб-сайту. В рамках першого тесту перевірятиметься адаптивність сайту. Для перевірки адаптації сторінок потрібно завантажити їх на декількох пристроях з різними розмірами екрана. Для першого тесту було обрано планшет(768px) – рисунок 3.22

		Коренівський М.О.			ІПЗ.КР.Б-121-23-ПЗ	Арк.
		Морозов А.В.				67
Змн.	Арк.	№ докум.	Підпис	Дата		

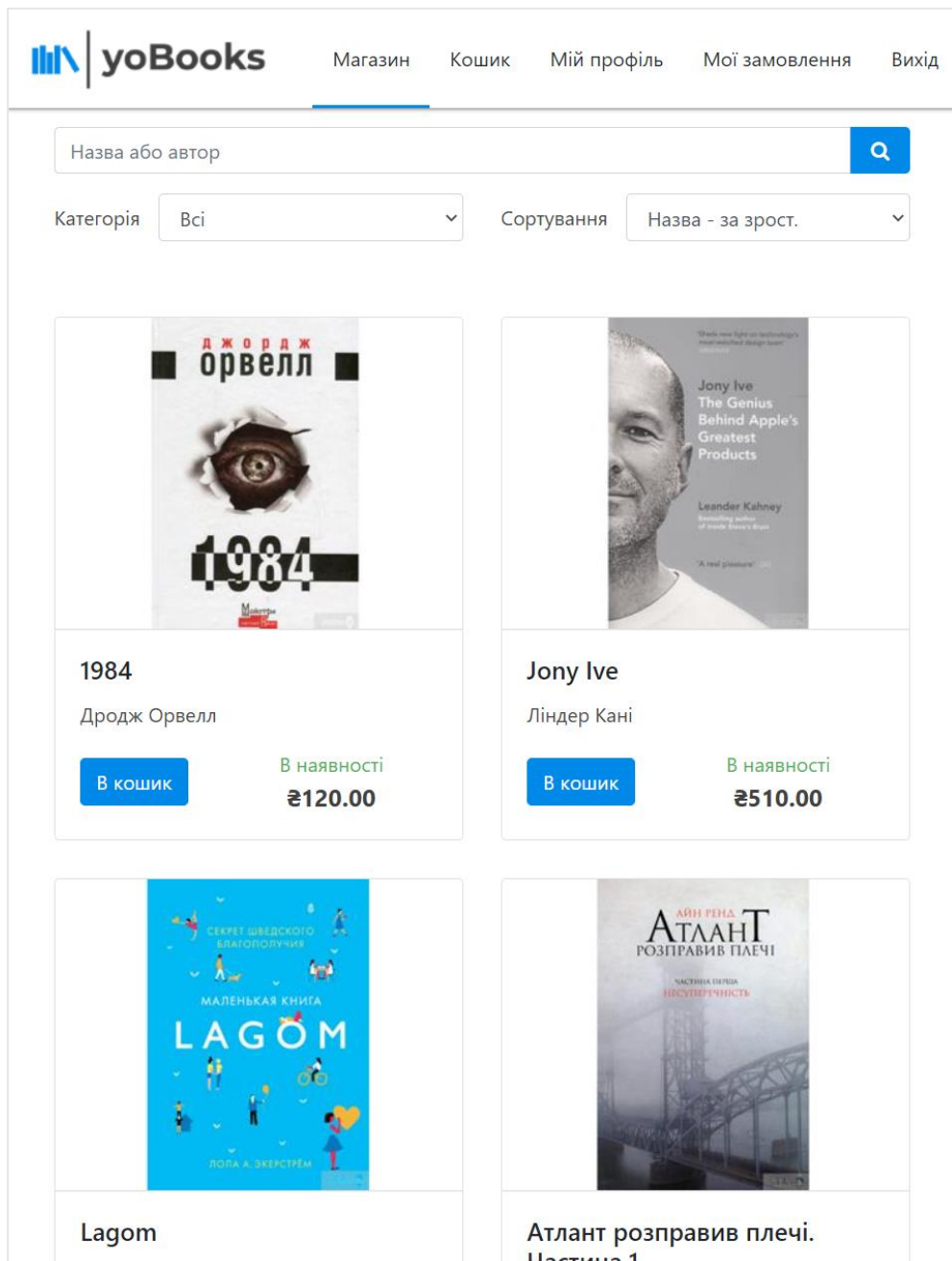


Рис. 3.22 – Тест адаптивності сторінки на планшеті(768px)

При зменшенні розмірів екрану всі об'єкти рівно та чітко відображаються на сторінці.

Наступним тестом є перевірка адаптивності на екрані телефону розміром 425px. Результат тесту зображено на рисунку 3.23. В цьому випадку всі елементи зберегли свої позиції, а деякі елементи було переміщено відносно сторінки задля їх кращого відображення.

		Коренівський М.О.		
		Морозов А.В.		
Змн.	Арк.	№ докум.	Підпис	Дата

ІПЗ.КР.Б-121-23-ПЗ

Арк.

68

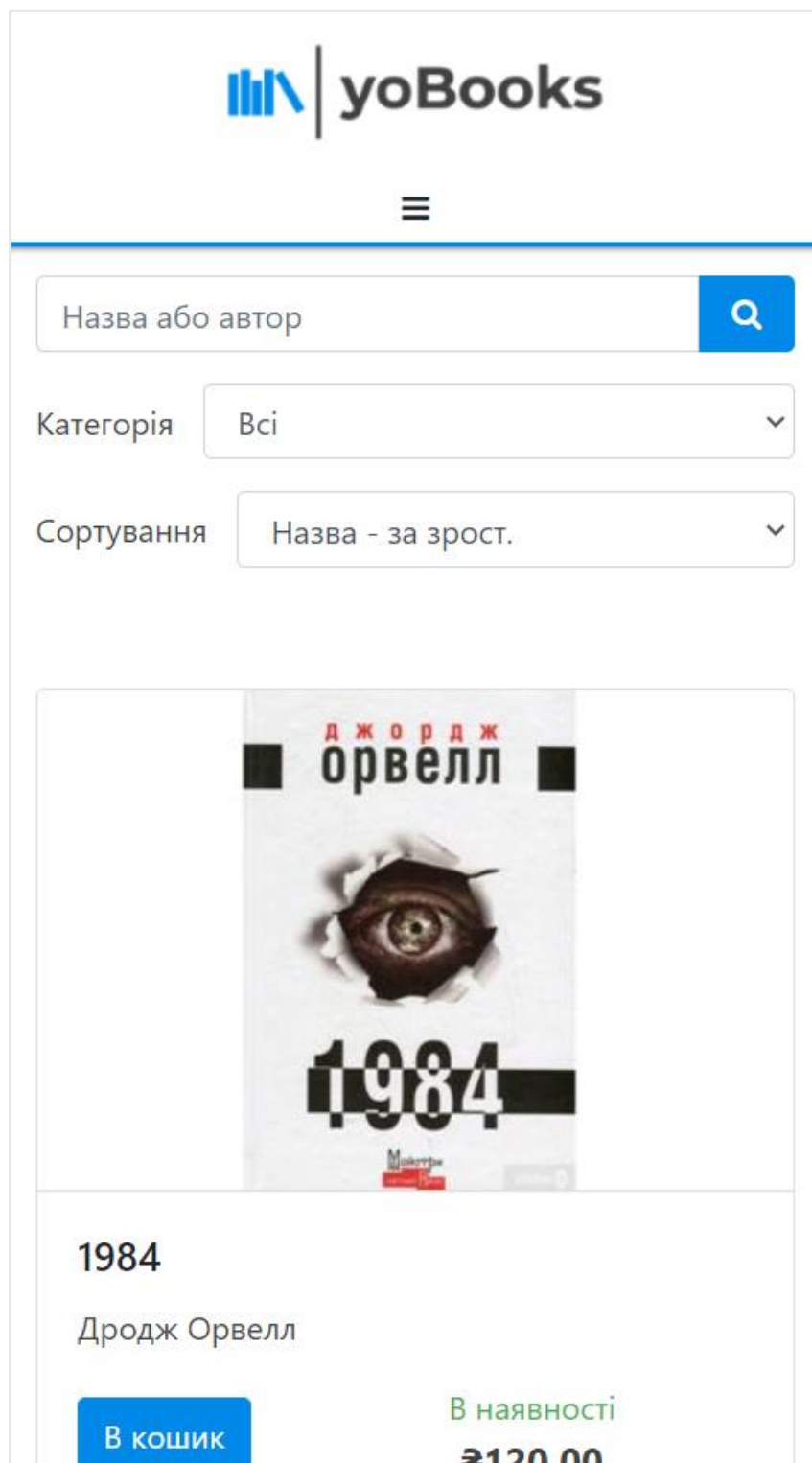


Рис. 3.23 – Тест на телефоні(425px)

Наступним розміром екрану для перевірки адаптивності веб-сайту було обрано телефон 360px. Сторінка обрана для тесту зображена на рисунку 3.24. В результаті тесту виявлено, що форма змінила свій розмір та помістилася на сторінку, не виходячи за її межі.

		Коренівський М.О.		
		Морозов А.В.		
Змн.	Арк.	№ докум.	Підпис	Дата

ІПЗ.КР.Б-121-23-ПЗ

Арк.

69



≡

Деталі профілю

E-mail*

Пароль

Підтвердження пароля

Ім'я*

Прізвище*

Номер телефону

Рис. 3.24 – Тест на телефоні(360px)

В рамках наступного етапу тестування перевірятиметься коректність введення даних. У разі введення неправильних у форму зміни даних облікового запису користувача, програма повідомить про помилку та буде виведено відповідне повідомлення рисунок 3.25.

		Коренівський М.О.			ІПЗ.КР.Б-121-23-ПЗ	Арк.
		Морозов А.В.				70
Змн.	Арк.	№ докум.	Підпис	Дата		

Деталі профілю

Е-mail*

Пароль

Підтвердження пароля

Пароль має містити мінімум 6 символів, букви та цифри

Ім'я*

Прізвище*

Це обов'язкове поле

Номер телефону

Рис. 3.25 – Тест валідації форми редагування облікового запису

Аналогічний тест на валідацію проведено і на формі авторизації – якщо пароль та логін не співпадають, вхід не відбувається, доки користувач не введе правильні дані. В разі введення некоректних даних користувач побачить повідомлення про помилку. Результат тестування форми авторизації зображено на рисунку 3.26.

Ласкаво просимо

sometestdata

.....

Невірне ім'я користувача та/або пароль

Вхід

Реєстрація

Рис. 3.26 – Тест валідації форми авторизації

		Коренівський М.О.			ІПЗ.КР.Б-121-23-ПЗ	Арк.
		Морозов А.В.				71
Змн.	Арк.	№ докум.	Підпис	Дата		

В результаті тестування веб-додатку було зроблено висновок, що розроблений веб-сайт задовольняє усі вимоги до адаптивного сайту. В рамках тестування було перевірено вірне написання всіх слів та текстів, коректність всіх посилань. Також було протестовано наявність валідації та відповідних повідомлень, які виводяться користувачеві. Під час тестування було виправлено усі помилки в роботі системи та налагоджено роботу сайту.

Висновки до третього розділу

У цьому розділі було описано процес встановлення та налаштування додатку, а також визначено мінімальні системні вимоги для його використання. Крім того, була наведена діаграма розгортання, що показує конфігурацію системи з точки зору зберігання та обробки даних.

Детально проаналізовано функціонал веб-додатку, його особливості та можливості, покроково описано процес роботи з додатком, Також проведено перевірку на наявність помилок та виконано кілька тестів, за успішними результатами можна стверджувати що розроблений програмний додаток має високу якість.

		Коренівський М.О.			ІПЗ.КР.Б-121-23-ПЗ	Арк.
		Морозов А.В.				72
Змн.	Арк.	№ докум.	Підпис	Дата		

ВИСНОВКИ

В рамках кваліфікаційної роботи бакалавра було спроектовано та реалізовано веб-орієнтовану систему книжкового інтернет-магазину, задля чого проведено ряд досліджень.

В першому розділі роботи було проведено аналіз предметної області, що спрягло визначенню основних аспектів, що мають бути втілені у веб-додаток. За результатами аналізу аналогів системи за тематикою кваліфікаційної роботи було окреслено критерії успішної реалізації книжкового інтернет-магазину, а також визначено переваги системи, розробленої в рамках кваліфікаційної роботи: зручний інтерфейс, адаптивність до різних пристроїв та розмірів екрану, створення облікового запису користувача та можливість онлайн-оплати замовлення. Також визначено архітектуру веб-орієнтованої системи книжкового інтернет-магазину, що відповідає патерну проєктування MVC, та визначено вимоги до апаратного забезпечення.

В другому розділі було спроектовано загальний алгоритм роботи програми, визначено об'єктно-орієнтовану модель системи. Для структурування усіх даних системи було спроектовано базу даних. Під час проєктування інтерфейсу веб-додатку було досліджено різноманітні підходи до розробки системи інтернет-магазину, визначено алгоритми, що будуть основою функціоналу.

В третьому розділі було детально описано роботу з веб-додатком, проведено налаштування параметрів системи. Було розглянуто основні особливості та можливості веб-додатку, покроково описано процес роботи з додатком. Також в цьому розділі було проведено тестування, виявлення та виправлення помилок, відлагодження усіх компонентів системи. Тестування показало, що інформаційна веб-система успішно виконує свої функції, є надійною та високоякісною.

Веб-орієнтована система книжкового інтернет-магазину, розроблена в рамках кваліфікаційної роботи, пройшла успішне тестування та продемонструвала високу якість. Додаток відповідає всім поставленим вимогам

		Коренівський М.О.			ІПЗ.КР.Б-121-23-ПЗ	Арк.
		Морозов А.В.				73
Змн.	Арк.	№ докум.	Підпис	Дата		

та потребам користувачів, забезпечуючи зручний спосіб придбання книг. Розроблений веб-додаток може бути розширений в рамках майбутньої розробки, завдяки якій можна впровадити нові функції та покращити взаємодію з користувачами.

		Коренівський М.О.			ІПЗ.КР.Б-121-23-ПЗ	Арк.
		Морозов А.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		74

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Перевозчикова О. Інформаційні системи і структури даних / Ольга Перевозчикова., 2007. – 288 с.
2. Коноваленко І. Програмування мовою С# 6.0 / Ігор Коноваленко. – Тернопіль: ТНТУ, 2016. – 226 с.
3. Мартін Р. Чистий код. Створення і рефакторинг за допомогою Agile / Роберт Сесіл Мартін., 2009. – 448 с.
4. McConnell S. Code Complete, 2nd Edition / Steve McConnell.. – 857 с.
5. UML @ Classroom: An Introduction to Object-Oriented Modeling (Undergraduate Topics in Computer Science) / M.Seidl, M. Scholz, C. Huemer, C. Kappel., 2015. – 218 с.
6. Petkovic D. Microsoft SQL Server 2019: A Beginner's Guide, Seventh Edition / Dusan Petkovic., 2020. – 864 с.
7. Korotkevitch D. Expert SQL Server Transactions and Locking: Concurrency Internals for SQL Server Practitioners / Dmitri Korotkevitch., 2018. – 320 с.
8. Duckett J. JavaScript and jQuery / Jon Duckett., 2014. – 620 с.
9. Flanagan D. JavaScript: The Definitive Guide: Activate Your Web Pages (Definitive Guides) 6th Edition / David Flanagan., 2011. – 1096 с.
10. Babbista G. Software Architecture with C# 10 and .NET 6: Develop software solutions using microservices, DevOps, EF Core / G. Babbista, F. Abbresesse., 2022. – 1203 с.
11. Boehm A. Murach's C# 2015 6th Edition / A. Boehm, J. Murach., 2016. – 906 с.
12. Troelsen A. Pro C# 7: With .NET and .NET Core / A. Troelsen, P. Japkse., 2017. – 1437 с.
13. Ackermann P. JavaScript: The Comprehensive Guide to Learning Professional JavaScript Programming / Philip Ackermann., 2022. – 982 с.
14. Мартін Р. Чиста архітектура / Роберт Мартін., 2020. – 368 с.
15. Уроки для вивчення С# [Електронний ресурс] – Режим доступу до ресурсу: <https://programer.in.ua/index.php/uroky/uroky-dlia-vyvchennia-c>.

		Коренівський М.О.			ІПЗ.КР.Б-121-23-ПЗ	Арк.
		Морозов А.В.				75
Змн.	Арк.	№ докум.	Підпис	Дата		

16. Working with Files & Directories in C# [Електронний ресурс] – Режим доступу до ресурсу: <https://www.tutorialsteacher.com/csharp/csharp-file>.
- 17.10 Good Practices for ASP.NET MVC Apps [Електронний ресурс] – Режим доступу до ресурсу: <https://www.codemag.com/article/1405071/10-Good-Practices-for-ASP.NET-MVC-Apps>.
18. SQL Server Tips, Articles and Training [Електронний ресурс] – Режим доступу до ресурсу: <https://www.mssqltips.com/>.
19. Using LINQ with C# [Електронний ресурс] – Режим доступу до ресурсу: <https://www.c-sharpcorner.com/UploadFile/84c85b/using-linq-with-C-Sharp-2012/>.
20. Building a Better ORM With NHibernate For .NET [Електронний ресурс] – Режим доступу до ресурсу: <https://www.c-sharpcorner.com/UploadFile/d6707e/building-a-better-orm-with-nhibernate-for-net/>.
21. What's new in C# 9.0 [Електронний ресурс] – Режим доступу до ресурсу: <https://learn.microsoft.com/en-us/dotnet/csharp/whats-new/csharp-9>.
22. What's new in C# 10 [Електронний ресурс] – Режим доступу до ресурсу: <https://learn.microsoft.com/en-us/dotnet/csharp/whats-new/csharp-10>.
23. C# documentation [Електронний ресурс] – Режим доступу до ресурсу: <https://learn.microsoft.com/en-us/dotnet/csharp/>.
24. NHibernate Reference Documentation [Електронний ресурс] – Режим доступу до ресурсу: <https://nhibernate.info/doc/nhibernate-reference/index.html>.
25. Tutorials for SQL Server [Електронний ресурс] – Режим доступу до ресурсу: <https://learn.microsoft.com/en-us/sql/sql-server/tutorials-for-sql-server-2016?view=sql-server-ver16>.
26. How to serialize and deserialize (marshal and unmarshal) JSON in .NET [Електронний ресурс] – Режим доступу до ресурсу: <https://learn.microsoft.com/en-us/dotnet/standard/serialization/system-text-json/how-to?pivots=dotnet-8-0>.

		Корнівський М.О.			ІПЗ.КР.Б-121-23-ПЗ	Арк.
		Морозов А.В.				76
Змн.	Арк.	№ докум.	Підпис	Дата		

27. Платіжний віджет LiqPay - Документація API [Електронний ресурс] – Режим доступу до ресурсу: <https://www.liqpay.ua/documentation/api/aquiring/widget/doc>.
28. How to use data annotations in C# [Електронний ресурс] – Режим доступу до ресурсу: <https://www.infoworld.com/article/3543302/how-to-use-data-annotations-in-c-sharp.html>.
29. Understanding HTML Helpers in ASP.NET MVC [Електронний ресурс] – Режим доступу до ресурсу: <https://www.dotnettricks.com/learn/mvc/understanding-html-helpers-in-aspnet-mvc>.
30. Fundamental Data Structures and Algorithms in C# [Електронний ресурс] – Режим доступу до ресурсу: <https://dev.to/adavidoaiei/fundamental-data-structures-and-algorithms-in-c-4ocf>.

		Коренівський М.О.			ІПЗ.КР.Б-121-23-ПЗ	Арк.
		Морозов А.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		77

ДОДАТКИ

		Коренівський М.О.			ІПЗ.КР.Б-121-23-ПЗ	Арк.
		Морозов А.В.				78
Змн.	Арк.	№ докум.	Підпис	Дата		

Лістинг програми:

EntityProvider:

```

public class EntityProvider<TEntity> : IEntityProvider<TEntity> where TEntity : class
{
    public virtual void Delete(object id)
    {
        Transaction(session => session.Delete(GetById(id)));
    }

    public virtual void Delete(TEntity entity)
    {
        Transaction(session => session.Delete(entity));
    }

    public virtual TEntity GetById(object id)
    {
        return Query(session => session.Get<TEntity>(id), $"An error was occurred
while getting entity by id {id}");
    }

    public virtual void Save(TEntity entity)
    {
        Transaction(session => session.SaveOrUpdate(entity), $"An error was
occured while saving an entity {entity.GetType()}");
    }

    public virtual void Save(IEnumerable<TEntity> entities)
    {
        Transaction(session => entities.Each(e => session.SaveOrUpdate(e)), $"An
error was occurred while saving entities {entities.GetType()}");
    }

    public virtual IList<TEntity> GetList(IEntityListLoadOptions loadOptions)
    {
        return Query(session =>
        {
            var criteria = session.CreateCriteria<TEntity>();
            AddListLoadOptionsToCriteria(criteria, loadOptions);
            AddFieldFiltersToCriteria(criteria, loadOptions.FieldFilters);

            return criteria.List<TEntity>();
        });
    }

    public virtual int GetTotalCount(IEntityListLoadOptions loadOptions)
    {
        return Query(session =>
        {
            var criteria = session.CreateCriteria<TEntity>()
                .SetProjection(Projections.RowCount());
            AddFieldFiltersToCriteria(criteria, loadOptions.FieldFilters);

            return criteria.UniqueResult<int>();
        });
    }

    protected Tq Query<Tq>(Func<ISession, Tq> func, string customMessage = null)
    {
        try
        {
            using (ISession session = NHibernateHelper.OpenSession())

```

		Коренівський М.О.			ІПЗ.КР.Б-121-23-ПЗ	Арк.
		Морозов А.В.				79
Змн.	Арк.	№ докум.	Підпис	Дата		

```

        {
            return func(session);
        }
    }
    catch (Exception)
    {
        throw;
    }
}

protected void Transaction(Action<ISession> action, string customMessage =
null)
{
    try
    {
        using (ISession session = NHibernateHelper.OpenSession())
        {
            using (ITransaction transaction = session.BeginTransaction())
            {
                action(session);
                transaction.Commit();
            }
        }
    }
    catch (Exception ex)
    {
        throw;
    }
}

protected virtual void AddListLoadOptionsToCriteria(ICriteria criteria,
IEntityListLoadOptions loadOptions)
{
    int pageIndex = loadOptions.PageIndex > 0 ? loadOptions.PageIndex - 1 : 0;
    if (!loadOptions.SortingField.IsNullOrEmpty())
        criteria.AddOrder(loadOptions.SortAscending ?
Order.Asc(loadOptions.SortingField) : Order.Desc(loadOptions.SortingField));

    if (loadOptions.PageSize > 0)
        criteria.SetFirstResult(loadOptions.PageSize * pageIndex)
        .SetMaxResults(loadOptions.PageSize);
}

protected virtual void AddFieldFiltersToCriteria(ICriteria criteria,
IDictionary<string, string> fieldFilters)
{
    if (fieldFilters == null)
        return;

    fieldFilters.Where(f => !f.Value.IsNullOrEmpty()).Each(f =>
criteria.Add(Restrictions.InsensitiveLike(f.Key, f.Value, MatchMode.Anywhere)));
}

public void Delete(IEnumerable<TEntity> entities)
{
    Transaction(session => entities.Each(e => session.Delete(e)));
}
}

```

EntityListLoadOptions:

```

public class EntityListLoadOptions : IEntityListLoadOptions
{
    public int PageIndex { get; set; }

    public int PageSize { get; set; }
}

```

		Коренівський М.О.			ІІЗ.КР.Б-121-23-ПЗ	Арк.
		Морозов А.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		80

```

public string SortingField { get; set; }

public bool SortAscending { get; set; }

public IDictionary<string, string> FieldFilters { get; set; }

public EntityListLoadOptions()
{
    this.FieldFilters = new Dictionary<string, string>();
}
}

```

HtmlExtensions:

```

public static class HtmlExtensions
{
    public static IHtmlString GetText(this HtmlHelper htmlHelper, string key)
    {
        return WebsiteManager.Text.GetText(key).AsHtmlString();
    }

    public static IHtmlString AsHtmlString(this string stringToConvert, bool
encode = false)
    {
        if (stringToConvert == null)
            return new HtmlString(string.Empty);

        return new HtmlString(stringToConvert);
    }

    public static MvcHtmlString LinkButton(this HtmlHelper html, string url,
IHtmlString text, string cssClass = null, object htmlAttributes = null)
    {
        return html.LinkButton(url, text.ToHtmlString(), cssClass,
htmlAttributes);
    }

    public static MvcHtmlString LinkButton(this HtmlHelper html, string url,
string text, string cssClass = null, object htmlAttributes = null)
    {
        TagBuilder a = new TagBuilder("a");
        a.AddCssClass(string.Join(" ", "btn", cssClass));
        a.InnerHtml = text;

        a.MergeAttributes(HtmlHelper.AnonymousObjectToHtmlAttributes(htmlAttributes));
        a.MergeAttribute("href", url);

        return new MvcHtmlString(a.ToString());
    }

    public static MvcHtmlString Icon(this HtmlHelper html, string name, string
cssClass = null, object htmlAttributes = null)
    {
        TagBuilder i = new TagBuilder("i");
        i.AddCssClass(string.Join(" ", "fa-", "fa-" + name, cssClass));

        i.MergeAttributes(HtmlHelper.AnonymousObjectToHtmlAttributes(htmlAttributes));

        return new MvcHtmlString(i.ToString());
    }

    public static MvcHtmlString SubmitButton(this HtmlHelper html, IHtmlString
text, string cssClass = null, object htmlAttributes = null)
    {
        TagBuilder btn = new TagBuilder("button");

```

		Коренівський М.О.			ІПЗ.КР.Б-121-23-ПЗ	Арк.
		Морозов А.В.				81
Змн.	Арк.	№ докум.	Підпис	Дата		

```

        btn.AddCssClass(string.Join(" ", "btn", cssClass));
        btn.InnerHtml = text.ToHtmlString();

        btn.MergeAttributes(HtmlHelper.AnonymousObjectToHtmlAttributes(htmlAttributes));
        btn.MergeAttribute("type", "submit");

        return new MvcHtmlString(btn.ToString());
    }

    public static MvcHtmlString Button(this HtmlHelper html, IHtmlString text,
    string cssClass = null, object htmlAttributes = null)
    {
        TagBuilder btn = new TagBuilder("button");
        btn.AddCssClass(string.Join(" ", "btn", cssClass));
        btn.SetInnerText(text.ToHtmlString());

        btn.MergeAttributes(HtmlHelper.AnonymousObjectToHtmlAttributes(htmlAttributes));
        btn.MergeAttribute("type", "button");

        return new MvcHtmlString(btn.ToString());
    }

    public static string FormatWith(this IHtmlString htmlString, params object[]
    args) => htmlString.ToHtmlString().FormatWith(args);

    public static MvcHtmlString ProductImage(this HtmlHelper html, int? productId,
    Constants.ImageSize imageSize, string cssClass = null, object htmlAttributes = null)
    {
        TagBuilder img = new TagBuilder("img");
        string imageFolder = imageSize switch
        {
            Constants.ImageSize.Small => "small",
            Constants.ImageSize.Medium => "medium",
            Constants.ImageSize.Large => "large",
            _ => string.Empty
        };

        string imagesPath = $"/Content/images/products/{imageFolder}";
        string imagePath = Path.Combine(imagesPath, productId.ToString() +
        ".jpg");
        if (!File.Exists(HttpContext.Current.Server.MapPath(imagePath)))
        {
            imagePath = Path.Combine(imagesPath, "no-image.jpg");
            img.AddCssClass("no-image");
        }

        img.MergeAttribute("src", imagePath);

        img.MergeAttributes(HtmlHelper.AnonymousObjectToHtmlAttributes(htmlAttributes));
        img.AddCssClass(string.Join(" ", $"img-{imageFolder}", cssClass));
        return new MvcHtmlString(img.ToString());
    }

    public static string ToPriceFormat(this decimal price) =>
    price.ToString("¤#0.00");

    public static IDictionary<string, object> MergeAnonymous(this HtmlHelper html,
    object obj1, object obj2)
    {
        var dict1 = obj1 as RouteValueDictionary ?? new
        RouteValueDictionary(obj1);
        var dict2 = obj2 as RouteValueDictionary ?? new
        RouteValueDictionary(obj2);
        IDictionary<string, object> result = new Dictionary<string, object>();

        foreach (var pair in dict1.Concat(dict2))

```

		Коренівський М.О.			ІІЗ.КР.Б-121-23-ІІЗ	Арк.
		Морозов А.В.				82
Змн.	Арк.	№ докум.	Підпис	Дата		

```

        {
            result.Add(pair);
        }

        return result;
    }
}

```

ImageHelper:

```

public static class ImageHelper
{
    public static void UploadImage(HttpPostedFileBase imageFile, string fileName,
    ImageSize imageSize)
    {
        var image = Image.FromStream(imageFile.InputStream, true, true);
        int width, height;
        width = height = imageSize switch
        {
            ImageSize.Small => 80,
            ImageSize.Medium => 250,
            ImageSize.Large => 500,
            _ => 0
        };

        string imageFolder = imageSize switch
        {
            Constants.ImageSize.Small => "small",
            Constants.ImageSize.Medium => "medium",
            Constants.ImageSize.Large => "large",
            _ => string.Empty
        };

        string imagesPath = $"Content/images/products/{imageFolder}";

        var destRect = new Rectangle(0, 0, width, height);
        var destImage = new Bitmap(width, height);

        destImage.SetResolution(image.HorizontalResolution,
        image.VerticalResolution);

        using (var graphics = Graphics.FromImage(destImage))
        {
            graphics.CompositingMode = CompositingMode.SourceCopy;
            graphics.CompositingQuality = CompositingQuality.HighQuality;
            graphics.InterpolationMode = InterpolationMode.HighQualityBicubic;
            graphics.SmoothingMode = SmoothingMode.HighQuality;
            graphics.PixelOffsetMode = PixelOffsetMode.HighQuality;

            using (var wrapMode = new ImageAttributes())
            {
                wrapMode.SetWrapMode(WrapMode.TileFlipXY);
                graphics.DrawImage(image, destRect, 0, 0, image.Width,
                image.Height, GraphicsUnit.Pixel, wrapMode);
            }
        }

        destImage.Save(AppDomain.CurrentDomain.BaseDirectory +
        $"\\{imagesPath}\\{fileName}.jpg", ImageFormat.Jpeg);
    }
}

```

ProfileController:

```

public class ProfileController : Controller
{
    [HttpGet]

```

		Коренівський М.О.			ІІЗ.КР.Б-121-23-ІІЗ	Арк.
		Морозов А.В.				83
Змн.	Арк.	№ докум.	Підпис	Дата		

```

public ActionResult Login()
{
    var model = ObjectManager.GetInstance<LoginModel>();
    return View(model);
}

[HttpPost]
[ValidateAntiForgeryToken]
public ActionResult Login(LoginModel model)
{
    if (!ModelState.IsValid)
        return View(model);

    bool isAdmin =
WebsiteManager.Authentication.AuthenticateAdmin(model.Username, model.Password);
    bool isUser = false;
    if (!isAdmin)
        isUser =
WebsiteManager.Authentication.AuthenticateUser(model.Username, model.Password);
    if (!isAdmin && !isUser)
    {
        ModelState.AddModelError("password",
WebsiteManager.Text.GetText("ValidationError_LoginFailed"));
        return View(model);
    }
    FormsAuthentication.SetAuthCookie(model.Username, true);

    if (isUser)
        return RedirectToAction("Index", "Products", new { Area = "" });
    if (isAdmin)
        return RedirectToAction("Index", "Orders", new { Area = "Admin" });
    return new EmptyResult();
}

[HttpGet]
public ActionResult Register()
{
    var model = ObjectManager.GetInstance<RegisterModel>();
    return View(model);
}

[HttpPost]
[ValidateAntiForgeryToken]
public ActionResult Register(RegisterModel model)
{
    if (!ModelState.IsValid)
        return View(model);

    var existingUser = WebsiteManager.Users.GetByEmail(model.Email);

    if (existingUser != null)
    {
        ModelState.AddModelError("email",
WebsiteManager.Text.GetText("ValidationError_NotUniqueEmail"));
        return View(model);
    }

    var user = ObjectManager.GetInstance<User>();
    model.Update(user);
    user.CreationDate = DateTime.Now;
    WebsiteManager.Users.Save(user);
    FormsAuthentication.SetAuthCookie(user.Email, true);

    return RedirectToAction("Index", "Products", new { Area = "" });
}

```

		Коренівський М.О.			ІПЗ.КР.Б-121-23-ПЗ	Арк.
		Морозов А.В.				84
Змн.	Арк.	№ докум.	Підпис	Дата		

```

[HttpGet]
[Authorize(Roles = "User")]
public ActionResult Details()
{
    var user = WebsiteManager.Users.GetByEmail(User.Identity.Name);
    if (user == null)
        return HttpNotFound();
    var model = ObjectManager.GetInstance<ProfileDetailsModel>();
    model.Initialize(user);

    return View(model);
}

[HttpPost]
[ValidateAntiForgeryToken]
[Authorize(Roles = "User")]
public ActionResult Details(ProfileDetailsModel model)
{
    if (!ModelState.IsValid)
        return View(model);

    var existingUser = WebsiteManager.Users.GetById(model.Id);
    bool emailIsNotUnique = existingUser != null && existingUser.Id !=
model.Id && model.Email.Equals(existingUser.Email);
    if (emailIsNotUnique)
    {
        ModelState.AddModelError("email",
WebsiteManager.Text.GetText("ValidationError_NotUniqueEmail"));
        return View(model);
    }

    if (existingUser != null)
    {
        model.Update(existingUser);
        WebsiteManager.Users.Save(existingUser);
    }

    return View(model);
}

[HttpGet]
public ActionResult Logout()
{
    FormsAuthentication.SignOut();
    return RedirectToAction("Login");
}
}

```

BasketController:

```

[Authorize(Roles = Constants.Roles.User)]
public class BasketController : Controller
{
    [HttpPost]
    [ValidateAntiForgeryToken]
    public ActionResult AddToBasket(int productId)
    {
        var basketLine = ObjectManager.GetInstance<BasketLine>();
        var currentUserId =
WebsiteManager.Users.GetByEmail(User.Identity.Name).Id;
        var existingLine =
WebsiteManager.BasketLines.GetByUserAndProduct(currentUserId, productId);
        bool success = true;
        if (existingLine != null)
        {
            var product = WebsiteManager.Products.GetById(existingLine.ProductId);

```

		Коренівський М.О.			ІІЗ.КР.Б-121-23-ІІЗ	Арк.
		Морозов А.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		85

```

        if (product.Stock == existingLine.Quantity)
            success = false;

        basketLine = existingLine;
        basketLine.Quantity++;
    }
    else
    {
        basketLine.ProductId = productId;
        basketLine.Quantity = 1;
        basketLine.UserId = currentUserId;
    }

    if (success)
        WebsiteManager.BasketLines.Save(basketLine);

    return Json(new { success = success });
}

[HttpGet]
public ActionResult Details()
{
    var currentUserId =
WebsiteManager.Users.GetByEmail(User.Identity.Name).Id;
    var basketLines = WebsiteManager.BasketLines.GetByUserId(currentUserId);
    var productsInBasket =
WebsiteManager.Products.GetByIdList(basketLines.Select(l => l.ProductId));

    var basketModel = ObjectManager.GetInstance<BasketModel>();

    foreach (var line in basketLines)
    {
        var basketLineModel = ObjectManager.GetInstance<BasketLineModel>();
        basketLineModel.Initialize(line, productsInBasket.Single(p => p.Id ==
line.ProductId));
        basketModel.BasketLines.Add(basketLineModel);
    }

    return View(basketModel);
}

[HttpPost]
[ValidateAntiForgeryToken]
public ActionResult Delete(int id)
{
    if (!Request.IsAjaxRequest())
        return HttpNotFound();

    var basketLine = WebsiteManager.BasketLines.GetById(id);
    if (basketLine == null)
        return HttpNotFound();

    WebsiteManager.BasketLines.Delete(basketLine);
    return Json(new { result = "Redirect", url = Url.BackOrToReferrer() });
}
}

```

OrdersController.cs:

```

public class OrdersController : Controller
{
    [HttpGet]
    public ActionResult Index(OrderListParametersModel listParameters)
    {
        var currentUserId =
WebsiteManager.Users.GetByEmail(User.Identity.Name).Id;

```

		Коренівський М.О.			ІПЗ.КР.Б-121-23-ПЗ	Арк.
		Морозов А.В.				86
Змн.	Арк.	№ докум.	Підпис	Дата		

```

        if (listParameters == null)
            listParameters =
ObjectManager.GetInstance<OrderListParametersModel>();
        var loadOptions = listParameters.ToEntityLoadOptions();
        loadOptions.FieldFilters.Add("UserId", currentUserId.ToString());
        loadOptions.PageSize = Constants.DefaultPageSizes.Orders;

        var totalCount = WebsiteManager.Orders.GetTotalCount(loadOptions);
        if (totalCount < loadOptions.PageSize)
            listParameters.PageIndex = loadOptions.PageIndex = 1;

        var orders = WebsiteManager.Orders.GetList(loadOptions);
        var model = CreateOrderListModel(orders, totalCount, listParameters);

        return View(model);
    }

    [HttpGet]
    public ActionResult Details(int id)
    {
        var currentUserId =
WebsiteManager.Users.GetByEmail(User.Identity.Name).Id;
        var order = WebsiteManager.Orders.GetByIdAndUserId(id, currentUserId);
        if (order == null)
            return HttpNotFound();

        var model = ObjectManager.GetInstance<OrderEditModel>();
        foreach (var orderLine in order.OrderLines)
        {
            var orderLineModel = ObjectManager.GetInstance<OrderLineModel>();
            orderLineModel.Initialize(orderLine);
            model.OrderLines.Add(orderLineModel);
        }
        model.Initialize(order);

        return View(model);
    }

    private EntityListModel<OrderListItemModel, OrderListParametersModel>
CreateOrderListModel(IList<Order> orders, int totalCount, OrderListParametersModel
listParameters = null)
    {
        var model = ObjectManager.GetInstance<EntityListModel<OrderListItemModel,
OrderListParametersModel>>();
        model.ListParameters = listParameters;
        model.ListParameters.InitializeSortingOptions();
        model.TotalCount = totalCount;

        foreach (var order in orders)
        {
            var listItemModel = ObjectManager.GetInstance<OrderListItemModel>();
            listItemModel.Initialize(order);
            model.Items.Add(listItemModel);
        }

        return model;
    }
}

```

ProductsController.cs:

```

[Authorize(Roles = Constants.Roles.User)]
public class ProductsController : Controller
{
    [HttpGet]
    public ActionResult Index(ProductListParametersModel listParameters)

```

		Коренівський М.О.			ІІЗ.КР.Б-121-23-ІІЗ	Арк.
		Морозов А.В.				87
Змн.	Арк.	№ докум.	Підпис	Дата		

```

        {
            if (listParameters == null)
                listParameters =
ObjectManager.GetInstance<ProductListParametersModel>();
            var loadOptions = listParameters.ToEntityLoadOptions();
            loadOptions.PageSize = Constants.DefaultPageSizes.ShopProducts;

            var totalCount = WebsiteManager.Products.GetTotalCount(loadOptions);
            if (totalCount < loadOptions.PageSize)
                listParameters.PageIndex = loadOptions.PageIndex = 1;

            var products = WebsiteManager.Products.GetList(loadOptions);
            var productCategories =
WebsiteManager.ProductCategories.GetList(ObjectManager.GetInstance<IEntityListLoadOptions>());
            var model = CreateProductListModel(products, productCategories,
totalCount, listParameters);
            return View(model);
        }

        private EntityListModel<ProductListItemModel, ProductListParametersModel>
CreateProductListModel(IList<Product> products,
                        IList<ProductCategory> productCategories,
                        int totalCount,
                        ProductListParametersModel listParameters = null)
        {
            var model =
ObjectManager.GetInstance<EntityListModel<ProductListItemModel,
ProductListParametersModel>>();
            model.ListParameters = listParameters;
            model.ListParameters.InitializeSortingOptions();
            model.TotalCount = totalCount;

            foreach (var product in products)
            {
                var listItemModel = ObjectManager.GetInstance<ProductListItemModel>();
                listItemModel.Initialize(product);
                model.Items.Add(listItemModel);
            }

            return model;
        }

        [HttpGet]
        public ActionResult Details(int id)
        {
            var product = WebsiteManager.Products.GetById(id);
            if (product == null)
                return HttpNotFound();

            var model = ObjectManager.GetInstance<ProductDetailsModel>();
            model.Initialize(product);
            return View(model);
        }
    }
}

```

AppInitializer.cs:

```

public static class AppInitializer
{
    public static void Initialize()
    {
        RegisterTypes();
    }

    public static void RegisterTypes()

```

		Коренівський М.О.			ІІЗ.КР.Б-121-23-ІІЗ	Арк.
		Морозов А.В.				88
Змн.	Арк.	№ докум.	Підпис	Дата		

```

    {
        RegisterProviders();
        RegisterManagers();

        ObjectManager.RegisterType<IEntityListLoadOptions,
EntityListLoadOptions>();
    }

    public static void RegisterProviders()
    {
        ObjectManager.RegisterTypeSingleton<IAdminUserProvider,
AdminUserProvider>();
        ObjectManager.RegisterTypeSingleton<IUserProvider, UserProvider>();
        ObjectManager.RegisterTypeSingleton<IProductCategoryProvider,
ProductCategoryProvider>();
        ObjectManager.RegisterTypeSingleton<IProductProvider, ProductProvider>();
        ObjectManager.RegisterTypeSingleton<IBasketLineProvider,
BasketLineProvider>();
        ObjectManager.RegisterTypeSingleton<IOrderProvider, OrderProvider>();
        ObjectManager.RegisterTypeSingleton<IOrderLineProvider,
OrderLineProvider>();
    }

    public static void RegisterManagers()
    {
        ObjectManager.RegisterTypeSingleton<IAdminUserManager,
AdminUserManager>();
        ObjectManager.RegisterTypeSingleton<ITextManager, TextManager>();
        ObjectManager.RegisterTypeSingleton<IAuthenticationManager,
AuthenticationManager>();
        ObjectManager.RegisterTypeSingleton<ICacheManager, CacheManager>();
        ObjectManager.RegisterTypeSingleton<IUserManager, UserManager>();
        ObjectManager.RegisterTypeSingleton<IProductCategoryManager,
ProductCategoryManager>();
        ObjectManager.RegisterTypeSingleton<IProductManager, ProductManager>();
        ObjectManager.RegisterTypeSingleton<IBasketLineManager,
BasketLineManager>();
        ObjectManager.RegisterTypeSingleton<IOrderLineManager,
OrderLineManager>();
        ObjectManager.RegisterTypeSingleton<IOrderManager, OrderManager>();
    }
}

```

UrlExtensions:

```

public static class UrlExtensions
{
    public static string BackOrAction(this UrlHelper urlHelper, string action,
string controller = null)
    {
        var returnUrl = urlHelper.GetReturnUrl();
        if (!returnUrl.IsNullOrEmptyOrWhiteSpace())
            return returnUrl;

        return urlHelper.Action(action, controller);
    }

    public static string BackOrToReferrer(this UrlHelper urlHelper)
    {
        var returnUrl = urlHelper.GetReturnUrl();
        if (!returnUrl.IsNullOrEmptyOrWhiteSpace())
            return returnUrl;

        return
urlHelper.RequestContext.HttpContext.Request.UrlReferrer.PathAndQuery;
    }
}

```

		Коренівський М.О.			ІІЗ.КР.Б-121-23-ІІЗ	Арк.
		Морозов А.В.				89
Змн.	Арк.	№ докум.	Підпис	Дата		

```

        public static string GetReturnUrl(this UrlHelper urlHelper) =>
urlHelper.RequestContext.HttpContext.Request.QueryString["returnurl"];

        public static string Current(this UrlHelper urlHelper) =>
urlHelper.RequestContext.HttpContext.Request.Url.PathAndQuery;

        public static string ToItemListPage(this UrlHelper urlHelper, int? pageIndex)
        {
            return urlHelper.ModifyQueryString((qs) =>
            {
                if (pageIndex.HasValue)
                {
                    if (pageIndex == 0)
                        qs.Remove(QueryString.PageIndexParameterName);
                    else
                        qs[QueryString.PageIndexParameterName] = pageIndex.ToString();
                }
            });
        }

        public static string ModifyQueryString(this UrlHelper urlHelper,
Action<NameValueCollection> action, string originalUrl = null)
        {
            var uri = urlHelper.RequestContext.HttpContext.Request.Url;
            var qs = HttpUtility.ParseQueryString(uri.Query);
            action(qs);
            var query = qs.ToString();
            if (query.IsNullOrEmpty())
                return uri.AbsolutePath;
            else
                return uri.AbsolutePath + '?' + query;
        }
    }
}

```

basket.js:

```

$(document).on('click', '.add-to-cart', function () {
    var $buttonContainerPage;
    if ($('.product-list').length)
        $buttonContainerPage = $('.product-list');
    if ($('.product-details').length)
        $buttonContainerPage = $('.product-details');

    var url = $buttonContainerPage.data('addtobasketurl');
    var successMessage = $buttonContainerPage.data('product-added-message');
    var failMessage = $buttonContainerPage.data('product-not-added-message');
    data = {
        __RequestVerificationToken:
$buttonContainerPage.find('input[name="__RequestVerificationToken"]').val(),
        productId: $(this).data('productid')
    }
    $.post(url, data, function (data) {
        if (data.success)
            toastr.info(successMessage);
        else {
            toastr.info(failMessage);
        }
    });
});

```

deleteEntityPopup.js:

```

var popupSelector = '#deletePopup'

$(document).on('click', '.delete-entity-trigger', function () {
    var id = $(this).data('id');

```

		Коренівський М.О.			ІПЗ.КР.Б-121-23-ПЗ	Арк.
		Морозов А.В.				90
Змн.	Арк.	№ докум.	Підпис	Дата		

```

        $(popupSelector).data('id', id);
        $(popupSelector).modal(true);
    });

    $(popupSelector).on('click', '.btn-modal-proceed', function () {
        var url = $(popupSelector).data('url');
        var id = $(popupSelector).data('id');
        var data = {
            __RequestVerificationToken:
$(popupSelector).find('input[name="__RequestVerificationToken"]').val(),
            id: id
        }

        $.post(url, data, function (response) {
            if (response.result == "Redirect")
                window.location = response.url;
        })
    });

```

productList.js:

```

$(document).on('change', '.shop-product-filter-sorting', function () {
    $(this).closest('form').submit();
});

$(document).on('change', '.shop-product-filter-category', function () {
    var $form = $(this).closest('form');
    $form.find('#PageIndex').val(1);
    $form.submit();
});

```

bookEditPage.js:

```

$(".imageUploader").on('change', function () {
    var files = this.files;
    if (files && files[0]) {
        var reader = new FileReader();

        reader.onload = function (e) {
            $('product-image').attr('src', e.target.result).width(250).height(250);
        };

        reader.readAsDataURL(files[0]);
    }
});

```

Shop.hbm.xml:

```

<hibernate-mapping namespace="BusinessLayer.DomainModels.Shop"
assembly="BusinessLayer" auto-import="true" xmlns="urn:hibernate-mapping-2.2">
    <class table="ProductCategories" name="ProductCategory">
        <id name="Id" type="int" column="Id">
            <generator class="trigger-identity"/>
        </id>
        <property name="Name" type="string"/>
    </class>

    <class table="Products" name="Product">
        <id name="Id" type="int" column="Id">
            <generator class="increment"/>
        </id>
        <property name="Name" type="string"/>
        <property name="Author" type="string"/>
        <property name="CategoryId" type="int"/>
        <property name="Description" type="string"/>
        <property name="Price" type="decimal"/>
        <property name="Stock" type="int"/>
    </class>

```

		Коренівський М.О.			ІПЗ.КР.Б-121-23-ПЗ	Арк.
		Морозов А.В.				91
Змн.	Арк.	№ докум.	Підпис	Дата		

```

    <property name="Cover" type="CoverType"/>
    <property name="PagesCount" type="int"/>
</class>

<class table="BasketLines" name="BasketLine">
    <id name="Id" type="int" column="Id">
        <generator class="trigger-identity"/>
    </id>
    <property name="ProductId" type="int"/>
    <property name="UserId" type="guid"/>
    <property name="Quantity" type="int"/>
</class>
<class table="Orders" name="Order">
    <id name="Id" type="int" column="Id">
        <generator class="increment"/>
    </id>
    <property name="UserId" type="guid"/>
    <property name="ReceiverName" type="string"/>
    <property name="ReceiverSurname" type="string"/>
    <property name="ReceiverEmail" type="string"/>
    <property name="ReceiverPhoneNumber" type="string"/>
    <property name="ReceiverAddress" type="string"/>
    <property name="OrderDate" type="datetime"/>
    <property name="Status" type="OrderStatus"/>
    <property name="PaymentMethod" type="PaymentMethod"/>
    <bag name="OrderLines" table="OrderLines" lazy="false">
        <key column="OrderId"/>
        <one-to-many class="OrderLine"/>
    </bag>
</class>
<class table="OrderLines" name="OrderLine">
    <id name="Id" type="Guid" column="Id">
        <generator class="guid"/>
    </id>
    <property name="Quantity" type="int"/>
    <property name="Title" type="string"/>
    <property name="Author" type="string"/>
    <property name="Price" type="decimal"/>
    <property name="OrderId" type="int"/>
</class>
</hibernate-mapping>

```

Users.hbm.xml:

```

<hibernate-mapping namespace="BusinessLayer.DomainModels.Profile"
assembly="BusinessLayer" auto-import="true" xmlns="urn:nhibernate-mapping-2.2">
    <class table="AdminUsers" name="AdminUser" lazy="false">
        <id name="Id" type="Guid" column="Id">
            <generator class="guid"/>
        </id>
        <property name="Email" type="string"/>
        <property name="Password" type="string"/>
        <property name="Salt" type="string"/>
    </class>

    <class table="Users" name="User" lazy="false">
        <id name="Id" type="Guid" column="Id">
            <generator class="guid"/>
        </id>
        <property name="Name" type="string"/>
        <property name="Surname" type="string"/>
        <property name="Email" type="string"/>
        <property name="Password" type="string"/>
        <property name="Salt" type="string"/>
        <property name="Phone" type="string"/>
        <property name="Address" type="string"/>
    </class>

```

		Коренівський М.О.			ІІЗ.КР.Б-121-23-ІІЗ	Арк.
		Морозов А.В.				92
Змн.	Арк.	№ докум.	Підпис	Дата		

```

        <property name="CreationDate" type="Timestamp"/>
    </class>
</hibernate-mapping>

```

ProductInputModel.cshtml

```

@model ProductInputModel
<div class="row">
    <div class="form-group col-auto">
        @Html.ProductImage(Model.Id, Constants.ImageSize.Medium, "img-thumbnail img-fluid product-image")
    </div>
    <div class="col">
        <div class="form-group">
            @Html.LabelFor(m => m.Name)
            @Html.EditorFor(m => m.Name)
            <div class="invalid-feedback">@Html.ValidationMessageFor(m => m.Name)</div>
        </div>
        <div class="form-group">
            @Html.LabelFor(m => m.Author)
            @Html.EditorFor(m => m.Author)
            <div class="invalid-feedback">@Html.ValidationMessageFor(m => m.Author)</div>
        </div>
        <div class="form-group">
            @Html.LabelFor(m => m.CategoryId)
            @Html.DropDownListFor(m => m.CategoryId, Model.CategoryOptions,
            Html.GetText("SelectCategory").ToHtmlString(), new { @class = "form-control" })
            <div class="invalid-feedback">@Html.ValidationMessageFor(m => m.CategoryId)</div>
        </div>
    </div>
</div>
<div class="form-group">
    @Html.LabelFor(m => m.Description)
    @Html.EditorFor(m => m.Description, new { @class = "form-control" })
    <div class="invalid-feedback">@Html.ValidationMessageFor(m => m.Description)</div>
</div>
<div class="form-group">
    @Html.LabelFor(m => m.Price)
    @Html.EditorFor(m => m.Price, new { htmlAttributes = new { @class = "form-control", type = "number" } })
    <div class="invalid-feedback">@Html.ValidationMessageFor(m => m.Price)</div>
</div>
<div class="form-group">
    @Html.LabelFor(m => m.Cover)
    @Html.DropDownListFor(m => m.Cover, Model.CoverOptions, new { @class = "form-control" })
    <div class="invalid-feedback">@Html.ValidationMessageFor(m => m.Cover)</div>
</div>
<div class="form-group">
    @Html.LabelFor(m => m.PagesCount)
    @Html.EditorFor(m => m.PagesCount, new { htmlAttributes = new { @class = "form-control" } })
    <div class="invalid-feedback">@Html.ValidationMessageFor(m => m.PagesCount)</div>
</div>
<div class="form-group">
    @Html.LabelFor(m => m.Stock)
    @Html.EditorFor(m => m.Stock, new { htmlAttributes = new { @class = "form-control" } })
    <div class="invalid-feedback">@Html.ValidationMessageFor(m => m.Stock)</div>
</div>
<div class="form-group">
    @Html.LabelFor(m => m.Image, new { @class = "btn btn-primary" })

```

		Коренівський М.О.			ІПЗ.КР.Б-121-23-ПЗ	Арк.
		Морозов А.В.				93
Змн.	Арк.	№ докум.	Підпис	Дата		

```

        @Html.TextBoxFor(m => m.Image, new { type = "file", @class = "d-none
imageUploader" })
        <div class="invalid-feedback">@Html.ValidationMessageFor(m => m.Image)</div>
</div>

```

BaseLayout.cshtml

```

<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>@ViewBag.Title</title>
    @Scripts.Render("~/bundles/modernizr")
    @Styles.Render("~/Content/css")
    @Styles.Render("~/Content/admin/css")
</head>
<body class="d-flex flex-column">
    @RenderBody()

    @Scripts.Render("~/bundles/jquery")
    @Scripts.Render("~/bundles/jqueryval")
    @Scripts.Render("~/bundles/scripts")
    @Scripts.Render("~/bundles/bootstrap")
    @RenderSection("scripts", required: false)
</body>
</html>

```

ItemListFooter.cshtml

```

@model (int, int, int)

@{
    int totalItems = Model.Item1;
    int pageIndex = Model.Item2;
    int pageCount = (int)Math.Ceiling((decimal)(1.0 * totalItems / Model.Item3));
    pageCount = pageCount == 0 ? 1 : pageCount;
    string backButtonClass = "btn-primary" + (pageIndex == 1 ? " disabled" :
string.Empty);
    string nextButtonClass = "btn-primary" + (pageIndex == pageCount ? " disabled" :
string.Empty);
}

<div class="row items-table-footer">
    <div class="col">
        <p class="text-
secondary">@Html.GetText("TotalRecords").FormatWith(totalItems)</p>
    </div>
    @if (totalItems > 0)
    {
        <div class="col text-right">
            <p class="text-secondary d-inline pr-
2">@Html.GetText("PagingInformation").FormatWith(pageIndex, pageCount)</p>
            @Html.LinkButton(Url.ToItemListPage(pageIndex - 1), Html.Icon("chevron-
left"), backButtonClass)
            @Html.LinkButton(Url.ToItemListPage(pageIndex + 1), Html.Icon("chevron-
right"), nextButtonClass)
        </div>
    }
</div>

```

		Коренівський М.О.			ІПЗ.КР.Б-121-23-ІЗ	Арк.
		Морозов А.В.				
Змн.	Арк.	№ докум.	Підпис	Дата		94