

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»

Факультет комп'ютерних наук і технологій

(повне найменування факультету)

Кафедра програмних засобів

(повне найменування кафедри)

Пояснювальна записка

до дипломного проєкту (роботи)

бакалавр

(ступінь вищої освіти)

на тему РОЗРОБКА ВЕБСАЙТУ ДЛЯ ОБЛІКУ ТОВАРІВ НА СКЛАДІ
ТА ЗАБЕЗПЕЧЕННЯ ВНУТРІШНЬОЇ ВЗАЄМОДІЇ З ПІДПРИЄМСТВАМИ
WEBSITE DEVELOPMENT FOR INVENTORY MANAGEMENT AND
INTERNAL INTERACTION WITH ENTERPRISES

Виконав: студент 4 курсу, групи КНТ-111

Спеціальності 121 Інженерія програмного

(код і найменування спеціальності)

забезпечення

Освітня програма (спеціалізація)

Інженерія програмного забезпечення

ЩЕРБИНА Д.С.

(ПРИЗВИЩЕ та ініціали)

Керівник ПОДКОВАЛІХІНА О.О.

(ПРИЗВИЩЕ та ініціали)

Рецензент ГОЛУБ Т.В.

(ПРИЗВИЩЕ та ініціали)

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»
 (повне найменування закладу вищої освіти)

Факультет КНТ
 Кафедра програмних засобів
 Ступінь вищої освіти бакалавр
 Спеціальність 121 Інженерія програмного забезпечення
(код і найменування)
 Освітня програма (спеціалізація) Інженерія програмного забезпечення
(назва освітньої програми (спеціалізації))

ЗАТВЕРДЖУЮ

Завідувач кафедри ПЗ, д.т.н, проф.
Сергій СУББОТІН
 “ ” 2025 року

З А В Д А Н Н Я

НА ДИПЛОМНУ РОБОТУ СТУДЕНТА

ЩЕРБИНИ Дмитра Сергійовича
(ПРИЗВИЩЕ, ім'я, по батькові)

1. Тема проєкту (роботи) Розробка вебсайту для обліку товарів на складі та забезпечення внутрішньої взаємодії з підприємствами. Website Development for Inventory Management and Internal Interaction with Enterprises

керівник проєкту (роботи) к.ф.-м.н., доцент, ПОДКОВАЛІХІНА Олена Олександрівна,
(науковий ступінь, вчене звання, ПРИЗВИЩЕ, ім'я, по батькові)

затверджені наказом закладу вищої освіти від “28” квітня 2025 року № 209

2. Строк подання студентом проєкту (роботи) 02 червня 2025 року

3. Вихідні дані до проєкту (роботи) рекомендована література, технічне завдання

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити) 1. Аналіз предметної області. 2. Вибір та обґрунтування структури системи, яка проєктується. 3. Основні рішення щодо реалізації компонентів системи. 4. Керівництво програміста. 5. Керівництво оператора. 6. Тестування і результати

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень, кількість слайдів, плакатів)

Слайди презентації

6. Консультанти розділів проєкту (роботи)

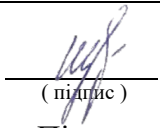
Розділ	ПРИЗВИЩЕ, ініціали та посада консультанта	Підпис, дата	
		завдання видав	прийняв виконане завдання
1-6 Основна частина	ПОДКОВАЛІХІНА О.О., доцент		
Нормоконтроль	БСЛОВА А.В., асистент		

7. Дата видачі завдання «14» квітня 2025 року.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломного проєкту (роботи)	Строк виконання етапів проєкту (роботи)	Примітка
1	Постановка завдання роботи.	1 тиждень	Завдання, ТЗ
2	Аналіз предметної області.	1 тиждень	Розділ 1
3	Вибір мови програмування та інших технологій розробки.	2 тиждень	Розділ 2
4	Розробка архітектури програми.	2 тиждень	Розділ 3
5	Розробка програми.	3-4 тижні	Розділ 3, 4
6	Тестування та експериментальне дослідження програмного забезпечення.	5 тиждень	Розділ 5, 6
7	Оформлення пояснювальної записки та документів до неї.	6 тиждень	Додатки
8	Нормоконтроль та рецензування.	7 тиждень	
9	Захист роботи.	8 тиждень	

Студент


(підпис)

Дмитро ЩЕРБИНА
(Ім'я ПРИЗВИЩЕ)

Керівник проєкту (роботи)

Підпис
(підпис)

Олена ПОДКОВАЛІХІНА
(Ім'я ПРИЗВИЩЕ)

РЕФЕРАТ

Пояснювальна записка до дипломної кваліфікаційної роботи бакалавра: 133 с., 20 табл., 35 рис., 4 дод., 18 джерел.

ВЕБСАЙТ, ОБЛІК ТОВАРІВ, ПІДПРИЄМСТВА-ПАРТНЕРИ, ЗАМОВЛЕННЯ, СКЛАД, C#, ASP.NET CORE MVC, MICROSOFT AZURE.

Об'єкт дослідження: процес розробки програмного забезпечення для автоматизації складського обліку та внутрішньої комунікації підприємств.

Предмет дослідження: методи та програмні засоби реалізації вебсайту для обліку товарів на складі та забезпечення внутрішньої взаємодії з підприємствами.

Мета роботи: підвищення ефективності обліку товарів на складі та оптимізація внутрішньої комунікації для мікро та малих підприємств з обмеженими ресурсами шляхом розробки спеціалізованого вебсайту.

Матеріали, методи та технічні засоби: об'єктно-орієнтоване програмування, мови програмування C# та JavaScript, середовище розробки Visual Studio 2022, систему керування базою даних MS SQL Server, фреймворки ASP.NET CORE MVC та EF Core, сервіс для розгортання вебсайту Microsoft Azure.

Результати. Створено вебзастосунок, що дає можливість підприємствам швидко і зручно організувати електронну комерцію для керування та реалізації власних виробів.

Висновки. Розроблено універсальний вебсайт для мікро та малих підприємств з обмеженими ресурсами, що надає основний функціонал для вирішення проблем з обліком товарів на складі та розповсюдженням своєї продукції підприємствами-партнерами. Для застосування наявні можливості для подальшого масштабування.

Галузь використання – організація електронної комерції малими та мікропідприємствами, що виробляють меблі з натуральної деревини.

ABSTRACT

Explanatory note to the diploma qualifying work of the bachelor: 133 pages, 20 tables, 35 figures, 4 appendixes, 18 sources.

WEBSITE, GOODS ACCOUNTING, PARTNER COMPANIES, ORDERS, WAREHOUSE, C#, ASP.NET CORE MVC, MICROSOFT AZURE.

Object of study is the process of developing software for automating warehouse accounting and internal communication of enterprises.

Subject of study are methods and software tools for implementing a website for accounting for goods in the warehouse and ensuring internal interaction with enterprises.

The purpose of the work is to improving the efficiency of warehouse accounting and optimizing internal communication for micro and small businesses with limited resources by developing a specialized website.

Materials, methods, and tools: object-oriented programming, C# and JavaScript programming languages, Visual Studio 2022 development environment, MS SQL Server database management system, ASP.NET CORE MVC and EF Core frameworks, Microsoft Azure website deployment service.

Results. A web application has been created that allows enterprises to quickly and conveniently organize e-commerce for managing and selling their own products.

Conclusions. A universal website for micro and small enterprises with limited resources has been developed, which provides basic functionality for solving problems with inventory accounting and distribution of their products by partner enterprises. The application has the potential for further scaling.

Field of use - organization of e-commerce by small and micro enterprises that produce furniture made of natural wood.

ПЕРЕЛІК СКОРОЧЕНЬ ТА УМОВНИХ ПОЗНАК

CSS – Cascading Style Sheets;
GUI – Graphical User Interface;
HTML – HyperText Markup Language;
HTTP – HyperText Transfer Protocol;
IDE – Integrated Development Environment;
LINQ – Language Integrated Query;
MVC – Model-View-Controller;
ORM – Object-Relational Mapping;
SQL – Structured Query Language;
WMS – Warehouse Management System;
БД – база даних;
Вт – ват;
ГБ – гігабайт;
ГГц – гігагерц;
ОС – операційна система;
ПЗ – програмне забезпечення;
СКБД – система керування базами даних;
ТЗ – технічне завдання.

ЗМІСТ

	С.
Перелік скорочень та умовних познач.....	7
Вступ.....	10
1 Аналіз предметної області.....	11
1.1 Огляд існуючих методів вирішення завдань.....	11
1.2 Огляд існуючих програмних засобів, що вирішують аналогічні завдання	14
1.2.1 Онлайн-сервіс USAP-Online	14
1.2.2 Онлайн-сервіс RemOnline	15
1.2.3 Інтернет-магазин «ТеМП-Меблі».....	16
1.2.4 Інтернет-магазин «MASIO»	17
1.3 Розробка ТЗ.....	20
1.3.1 Мета розробки та призначення.....	20
1.3.2 Основні вимоги	21
1.3.3 Вимоги до технічних засобів	22
1.3.4 Вимоги до програмної документації.....	23
1.3.5 Тестування	23
1.4 Висновки за розділом 1	24
2 Вибір та обґрунтування структури системи, яка проєктується	25
2.1 Вибір інструментарію розробки програмного забезпечення	25
2.1.1 Огляд особливостей мови програмування	26
2.1.2 Огляд особливостей середовища розробки.....	27
2.1.3 Огляд особливостей технології для розробки.....	28
2.1.4 Огляд особливостей СКБД.....	29
2.2 Структурна схема програми.....	31
2.3 Опис модулів системи, що розробляється.....	32
2.4 Висновки за розділом 2	33
3 Основні рішення щодо реалізації компонентів системи.....	35

	9
3.1 Огляд обраного шаблону проєктування	35
3.2 Опис реалізованих класів	36
3.3 Опис реалізованих представлень.....	50
3.4 Проєктування бази даних	56
3.5 Розгортання вебсайту.....	60
3.6 Висновки за розділом 3	66
4 Керівництво програміста.....	67
4.1 Призначення й умови застосування програми.....	67
4.2 Характеристики програми.....	67
4.3 Звертання до програми	69
4.4 Початкові та вихідні дані	72
4.5 Повідомлення програмісту.....	73
5 Керівництво оператора	74
5.1 Призначення програми	74
5.2 Умови виконання програми	74
5.3 Виконання програми.....	75
5.4 Повідомлення оператору	79
6 Тестування і результати.....	82
6.1 Опис методики проведення тестування.....	82
6.2 Результати використання розробленого програмного забезпечення	82
Висновки	92
Перелік джерел посилання	94
Додаток А Технічне завдання	96
Додаток Б Опис програми	104
Додаток В Текст програми	114
Додаток Г Слайди презентації	125

ВСТУП

У сучасних умовах цифровізації економіки важливу роль відіграє впровадження інформаційних технологій у діяльність бізнесу. Особливо це стосується мікро- та малих підприємств, які, на відміну від великих компаній, мають обмежені ресурси для автоматизації бізнес-процесів, зокрема в сфері виробництва, зберігання та збуту продукції. Одним із ефективних способів підвищення конкурентоспроможності таких підприємств є впровадження в роботу програмного забезпечення для виконання поставлених завдань.

Зокрема, слід звернути увагу на виробництво і продаж меблів з деревини. По-перше, деревообробка є традиційною галуззю в Україні, яка активно розвивається. По-друге, продукція з натурального дерева є затребуваною на ринку завдяки екологічності, довговічності та естетичній привабливості. Малі виробники дерев'яних меблів зазвичай працюють на індивідуальні замовлення або малими серіями, що потребує тіснішого контакту з клієнтом, оперативного оновлення асортименту й гнучких умов продажу — саме ті завдання, які можна ефективно вирішити за допомогою розробленого програмного забезпечення [1]-[3].

Актуальність створення спеціалізованого вебсайту полягає в тому, що більшість мікропідприємств не мають можливості використовувати складні та коштовні ІТ-рішення, а тому потребують простих, доступних, але водночас функціональних інструментів для організації електронної комерції.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Огляд існуючих методів вирішення завдань

Терміном «складський облік» називають процес збирання та опрацювання актуальної інформації про товари, які присутні на складі. Завдяки йому можна точно знати, який вантаж розташований у певних зонах, його кількість, вартість та ін. Ця інформація вкрай важлива для нормальної роботи бізнесу. Володіючи точними даними, з'являється можливість оптимізувати бізнес-процеси, підвищити загальний рівень ефективності працівників, а також досягти того, щоб замовлення оброблялися максимально швидко та без затримок [4].

Складський облік дозволяє:

- контролювати кількість вантажів, які знаходяться на складі. Власне, це є основним завданням складського обліку. Тобто фактична кількість товарів завжди збігатиметься з обліковим, тому не виникне ситуації, коли клієнт зробив замовлення, а на складі немає необхідної продукції;
- грамотно планувати закупівлю. Оскільки в будь-який момент можна переглянути інформацію про те, скільки та яких товарів є на складі, перед підприємством відкриваються можливості для детального планування;
- спростити пошук необхідної продукції та проведення інвентаризації. Системи складського обліку нададуть всю необхідну інформацію, тому співробітникам буде легше шукати потрібні товари та формувати замовлення;
- створювати повноцінні довідники продукції. Якщо на складі зберігається багато різноманітних товарів, то можна об'єднати їх у групи та категорії для спрощення обліку на складі.
- знизити кількість помилок та фактів розкрадання товарів. Після впровадження програми складського обліку забезпечується можливість комплексного контролю всіх процесів, пов'язаних із товарними операціями, зокрема оприбуткування, переміщення та списання. Це сприяє підвищенню

точності обліку та мінімізації ймовірності виникнення помилок або недоліків у складських операціях [4].

Основними варіантами забезпечення виконання вищезазначених функцій є використання десктопних та веборієнтованих систем.

Вибір між хмарною та локальною WMS залежить від потреб бізнесу, бюджету та операційних пріоритетів. Для компаній, які прагнуть гнучкості, масштабованості та мінімального залучення ІТ-спеціалістів, хмарна WMS часто є кращим вибором. Вона забезпечує віддалений доступ, безперебійне оновлення та менші початкові витрати, що робить її ідеальним рішенням для зростаючих компаній та компаній з розподіленими командами. Однак, якщо компанія надає перевагу повному контролю над даними, працює в регіонах з обмеженим доступом до Інтернету або обробляє конфіденційну інформацію, локальна система управління складом може бути більш прийнятною. Цей варіант гарантує, що дані залишаються на місці, надаючи повний нагляд і контроль над безпекою та налаштуваннями [5].

Окрім ведення складського обліку, однією з основних цілей будь-якого бізнесу є розширення клієнтської бази та збільшення обсягів продажу. В умовах цифрової трансформації підприємства отримують унікальну можливість виходу на глобальні ринки за допомогою електронної комерції. Зокрема, створення інтернет-магазину стає ефективним інструментом для досягнення цієї мети.

За статистичними даними, у 2022 році обсяг онлайн-продажів в Україні сягнув 151 млрд грн, що свідчить про стрімке зростання значення електронної торгівлі. Відтак, цифрова присутність у мережі Інтернет виступає одним із ключових чинників конкурентоспроможності суб'єкта господарювання на сучасному ринку [6].

Нижче наведено додаткові статистичні дані, що підтверджують переваги наявності інтернет-магазину електронної торгівлі:

- очікується, що до 2040 року 95% покупок світу будуть відбуватися через електронну торгівлю;

– 79% українців купували товари або послуги в Інтернеті в 2020 році.

На основі цих даних можна зробити висновок, що електронна торгівля відіграє фундаментальну роль у збільшенні продажів. Це означає, що створення інтернет-магазину є одним із найбільш перспективних напрямків не лише для нових підприємств, але й для вже існуючого бізнесу [7].

Інтернет-магазин являє собою електронний ресурс у вигляді вебсайту з інтегрованим каталогом продукції, який забезпечує можливість прямого продажу товарів кінцевому споживачеві з урахуванням організації доставки. Уся взаємодія – розміщення інформації, оформлення замовлення та укладення угоди – відбувається безпосередньо на вебплатформі [8].

Функціонування сучасного інтернет-магазину вимагає наявності чітко визначених функціональних блоків, що забезпечують зручність використання ресурсу для потенційних користувачів. До основних інформаційних компонентів належать: перелік товарів та послуг, прайс-листи, контактні дані (зокрема адреса, номер телефону, факс, електронна пошта), система пошуку по сайту, опис умов співпраці, варіанти оплати та способи доставки. Крім того, ефективне функціонування інтернет-магазину неможливе без реалізації таких елементів, як можливість реєстрації користувачів, створення особистого кабінету, функція додавання товарів до «Кошика», сторінка порівняння товарів, а також інтеграція з онлайн-консультантом для оперативного зворотного зв'язку [8].

На основі вищесказаного можна зробити висновок, що для мікро/малих підприємств-виробників з обмеженими ресурсами слід розглядати варіант використання веборієнтованого програмного застосунку для забезпечення зручного ведення як складського обліку так і продажу товарів. При цьому, розповсюдження виготовленої продукції слід організувати через підприємства-партнерів, які мають статус офіційних дилерів підприємства-виробника. Це обумовлено тим, що виробник має обмежені ресурси, в тому числі людські, і при малих масштабах виробництва не може забезпечити самостійне розповсюдження своєї продукції кожному клієнту-користувачу.

1.2 Огляд існуючих програмних засобів, що вирішують аналогічні завдання

При огляді існуючих засобів для вирішення поставленого завдання було розглянуто різні типи вебзастосунків – вебсервіси для обліку товарів та Інтернет-магазини.

1.2.1 Онлайн-сервіс USAP-Online

USAP-Online [9] — це веб-сервіс для автоматизації обліку товарів, складів і поставчань, орієнтований переважно на підприємства з фізичними товарними залишками (рис. 1.1). Підходить для малого та середнього бізнесу в торгівлі, дистрибуції та виробництві.

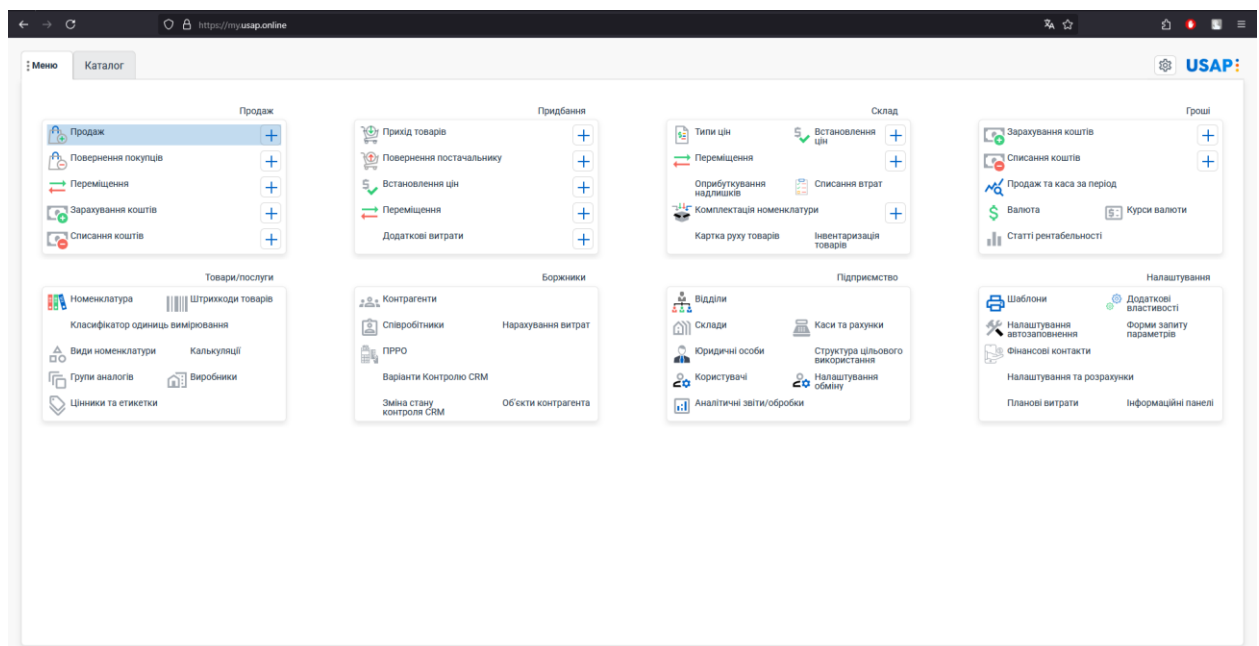


Рисунок 1.1 – Головне меню вебсервісу USAP-Online [9]

Серед переваг сервісу можна виділити наступні пункти:

— наявність основних напрямів обліку (продаж, придбання, товари, контрагенти, склад, підприємство, гроші), що особливо цінно для ведення середнього бізнесу;

- можливість формувати звіти та передивлятися аналітику;
- можливість відслідковувати переміщення товарів (як замовлень так і поставок);
- врахування об'єму товарів.

Серед недоліків можна виділити наступне:

- відсутність автоматизації, усі операції слід виконувати самостійно;
- користувацький інтерфейс занадто навантажений і незрозумілий;
- не враховується наповненість складу при веденні обліку товарів.

1.2.2 Онлайн-сервіс RemOnline

RemOnline [10] — це сучасна хмарна система для управління бізнесом, орієнтована на сервісні компанії, виробництво під замовлення, роздрібну торгівлю й індивідуальні послуги (рис. 1.2). Підходить для бізнесів, де важливе управління замовленнями, працівниками, складом та фінансами.

Серед переваг сервісу можна виділити наступні пункти:

- зручний і достатньо зрозумілий інтерфейс, особливо для нових користувачів;
- можливість формувати звіти та передивлятися аналітику;
- можливість записувати переміщення товарів.

Серед недоліків можна виділити наступне:

- відсутність автоматизації, усі операції слід виконувати самостійно;
- не враховується об'єм товарів при веденні обліку;
- не враховується наповненість складу при веденні обліку товарів.

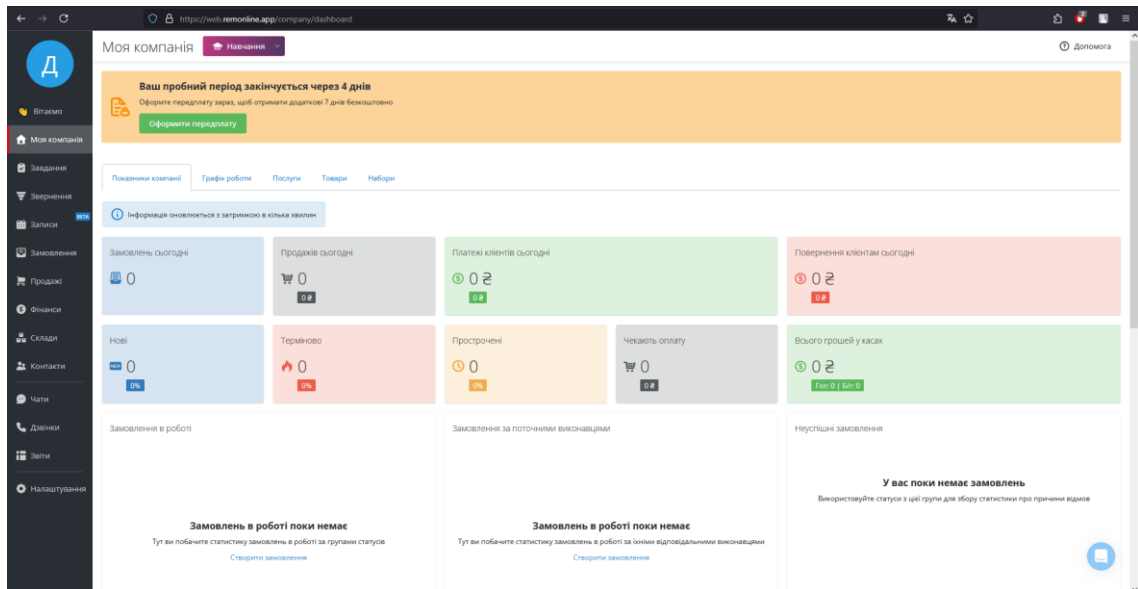


Рисунок 1.2 – Головне меню вебсервісу RemOnline [10]

1.2.3 Інтернет-магазин «ТеМП-Меблі»

Інтернет-магазин «ТеМП-Меблі» [11] представляє собою класичний вебсайт для розповсюдження товарів обраної тематики. В даному випадку це меблі, що виготовлені з деревини. Головна сторінка сайту зображена на рисунку 1.3.

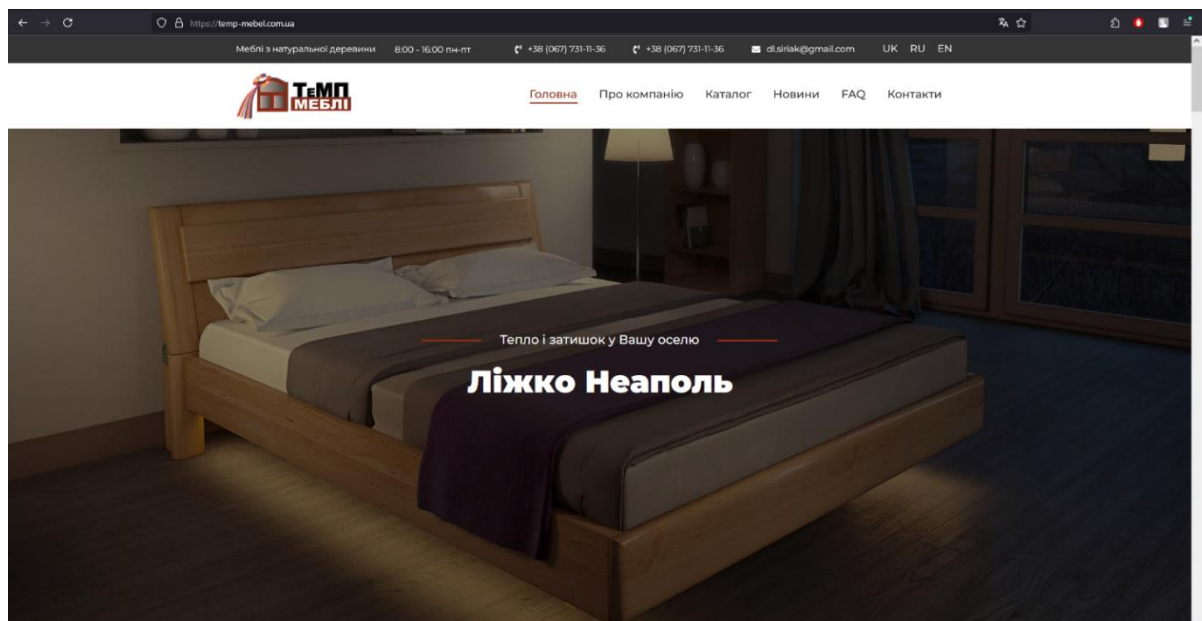


Рисунок 1.3 – Головна сторінка інтернет-магазину «ТеМП-Меблі» [11]

Серед переваг магазину можна виділити наступне:

- простий зрозумілий інтерфейс;
- зручна навігація в каталозі товарів;
- наявність детальних характеристик для товарів різних категорій;
- гарна швидкодія сайту.

Серед недоліків слід зазначити:

- відсутність пошуку товарів за ключовим словом;
- не відображається наявність/відсутність товару;
- відсутня реєстрація для оформлення замовлень;
- відсутність валідації полів при оформленні замовлень;
- неможливо продивитись зроблені замовлення.

1.2.4 Інтернет-магазин «MASIO»

Інтернет-магазин «MASIO» [12] являє собою типовий вебсайт для реалізації товарів певної категорії — зокрема, меблів, виготовлених з дерева. Головна сторінка сайту зображена на рисунку 1.4.

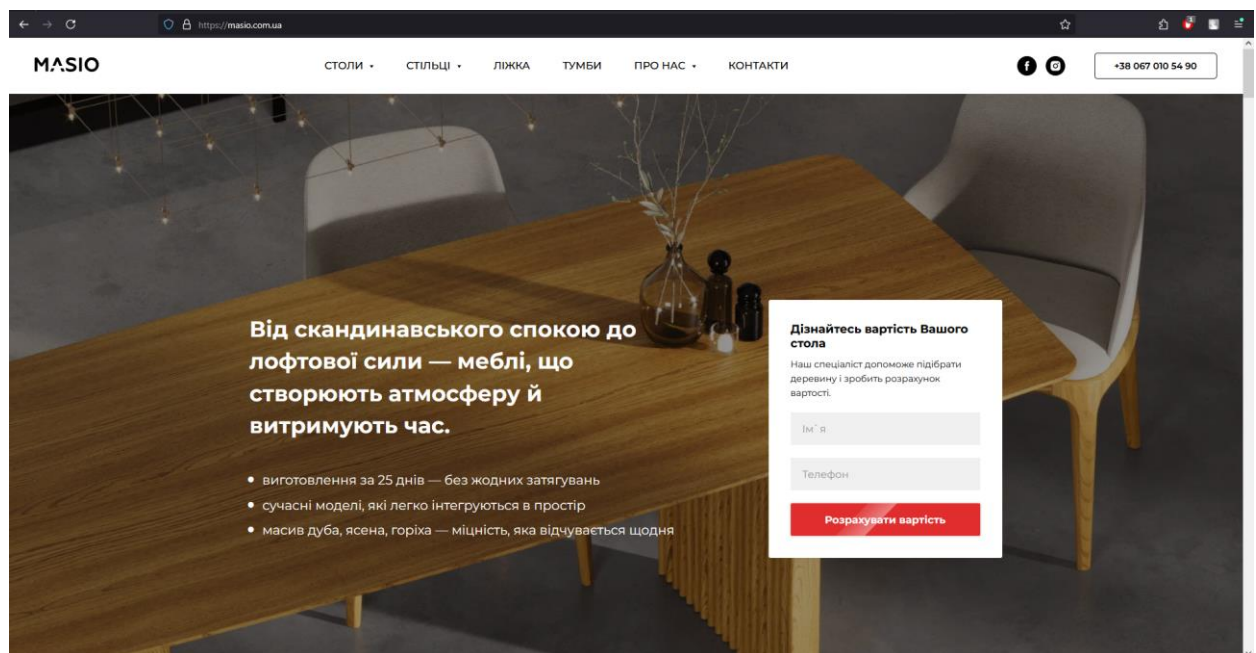


Рисунок 1.4 – Головна сторінка інтернет-магазину «MASIO» [12]

Серед переваг магазину можна виділити наступне:

- простий зрозумілий інтерфейс;
- зручна навігація в каталозі товарів;
- наявність детальних характеристик для товарів різних категорій;
- гарна швидкодія сайту.

Серед недоліків слід зазначити:

- відсутність пошуку товарів за ключовим словом;
- не відображається наявність/відсутність товару;
- відсутня реєстрація для оформлення замовлень;
- відсутність валідації полів при оформленні замовлень;
- неможливо продивитись зроблені замовлення.

1.2.5 Порівняльні таблиці існуючих програмних продуктів із запропонованим рішенням

В таблиці 1.1 наведено порівняння продукту, що розробляється, та існуючих вебсервісів для ведення обліку товарів.

Таблиця 1.1 – Порівняльна таблиця вебсервісів для ведення обліку товарів

	Сервіс USAP-Online	Сервіс RemOnline	Вебзастосунок, що розроблюється
1	2	3	4
Врахування об'єму товарів	+	—	+
Врахування наповненості складу	—	—	+
Інтуїтивно зрозумілий інтерфейс	— (дуже важкий для сприйняття)	+	+

Кінець таблиці 1.1

1	2	3	4
Автоматичне формування замовлення	—	—	+ (За рахунок поєднання з Інтернет-магазином)
Автоматичне внесення змін при оформленні замовлення	—	—	+ (За рахунок поєднання з Інтернет-магазином)
Керування переміщеннями	+	+	— (Товар самостійно виробляється, а при покупці вивозиться замовниками, тому відстеження переміщень не потрібне)
Налаштування списку партнерів	+	+	+
Звітність	+	+	+
Аналітика	+	+	— (Планується на перших етапах масштабування)

В таблиці 1.2 наведено порівняння продукту, що розробляється та існуючих Інтернет-магазинів для розповсюдження товарів.

Таблиця 1.2 – Порівняльна таблиця інтернет-магазинів для реалізації виготовленої продукції

	Інтернет-магазин «ТеМП-Меблі»	Інтернет-магазин «Masio»	Вебзастосунок, що розроблюється
1	2	3	4
Пошук товарів	—	—	+
Перегляд каталогу товарів за категоріями	+	+	—

Кінець таблиці 1.2

1	2	3	4
Інтуїтивно зрозумілий інтерфейс	+	+	+
Надання детальних характеристик	+	+	+
Відображення наявності товару	—	—	+
Необхідність реєстрації для замовлення	— (Окрім цього, дані що вводяться для оформлення замовлення не валідуються)		+
Перегляд власних замовлень та керування ними	—	—	+
Швидкодія сайту	+	+	+

1.3 Розробка ТЗ

1.3.1 Мета розробки та призначення

Метою розробки даного вебсайту є створення рішення, що включало б у собі функціонал для керування складом та забезпечення взаємодії з партнерами-підприємствами, тобто, розробка універсального вебсайту для мікро та малих підприємств з обмеженими ресурсами. Даний застосунок буде містити в собі функціонал як застосунків для обліку товарів на складі так і Інтернет-магазинів для продажу виготовленої продукції. Таким чином, підприємство отримає вебсайт з наданням основного функціоналу для вирішення проблем з обліком товарів, оформленням замовлень, розповсюдженням своєї продукції підприємствами-партнерами. При цьому залишається можливість масштабування застосунку у різних напрямках.

1.3.2 Основні вимоги

Програмне забезпечення має надавати можливість авторизації на сайті та розподіляти користувачів на три ролі: адміністратор, партнери та гості.

Гості: користувачі, що не мають власного акаунта на сайті. Мають можливість переглядати каталог доступної продукції та детальні характеристики обраного товару.

Партнери: користувачі-підприємства, які мають зареєстрований адміністратором акаунт на сайті. Мають можливість переглядати каталог доступної продукції та детальні характеристики обраного товару, а також оформлювати замовлення на товар в декілька кліків. Список замовлень та детальну інформацію про них можна буде передивитись на відповідній сторінці сайту. За бажанням – партнер зможе скасувати замовлення.

Адміністратор — користувач від підприємства-власника сайту, що має права на повне керування сайтом, а саме:

- реєстрація акаунтів для підприємств-партнерів;
- можливість переглядати каталог доступної продукції та детальні характеристики, а також редагувати інформацію про обраний товар;
- можливість реєструвати новий товар у каталозі;
- можливість переглядати інформацію про наповненість складу, та додавати нові товари на склад. Максимальна кількість товару для додавання обмежується вільним місцем на складі та об'ємом однієї одиниці обраного товару;
- переглядати історію керування товарами, яка формується при виконанні таких дій, як: реєстрація, редагування і додавання товару на склад;
- передивлятись усі дійсні акаунти, детальну інформацію про них та виконувати з ними наступні дії: редагувати інформацію акаунту, змінювати пароль від акаунту та деактивувати акаунт;
- можливість продивитись список всіх замовлень та детальну інформацію про них. Актуальні замовлення можна позначити як виконані.

При взаємодії адміністратора з акаунтом партнера слід забезпечити надсилання email-повідомлень відповідному підприємству про виконання конкретних дій адміністратором.

Інтерфейс вебсайту має бути простим та інтуїтивно зрозумілим без необхідності звертатись до додаткової документації щодо експлуатації застосунку.

1.3.3 Вимоги до технічних засобів

У разі використання власного сервера для роботи вебсайту, до серверної частини висуваються наступні вимоги:

- операційна система: Windows Server 2022;
- процесор: рекомендується процесор Intel Xeon E5-4620@8x2,2GHz або краще;
- пам'ять: оперативна пам'ять має бути достатньою для запуску та обробки запитів, залежить від обсягу даних і навантаження на сервер. Рекомендується не менше 16 ГБ оперативної пам'яті;
- сховище даних: у разі можливості, встановлення бази даних на SSD може значно збільшити продуктивність роботи програми. Рекомендований обсяг накопичувача становить 500ГБ;
- середовище виконання: необхідне встановлене середовище виконання .NET 5/6/7 або .NET Core на сервері для запуску серверної частини додатка.

При використанні хмарного хостингу або платформи як сервісу (PaaS - Platform as a Service), відповідальність за апаратні ресурси лягає на провайдера, і вимоги можуть відрізнятися.

Для якісної роботи клієнтської частини програми рекомендується персональний комп'ютер до складу якого входять наступні компоненти та технічні засоби:

- процесор Intel Core i3-8100 (3.6 ГГц)(Intel UHD Graphics 630) з тактовою частотою, ГГц – 3,6;
- материнська плата Asus Prime H310M-R R2.0 (s1151 v2, Intel H310, PCI-Ex16);
- оперативна пам'ять DDR3 з об'ємом 4ГБ;
- блок живлення 450 Вт;
- об'єм логічної пам'яті SSD – 120 ГБ;
- інструменти вводу інформації – клавіатура та миша;
- засоби виводу інформації – монітор з роздільною здатністю 1920x1080.

Для запуску програми достатньо мати встановлену на комп'ютері будь-яку версію ОС та один із встановлених браузерів (Google Chrome, Mozilla Firefox, Opera, Microsoft Edge).

1.3.4 Вимоги до програмної документації

Для розробленого програмного забезпечення має бути оформлена наступна документація: технічне завдання, керівництво програміста, керівництво оператора, опис програми, тексти основних функціональних алгоритмів програми.

1.3.5 Тестування

Тестування програмного забезпечення включає в себе:

- ручне тестування функціоналу шляхом практичного використання сайту;
- тестування верстки за допомогою вбудованих у браузер інструментів розробника;
- виправлення виявлених помилок.

1.4 Висновки за розділом 1

В першому розділі було проаналізовано предметну область для подальшої розробки вебсайту для ведення обліку товарів та забезпечення внутрішньої взаємодії з підприємствами.

В ході аналізу було визначено цільову аудиторію, тобто, для яких підприємств буде розроблятися програмне забезпечення. Після цього було розглянуто існуючі методи вирішення завдань, проведено порівняння аналогічних за призначенням програмних засобів.

На підставі отриманих даних визначено вимоги до нового вебсайту.

Актуальність розробки програмного забезпечення була підтверджена тим фактом, що незважаючи на важливість правильної організації електронної комерції, все ж таки існують мікро та малі підприємства, що мають обмежені ресурси, але все одно потребують доступних і універсальних інструментів для ведення обліку товарів на складі та реалізації власної продукції.

2 ВИБІР ТА ОБГРУНТУВАННЯ СТРУКТУРИ СИСТЕМИ, ЯКА ПРОЄКТУЄТЬСЯ

2.1 Вибір інструментарію розробки програмного забезпечення

У рамках цього проєкту було ухвалено рішення використовувати специфічний стек технологій, спираючись на низку факторів і вимог. Перед вибором набору інструментів було враховано особливості завдання, а також потреби в ефективній розробці та функціональності додатка.

На підставі аналізу вимог було обрано для back-end частини мову програмування C#, яка забезпечує потужні можливості розроблення, велику кількість інструментів та широку підтримку з боку Microsoft. Для front-end частини обрано JavaScript через невелику динаміку на сторінках сайтах та скрипти, що не потребують використання додаткових інструментів, таких як React чи Angular.

Обраною технологією для розробки став ASP .NET Core (MVC) - сучасний кросплатформний фреймворк для створення веб-додатків, який надає розробникам можливість створювати масштабовані, продуктивні та безпечні веб-додатки. ASP.NET Core MVC реалізує шаблон проєктування Model-View-Controller, що сприяє чіткому розділенню відповідальностей у структурі додатку. Такий підхід дозволяє легше підтримувати код, тестувати окремі компоненти, а також забезпечує гнучкість у розвитку програмного продукту.

Для зберігання даних було обрано Microsoft SQL Server, що зумовлено його широким розповсюдженням, надійністю, масштабованістю та можливостями інтеграції з обраними технологіями.

Нарешті, як ORM фреймворк було обрано Entity Framework Core. EF Core забезпечує зручну взаємодію з базами даних і дає змогу зосередитися на логіці застосунку, оминаючи необхідність вручну писати SQL-запити.

2.1.1 Огляд особливостей мов програмування

В таблиці 2.1 наведено порівняння мов програмування для розробки back-end частини.

В результаті порівняння було обрано мову програмування C#, використання якої забезпечить можливість масштабування вебсайту як в напрямі обліку так і в напрямі продажу товарів. Багатий вибір бібліотек та фреймворків забезпечить надійний процес розробки, а повна інтеграція з Microsoft 365 дозволить формувати якісні звіти та забезпечить комфортну роботу з даними. Крім того, використання платформи .NET Core гарантує високу продуктивність і безпеку, що робить C# оптимальним вибором для створення стабільного та вебзастосунку, що буде розширюватись.

Таблиця 2.1 – Порівняння мов програмування для back-end частини

	C#	Java	Python
Продуктивність і оптимізація	Висока	Середня	Низька
Простота вивчення	Середня	Середня	Висока
Статична типізація	Так	Так	Ні
Швидкість розробки	Середня	Середня	Висока
Підтримка + документація	Висока	Висока	Висока
Оцінка наявності обширних бібліотек та фреймворків для створення вебзастосунків	Висока	Висока	Середня
Масштабованість	Висока	Висока	Середня
Безпека	Висока	Висока	Середня
Автоматичне керування пам'яттю	Так	Так	Так
Інтеграція з Microsoft 365	Повна	Обмежена	Обмежена

Зважаючи на те, що вебсайт розробляється як універсальний і поєднує в собі функціонал інтернет-магазину та вебсервісу для ведення обліку товарів на складі, під час розробки слід зосередити увагу саме на функціональних вимогах. До того ж, динаміка на сторінках сайту низька і не потребує використання надпотужних інструментів, таких як React або Angular. В такому

разі front-end частині буде приділено менше часу. Тому для оформлення зовнішнього інтерфейсу вебсайту буде використано класичний набір інструментів, що включає в собі: мову розмітки HTML для структурування вмісту веб-сторінок, CSS для стилізації та візуального оформлення елементів інтерфейсу, і мову програмування JavaScript для виконання скриптів та часткового оновлення сторінки (коли це необхідно).

2.1.2 Огляд особливостей середовища розробки

В таблиці 2.2 наведено порівняння найпоширеніших IDE для розробки вебсайтів мовою C#.

В результаті порівняння було обрано інтегроване середовище розробки Visual Studio 2022. Розробник даної IDE та мови C# один – Microsoft. Це робить зроблений вибір більш доцільним через отримання підтримки від офіційного розробника одразу для обох інструментів. Обране IDE є найфункціональнішим серед існуючих аналогів, що дозволяє комфортно вести розробку незважаючи на високу ресурсоємність, а великі можливості при відлагодженні коду полегшать написання та підтримку коду.

Таблиця 2.2 – Порівняння існуючих IDE для розробки мовою C#

	Visual Studio 2022	JetBrains Rider
1	2	3
Кросплатформеність	Ні, тільки Windows	Так
Функціональність	Висока	Середня
Продуктивність	Висока	Висока
Ресурсоємність	Висока	Середня
Підтримка NuGet-пакетів	Повна	Повна
Інструменти для роботи з базами даних	Вбудовані	Вбудовані
Можливості для відлагодження	Високі	Середні
Інтеграція з Git	Вбудована	Вбудована

Кінець таблиці 2.2

1	2	3
Підтримка спільнотою	Активна підтримка ком'юніті. Офіційна підтримка від Microsoft	Активна підтримка ком'юніті
Ціна	Безкоштовна тільки для некомерційного використання	

2.1.3 Огляд особливостей технології для розробки

В таблиці 2.3 наведено порівняння існуючих технологій для розробки вебсайту мовою C#.

В результаті порівняння було обрано ASP.Net Core (MVC). Такий вибір забезпечить гарну структурованість коду завдяки використанню шаблону MVC(Model-View-Controller). При подальшому розширенні вебсайту структурований код полегшить написання і редагування коду. Як вже було зазначено при виборі мови програмування – сторінки сайту мають досить невелику динамічність, тому розробка односторінкового застосунку не є необхідним, відповідно і використання Blazor втрачає свою актуальність.

Таблиця 2.3 – Порівняння існуючих технологій для розробки вебсайту мовою C#

	Razor Pages	ASP .Net Core (MVC)	Blazor
1	2	3	4
Тип застосунку	Серверний вебзастосунок	Серверний вебзастосунок	SPA (односторінковий застосунок)
Рендерінг	На сервері	На сервері	На клієнті або на сервері
Продуктивність	Висока	Висока	Середня
Безпека	Висока	Висока	Середня
Складність розробки	Низька	Середня	Висока
Структурованість коду	Низька	Висока	Низька

Кінець таблиці 2.3

1	2	3	4
Складність масштабування	Висока	Низька	Середня
Використання C# для front-end частини	Ні	Ні	Так
Рейтинг в спільноті	Високий	Високий	Низький

2.1.4 Огляд особливостей СКБД

В таблиці 2.4 наведено порівняння існуючих СКБД для взаємодії бази даних з вебсайтом.

В результаті порівняння було прийнято рішення використовувати MS SQL Server. Даний вибір обумовлено тим, що СКБД має високу продуктивність, помірну складність налаштування та забезпечує високу безпеку даних. Додатковим плюсом, є те, що SQL Server розроблений та активно підтримується компанією Microsoft, а зважаючи на те, що раніше обрані мова програмування, IDE та фреймворк для розробки мають того ж розробника – це підвищує надійність системи та зменшує шанси на появу проблем при розробці. А це в свою чергу, підвищує швидкість розробки вебсайту.

Таблиця 2.4 – Порівняння СКБД

	MySQL	PostgreSQL	MS SQL Server
1	2	3	4
Тип ліцензії	Open Source	Open Source	Комерційна, з безкоштовною версією
Продуктивність	Середня	Висока	Висока
Складність налаштування	Середня	Висока	Середня

Кінець таблиці 2.4

1	2	3	4
Здатність до масштабування	Середня	Висока	Висока
Типи даних	Обмежений вибір	Широкий вибір (включаючи користувацькі типи)	
Підтримка ACID	Так	Так	Так
Безпека	Низька	Середня	Висока
Оцінка можливостей відновлення резервних даних	Середня	Висока	Висока
Документація і спільнота	Широка спільнота, багато ресурсів	Відмінна документація і спільнота	Відмінна підтримка Microsoft, велика спільнота

Додатково, було порівняно існуючі ORM-фреймворки. Результати порівняння наведено в таблиці 2.5.

В результаті порівняння для використання було обрано Entity Framework Core. Даний вибір дозволить пришвидшити налаштування БД та побудову запитів за рахунок використання LINQ замість SQL, покращеної роботи зі зв'язками між сутностями та підтримки міграцій.

Таблиця 2.5 – Порівняння існуючих ORM-фреймворків

	Entity Framework Core	Drapper
1	2	3
Тип	ORM	Micro-ORM
Продуктивність	Середня	Висока
Написання запитів вимагає використання SQL	Ні (використовується LINQ)	Так
	Entity Framework Core	Drapper
Автоматичне управління зв'язками	Так	Ні
Підтримка навігаційних властивостей	Так	Ні

Кінець таблиці 2.5

Легкість роботи зі зв'язками	Висока	Низька
Підтримка міграцій	Так	Ні

2.2 Структурна схема програми

На рисунку 2.1 наведено структурну схему програми, на якій видно розподіл функціоналу по модулях та по ролях користувачів. Жовтими стрілками наведено обмін даними між користувачем та БД, червоними – завантаження даних до БД, синіми – завантаження даних з БД.

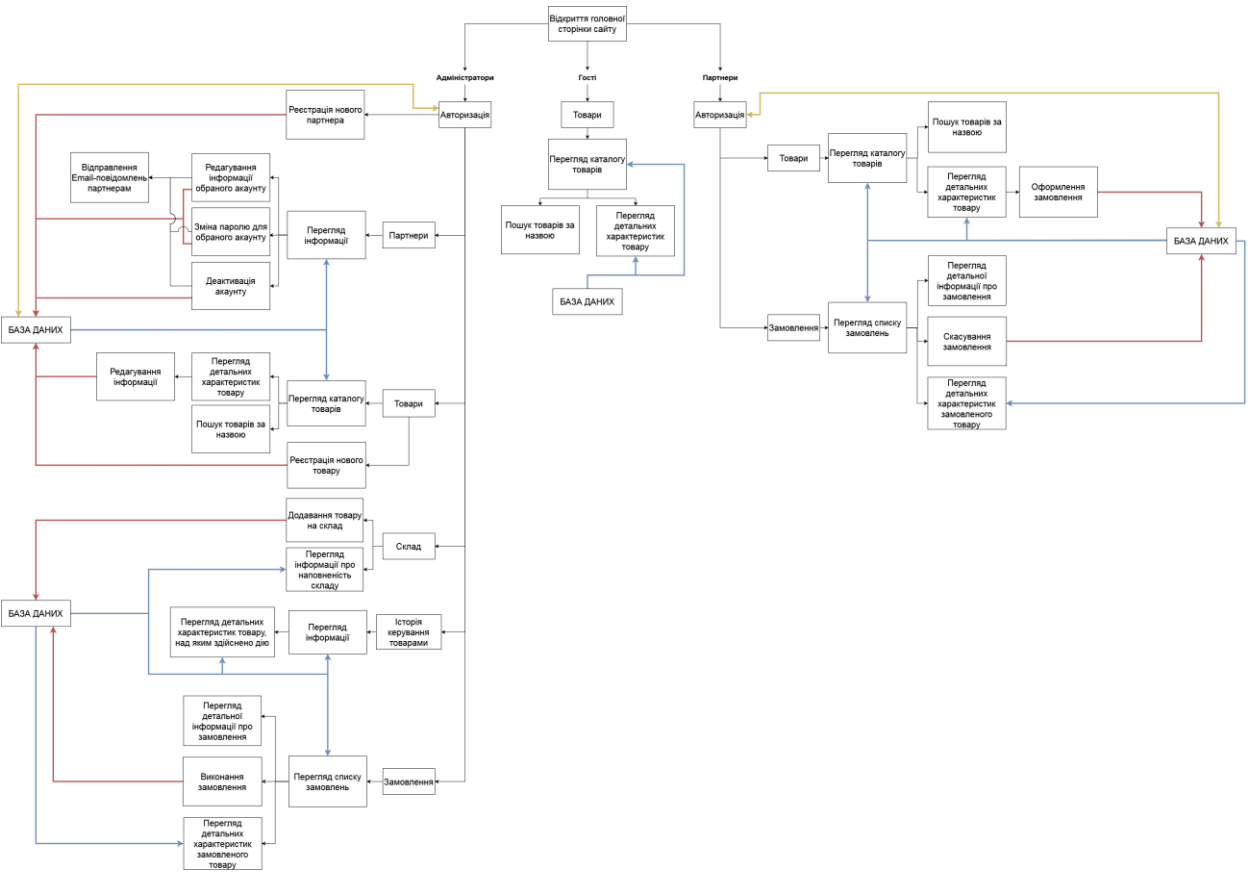


Рисунок 2.1 – Структурна схема ПЗ

2.3 Опис модулів системи, що розробляється

У таблиці 2.5 наведено опис модулів ПЗ з визначенням їх основних функцій.

Таблиця 2.5 – Опис модулів ПЗ з визначенням основних функцій

Модуль	Опис
1	2
Авторизація	Даний модуль охоплює усі сторінки графічного інтерфейсу з прилеглими елементами керування, запити до БД, контролери та моделі, що дають можливість проводити валідацію вхідних даних і реєструвати нових партнерів (для адміністраторів) та забезпечити безпечний вхід в систему тільки для зареєстрованих користувачів.
Партнери	Даний модуль охоплює усі сторінки графічного інтерфейсу з прилеглими елементами керування, запити до БД, контролери для представлень та моделі даних, що дають можливість проводити валідацію вхідних даних і виконувати адміністраторам наступні дії із списком партнерів: перегляд списку наявних партнерів та детальної інформації про них, редагування інформації про партнера, зміна паролю від акаунту, деактивація акаунту.
Товари	Даний модуль охоплює усі сторінки графічного інтерфейсу з прилеглими елементами керування, запити до БД, контролери для представлень та моделі даних, що дають можливість проводити валідацію вхідних даних і виконувати наступні дії із товарами: для усіх користувачів – переглядати каталог товарів, шукати товари за введеною назвою, переглядати детальні характеристики обраного товару. Для адміністраторів – реєструвати новий товар у каталозі, редагувати інформацію про обраний товар.
Склад	Даний модуль охоплює усі сторінки графічного інтерфейсу з прилеглими елементами керування, запити до БД, контролери для представлень та моделі даних, що дають можливість проводити валідацію вхідних даних і виконувати наступні дії із товарами: для адміністраторів - додавати товар на склад, враховуючи наповненість складу та об'єм товару, що додається.

Кінець таблиці 2.5

1	2
Історія керування товарами	Даний модуль охоплює усі сторінки графічного інтерфейсу з прилеглими елементами керування, запити до БД, контролери для представлень та моделі даних, що дають адміністраторам можливість переглядати історію виконання таких дій над товарами, як реєстрація, редагування, додавання товару на склад. Також надається можливість швидко перейти до детальних характеристик обраного товару. Робота цього модулю залежить від взаємодії адміністратора з модулями «Товари» і «Склад».
Замовлення	Даний модуль охоплює усі сторінки графічного інтерфейсу з прилеглими елементами керування, запити до БД, контролери для представлень та моделі даних, що дають можливість проводити валідацію вхідних даних і виконувати наступні дії при роботі з вебсайтом: для партнерів - оформлення замовлення обраного товару, перегляд списку оформлених замовлень та детальної інформації про них, скасування обраного замовлення, перегляд детальних характеристик замовленого товару. Для адміністраторів – перегляд списку усіх замовлень та детальної інформації про них, позначення замовлення як «Виконане».
Email-повідомлення	Даний модуль забезпечує відправку email-повідомлень зареєстрованому партнеру, над акаунтом якого адміністратор виконує такі дії як, редагування інформації, зміна паролю та деактивація акаунту.
База даних	Даний модуль забезпечує зв'язок з БД, визначення сутностей БД, зчитування, запис, оновлення та видалення даних у БД, виклик зформованих запитів до БД, кешування даних та взаємодію з кешем.

2.4 Висновки за розділом 2

В рамках даного розділу було обґрунтовано вибір відповідного технологічного стеку на основі проведеного аналізу вимог та специфіки проєкту. У якості мови програмування для серверної частини застосунку обрано C#, що забезпечує високий рівень продуктивності та широкі можливості для розробки. Для реалізації клієнтської частини використано JavaScript, оскільки динаміка сторінок у межах даного проєкту є незначною, а

написання скриптів для комфортної взаємодії з інтерфейсом не потребує використання таких фреймворків, як React або Angular.

У якості основної технологічної платформи для побудови вебзастосунку обрано ASP.NET Core (MVC) — сучасне кросплатформний фреймворк, що реалізує архітектурну модель Model-View-Controller (MVC). Такий підхід сприяє логічному розмежуванню компонентів вебсайту, полегшує тестування, підтримку та масштабування застосунку.

Для організації зберігання та обробки даних використано Microsoft SQL Server — надійну, масштабовану СКБД, яка характеризується високим рівнем інтеграції з обраними інструментами розробки. Як засіб об'єктно-реляційного відображення було використано Entity Framework Core, що забезпечує ефективну взаємодію з базою даних і дозволяє зосередити зусилля на реалізації бізнес-логіки, мінімізуючи потребу у написанні низькорівневих SQL-запитів.

Таким чином, обраний стек технологій повністю відповідає вимогам проєкту, забезпечуючи ефективну реалізацію, підтримуваність та подальший розвиток веб-застосунку.

3 ОСНОВНІ РІШЕННЯ ЩОДО РЕАЛІЗАЦІЇ КОМПОНЕНТІВ СИСТЕМИ

3.1 Огляд обраного шаблону проєктування

При розробці вебсайту було використано шаблон MVC(Model-View-Controller), так як даний шаблон використовується при розробці програмного забезпечення за допомогою фреймворку ASP.NET Core MVC.

Обраний шаблон архітектури передбачає поділ вебзастосунку на три ключові компоненти: модель, представлення та контролер. Цей підхід сприяє чіткому розподілу завдань у програмному забезпеченні. У рамках обраного патерну запити користувача спрямовуються до контролера, який виконує взаємодію з моделлю для реалізації дій користувача або отримання необхідних результатів. Контролер, у свою чергу, обирає відповідне подання, яке буде відображене користувачу, і передає йому всі необхідні дані, що належать до моделі [13].

Як подання, так і контролер залежать від моделі, тоді як сама модель не має залежностей ані від подання, ані від контролера. Це є однією з основних переваг такого розподілу компонентів. Такий підхід забезпечує можливість незалежної розробки та тестування моделі без необхідності враховувати деталі візуального представлення [13].

Серед характерних рис обраного шаблону проєктування доцільно виокремити такі особливості:

- інтегрована підтримка ін'єкції залежностей, що сприяє спрощенню впровадження зовнішніх сервісів у контролери, представлення та інші компоненти додатка;
- система маршрутизації, яка передбачає використання як атрибутивної, так і конвенційної маршрутизації для визначення шаблонів URL, що дозволяє реалізовувати гнучку та масштабовану структуру навігації між ресурсами вебсайту;

- технологія Tag Helpers, що забезпечує зручний механізм генерації HTML-розмітки у Razor-представленнях шляхом інтеграції серверної логіки безпосередньо в розмітку, підвищуючи при цьому читабельність та підтримуваність коду;
- підтримка асинхронного програмування, яка дозволяє ефективно управляти операціями, пов'язаними з введенням/виведенням (наприклад, звернення до бази даних), завдяки чому досягається підвищена продуктивність та масштабованість вебсайту;
- концепція ланцюга посередників (middleware pipeline), яка дає змогу конфігурувати послідовність обробки HTTP-запитів, включаючи додавання власних посередників для реалізації таких функцій, як логування, автентифікація, обробка виключень та інші аспекти обробки запитів;
- вбудовані механізми безпеки, які охоплюють примусове використання захищеного протоколу HTTPS, автоматизований захист від атак типу XSS (міжсайтове впровадження скриптів), а також реалізацію захисту від CSRF-атак шляхом використання антифальсифікаційних токенів. Ці засоби забезпечують належний рівень безпеки при обробці запитів і збереженні даних користувачів [14].

3.2 Опис реалізованих класів

Реалізовані класи відповідно до шаблону проєктування поділяються на контролери та моделі. Усі методи в класах-контролерах, які взаємодіють з БД, завантажують дані з кешу або у разі його відсутності – з бази даних. При зміні даних, відповідно, оновлюється як кеш так і інформація у БД.

Нижче наведено опис класів-контролерів та їх методів при роботі з даними, результатом такої роботи є повернення користувачу повного або часткового представлення з відповідними даними.

Клас `CriticalErrorController` призначений для відображення користувачам критичних помилок при роботі з вебсайтом. Метод `public`

`ActionResult Index(bool DbUpdateConcurrencyException = false)` відповідає за формування тексту помилки. Якщо виникла помилка паралельної зміни одних і тих самих даних – виводиться повідомлення «ДАНІ, З ЯКИМИ ВИ ПРАЦЮЄТЕ БУЛИ ЗМІНЕНІ. БУДЬ-ЛАСКА, ПЕРЕЗАВАНТАЖТЕ СТОРІНКУ ТА СПРОБУЙТЕ ПІЗНІШЕ.». В інших випадках текст повідомлення такий: «СЕРВЕР НЕ ВІДПОВІДАЄ. БУДЬ-ЛАСКА, СПРОБУЙТЕ ПІЗНІШЕ.».

Клас `AddingItemsController` призначений для обробки усіх вхідних HTTP-запитів від адміністраторів, пов'язаних з додаванням товару на склад (пошук необхідного товару, заповнення складу).

Даний клас містить такі методи:

- `public AddingItemsController(IMemoryCache memoryCache, DataBaseContext db)` – конструктор класа, що ініціалізує залежності, які необхідні для роботи з кешем та для виконання запитів до БД;

- `public async Task<ActionResult> Index(bool success = false, bool fail = false, string searchQuery = "")` – відповідає за завантаження переліку товарів, які можна додати на склад. Якщо метод викликається після спроби додавання товару, додатково відображає повідомлення про успіх або помилку при виконанні операції;

- `public async Task<ActionResult> AddItemInWarehouse()` – відповідає за додавання товару на склад. Якщо на складі достатньо місця для додавання визначеної кількості обраного товару – товар додається, а наповненість складу оновлюється, адміністратору повертаються оновлені дані для відображення та повідомлення про успіх. Якщо на складі недостатньо місця – операція скасовується, а адміністратору передається відповідне повідомлення. Якщо при здійсненні операції виникає помилка паралелізму, пов'язана із зміною кількості товарів на складі – операція скасовується і користувачу повертається повідомлення за допомогою класу `CriticalErrorController`;

- `public async Task<IActionResult> Search(string SearchQuery)` – відповідає за завантаження відфільтрованого каталогу товарів згідно пошукового запиту адміністратора;

- `public async Task<List<CatalogModel>> WarehouseInfo(string SearchQuery)` – допоміжний метод, що завантажує перелік товарів згідно пошукового запиту (або весь перелік, якщо запиту немає) та інформацію про наповненість складу. Використовується для уникнення повторного написання однакового коду.

Клас `CatalogController` призначений для обробки усіх вхідних HTTP-запитів від користувачів, пов'язаних з переглядом каталогу товарів.

Даний клас містить такі методи:

- `public CatalogController(IMemoryCache memoryCache, DbContext db)` - конструктор класа, що ініціалізує залежності, які необхідні для роботи з кешем та для виконання запитів до БД;

- `public async Task<IActionResult> Index()` – відповідає за завантаження повного каталогу товарів;

- `public async Task<IActionResult> Search(string SearchQuery)` – відповідає за завантаження переліку товарів, що відповідають пошуковому запиту користувача. Якщо, запит пустий – відображається весь перелік.

Клас `EditingItemController` призначений для обробки усіх вхідних HTTP-запитів від адміністраторів, пов'язаних з редагуванням товару.

Даний клас містить такі методи:

- `public EditingItemController(IMemoryCache memoryCache, IWebHostEnvironment appEnvironment, DbContext db)` - конструктор класа, що ініціалізує залежності, які необхідні для роботи з кешем і файловою системою та для виконання запитів до БД;

- `public async Task<IActionResult> Index(int ID, bool editItem = false, bool editImage = false)` - відповідає за завантаження детальних характеристик обраного для редагування товару. Якщо метод викликається після спроби

редагування – додатково відображає повідомлення про успішну зміну зображення або інших характеристик;

- `public async Task<IActionResult> EditImage(EditableItem model)` – відповідає за редагування зображення товару. Спочатку проводиться перевірка, щоб назва фото була не більше 35 символів. Далі перевіряється, що нове фото унікальне і відсутнє в інших товарах. Якщо зображення унікальне – редагування проходить успішно і товар оновлюється. У випадку непроходження однієї з визначених перевірок – операція скасовується, а користувачу повертається відповідне повідомлення про помилку;

- `public async Task<IActionResult> EditItem(EditableItem model)` – відповідає за редагування характеристик товару. Спочатку проводиться валідація полів. Після цього відбувається перевірка того, що нова назва товару є унікальною, і якщо перевірку пройдено – дані товару оновлюються. У випадку непроходження однієї з визначених перевірок – операція скасовується, а користувачу повертається відповідне повідомлення про помилку.

Клас `HomeController` призначений для обробки усіх вхідних HTTP-запитів від користувачів, пов'язаних з перебуванням на сайті.

Даний клас містить такі методи:

- `public IActionResult Index()` – відповідає за відображення головної сторінки сайту;

- `public async Task<IActionResult> Logout()` – відповідає за вихід з системи. Відбувається видалення даних про активну сесію та перенаправлення користувача на сторінку входу;

- `public IActionResult Error()` – відповідає за відображення повідомлень про необроблені помилки. Метод автоматично генерується при створенні проєкту.

Клас `ItemDetailsController` призначений для обробки усіх вхідних HTTP-запитів від користувачів, пов'язаних з переглядом детальних характеристик товару.

Даний клас містить такі методи:

- `public ItemDetailsController(IMemoryCache memoryCache, DataBaseContext db)` - конструктор класа, що ініціалізує залежності, які необхідні для роботи з кешем та для виконання запитів до БД;
- `public async Task<IActionResult> Index(int ID, bool order = false)` – відповідає за завантаження інформації про характеристики товару. Якщо метод викликається після спроби оформлення замовлення партнером – додатково відображає повідомлення про успішне оформлення замовлення;
- `public async Task<IActionResult> Order(int ID, int quantity, string Name, decimal Price, decimal Volume)` – відповідає за оформлення замовлення на товар. Дані активного користувача зберігаються в замовленні, загальна кількість товару зменшується, а кількість заброньованого товару відповідно збільшується. Якщо при здійсненні операції виникає помилка паралелізму, пов'язана із зміною кількості товарів на складі – операція скасовується і користувачу повертається повідомлення за допомогою класу `CriticalErrorController`;
- `public IActionResult EditingItem(int ID)` – відповідає за перехід на сторінку редагування товару при натисканні відповідної кнопки адміністратором.

Клас `ItemRegistrationController` призначений для обробки усіх вхідних HTTP-запитів від користувачів, пов'язаних з переглядом детальних характеристик товару.

Даний клас містить такі методи:

- `public ItemRegistrationController(IMemoryCache memoryCache, IWebHostEnvironment appEnvironment, DataBaseContext db)` - конструктор класа, що ініціалізує залежності, які необхідні для роботи з кешем і файловою системою та для виконання запитів до БД;
- `public async Task<IActionResult> Index(bool addItem = false)` – відповідає за завантаження переліку категорій та атрибутів, що з ними

пов'язані. Якщо метод викликається після спроби реєстрації товару – додатково відображає відповідне повідомлення про успішну операцію;

- `public async Task<IActionResult> ChangeCategory(int categoryID)` – відповідає за формування переліку атрибутів, які мають відображатись при виборі відповідної категорії товарів користувачем;

- `public async Task<IActionResult> AddItem(NewItemModel model)` – відповідає за реєстрацію нового товару. Спочатку проводиться валідація усіх полів. Далі, проводиться перевірка унікальності назви та фото товару. Якщо дані унікальні – відбувається реєстрація товару. У випадку непроходження однієї з визначених перевірок – операція скасовується, а користувачу повертається відповідне повідомлення про помилку;

- `public async Task TmpFunc(NewItemModel model)` – допоміжний метод, що викликається у разі виникнення помилок при реєстрації товару. Відповідає за завантаження категорій та їх атрибутів. Використовується для уникнення повторного написання однакового коду.

Клас `LoginController` призначений для обробки усіх вхідних HTTP-запитів від користувачів, пов'язаних з входом в систему.

Даний клас містить такі методи:

- `public LoginController(ICaptchaValidator captchaValidator, DbContext db)` - конструктор класа, що ініціалізує залежності, які необхідні для роботи з капчею та для виконання запитів до БД;

- `public IActionResult Index()` – відповідає за відображення головної сторінки сайту;

- `public async Task<IActionResult> Login(LoginModel model)` – відповідає за вхід в акаунт. Спочатку відбувається перевірка коректно пройденої капчі, далі здійснюється валідація полів. Після цього перевіряється коректність даних для входу в акаунт. Якщо дані коректні – відбувається збереження активної сесії за допомогою `claims` і користувач перенаправляється на головну сторінку. У випадку непроходження однієї з

визначених перевірок – операція скасовується, а користувачу повертається відповідне повідомлення про помилку.

Клас `OrdersController` призначений для обробки усіх вхідних HTTP-запитів від користувачів, пов'язаних із списком замовлень.

Даний клас містить такі методи:

- `public OrdersController(IMemoryCache memoryCache, ICompositeViewEngine viewEngine, DataBaseContext db)` - конструктор класа, що ініціалізує залежності, які необхідні для роботи з кешем, рендерінгу представлень та для виконання запитів до БД;

- `public async Task<IActionResult> IndexForPartner()` – відповідає за завантаження списку замовлень активного користувача;

- `public async Task<IActionResult> IndexForAdmin()` – відповідає за завантаження списку усіх замовлень для подальшої роботи з ним адміністратором;

- `public async Task<IActionResult> ChangeStatus([FromBody] OrderRequest request)` – відповідає за зміну адміністратором статусу замовлення на «Виконано». Спочатку відбувається пошук обраного замовлення у списку. Якщо під час виконання операції статус замовлення не було змінено на «Скасовано» або «Виконано» іншим користувачем - операція успішно виконується. Кількість замовленого товару зписується зі складу. Якщо перевірка не була пройдена – операція скасовується, а користувач отримує відповідне повідомлення про помилку. Якщо при здійсненні операції виникає помилка паралелізму, пов'язана із зміною кількості товарів на складі – операція скасовується і користувачу повертається повідомлення за допомогою класу `CriticalErrorController`;

- `public async Task<IActionResult> CancelOrder([FromBody] OrderRequest request)` – відповідає за зміну користувачем-партнером статусу замовлення на «Скасовано». Спочатку відбувається пошук обраного замовлення у списку. Якщо під час виконання операції статус замовлення не було змінено на «Скасовано» або «Виконано» іншим користувачем - операція

успішно виконується. Кількість замовленого товару додається до загальної кількості товару. Якщо перевірка не була пройдена – операція скасовується, а користувач отримує відповідне повідомлення про помилку. Якщо при здійсненні операції виникає помилка паралелізму, пов'язана із зміною кількості товарів на складі – операція скасовується і користувачу повертається повідомлення за допомогою класу `CriticalErrorController`;

- `public string RenderPartialViewToString(string viewName, object model)` – допоміжний метод, що здійснює рендерінг представлення у вигляді рядка для його повернення у `fetch`-запиті. При рендерінгу усі дані з моделі передаються до представлення.

Клас `PartnerController` призначений для обробки усіх вхідних HTTP-запитів від адміністраторів, пов'язаних із списком зареєстрованих користувачів.

Даний клас містить такі методи:

- `public PartnerListController(IMemoryCache memoryCache, ICompositeViewEngine viewEngine, DbContext db, IEmailSender emailSender)` - конструктор класа, що ініціалізує залежності, які необхідні для роботи з кешем, рендерінгу представлень, виконання запитів до БД та для відправлення email-повідомлень;

- `public async Task<IActionResult> Index()` – відповідає за завантаження списку зареєстрованих користувачів;

- `public async Task<IActionResult> EditInfo([FromBody] PartnerModel partnerModel)` – відповідає за редагування інформації обраного користувача. Спочатку відбувається валідація заповнених полів. Якщо дані коректні – відбувається перевірка даних на унікальність. Якщо оновлена інформація не збігається з іншими користувачами – операція успішно виконується, а користувачу на пошту, що була зазначена до редагування даних, надсилається email-повідомлення про виконану дію. У випадку, якщо одна із зазначених перевірок не була пройдена – операція скасовується, а користувачу повертається відповідне повідомлення про помилку;

– `public async Task<IActionResult> DeleteAccount([FromBody] UserDeleteRequest request)` – відповідає за деактивацію акаунта користувача. Спочатку відбувається зміна статусу акаунта на «Неактивний». Після цього відбувається пошук усіх наявних замовлень обраного користувача, і їх статус змінюється на «Скасовано», а на складі відповідно оновлюється кількість товарів. Після виконання операції користувачу на пошту надсилається email-повідомлення про виконану дію. Якщо при здійсненні операції виникає помилка паралелізму, пов'язана із зміною кількості товарів на складі – операція скасовується і користувачу повертається відповідне повідомлення за допомогою класу `CriticalErrorController`;

– `public async Task<IActionResult> ChangePassword([FromBody] ChangePasswordModel passwords)` – відповідає за зміну адміністратором паролю від акаунту обраного користувача. Спочатку відбувається валідація заповнених полів. Якщо дані коректні – відбувається спроба зміни паролю. Якщо поточний пароль було введено вірно – операція успішно виконується, а користувачу на пошту надсилається email-повідомлення. У випадку, коли одна із зазначених перевірок не була пройдена – операція скасовується, а користувач отримує відповідне повідомлення про помилку;

– `public string RenderPartialViewToString(string viewName, object model)` - допоміжний метод, що був описаний в класі-контролері `OrdersController`. Виконує такі самі функції.

Клас `RegistrationController` призначений для обробки усіх вхідних HTTP-запитів від адміністраторів, пов'язаних із реєстрацією акаунтів.

Даний клас містить такі методи:

– `public RegistrationController(IMemoryCache memoryCache, DbContext db)` - конструктор класа, що ініціалізує залежності, які необхідні для роботи з кешем та для виконання запитів до БД;

– `public IActionResult Index()` – відповідає за відображення сторінки реєстрації;

– `public async Task<IActionResult> Registration(RegModel model)` – відповідає за реєстрацію нового користувача. Спочатку відбувається валідація заповнених полів. Якщо дані коректні – вони перевіряються на унікальність. У разі проходження перевірки на унікальність – відбувається реєстрація нового користувача, а адміністратор отримує відповідне повідомлення про успіх. У випадку непроходження однієї з зазначених перевірок – операція скасовується, а адміністратор отримує відповідне повідомлення про помилку.

Клас `TransactionHistoryController` призначений для обробки усіх вхідних HTTP-запитів від адміністраторів, пов'язаних із історією керування товарами.

Даний клас містить такі методи:

– `public TransactionHistoryController(IMemoryCache memoryCache, DataBaseContext db)` - конструктор класа, що ініціалізує залежності, які необхідні для роботи з кешем та для виконання запитів до БД;

– `public async Task<IActionResult> Index()` – відповідає за відображення списку операцій, що були здійснені із товарами. Сюди входять: реєстрація товару, редагування характеристик товару, додавання товару на склад.

Нижче наведено опис класів, що представляють собою моделі для взаємодії БД з кешом і представленнями. Варто зазначити, що до моделей також відносяться ті класи, які формують таблиці БД. Ці класи будуть розглянуті нижче при описі сутностей БД.

Клас `AttributeValueModel` – модель, що відповідає за збереження даних про значення унікальних характеристик для кожного товару. Містить такі поля: `int ItemID`, `int AttributeID`, `string AttributeName`, `string Value`. Використовується при перегляді та при редагуванні детальних характеристик товару.

Клас `CatalogModel` – модель, що відповідає за збереження даних про стислі характеристики товарів. Містить такі поля: `int ItemID`, `string Name`, `string Photo`, `decimal Price`, `decimal Volume`, `int Quantity`. Використовується при

перегляді каталогу товарів та при додаванні товару на склад. Значення «Volume» відображається тільки на сторінці додавання товару на склад, а значення «Price» тільки на сторінці каталогу товарів.

Клас `CategoriesWithAttributesModel` – модель, що відповідає за збереження даних про зв'язок категорій з унікальними характеристиками (атрибутами). Містить такі поля: `int CategoryID`, `int AttributeID`, `string AttributeName`. Використовується при реєстрації нового товару.

Клас `CategoryModel` – модель, що відповідає за збереження даних про наявні категорії. Містить такі поля: `int CategoryID`, `string CategoryName`. Використовується при реєстрації нового товару.

Клас `EditableItem` – модель, що відповідає за збереження даних про оновлені характеристики товару. Містить в собі такі поля: `int ItemID`, `string Name`, `IFormFile Photo` (необхідне для отримання нового зображення через форму), `string PhotoPath`, `string CategoryName`, `string WoodType`, `string Color`, `string Size`, `decimal Weight`, `decimal Volume`, `string Description`, `decimal SalePrice`, `List<ItemAttribute> Attributes`. Використовується при редагуванні інформації про товар.

Клас `ErrorViewModel` – автоматично згенерована модель, що використовується при виникненні необроблених помилок. Містить такі поля: `string? RequestId`, `bool ShowRequestId`.

Клас `ItemAttribute` – модель, що відповідає за збереження даних про унікальні характеристики (атрибути) товару та їх значення. Містить такі поля: `int AttributeID`, `string Value`, `string Name`. Дана модель є допоміжною і є частиною інших моделей: `ItemDetailsModel`, `EditableItem`, `NewItemModel`. Відповідно використовується при реєстрації нового товару, перегляді та редагуванні характеристик існуючого товару.

Клас `ItemDetailsModel` – модель, що відповідає за збереження інформації про товар. Містить в собі такі поля: `ItemModel Item`, `List<ItemAttribute> Attributes`. Використовується при перегляді детальної інформації про товар.

Клас `ItemModel` – модель, що відповідає за збереження даних про детальні характеристики товару. Містить в собі такі поля: `int ItemID`, `string Name`, `string Photo`, `string CategoryName`, `string WoodType`, `string Color`, `string Size`, `decimal Weight`, `decimal Volume`, `string Description`, `int QuantityInStock`, `int QuantityForOrder`, `decimal SalePrice`. Використовується в моделі `ItemDetailsModel` та для збереження даних про товари в кеші. Кеш, в свою чергу використовується при додаванні товарів на склад, при завантаженні каталогу товарів, при редагуванні інформації про товар, при перегляді детальних характеристик товару, при роботі із замовленнями, при реєстрації нового товару та при деактивації акаунту користувача.

Клас `ItemValidationModel` – модель, що відповідає за збереження даних про товар, які мають бути унікальними для кожного товару. Містить такі поля: `int ItemID`, `string ImagePath`, `string ItemName`. Використовується при реєстрації нового товару та при редагуванні інформації про існуючий товар.

Клас `LoginModel` – модель, що відповідає за збереження введених користувачем даних при вході в акаунт. Містить такі поля: `string Username`, `string Password`, `string CaptchaToken`.

Клас `NewItemModel` – модель, що відповідає за збереження даних про товар, який реєструють. Містить такі поля: `NewItem NewItem`, `List<ItemAttribute> ItemAttributes`. Використовується тільки при реєстрації нового товару.

Клас `NewItem` – модель, що відповідає за збереження детальних характеристик товару, що реєструють. Дана модель описана в одному файлі з `NewItemModel`. Містить такі поля: `int ItemID`, `string Name`, `IFormFile Photo` (необхідне для отримання нового зображення через форму), `int CategoryID`, `string WoodType`, `string Color`, `string Size`, `decimal Weight`, `decimal Volume`, `string Description`, `decimal SalePrice`. Модель є допоміжною і використовується всередині моделі `NewItemModel`.

Клас `OperationModel` – модель, що відповідає за збереження історії керування товарами. Містить такі поля: `int ItemID`, `string ItemName`, `DateTime`

OperationDate, string OperationType. Використовується тільки при перегляді історії керування товарами (реєстрація, редагування, додавання товарів на склад).

Клас OrderInfoModel – модель, що відповідає за збереження даних про оформлені, скасовані, виконані замовлення. Дана модель описана в одному файлі з OrderModel. Містить такі поля: OrderModel OrderInfo, List<OrderModel> Orders. Використовується при роботі із списком замовлень.

Клас OrderModel – модель, що відповідає за збереження детальної інформації про замовлення. Містить такі поля: int OrderID, int UserID, string UserName, int ItemID, string ItemName, DateTime OrderTime, int QuantityOrdered, decimal ItemsVolume, string DeliveryMethod, decimal OrderCost, string PaymentMethod, bool OrderStatusTrue, bool OrderStatusCancel. Використовується в моделі OrderInfoModel, при оформленні замовлення, перегляді детальної інформації про замовлення, при деактивації акаунту користувача.

Клас PartnerInfoModel – модель, що відповідає за збереження даних про список партнерів. Містить такі поля: PartnerModel PartnerModel, ChangePasswordModel ChangePasswordModel, List<PartnerModel> Partners. Використовується при роботі із списком партнерів.

Клас PartnerModel – модель, що відповідає за збереження детальної інформації про користувача. Дана модель описана в одному файлі з PartnerInfoModel. Містить такі поля: int UserID, string Login, string CodeUSREOU, string CompanyName, string CompanyAddress, string CompanyPhoneNumber, string CompanyEmail, string AccountRole, bool isActive. Використовується в моделі PartnerInfoModel, при реєстрації нового користувача, при перегляді детальної інформації про користувача.

Клас ChangePasswordModel – модель, що відповідає за збереження введених даних при зміні адміністратором паролю від акаунта користувача. Дана модель описана в одному файлі з PartnerInfoModel. Містить такі поля: int UserID, string OldPassword, string NewPassword, string ConfirmPassword.

Клас `RegModel` – модель, що відповідає за збереження введених даних при реєстрації адміністратором нового користувача. Містить такі поля: `string Username`, `string Email`, `string Password`, `string ConfirmPassword`, `string CodeUSREOU`, `string CompanyName`, `string CompanyAddress`, `string CompanyPhoneNumber`.

Окремо слід розглянути такі класи: `DataBaseContext`, `ReCaptchaValidator` (файл `CaptchaValidator.cs`), `EmailSender` (файл `ISender.cs`).

`DataBaseContext` – це контекст бази даних, через який визначаються таблиці БД, встановлюється підключення до БД, формуються запити для роботи з даними, налаштовуються специфічні обмеження або поведінка для побудованих сутностей (встановлення стовпцям значення за замовчуванням, створення індексів та визначення унікальності для стовпців таблиці).

`ReCaptchaValidator` – клас, що реалізує інтерфейс `ICaptchaValidator` і відповідає за перевірку пройденої користувачем ReCaptcha. Метод `private async Task<JObject> GetCaptchaResultDataAsync(string token)` забезпечує відправку секретного ключа та токена капчі, отриманого з клієнтської сторони, до сервера Google reCAPTCHA. Отримана відповідь передається до методу `public async Task<bool> IsCaptchaPassedAsync(string token)`, звідки отриманий результат передається вже до контролера, який викликав перевірку капчі. Таким чином забезпечується захист від ботів та автоматичних запитів у форму входу в акаунт.

`EmailSender` – клас, що реалізує інтерфейс `ISender` і відповідає за формування і надсилання email-повідомлень. Конструктор класу `public EmailSender(ILogger<EmailSender> logger, IConfiguration configuration)` ініціалізує залежності, які необхідні для роботи з логуванням та для отримання параметрів із файлу конфігурацій. Метод класу `public async Task SendEmailAsync(string messageTo, string messageBody)` забезпечує отримання параметрів для відправки повідомлення, формування листа з вказанням

відправника, отримувача, теми, основного тексту та подальше відправлення повідомлення.

Окрім цього, в проєкті наявні файли `appsettings.json` і `Program.cs`.

Файл `appsettings.json` містить в собі налаштування для різних компонентів вебсайту: `ReCaptcha` – значення ключів для проходження капчі, `ConnectionStrings` – рядки для підключення до БД, `SmtpSettings` – параметри для відправлення email-повідомлень, `Logging` – налаштування логування, `AllowedHosts` – дозволені хости.

Файл `Program.cs` відповідає за конфігурацію та запуск вебсайту. Тут визначається: які служби будуть використовуватись, як підключатися до бази даних, як обробляти запити користувачів, і як забезпечувати безпеку сайту. Саме в цьому файлі налаштовується система аутентифікації та авторизації, підключається логування, вмикається обробка помилок, визначаються правила маршрутизації, а також конфігурується робота зі статичними файлами.

3.3 Опис реалізованих представлень

Реалізовані представлення дають змогу користувачам взаємодіяти з даними та використовувати увесь розроблений функціонал вебсайту. Кожне представлення пов'язано з однією із моделей, описаних в пункті 3.2. Також, наявні допоміжні представлення, що відповідають за налаштування вебсторінок.

Представлення `_Layout.cshtml` – майстер-сторінка, що реалізує ті елементи сторінок, які повторюються. Сюди входять шапка і футер сайту, та навігаційне меню, що розташоване в лівій частині сторінки. Шапка містить в собі клікабельну назву сайту, яка переводить користувача на головну сторінку. В правій частині шапки знаходиться кнопка переходу на сторінку входу, або, якщо користувач вже авторизован – відображається логін та кнопка виходу з акаунта. У футері сайту знаходиться інформація про контакти адміністрації

сайту. Також, тут ініціалізується токен захисту від підробок для його використання на інших сторінках.

Представлення `_Layout2.cshtml` – майстер-сторінка, аналогічна до `_Layout.cshtml`. Відрізняється тим, що тут відсутнє навігаційне меню, так як дана сторінка використовується при реєстрації та вході в акаунт.

Представлення `Index.cshtml` – головна сторінка, яку бачить користувач при переході на вебсайт. Містить в собі текст, що ознайомлює відвідувачів з тематикою і призначенням сайту.

Представлення `LogIn.cshtml` – сторінка входу в акаунт. Для роботи з даними використовується модель `LoginModel`. Окрім цього, тут застосовується капча для захисту від ботів. Сама сторінка складається з полів для введення логіну і паролю, капчі та кнопки входу.

Представлення `_ValidationScriptsPartial.cshtml` – автоматично згенероване часткове представлення, що призначене для підключення валідації форм на стороні клієнта.

Представлення `Error.cshtml` – автоматично згенероване представлення, що використовується для відображення повідомлень про необроблені помилки.

Представлення `_ViewImports.cshtml` – автоматично згенерований файл, що спрощує підключення простору імен та імпорт тег-хелперів.

Представлення `_ViewStart.cshtml` – автоматично згенерований файл, що дозволяє визначити загальні налаштування для усіх представлень. За замовчуванням, тут вказано, що для всіх сторінок слід використовувати майстер-сторінку `_Layout.cshtml`.

Представлення `AddingItems.cshtml` – сторінка, яка відповідає за взаємодію користувача зі складом, а саме, додаванням товарів. Для роботи з даними використовується модель `List<CatalogModel>`. Дана сторінка складається з рядку пошуку, вікна для переліку товарів та нижньої панелі, де відображається кількість товару, що додається, його об'єм, та наповненість складу. Відповідно, для додавання товару визначена кнопка. Реалізовані

скрипти дозволяють обмежити максимальну кількість товару для додавання, в залежності від наповненості складу та об'єму самого товару, розрахувати об'єм товарів, який буде додано на склад, заборонити додавання товару, якщо з переліку нічого не було обрано.

Представлення `Catalog.cshtml` – сторінка, яка відповідає за взаємодію користувача з переліком товарів, а саме їх перегляд і пошук. Товари, яких немає в наявності відображаються сірим кольором. Для роботи з даними використовується модель `List<CatalogModel>`. Дана сторінка складається з рядку пошуку та вікна для відображення переліку товарів.

Представлення `EditingItem.cshtml` – сторінка, що відповідає за взаємодію користувача з товаром, який слід відредагувати. Для роботи з даними використовується модель `EditableItem`. Дана сторінка складається з двох форм для редагування. Перша знаходиться в верхній частині і відповідає за зміну зображення товару, також тут відображається поточне зображення і наявна кнопка для прийняття змін. В нижній частині знаходиться друга форма, що відповідає за зміну інших характеристик, таких як назва товару, ціна, опис тощо. Тут, для редагування також визначена окрема кнопка. Повідомлення користувачу відображаються вгорі сторінки над формами. Предперегляд обраного зображення відбувається за допомогою реалізованого скрипта.

Представлення `ItemDetails.cshtml` – сторінка, що відповідає за взаємодію з характеристиками обраного товару. Для роботи з даними використовується модель `ItemDetailsModel`. Сторінка складається з двох стовпців. В лівому знаходиться назва товару, фото, ціна, інформація про наявність товару та опис. В правому стовпці розташовані загальні та унікальні характеристики. Неавторизованим користувачам дозволено лише переглядати основну інформацію про товар. Адміністратори, мають можливість перейти до редагування на сторінку `EditingItem.cshtml`, а в характеристиках вони додатково бачать об'єм товару, кількість товару, що було заброньовано на замовлення, та кількість товару на складі. Користувачі-партнери мають змогу побачити точну кількість товару на складі та оформити замовлення, якщо

товар є. При оформленні відображається інформація про кількість товару, ціну замовлення, спосіб доставки та спосіб оплати. Повідомлення при оформленні замовлення відображається в верхній частині сторінки. Реалізовані скрипти дозволяють обмежити максимальну кількість товару для замовлення, в залежності від кількості на складі, розрахувати ціну замовлення.

Представлення `ItemRegistration.cshtml` – сторінка, яка відповідає за взаємодію користувача з каталогом, а саме за реєстрацію нового товару. Для роботи з даними використовується модель `NewItemModel`. Сторінка складається з однієї форми, яку варто заповнити для реєстрації товару. В нижній частині даної форми розташована кнопка для здійснення операції. Повідомлення користувачу відображаються у верхній частині сторінки. Реалізовані скрипти дозволяють переглядати обране для товару зображення та змінювати категорію товару разом з пов'язаними характеристиками.

Представлення `ItemRegistraion_Partial.cshtml` – часткове представлення, що оновлюється завдяки скрипту, реалізованому в `ItemRegistration.cshtml`. Для роботи з даними використовується модель `NewItemModel`. Відповідає за відображення унікальних атрибутів, в залежності від обраної категорії для нового товару. Дане представлення є допоміжним і викликається всередині `ItemRegistration.cshtml`.

Представлення `NotFound.cshtml` – сторінка, що відповідає за відображення повідомлення про критичну помилку при роботі вебсайту.

Представлення `Orders.cshtml` – сторінка, що відповідає за взаємодію користувачів-партнерів із списком замовлень. Для роботи з даними використовується модель `OrderInfoModel`. Сторінка складається з двох стовпців. В лівому знаходиться список замовлень користувача, де вказані назва, кількість замовленого товару та надається можливість замовнику скасувати замовлення (для скасованого замовлення немає доступних дій). Назви товарів є клікабельними і перенаправляють користувача у представлення `ItemDetails.cshtml` для перегляду детальних характеристик товару. При виборі із списку замовлення застосовується правий стовпець, в

якому відображається детальна інформація, доступна тільки для перегляду. Список замовлень візуально поділяється на три групи – білим виділено актуальні замовлення, сірим – скасовані, а зеленим – виконані. Реалізовані скрипти відповідають за формування зручної таблиці із списком замовлень, за динамічне оновлення блоку з інформацією при виборі замовлення та за динамічне оновлення таблиці при скасуванні замовлення. Для користувача реалізовані спливаючі повідомлення.

Представлення `Orders_Partial.cshtml` – часткове представлення, що оновлюється завдяки скрипту, реалізованому в `Orders.cshtml`. Для роботи з даними використовується модель `List<OrderModel>`. Відповідає за відображення списку замовлень. Дане представлення є допоміжним і викликається всередині `Orders.cshtml`.

Представлення `OrdersForAdmin.cshtml` – сторінка, що відповідає за взаємодію адміністраторів із списком замовлень. Для роботи з даними використовується модель `OrderInfoModel`. Сторінка складається з двох стовпців. В лівому знаходиться список замовлень усіх користувачів, де вказані назва товару, замовник, дата замовлення та надається можливість адміністратору позначити замовлення як виконане (для виконаного замовлення немає доступних дій). Назви товарів є клікабельними і перенаправляють користувача у представлення `ItemDetails.cshtml` для перегляду детальних характеристик товару. При виборі із списку замовлення застосовується правий стовпець, в якому відображається детальна інформація, доступна тільки для перегляду. Список замовлень візуально поділяється на три групи – білим виділено актуальні замовлення, сірим – скасовані, а зеленим – виконані. Реалізовані скрипти відповідають за формування зручної таблиці із списком замовлень, за динамічне оновлення блоку з інформацією при виборі замовлення та за динамічне оновлення таблиці при виконанні замовлення. Для користувача реалізовані спливаючі повідомлення.

Представлення `OrdersForAdmin_Partial.cshtml` – часткове представлення, що оновлюється завдяки скрипту, реалізованому в

OrdersForAdmin.cshtml. Для роботи з даними використовується модель `List<OrderModel>`. Відповідає за відображення списку замовлень. Дане представлення є допоміжним і викликається всередині `OrdersForAdmin.cshtml`.

`PartnerList.cshtml` – сторінка, що відповідає за взаємодію адміністратора із списком партнерів. Для роботи з даними використовується модель `PartnerInfoModel`. Сторінка візуально поділяється на два стовпці. В лівому знаходиться таблиця із списком усіх зареєстрованих користувачів. Тут відображається інформація про логін користувача, назву підприємства, до якого він відноситься та надається можливість адміністратору деактивувати обраний акаунт. Деактивовані акаунти не відображаються у списку. При виборі із списку користувача застосовується правий стовпець, в якому відображається детальна інформація, доступна для редагування. Тут знаходяться дві форми: в верхній частині – форма з інформацією про акаунт та кнопка для редагування. В нижній частині – форма для зміни паролю, що складається з полів для введення старого та нового паролів та кнопка для здійснення операції. Реалізовані скрипти відповідають за формування зручної таблиці із списком користувачів, за динамічне оновлення блоку з інформацією при виборі користувача та при редагуванні інформації. Для адміністратора реалізовані спливаючі повідомлення.

Представлення `PartnerList_Partial.cshtml` - часткове представлення, що оновлюється завдяки скрипту, реалізованому в `PartnerList.cshtml`. Для роботи з даними використовується модель `List<PartnerModel>`. Відповідає за відображення списку користувачів. Дане представлення є допоміжним і викликається всередині `PartnerList.cshtml`.

Представлення `PartnerRegistration.cshtml` – сторінка, що надає змогу адміністратору реєструвати нових користувачів. Для роботи з даними використовується модель `RegModel`. Сторінка складається з полів для заповнення інформації про акаунт та кнопки реєстрації.

Представлення `TransactionHistory.cshtml` – сторінка, що відповідає за взаємодію з історією керування товарами. Для роботи з даними

використовується модель List<OperationModel>. Сторінка складається з однієї таблиці, що відображає список операцій, що відбувались із товарами. Інформація доступна тільки для перегляду. Поля відображають назву товару, тип операції і дату виконання операції. Назви товарів є клікабельними і перенаправляють користувача у представлення ItemDetails.cshtml для перегляду детальних характеристик товару. Реалізований скрипт відповідає за формування зручної таблиці із списком операцій.

3.4 Проектування бази даних

На рисунку 3.1 відображено схему розробленої бази даних зі всіма наявними зв'язками.

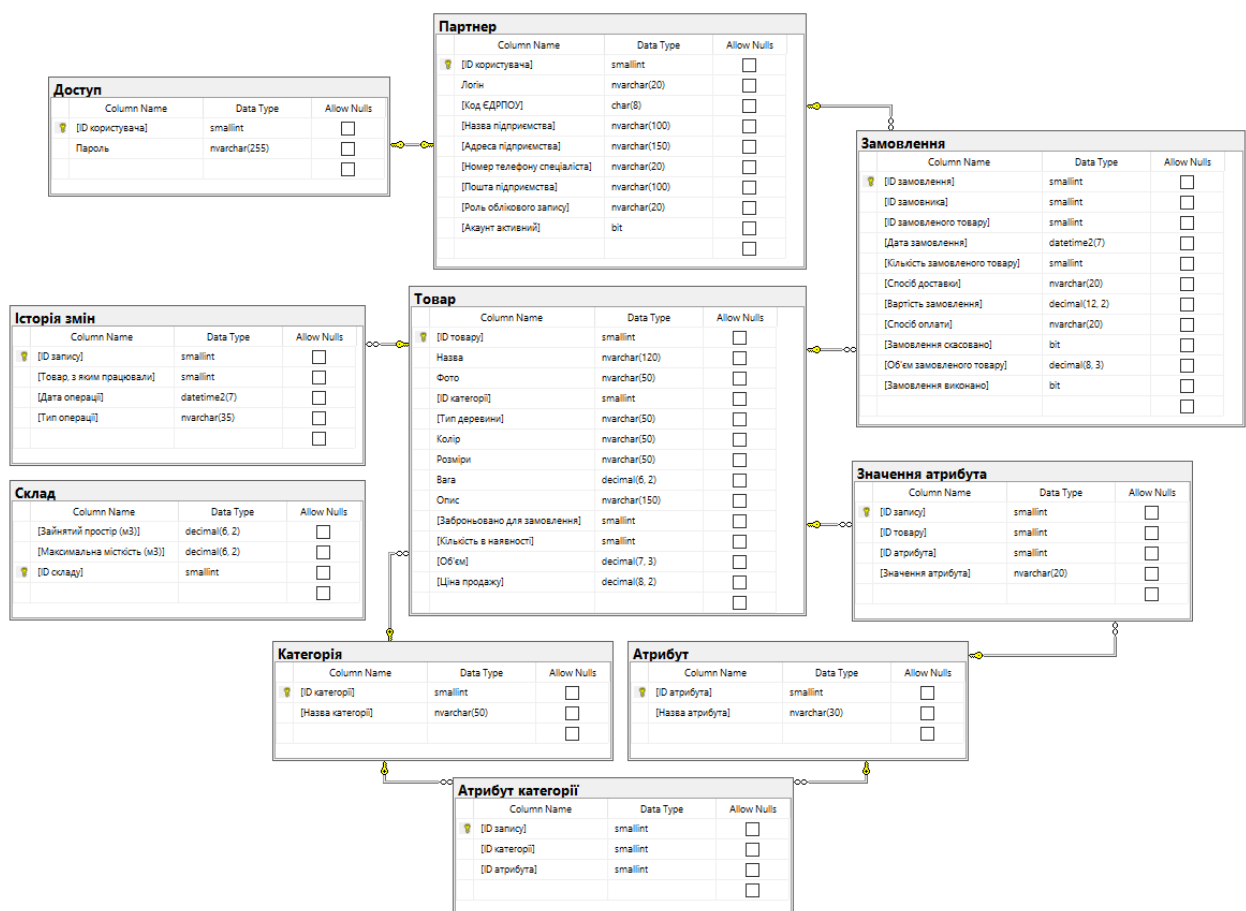


Рисунок 3.1 – Схема БД

В таблиці 3.1 наведено опис сутності «Партнер». Дана сутність дозволяє зберігати дані, що стосуються партнерів-підприємств та адміністраторів підприємства-виробника. Сутність має такі зв'язки: Доступ – 1:1, Замовлення – 1:М.

Таблиця 3.1 – Опис сутності «Партнер»

№	Атрибут	Тип даних
1	ID користувача	smallint
2	Логін	nvarchar(20)
3	Код ЄДРПОУ	char(8)
4	Назва підприємства	nvarchar(100)
5	Адреса підприємства	nvarchar(150)
6	Номер телефону спеціаліста	nvarchar(20)
7	Пошта підприємства	nvarchar(100)
8	Роль облікового запису	nvarchar(20)
9	Акаунт активний	bit

В таблиці 3.2 наведено опис сутності «Доступ». Дана сутність відповідає за збереження хешу паролей від зареєстрованих користувачів. Сутність має такі зв'язки: Партнер – 1:1.

Таблиця 3.2 – Опис сутності «Доступ»

№	Атрибут	Тип даних
1	ID користувача	smallint
2	Пароль	nvarchar(255)

В таблиці 3.3 наведено опис сутності «Замовлення». Дана сутність відповідає за збереження даних про оформлені, виконані, скасовані замовлення підприємств-партнерів. Сутність має такі зв'язки: Партнер – М:1, Товар – М:1.

Таблиця 3.3 – Опис сутності «Замовлення»

№	Атрибут	Тип даних
1	2	3
1	ID замовлення	smallint
2	ID замовника	smallint

Кінець таблиці 3.3

1	2	3
3	ID замовленого товару	smallint
4	Дата замовлення	datetime2(7)
5	Кількість замовленого товару	smallint
6	Спосіб доставки	nvarchar(20)
7	Вартість замовлення	decimal(12,2)
8	Спосіб оплати	nvarchar(20)
9	Об'єм замовленого товару	decimal(8,3)
10	Замовлення скасовано	bit
11	Замовлення виконано	bit

В таблиці 3.4 наведено опис сутності «Товар». Дана сутність відповідає за збереження даних про товари виробника. Сутність має такі зв'язки: Замовлення – 1:М, Історія змін – 1:М, Значення атрибута – 1:М, категорія – М:1.

Таблиця 3.4 – Опис сутності «Товар»

№	Атрибут	Тип даних
1	ID товару	smallint
2	Назва	nvarchar(120)
3	Фото	nvarchar(50)
4	ID категорії	smallint
5	Тип деревини	nvarchar(50)
6	Колір	nvarchar(50)
7	Розміри	nvarchar(50)
8	Вага	decimal(6,2)
9	Опис	nvarchar(150)
10	Заброньовано для замовлення	smallint
11	Кількість в наявності	smallint
12	Об'єм	decimal(7,3)
13	Ціна продажу	decimal(8,2)

В таблиці 3.5 наведено опис сутності «Історія змін». Дана сутність відповідає за збереження даних про історію взаємодії з товарами (реєстрація, редагування, додавання товару на склад). Сутність має такі зв'язки: Товар – М:1.

Таблиця 3.5 – Опис сутності «Історія змін»

№	Атрибут	Тип даних
1	ID запису	smallint
2	Товар, з яким працювали	smallint
3	Дата операції	datetime2(7)
4	Тип операції	nvarchar(35)

В таблиці 3.6 наведено опис сутності «Значення атрибута». Дана сутність відповідає за збереження даних про значення унікальних характеристик категорії для кожного з товарів. Сутність має такі зв'язки: Товар – М:1, Атрибут – М:1.

Таблиця 3.6 – Опис сутності «Значення атрибута»

№	Атрибут	Тип даних
1	ID запису	smallint
2	ID товару	smallint
3	ID атрибута	smallint
4	Значення атрибута	nvarchar(20)

В таблиці 3.7 наведено опис сутності «Категорія». Дана сутність відповідає за збереження даних про доступні категорії для товарів. Сутність має такі зв'язки: Товар – 1:М, Атрибут категорії – 1:М.

Таблиця 3.7 – Опис сутності «Категорія»

№	Атрибут	Тип даних
1	ID категорії	smallint
2	Назва категорії	nvarchar(50)

В таблиці 3.8 наведено опис сутності «Атрибут». Дана сутність відповідає за збереження даних про доступні характеристики, що не підходять кожному товару. Сутність має такі зв'язки: Атрибут категорії – 1:М, Значення атрибута – 1:М.

Таблиця 3.8 – Опис сутності «Атрибут»

№	Атрибут	Тип даних
1	ID атрибута	smallint
2	Назва атрибута	nvarchar(50)

В таблиці 3.9 наведено опис сутності «Атрибут категорії». Дана сутність відповідає за збереження даних про відношення атрибутів до категорій. Таким чином тут формуються унікальні характеристики для кожної з категорій. Сутність має такі зв'язки: Категорія – М:1, Атрибут – М:1.

Таблиця 3.9 – Опис сутності «Атрибут категорії»

№	Атрибут	Тип даних
1	ID запису	smallint
2	ID категорії	smallint
3	ID атрибута	smallint

В таблиці 3.10 наведено опис сутності «Склад». Дана сутність відповідає за збереження даних про склад підприємства-виробника та дозволяє контролювати наповненість складу товарами.

Таблиця 3.10 – Опис сутності «Склад»

№	Атрибут	Тип даних
1	ID складу	smallint
2	Зайнятий простір (м ³)	decimal(6,2)
3	Максимальна місткість (м ³)	decimal(6,2)

3.5 Розгортання вебсайту

Для розгортання вебсайту варто використовувати сторонній хостинг. Проте, усі вони є платними та не надають можливості продемонструвати розгортання вебсайту в рамках виконання дипломної кваліфікаційної роботи. Тому, єдиним варіантом стало використання сервісу Microsoft Azure за допомогою студентської ліцензії [15].

Розгортання вебсайт відбувалося таким чином:

- збирання версії проєкту для публікації за допомогою Visual Studio 2022;
- імпорт бази даних до хмарного сервісу;
- налаштування доступу до БД;
- публікація сайту.

Підготовка проєкту до публікації відбувалася за допомогою IDE Visual Studio 2022 [16]. Нижче наведено покрокове налаштування проєкту. На рисунку 3.2 видно, що спочатку було обрано місце для публікації проєкту. При використанні Microsoft Azure, відповідно було обрано перший пункт Azure.

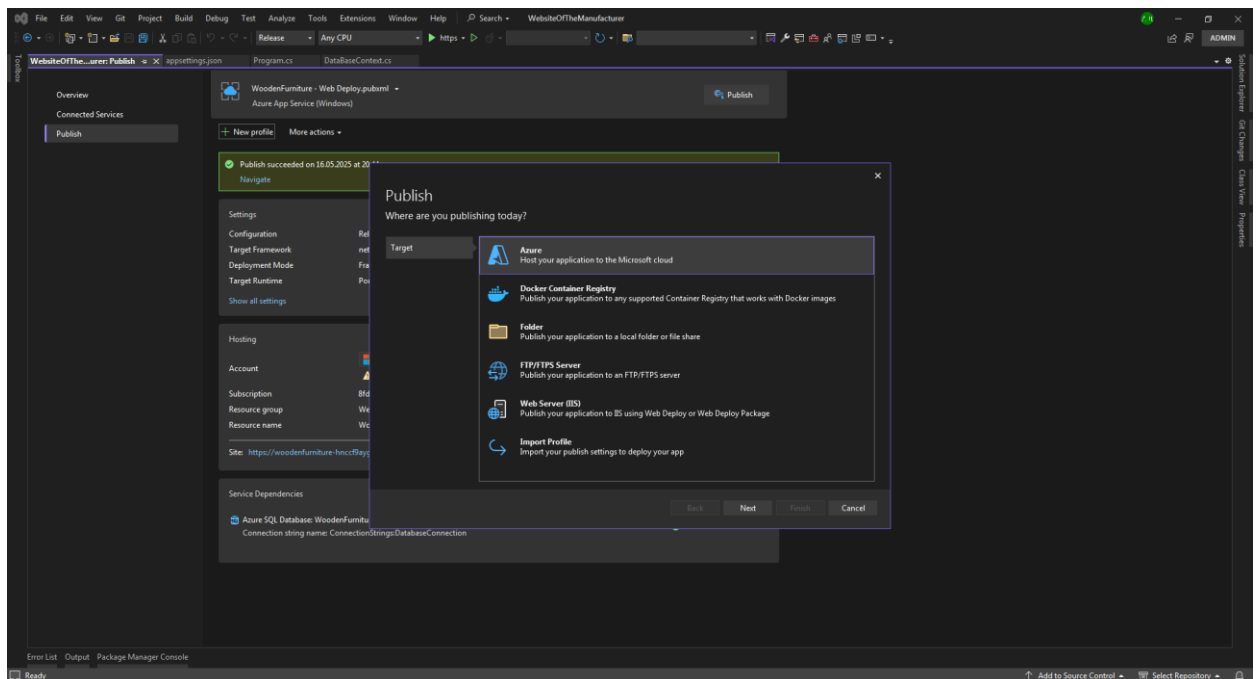


Рисунок 3.2 – Вибір місця публікації проєкту у Visual Studio 2022 [16]

Для публікації проєкту в Azure існує декілька варіантів сервісів. Для простого і швидкого розгортання було обрано Azure App Service (Windows). Цього достатньо для повноцінного функціонування вебсайту. Після вибору сервісу було створено групу ресурсів та сервіс додатків, через який відбувається керування розгорнутим проєктом. Сюди входить запуск і зупинення роботи вебсайту, налаштування рядків для підключення БД,

моніторинг активності, налаштування конфігурацій, створення бекапів тощо. На рисунку 3.3 показано меню створеного сервісу додатків. На цьому підготовка до розгортання проєкту майже завершена.

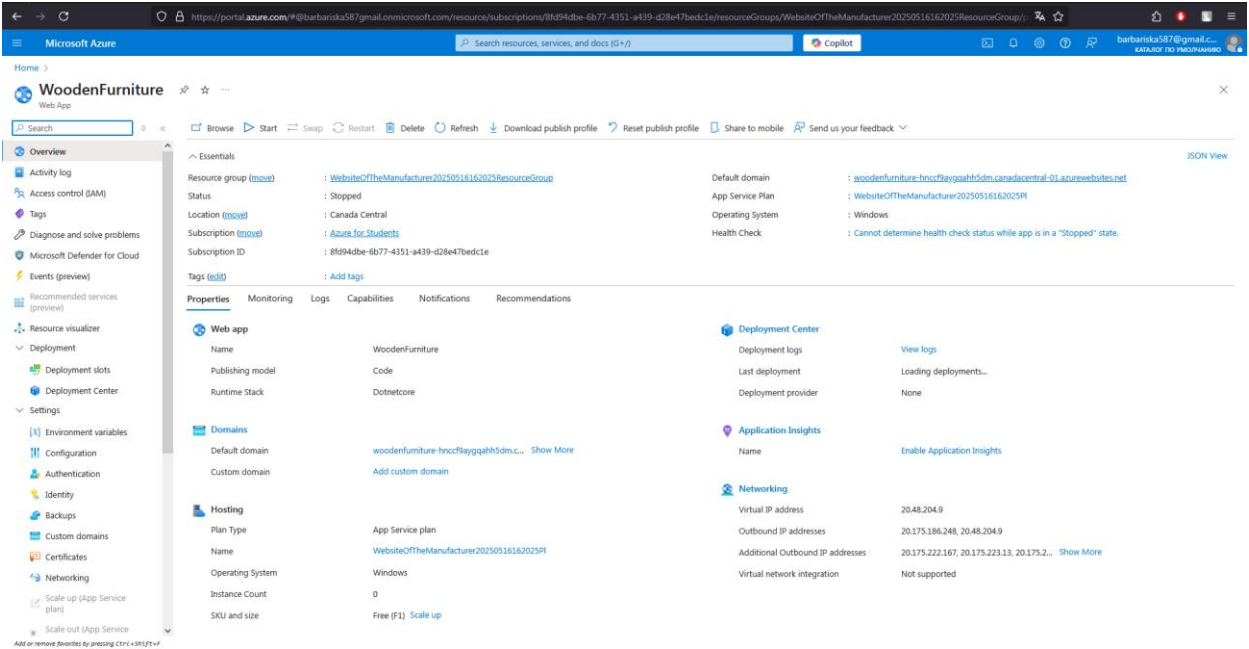


Рисунок 3.3 – Створений сервіс додатків у Microsoft Azure [15]

Після цього для готового профілю, зображеного на рисунку 3.4, було створено і додано базу даних Azure SQL для подальшої взаємодії із розгорнутим вебсайтом.

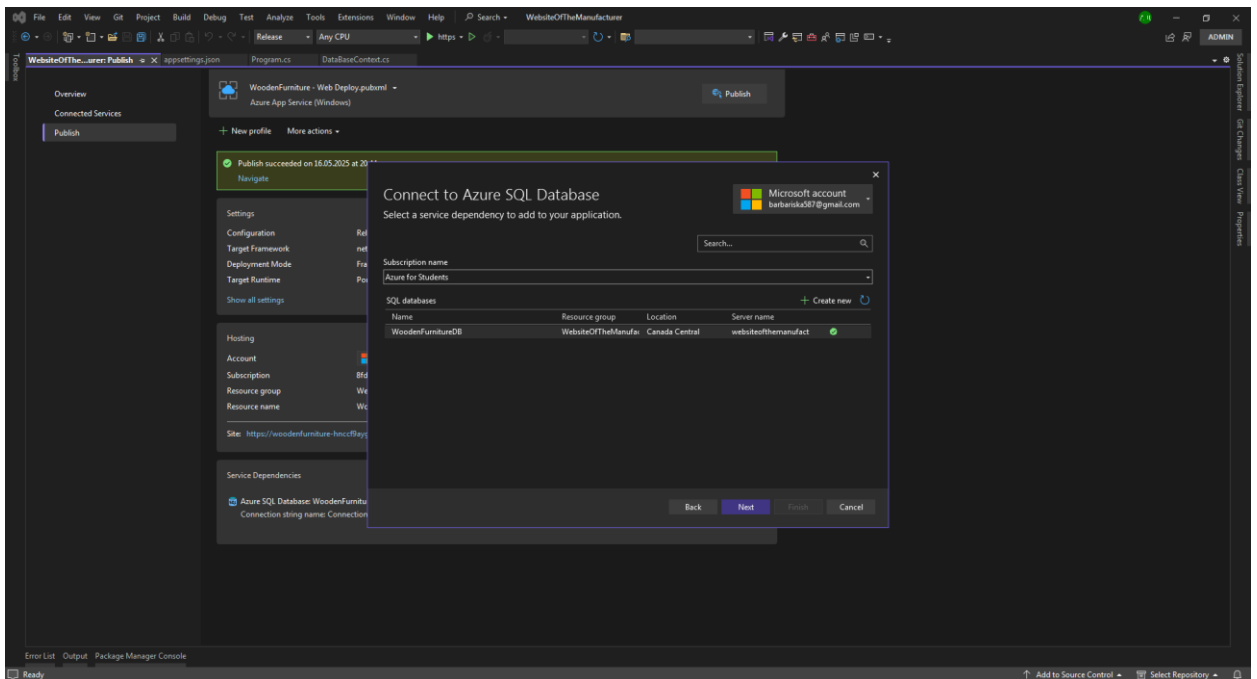


Рисунок 3.4 – Додавання хмарної бази даних до проєкту за допомогою Visual Studio 2022 [16]

Після цього для розгортання вебсайту залишається перенести локальну базу даних до хмари, налаштувати до неї доступ та запустити вебсайт.

Перенесення бази даних до Azure відбувалося за допомогою Microsoft Data Migration Assistant, який призначений для виконання саме таких задач [17]. Процес перенесення БД складається з 6 пунктів:

- вибір джерела;
- вибір цільового сервера;
- вибір об'єктів для переносу (таблиці, схеми, функції, процедури тощо.);
- генерація скрипта для виконання операції;
- вибір таблиць, з яких варто перенести фактичні дані, що знаходяться в локальній БД;
- виконання операції переносу даних.

Як видно на рисунку 3.5 – локальну базу даних було успішно перенесено на сервер.

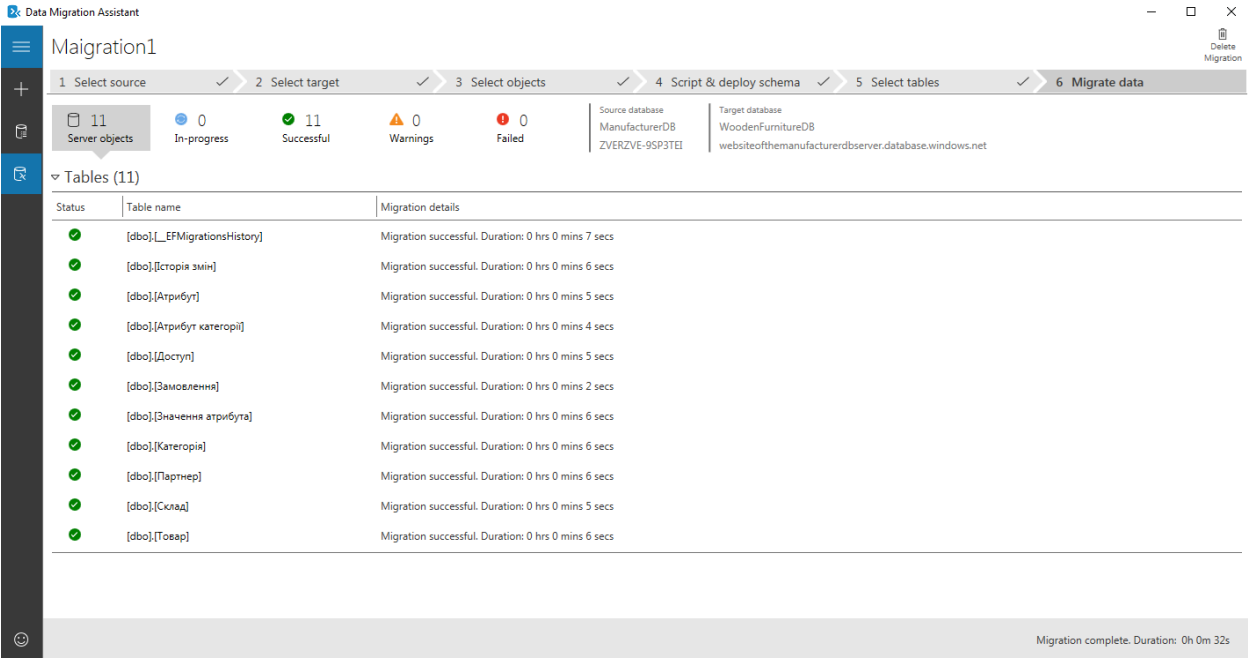


Рисунок 3.5 – Робота із Microsoft Data Migration Assistant [17]

На цьому перенесення даних було завершено, результат операції можна перевірити в Microsoft Azure в налаштуваннях створеної БД. Як видно на рисунку 3.6, усі таблиці успішно перенеслися, а виконаний запит демонструє збереження наявних даних.

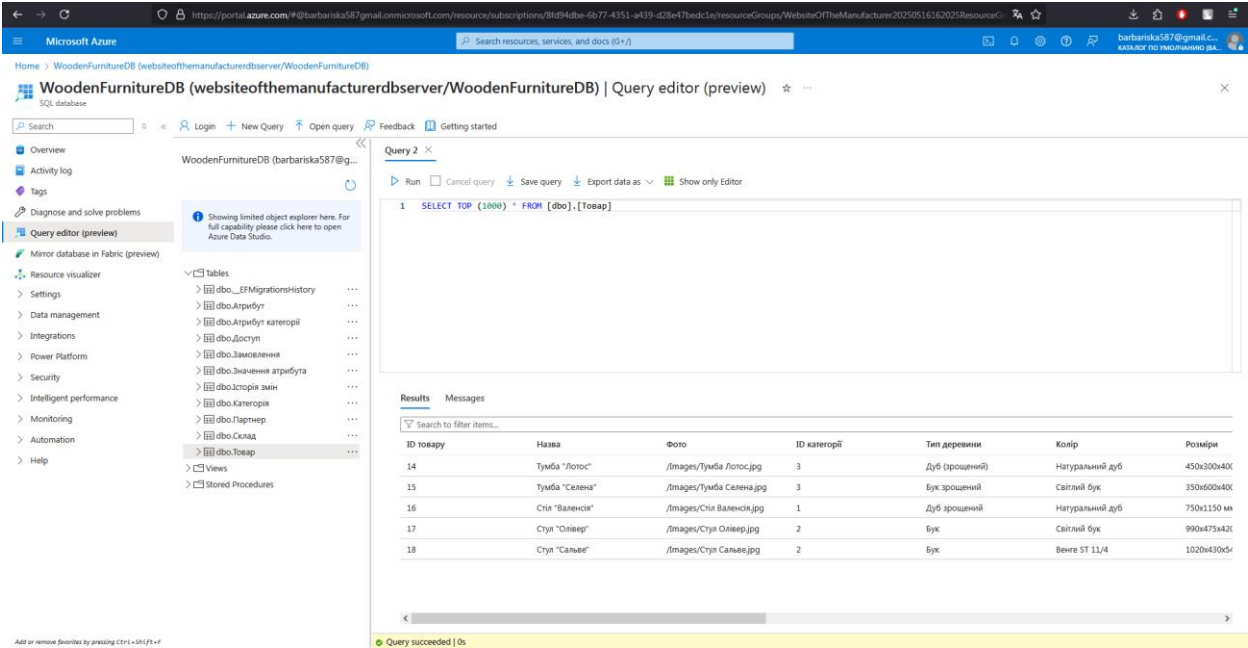


Рисунок 3.6 – Перегляд перенесеної бази даних у Microsoft Azure [15]

Для надання доступу для читання та редагування даних розгорнутому вебсайту було проведено підключення до хмарної БД за допомогою SQL Server Management Studio та виконано запити для надання доступу. Запити мають такий вигляд: `ALTER ROLE db_datareader ADD MEMBER WebAppUser`, `ALTER ROLE db_datawriter ADD MEMBER WebAppUser`.

Після цього, готовий проєкт було опубліковано за допомогою створеного профілю у Visual Studio 2022, який зображено на рисунку 3.7.

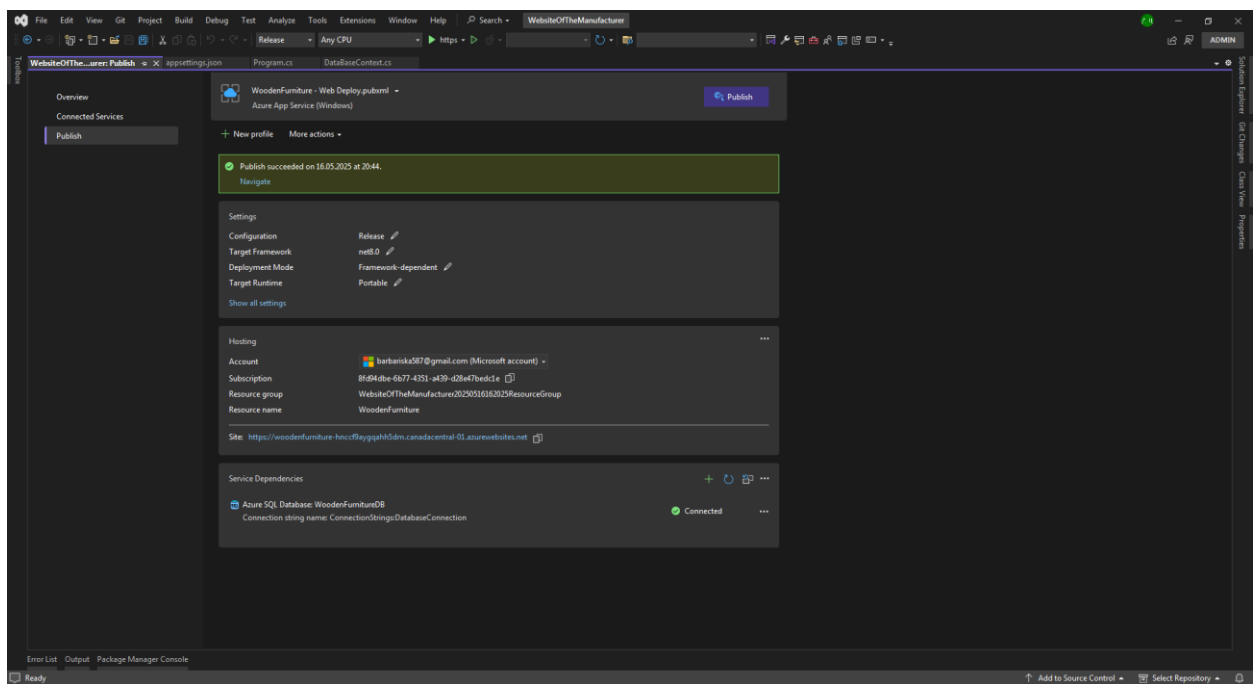


Рисунок 3.7 – Готовий до публікації профіль проєкту у Visual Studio 2022 [16]

Таким чином, розгортання проєкту було успішно виконано. Результат розгортання можна побачити на рисунку 3.8. Тут видно, що опублікований сайт доступний за власним посиланням, а БД успішно з'єдналася, так як каталог товарів завантажився без проблем.

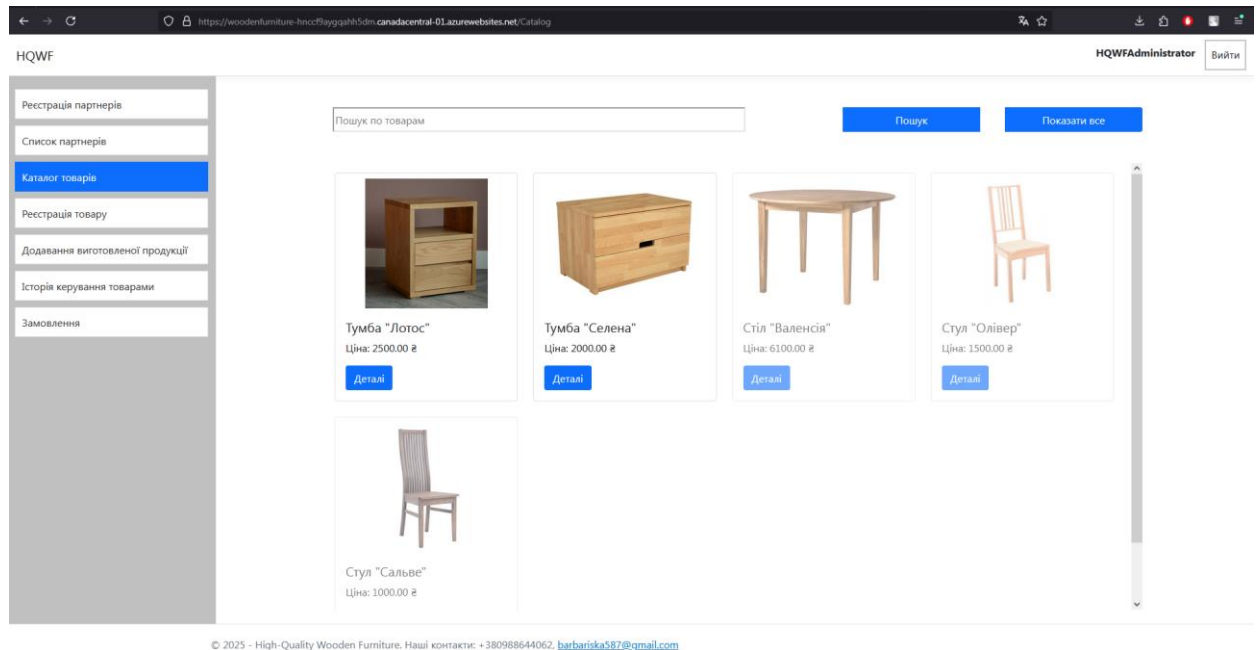


Рисунок 3.8 – Робота розгорнутого вебсайту

3.6 Висновки за розділом 3

В рамках даного розділу було описано обраний шаблон проєктування та наведено основні його особливості. Описано усі реалізовані класи, основна частина яких згідно із шаблоном проєктування поділилась на контролери та моделі. Також було описано представлення, які дають можливість користувачам взаємодіяти з інтерфейсом та використовувати усі визначені функції вебсайту. Після цього розглянуто зпроєктовану БД: наведено схему, описано усі сутності та зазначено усі зв'язки між ними. Таким чином, було здійснено опис всього розробленого проєкту. Готовий вебсайт було розгорнуто за допомогою Microsoft Azure, процес розгортання описано у відповідному пункті. На цьому основний етап розробки вебсайту завершився.

4 КЕРІВНИЦТВО ПРОГРАМІСТА

4.1 Призначення й умови застосування програми

Розроблений вебсайт призначений для використання у сфері обліку та продажу товарів, де має допомогти мікро та малим підприємствам з обмеженими ресурсами спростити процес ведення обліку товарів та реалізовувати власну продукцію при підтримці інших підприємств за допомогою простих і сучасних рішень, починаючи від використання БД і закінчуючи впровадженням автоматизації для керування процесами оформлення і виконання замовлень, додавання та зміни товарів тощо.

Ціль вебзастосунку:

- зменшити час, необхідний для ведення обліку товарів;
- спростити ведення обліку товарів на складі;
- спростити керування списками користувачів, замовлень, каталогом товарів;
- забезпечити збереження усіх необхідних для роботи даних;
- надати малим підприємствам доступне і водночас функціональне рішення для організації електронної комерції;
- спростити та вдосконалити якість аналізу роботи підприємства.

4.2 Характеристики програми

Вебсайт було розроблено на мові програмування C# (для back-end частини) та JavaScript (для front-end частини) з використанням фреймворку ASP.NET Core MVC. У якості СКБД обрано MS SQL Server, для доступу до бази даних використано ORM-фреймворк Entity Framework Core. Для розробки використовувалося IDE Microsoft Visual Studio 2022.

Нижче наведено вимоги до технічних та програмних засобів.

У разі використання власного сервера для роботи вебсайту, до серверної частини висуваються наступні вимоги:

- операційна система: Windows Server 2022;
- процесор: рекомендується процесор Intel Xeon E5-4620@8x2,2GHz або краще;
- пам'ять: оперативна пам'ять має бути достатньою для запуску та обробки запитів, залежить від обсягу даних і навантаження на сервер. Рекомендується не менше 16 ГБ оперативної пам'яті;
- сховище даних: у разі можливості, встановлення бази даних на SSD може значно збільшити продуктивність роботи програми. Рекомендований обсяг накопичувача становить 500ГБ;
- середовище виконання: необхідне встановлене середовище виконання .NET 5/6/7 або .NET Core на сервері для запуску серверної частини додатка.

При використанні хмарного хостингу або платформи як сервісу (PaaS - Platform as a Service), відповідальність за апаратні ресурси лягає на провайдера, і вимоги можуть відрізнятися.

Для якісної роботи клієнтської частини програми рекомендується персональний комп'ютер до складу якого входять наступні компоненти та технічні засоби:

- процесор Intel Core i3-8100 (3.6 ГГц)(Intel UHD Graphics 630) з тактовою частотою, ГГц – 3,6;
- материнська плата Asus Prime H310M-R R2.0 (s1151 v2, Intel H310, PCI-Ex16);
- оперативна пам'ять DDR3 з об'ємом 4ГБ;
- блок живлення 450 Вт;
- об'єм логічної пам'яті SSD – 120 ГБ;
- інструменти вводу інформації – клавіатура та миша;
- засоби виводу інформації – монітор з роздільною здатністю 1920x1080.

Для запуску програми достатньо мати встановлену на комп'ютері будь-яку версію ОС та один із встановлених браузерів (Google Chrome, Mozilla Firefox, Opera, Microsoft Edge).

4.3 Звертання до програми

Запуск вебсайту відбувається за посиланням [18]. При відкритті вебсайту користувач потрапляє на головну сторінку. Тут можна ознайомитись із призначенням і тематикою сайту, побачити контакти для зв'язку з адміністрацією (у футері). В шапці знаходиться кнопка для переходу до меню авторизації, а в лівій частині сторінки розташована навігаційна панель. З її допомогою користувачі можуть використовувати функціонал сайту в залежності від своїх прав. При переході до авторизації, можна зайти на свій акаунт, якщо користувач є адміністратором сайту або представником підприємства-партнера, для якого заздалегідь було зареєстровано акаунт адміністратором. Звичайні відвідувачі сайту не мають можливості увійти до акаунту.

Неавторизований користувач має можливість перейти лише до каталогу товарів. Такий каталог є спільним для усіх типів користувачів. При переході на сторінку товарів можна побачити рядок для пошуку товарів за назвою та кнопки для здійснення пошуку та показу усього списку. Нижче пошукового рядку розташовані картки існуючих товарів. При натисканні на кнопку «Деталі» користувач переходить до перегляду детальних характеристик товару. Гості сайту мають можливість тільки переглянути подану інформацію.

Користувачі від підприємств-партнерів мають розширені права при роботі на сайті. При перегляді детальних характеристик товару вони можуть побачити додаткову інформацію про кількість товару на складі та мають можливість оформити тут замовлення. Для цього варто обрати бажану кількість товару та натиснути кнопку для замовлення. Після цього замовлення

буде автоматично зформовано і додано до списку замовлень активного користувача. Додатково, при оформленні клієнт може побачити інформацію про вартість замовлення, спосіб оплати та спосіб доставки.

Оформлені замовлення користувач може продивитись за допомогою кнопки «Замовлення» у навігаційній панелі. При переході на дану сторінку відвідувач може побачити список своїх замовлень у вигляді таблиці та їх статус. Білим позначаються актуальні замовлення, сірим – скасовані, зеленим – виконані. При натисканні на рядок у списку в правій частині сторінки відображається детальна інформація про обране замовлення. Ці дані доступні тільки для читання. Якщо клієнт має бажання скасувати актуальне замовлення, він може зробити це за допомогою кнопки «Скасувати», що знаходиться у третьому стовпці таблиці. Виконання даної операції потребує підтвердження дії у спливаючому повідомленні. Якщо користувач дав згоду – замовлення змінює свій статус та колір на відповідний. На цьому можливості клієнтів-партнерів закінчуються.

Користувачі-адміністратори мають найбільші права при взаємодії з вебсайтом. Через навігаційну панель вони можуть перейти до реєстрації нового користувача. Тут варто ввести унікальні дані підприємства-партнера та натиснути кнопку «Реєстрація».

Наявні у базі даних акаунти можна продивитись за допомогою сторінки «Список партнерів». Тут у вигляді таблиці відображаються усі зареєстровані користувачі. Якщо є необхідність у видаленні акаунта, адміністратор має деактивувати його за допомогою кнопки «Деактивувати», що знаходиться у третьому стовпці таблиці із списком користувачів. Виконання даної операції потребує підтвердження дії у спливаючому повідомленні. Якщо користувач дав згоду – акаунт змінює свій статус і не відображається у списку. При натисканні на рядок у списку в правій частині сторінки відображається детальна інформація про підприємства, до якого відноситься обраний акаунт. Ці дані можна відредагувати, проте вони мають залишатись унікальними. За необхідністю, адміністратор може змінити пароль від обраного акаунта за

допомогою відповідної форми, що розташована в нижній правій частині сторінки.

При перегляді детальних характеристик обраного з каталогу товару, адміністратор має доступ до усієї інформації, що включає в себе об'єм товару, кількість на складі та кількість товару, що заброньовано для замовлень. Для редагування характеристик товару адміністратор може перейти на відповідну сторінку за допомогою кнопки «Редагувати товар», що знаходиться у правій нижній частині сторінки. Процес редагування поділяється на дві частини: можна змінити або зображення товару (форма для цього знаходиться у верхній частині сторінки, нове зображення має бути унікальним) або інші характеристики товару (форма для цього знаходиться у нижній частині сторінки, назва товару має бути унікальною). Виключенням є категорія, об'єм та розміри товару – ці характеристики змінювати не можна. При успішному редагуванні або виникненні помилок адміністратору відображаються відповідні повідомлення у верхній частині сторінки. При внесенні змін на даній сторінці зберігається запис про редагування товару.

За допомогою кнопки у навігаційному меню «Реєстрація товару» відбувається додавання до каталогу нового товару. При додаванні адміністратору варто заповнити усі наявні поля, обрати унікальне зображення та назву товару, обрати категорію та заповнити унікальні атрибути, пов'язані з нею. При успішній реєстрації товару або виникненні помилок адміністратору відображаються відповідні повідомлення у верхній частині сторінки. При виконанні операцій на даній сторінці зберігається запис про реєстрацію товару.

На сторінці «Додавання виготовленої продукції» адміністратор має змогу додавати товар на склад. Сторінка за своєю структурою повторює сторінку з каталогом товарів, тут наявний рядок пошуку та картки з товарами, проте в нижній частині ще додається блок для додавання товарів. Дана операція відбувається таким чином: адміністратор обирає картку товару, який хоче додати і в нижній частині сторінки обирає кількість товару для додавання

на склад. Тут також відображається об'єм товарів, що додається та наповненість складу. Максимальна кількість для додавання продукції обмежується вільним місцем на складі та об'ємом однієї одиниці товару. При успішному наповненні складу або виникненні помилок адміністратору відображаються відповідні повідомлення у верхній частині сторінки. При виконанні операцій на даній сторінці зберігається запис про додавання товару.

Сторінка «Історія керування товарами» дозволяє адміністратору продивитись усі операції, які були здійснені над товарами. Сюди входить реєстрація, редагування та додавання товару на склад. Дана інформація доступна тільки для читання. При редагуванні зберігається запис лише про сам факт редагування, без зазначення детальних змін. При додаванні товару зазначається кількість товару, яку було додано до складу.

При відкритті адміністратором сторінки «Замовлення» він може побачити список замовлень від усіх користувачів та їх статус. Білим позначаються актуальні замовлення, сірим – скасовані, зеленим – виконані. При натисканні на рядок у списку в правій частині сторінки відображається детальна інформація про обране замовлення. Ці дані доступні тільки для читання. Якщо адміністратору необхідно позначити виконаним актуальне замовлення, він може зробити це за допомогою кнопки «Виконати», що знаходиться у третьому стовпці таблиці. На цьому можливості адміністраторів закінчуються.

Вихід користувача з акаунту здійснюється тільки при натисканні відповідної кнопки у шапці сайту. При звичайному закритті вкладки, дані про активну сесію залишаються.

4.4 Початкові та вихідні дані

Початковими даними при розгортанні вебсайту будуть наявні в БД зареєстрований адміністратор (Логін: HQWFAdministrator, пароль:

PasswordForSite138), визначені категорії (столи, стільці, тумби), визначені унікальні атрибути для характеристик:

- столи (призначення стола, форма стола, механізм трансформації);
- стільці (призначення стільця, висота стільця, каркас);
- тумби (призначення тумби, кількість ящиків, тип відкриття ящиків).

Вихідними даними будуть списки з відповідною інформацією (користувачі, замовлення, історія керування товарами, товари). В залежності від обраних дій користувача та працюючих алгоритмів вихідними даними можуть бути результати оновлень БД, повідомлення про помилки та успішне виконання операцій, email-повідомлення, змінені згідно із фільтром списки даних.

4.5 Повідомлення програмісту

Для програміста окремих повідомлень передбачено не було.

5 КЕРІВНИЦТВО ОПЕРАТОРА

5.1 Призначення програми

Розроблений вебсайт орієнтований на використання у сфері обліку та продажу товарів, зокрема для мікро- та малих підприємств з обмеженими ресурсами. Його основна мета полягає у спрощенні процесів ведення обліку товарів та реалізації продукції шляхом інтеграції сучасних технологічних рішень, включаючи бази даних та автоматизацію управлінських процесів, таких як оформлення та виконання замовлень, додавання та оновлення інформації про товари.

Основні цілі вебзастосунку включають:

- зменшення часу, необхідного для ведення обліку товарів;
- оптимізацію процесів обліку товарів на складі;
- спрощення управління списками користувачів, замовлень та каталогами товарів;
- забезпечення надійного зберігання всіх необхідних для функціонування підприємства даних;
- надання малим підприємствам доступного та ефективного рішення для організації електронної комерції;
- удосконалення процесів аналізу діяльності підприємства з метою підвищення ефективності його роботи.

5.2 Умови виконання програми

Для якісної роботи клієнтської частини програми рекомендується персональний комп'ютер до складу якого входять наступні компоненти та технічні засоби:

- процесор Intel Core i3-8100 (3.6 ГГц)(Intel UHD Graphics 630) з тактовою частотою, ГГц – 3,6;

- материнська плата Asus Prime H310M-R R2.0 (s1151 v2, Intel H310, PCI-Ex16);
- оперативна пам'ять DDR3 з об'ємом 4ГБ;
- блок живлення 450 Вт;
- об'єм логічної пам'яті SSD – 120 ГБ;
- інструменти вводу інформації – клавіатура та миша;
- засоби виводу інформації – монітор з роздільною здатністю 1920x1080.

Для запуску програми достатньо мати встановлену на комп'ютері будь-яку версію ОС та один із встановлених браузерів (Google Chrome, Mozilla Firefox, Opera, Microsoft Edge).

5.3 Виконання програми

Запуск вебсайту здійснюється за посиланням [18]. Після його відкриття користувач потрапляє на головну сторінку, де надається інформація про основну мету та тематику ресурсу, а також контакти для зв'язку з адміністрацією, що розташовані у футері. У шапці сайту розміщена кнопка для переходу до меню авторизації, а ліворуч — навігаційна панель, що дає користувачам змогу взаємодіяти з функціоналом сайту залежно від їхніх прав.

При переході до розділу авторизації, користувач може увійти в свій акаунт, якщо він є адміністратором сайту або представником підприємства-партнера, акаунт якого був зареєстрований адміністрацією. Звичайні відвідувачі не мають можливості авторизації.

Гості можуть отримати доступ лише до загального каталогу товарів, який є доступним для всіх типів користувачів. Відвідувач може здійснити пошук товарів за назвою, використовуючи відповідний рядок пошуку. У нижній частині сторінки відображаються картки товарів, кожна з яких містить кнопку «Деталі», що дає змогу перейти до перегляду детальних характеристик товару. Гості сайту можуть лише переглядати інформацію про товари.

Користувачі, які представляють підприємства-партнери, володіють розширеними правами доступу до функціоналу вебсайту. При перегляді детальних характеристик товару вони отримують додаткову інформацію щодо наявності товару на складі та мають можливість оформити замовлення безпосередньо на сайті. Для цього користувач вибирає необхідну кількість товару та натискає кнопку для оформлення замовлення. Після цього замовлення автоматично генерується і додається до списку замовлень користувача. Крім того, при оформленні замовлення відображається інформація про його вартість, способи оплати та доставки.

Оформлені замовлення можна переглянути через кнопку «Замовлення», розташовану на навігаційній панелі. Після переходу на відповідну сторінку, користувач бачить список своїх замовлень у вигляді таблиці з відображенням їх статусів. Замовлення позначаються білим кольором, якщо вони актуальні, сірим — якщо скасовані, та зеленим — якщо виконані. При натисканні на конкретний рядок таблиці в правій частині сторінки з'являється детальна інформація про вибране замовлення, яка доступна тільки для перегляду. Якщо користувач бажає скасувати актуальне замовлення, він може здійснити це через кнопку «Скасувати», розташовану у третьому стовпці таблиці. Для підтвердження скасування з'являється спливаюче повідомлення, і після згоди користувача замовлення змінює свій статус і колір відповідно до нової категорії. На цьому права користувачів-партнерів обмежуються.

Користувачі-адміністратори мають найбільші повноваження при взаємодії з вебсайтом. Вони можуть перейти до реєстрації нового користувача через навігаційну панель, заповнивши унікальні дані підприємства-партнера та натиснувши кнопку «Реєстрація».

Перегляд наявних у базі даних користувацьких акаунтів здійснюється через сторінку «Список партнерів». У даному розділі всі зареєстровані користувачі відображаються у вигляді таблиці. У разі потреби адміністратор має можливість деактивувати обраний акаунт шляхом натискання відповідної

кнопки, розміщеної в третьому стовпці таблиці. Здійснення цієї операції супроводжується спливаючим повідомленням для підтвердження дії. Після підтвердження акаунт змінює свій статус і більше не відображається у загальному списку.

При виборі акаунту у правій частині інтерфейсу відображається розширена інформація про підприємство, до якого належить відповідний користувач. Ці дані можуть бути відредаговані адміністратором, однак мають залишатися унікальними в межах бази даних. За необхідності також передбачена можливість зміни пароля для обраного користувача — відповідна форма розміщена в нижньому правому куті сторінки.

При перегляді детальних характеристик товару, обраного з каталогу, адміністратор має доступ до всієї необхідної інформації, зокрема об'єму товару, кількості одиниць на складі та кількості товару, заброньованого для замовлень. Для здійснення редагування характеристик товару адміністратор може перейти на відповідну сторінку за допомогою кнопки «Редагувати товар», розташованої в правому нижньому куті сторінки. Процес редагування поділяється на дві частини: зміну зображення товару, що здійснюється через форму в верхній частині сторінки (нове зображення повинно бути унікальним), та зміну інших характеристик товару, що здійснюється через форму в нижній частині сторінки (назва товару має бути унікальною). Винятком становлять категорія та об'єм — ці характеристики не підлягають зміні. Після успішного редагування або виникнення помилок адміністратор отримує відповідні повідомлення в верхній частині сторінки. Внесення змін на даній сторінці автоматично фіксується в історії змін товарів у вигляді запису про редагування товару.

За допомогою кнопки в навігаційному меню «Реєстрація товару» здійснюється додавання нового товару до каталогу. Під час цього процесу адміністратор зобов'язаний заповнити всі необхідні поля, вибрати унікальне зображення та назву товару, обрати категорію і заповнити відповідні унікальні атрибути, пов'язані з цією категорією. У разі успішної реєстрації товару або

виникнення помилок, адміністратор отримує відповідне повідомлення в верхній частині сторінки. Всі операції на цій сторінці фіксуються в історії змін товарів як запис про реєстрацію товару.

Сторінка «Додавання виготовленої продукції» дозволяє адміністратору додавати товар до складу. Структурно вона є схожою на сторінку каталогу товарів і містить функціонал для пошуку та перегляду карток товарів. В нижній частині сторінки додано блок для додавання товарів на склад. Процес додавання здійснюється шляхом вибору картки товару, який необхідно додати, та визначення кількості одиниць товару для поповнення складу. Окрім того, на цій сторінці відображаються дані про об'єм товару, що додається, а також рівень заповнення складу. Кількість товару, яку можна додати, обмежується доступним місцем на складі та об'ємом однієї одиниці товару. Після успішного виконання операції або виникнення помилок адміністратор отримує відповідне повідомлення в верхній частині сторінки. Операції, виконані на цій сторінці, автоматично зберігаються в історії змін товарів як запис про додавання товару на склад.

Сторінка «Історія керування товарами» надає адміністратору можливість переглядати всі операції, виконані з товарами. До цих операцій відносяться реєстрація, редагування та додавання товарів на склад. Дана інформація доступна лише для перегляду. При редагуванні товару фіксується лише факт редагування без зазначення конкретних змін. У разі додавання товару вказується кількість одиниць, які були додані на склад.

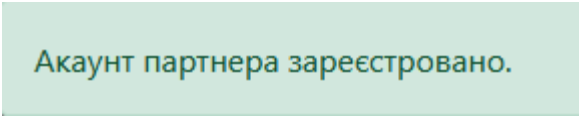
Сторінка «Замовлення» надає адміністратору змогу переглядати список всіх замовлень користувачів та їх статуси. Актуальні замовлення позначаються білим кольором, скасовані — сірим, а виконані — зеленим. При виборі запису у списку, у правій частині сторінки відображається детальна інформація про обране замовлення. Ці дані доступні тільки для перегляду. Якщо адміністратор має необхідність позначити актуальне замовлення як виконане, він може скористатися кнопкою «Виконати», що розташована в третьому стовпці таблиці. На цьому права адміністратора закінчуються.

Вихід користувача з акаунту можливий лише за допомогою натискання відповідної кнопки в шапці сайту. При звичайному закритті вкладки дані про активну сесію не зникають.

5.4 Повідомлення оператору

При виконанні усіх дій, наведених у пункті 5.3, користувачів будуть зустрічати різні повідомлення, що допоможуть при роботі з вебсайтом. Загалом, повідомлення стосуються або успішного виконання операції або виникнення проблем.

Повідомлення, що вказує користувачу на успішне виконання операції наведено на рисунку 5.1. Такі повідомлення будуть зустрічати користувача під час реєстрації користувачів, реєстрації та редагуванні товару, оформленні замовлення, додаванні товару на склад.



Акаунт партнера зареєстровано.

Рисунок 5.1 – Повідомлення про успішне виконання операції

Повідомлення, що вказує користувачу на виникнення проблем при виконанні операції наведено на рисунку 5.2. Такі повідомлення будуть зустрічати користувача під час входу в акаунт, реєстрації користувачів, реєстрації та редагуванні товару, додаванні товару на склад.

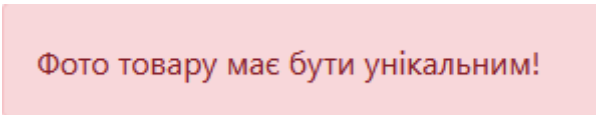


Фото товару має бути унікальним!

Рисунок 5.2 – Повідомлення про виникнення помилки при виконанні операції

При роботі із списками замовлень або зареєстрованих акаунтів, користувач може побачити спливаюче повідомлення про підтвердження дії.

Приклад такого повідомлення наведено на рисунку 5.3. Також, тут користувач може побачити повідомлення про виникнення помилок. Приклад такого повідомлення наведено на рисунку 5.4. На рисунку 5.5 показано спливаюче повідомлення, що вказує на успішну зміну паролю.

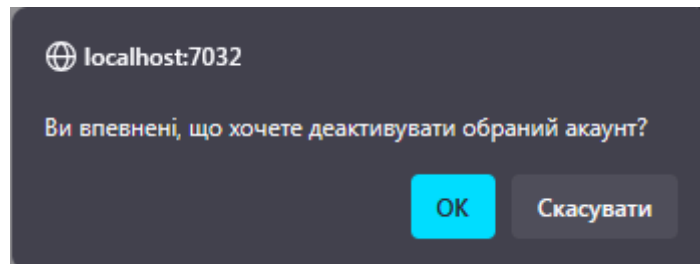


Рисунок 5.3 – Спливаюче повідомлення про підтвердження дії

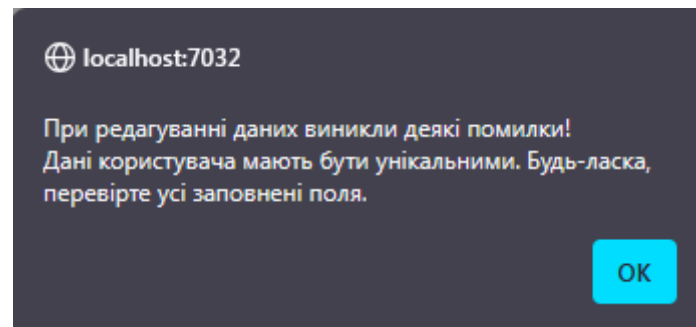


Рисунок 5.4 – Спливаюче повідомлення про виникнення помилки при виконанні операції

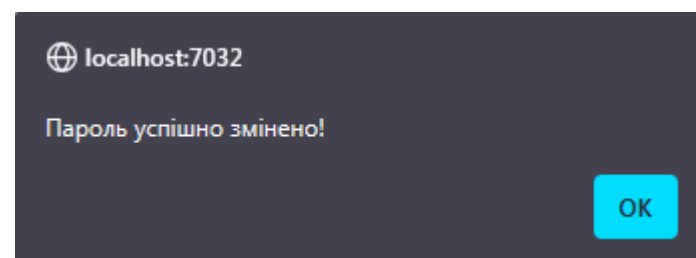


Рисунок 5.5 – Спливаюче повідомлення при успішній зміні паролю

Окрім цього, при редагуванні адміністратором даних акаунтів партнерів, їм надсилаються повідомлення про внесення змін. Приклад такого повідомлення наведено на рисунку 5.6.

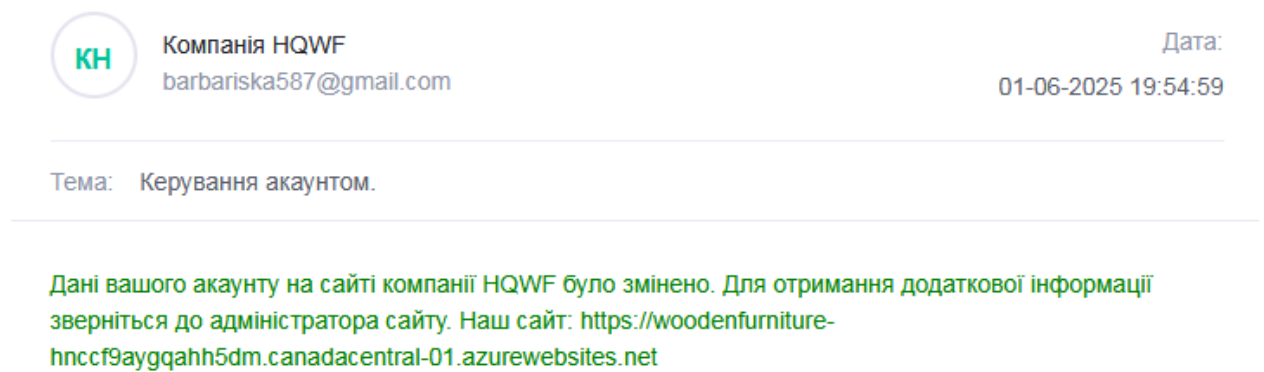


Рисунок 5.6 – Email-повідомлення від адміністратора

6 ТЕСТУВАННЯ І РЕЗУЛЬТАТИ

6.1 Опис методики проведення тестування

Тестування програмного забезпечення проводилося під час розробки функціоналу вебсайту. Завдяки такому методу тестування було виявлено та виправлено значну кількість помилок. Додатково, було проведено повне тестування вебсайту шляхом його практичного використання після завершення розробки. Таким чином, на момент релізу кількість багів та помилок мінімальна.

6.2 Результати використання розробленого програмного забезпечення

В таблиці 6.1 наведено перелік проведених тестів функціоналу та їх результати

Таблиця 6.1 – Результати тестування ПЗ

Тестування	Результат	Коментарі	Користувач, для якого проводилося тестування
1	2	3	4
Перехід на сайт за посиланням	OK	Відкривається головна сторінка сайту	Гість
Перехід до каталогу товарів	OK	Відкривається сторінка з каталогом товарів	Гість, Партнер, Адміністратор
Пошук товарів в каталозі	OK	Каталог фільтрується згідно пошукового запиту	Гість, Партнер, Адміністратор
Вхід в систему. Валідні дані	OK	Користувач увійшов у свій акаунт	Гість
Вхід в систему. Невалідні дані	OK	Користувач отримав повідомлення про помилку	Гість
Скасування фільтру пошуку	OK	Відображається повний каталог товарів	Гість, Партнер, Адміністратор

Продовження таблиці 6.1

1	2	3	4
Пошук товарів в каталозі за пустим полем	OK	Відображається повний каталог товарів	Гість, Партнер, Адміністратор
Перехід до детальних характеристик товару	OK	Відкривається сторінка з характеристиками товару	Гість, Партнер, Адміністратор
Коректне відображення характеристик товару в залежностей від ролі користувача	OK	Гість бачить тільки основну інформацію. Партнер додатково бачить інформацію про точну кількість товару на складі. Адміністратор бачить інформацію партнера та додатково об'єм одиниці та кількість товару, заброньованого для замовлень	Гість, Партнер, Адміністратор
Оформлення замовлень тільки партнером	OK	Гість не має таких прав. Адміністратор не може оформлювати замовлення. Окрім партнера, ніхто не бачить елементи для оформлення замовлень	Гість, Партнер, Адміністратор
Автоматичний розрахунок вартості замовлення при виборі кількості товару	OK	Вартість коректно розраховується	Партнер
Обмеження максимальної кількості товару для замовлення кількістю товару на складі	OK	Обмеження кількості товару для замовлення працює коректно	Партнер
Автоматичний розрахунок об'єму замовлення після оформлення	OK	Об'єм замовлення коректно розраховується	Партнер

Продовження таблиці 6.1

1	2	3	4
Перехід до списку замовлень	OK	Відкривається сторінка з переліком замовлень	Партнер, Адміністратор
Коректне відображення замовлень згідно з їх статусом	OK	Актуальні замовлення виділені білим кольором, скасовані - сірим, а виконані - зеленим	Партнер, Адміністратор
Скасування тільки актуального замовлення	OK	Виконані або вже скасовані замовлення не дають можливості їх скасувати	Партнер
Скасування замовлення	OK	Замовлення змінює свій статус	Партнер
Коректна зміна кількості товару після скасування	OK	Кількість товару на складі збільшилась. Кількість товару, заброньованого для замовлення зменшилась	Партнер
Оновлення списку замовлень після скасування	OK	Список замовлень коректно оновлюється	Партнер
Перегляд детальної інформації про обране замовлення	OK	Детальна інформація коректно відображається	Партнер, Адміністратор
Інформацію про обране замовлення неможливо редагувати	OK	Інформація доступна тільки для читання	Партнер, Адміністратор
Пошук замовлень у списку за наявними стовпцями	OK	Список замовлень коректно фільтрується	Партнер, Адміністратор
Сортування списку замовлень за наявними стовпцями	OK	Список замовлень коректно сортується	Партнер, Адміністратор

Продовження таблиці 6.1

1	2	3	4
Зміна статусу замовлення на «Виконано»	OK	Замовлення змінює свій статус	Адміністратор
Виконання тільки актуального замовлення	OK	Скасовані або вже виконані замовлення не дають можливості їх виконати	Адміністратор
Оновлення списку замовлень після виконання	OK	Список замовлень коректно оновлюється	Адміністратор
Перехід на сторінку реєстрації партнерів	OK	Відкривається сторінка для реєстрації партнерів	Адміністратор
Реєстрація партнера з валідними даними	OK	Партнера успішно зареєстровано	Адміністратор
Реєстрація партнера з невалідними даними	OK	Виводиться повідомлення про помилку	Адміністратор
Реєстрація партнера з неунікальними даними	OK	Виводиться повідомлення про помилку	Адміністратор
Перехід до списку партнерів	OK	Відкривається сторінка зі списком партнерів	Адміністратор
Відображення тільки активних акаунтів у списку	OK	Список партнерів відображається коректно	Адміністратор
Деактивація обраного акаунту	OK	Обраний акаунт деактивується. Список партнерів оновлюється	Адміністратор
Відображення детальної інформації про користувача	OK	Детальна інформація коректно відображається	Адміністратор

Продовження таблиці 6.1

1	2	3	4
Редагування даних користувача. Валідні дані	OK	Дані користувача коректно редагуються	Адміністратор
Редагування даних користувача. Невалідні дані	OK	Редагування даних не відбувається. Поля з невалідними даними візуально виділяються	Адміністратор
Редагування даних користувача. Унікальні дані	OK	Дані користувача коректно редагуються	Адміністратор
Редагування даних користувача. Неунікальні дані	OK	Редагування даних не відбувається. Виводиться повідомлення про помилку	Адміністратор
Зміна паролю обраного акаунту. Валідні дані	OK	Пароль від акаунту змінюється. Виводиться повідомлення про успішну операцію	Адміністратор
Зміна паролю обраного акаунту. Невалідні дані	OK	Зміна паролю не відбувається. Поля з невалідними даними візуально виділяються	Адміністратор
Зміна паролю обраного акаунту. Невірний поточний пароль	OK	Зміна паролю не відбувається. Виводиться повідомлення про помилку	Адміністратор
Відправка email-повідомлень при роботі з акаунтом	OK	Коли адміністратор редагує акаунт, відповідний партнер отримує повідомлення на вказану пошту	Партнер, Адміністратор
Пошук партнерів у списку за наявними стовпцями	OK	Список партнерів коректно фільтрується	Адміністратор

Продовження таблиці 6.1

1	2	3	4
Сортування списку партнерів за наявними стовпцями	OK	Список партнерів коректно сортується	Адміністратор
Перехід до редагування обраного товару тільки адміністратором	OK	Гості і партнери не мають необхідних прав і не бачать кнопки для редагування	Гість, Партнер, Адміністратор
Перехід до редагування товару	OK	Відкривається сторінка редагування товару	Адміністратор
Зміна зображення товару. Унікальне зображення	OK	Зображення товару змінюється. Адміністратор отримує відповідне повідомлення	Адміністратор
Зміна зображення товару. Неунікальне зображення	OK	Зображення товару не змінюється Адміністратор отримує відповідне повідомлення про помилку	Адміністратор
Зміна зображення товару. Зображення не обрано	OK	Операція не відбувається. Поле для вибору зображення візуально виділяється і повідомляє про необхідність вибору фото.	Адміністратор
Зміна характеристик товару. Валідні дані	OK	Характеристики товару змінюються. Адміністратор отримує відповідне повідомлення	Адміністратор
Зміна характеристик товару. Невалідні дані	OK	Характеристики товару не змінюються. Адміністратор отримує відповідне повідомлення про помилку	Адміністратор
Заборона зміни об'єму	OK	Дана інформація доступна тільки для читання	Адміністратор

Продовження таблиці 6.1

1	2	3	4
Зміна характеристик товару. Назва товару унікальна	OK	Характеристики товару змінюються. Адміністратор отримує відповідне повідомлення	Адміністратор
Перехід до реєстрації товару	OK	Відкривається сторінка реєстрації товару	Адміністратор
Реєстрація товару. Валідні дані	OK	Реєстрація товару проходить успішно. Адміністратор отримує відповідне повідомлення	Адміністратор
Реєстрація товару. Невалідні дані	OK	Реєстрація товару не відбувається. Адміністратор отримує відповідне повідомлення про помилку	Адміністратор
Реєстрація товару. Назва та зображення товару унікальні	OK	Реєстрація товару проходить успішно. Адміністратор отримує відповідне повідомлення	Адміністратор
Реєстрація товару. Назва або зображення товару неунікальні	OK	Реєстрація товару не відбувається. Адміністратор отримує відповідне повідомлення про помилку	Адміністратор
Перехід до додавання товару на склад	OK	Відкривається сторінка для додавання товару на склад	Адміністратор
Додавання товару на склад без вибору товару	OK	Кнопка додавання товару недоступна	Адміністратор
Максимальна кількість товару для додавання розраховується автоматично	OK	Максимальна кількість товару для додавання коректно розраховується в залежності від об'єму товару та наповненості складу	Адміністратор
Об'єм товарів, що додається на склад, розраховується автоматично	OK	Об'єм товарів коректно розраховується в залежності від об'єму одиниці товару та обраної кількості для додавання	Адміністратор

Кінець таблиці 6.1

1	2	3	4
Додавання товарів з об'ємом менше доступного місця на складі	OK	Додавання товару на склад успішно здійснюється. Адміністратор отримує відповідне повідомлення	Адміністратор
Додавання товарів з об'ємом більше доступного місця на складі	OK	Додавання товару на склад не відбувається. Адміністратор отримує відповідне повідомлення про помилку	Адміністратор
Відображення відфільтрованого списку товарів після додавання (Якщо список було відфільтровано)	OK	Після додавання товару на склад пошуковий запит адміністратора залишається і список товарів коректно відображається згідно фільтру	Адміністратор
Перехід до історії керування товарами	OK	Відкривається сторінка з історією керування товарами	Адміністратор
Пошук записів у списку операцій за наявними стовпцями	OK	Список операцій над товарами коректно фільтрується	Адміністратор
Сортування списку операцій над товарами за наявними стовпцями-	OK	Список операцій над товарами коректно сортується	Адміністратор
Вихід з акаунту	OK	Активна сесія завершується і користувач потрапляє на сторінку входу в акаунт	Партнер, Адміністратор

В таблиці 6.2 наведено перелік проведених тестів верстки сайту та їх результати

Таблиця 6.2 – Результати тестування верстки сайту

Тестування	Результат	Коментарі
Коректне відображення шапки і футеру сайту	BUG	Футер сайту при використанні майстер-сторінки _Layout2.cshtml (реєстрація та вхід в акаунт) має більшу висоту, ніж футер, що визначено в майстер-сторінці _Layout.cshtml
Наявна навігаційна панель на усіх сторінках, де це передбачено	OK	Для сторінок реєстрації та входу в акаунт відображення навігаційної панелі не передбачено
Елементи інтерфейсу адаптовано під роздільну здатність 1920x1080	OK	
Орфографія та граматика контенту сайту	OK	Помилки в орфографії та граматиці відсутні
Коректне вирівнювання елементів на сторінках	OK	
Коректне вирівнювання елементів на сторінках при відображенні повідомлень	OK	При відображенні валідаційних або повідомлень про успіх/помилки в здійсненні операцій усі елементи сторінок залишаються на своїх місцях
Коректне відображення таблиць із списками даних при їх наповненні	BUG	Якщо назва підприємства занадто велика – з'являється горизонтальна прокрутка при перегляді списку користувачів. Відповідно, перегляд усіх стовпців таблиці одразу стає неможливим.
Зміна курсору на pointer при наведенні на клікабельний елемент	OK	
Візуальне виділення відкритої сторінки у навігаційній панелі	OK	Відкрита сторінка виділяється синім кольором у навігаційній панелі
Коректне відображення стилів та шрифтів	OK	

Кінець таблиці 6.2

Тестування	Результат	Коментарі
Візуальне виділення обраного рядку у списку (там, де таке виділення доступне)	BUG	Якщо у списках партнерів або замовлень обрати рядок, перейти на другу сторінку таблиці і обрати ще один рядок, то при зміні сторінок таблиці на обох сторінках один рядок буде відображатись як виділений. Технічно виділеним залишається останній обраний рядок

Усі виявлені баги та помилки будуть якнайшвидше виправлені у наступній версії вебзастосунку.

ВИСНОВКИ

В ході виконання даної роботи було зроблено наступне.

Проведено аналіз предметної області – визначено цільову аудиторію, проведено порівняння існуючого програмного забезпечення для вирішення поставлених задач та на основі отриманої інформації зформовано вимоги до нового вебзастосунку.

Розроблено універсальний вебсайт за допомогою таких інструментів і технологій, як мови програмування C# та JavaScript, IDE Visual Studio 2022, ASP.NET Core MVC, Entity Framework Core та MS SQL Server.

Протестовано розроблене ПЗ. Визначено, що вебсайт працює коректно. Усі розроблені функції працюють без помилок, а застосунок забезпечує виконання усіх поставлених задач. При тестуванні верстки були виявлені незначні візуальні баги, які будуть ліквідовані в найближчому оновленні вебсайту при наданні техпідтримки.

Результатом створення нового вебсайту є підвищення ефективності ведення обліку товарів на складі та надання автоматизованого функціоналу для реалізації товарів через підприємства-партнерів. Завдяки розробленому вебзастосунку мікро та малі підприємства з обмеженими ресурсами отримали можливість легко і швидко керувати каталогом товарів, взаємодіяти з партнерами, керувати замовленнями. Завдяки вільному доступу, сайт буде поширювати свою популярність шляхом його відвідування випадковими користувачами та ознайомить потенційних клієнтів з новим виробником меблів.

На момент публікації вебзастосунок містить в собі весь необхідний функціонал. Однак, залишається можливість для масштабування системи у таких напрямках:

- інтернет-магазину (додавання кошику, збільшення варіантів фільтрації товарів, додавання нових категорій та характеристик, виготовлення ексклюзивних виробів на замовлення);

- обліку товарів (взаємодія з декількома складами, керування переміщеннями товарів між ними, імпорт списку товарів до каталогу з Excel-файлу, додавання користувачів-модераторів);
- взаємодії з партнерами (додавання функціоналу для керування власним акаунтом, складання та збереження угод між підприємствами, відновлення деактивованих акаунтів);
- замовлень (інтеграція сервісів для додавання нових способів оплати та доставки товарів, формування звітів за різні періоди роботи підприємства для складання статистики, облік грошових прибутків та витрат).

Окремо варто зазначити, що вдосконалення зовнішнього вигляду вебсайту теж є важливою частиною розробки, тому можна працювати і в цьому напрямку.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Лісова та деревообробна промисловість [Електронний ресурс]. – Режим доступу: <https://uahistory.co/zno/geography-zno-2020-kyzishin-comprehensive-edition/38.php>
2. «Навички для смарт-спеціалізації» для меблевиків та деревообробників [Електронний ресурс]. – Режим доступу: <https://uafm.com.ua/navychky-dlya-smart-spetsializatsiyi-dlya-meblevykiv-ta-derevoobrobnykiv/>
3. Магазин дерев'яних меблів Закарпаття від виробника «Meblisat» [Електронний ресурс]. – Режим доступу: meblisat.com.ua
4. Ефективний облік товарів на складі [Електронний ресурс]. – Режим доступу: <https://usap.online/blog/efektivnij-oblik-tovariv-na-skladi/>
5. Cloud WMS vs On-Premise WMS: Advantages and Key Differences [Electronic resource]. – Access mode: <https://wizefulfill.com/blog/cloud-wms/>
6. Чому створення інтернет-магазину це один з кращих методів розширення бізнесу [Електронний ресурс]. – Режим доступу: https://lb.ua/tech/2023/10/23/579979_chomu_stvorenniya_internetmagazinu_tse.html
7. Переваги інтернет-магазину [Електронний ресурс]. – Режим доступу: <https://www.legion-marketing.com.ua/chomu-varto-vidkrivati-internet-magazin/>
8. Інтернет-магазин: примха сучасності чи необхідність для вдалого бізнесу? [Електронний ресурс]. – Режим доступу: <https://web4u.in.ua/blog/nternet-magazin-primha-suchasnost-chi-neobh-dn-st-dlya-vdalogo-b-znesu-17>
9. USAP-Online [Електронний ресурс]. – Режим доступу: <https://my.usap.online/>
10. RemOnline [Електронний ресурс]. – Режим доступу: <https://web.remonline.app/warehouse/residue>

11. ТеМП-Меблі [Електронний ресурс]. – Режим доступу: <https://temp-mebel.com.ua/>
12. MASIO [Електронний ресурс]. – Режим доступу: <https://masio.com.ua/>
13. Overview of ASP.NET Core MVC [Electronic resource]. – Access mode: <https://learn.microsoft.com/en-us/aspnet/core/mvc/overview?view=aspnetcore-9.0>
14. Особливості ASP.Net Core MVC [Електронний ресурс]. – Режим доступу: <https://javascript.org.ua/osoblivosti-asp-net-core-mvc/>
15. Cloud Computing Services | Microsoft Azure [Electronic resource]. – Access mode: <https://azure.microsoft.com/en-us>
16. Visual Studio 2022 IDE [Electronic resource]. – Access mode: <https://visualstudio.microsoft.com/vs/>
17. Microsoft Data Migration Assistant v5.8 [Electronic resource]. – Access mode: <https://www.microsoft.com/en-us/download/details.aspx?id=53595>
18. High-Quality Wooden Furniture [Електронний ресурс]. – Режим доступу: <https://woodenfurniture-hnccf9aygqahh5dm.canadacentral-01.azurewebsites.net>

ДОДАТОК А
Технічне завдання

A.1 Вступ

A.1.1 Назва вебсайту

Назва вебсайту – High-Quality Wooden Furniture. Назва проєкту при розробці - WebsiteOfTheManufacturer.

A.1.2 Призначення та область застосування

Розроблений вебсайт призначений для використання у сфері обліку та продажу товарів, де має допомогти мікро та малим підприємствам з обмеженими ресурсами спростити процес ведення обліку товарів та реалізовувати власну продукцію при підтримці інших підприємств за допомогою простих і сучасних рішень, починаючи від використання БД і закінчуючи впровадженням автоматизації для керування процесами оформлення і виконання замовлень, додавання та зміни товарів тощо.

A.2 Підстави для розробки

Дипломна кваліфікаційна робота виконується за темою «Розробка вебсайту для обліку товарів на складі та забезпечення внутрішньої взаємодії з підприємствами» у відповідності з наказом від 28 квітня 2025 року № 209 за Національним університетом «Запорізька політехніка».

A.3 Призначення розробки

Ціль вебзастосування:

- зменшити час, необхідний для ведення обліку товарів;
- спростити ведення обліку товарів на складі;
- спростити керування списками користувачів, замовлень, каталогом товарів;

- забезпечити збереження усіх необхідних для роботи даних;
- надати малим підприємствам доступне і водночас функціональне рішення для організації електронної комерції;
- спростити та вдосконалити якість аналізу роботи підприємства.

A.4 Вимоги до програми

A.4.1 Вимоги до функціональних характеристик

Програмне забезпечення має надавати можливість авторизації на сайті та розподіляти користувачів на три ролі: адміністратор, партнери та гості.

Гості: користувачі, що не мають власного акаунта на сайті. Мають можливість переглядати каталог доступної продукції та детальні характеристики обраного товару.

Партнери: користувачі-підприємства, які мають зареєстрований адміністратором акаунт на сайті. Мають можливість переглядати каталог доступної продукції та детальні характеристики обраного товару, а також оформлювати замовлення на товар в декілька кліків. Список замовлень та детальну інформацію про них можна буде передивитись на відповідній сторінці сайту. За бажанням – партнер зможе скасувати замовлення.

Адміністратор — користувач від підприємства-власника сайту, що має права на повне керування сайтом, а саме:

- реєстрація акаунтів для підприємств-партнерів;
- можливість переглядати каталог доступної продукції та детальні характеристики, а також редагувати інформацію про обраний товар;
- можливість реєструвати новий товар у каталозі;
- можливість переглядати інформацію про наповненість складу, та додавати нові товари на склад. Максимальна кількість товару для додавання обмежується вільним місцем на складі та об'ємом однієї одиниці обраного товару;

- переглядати історію керування товарами, яка формується при виконанні таких дій, як: реєстрація, редагування і додавання товару на склад;
- передивлятися усі дійсні акаунти, детальну інформацію про них та виконувати з ними наступні дії: редагувати інформацію акаунту, змінювати пароль від акаунту та деактивувати акаунт;
- можливість продивитись список всіх замовлень та детальну інформацію про них. Актуальні замовлення можна позначити як виконані.

При взаємодії адміністратора з акаунтом партнера слід забезпечити надсилання email-повідомлень відповідному підприємству про виконання конкретних дій адміністратором.

А.4.2 Вимоги до надійності

Програмою мають оброблятися усі вказані та ті збійні ситуації, що можуть виникати при користуванні продуктом:

- відсутній зв'язок з базою даних;
- занесення до БД некоректних даних;
- неможливість коректного зчитування даних з БД.

Надійне функціонування вебсайту повинно бути забезпечено замовником у сукупності організаційно-технічних пунктів, які описані нижче:

- безперервне постачання живлення технічних пристроїв на яких буде відкрито вебсайт;
- забезпечення безперервного підключення до мережі Інтернет;
- використання ліцензованого програмного забезпечення;
- регулярним виконанням вимог ДСТУ 3396.0-96 захисту інформації.

А.4.3 Вимоги до технічних та програмних засобів

У разі використання власного сервера для роботи вебсайту, до серверної частини висуваються наступні вимоги:

- операційна система: Windows Server 2022;
- процесор: рекомендується процесор Intel Xeon E5-4620@8x2,2GHz або краще;
- пам'ять: оперативна пам'ять має бути достатньою для запуску та обробки запитів, залежить від обсягу даних і навантаження на сервер. Рекомендується не менше 16 ГБ оперативної пам'яті;
- сховище даних: у разі можливості, встановлення бази даних на SSD може значно збільшити продуктивність роботи програми. Рекомендований обсяг накопичувача становить 500ГБ;
- середовище виконання: необхідне встановлене середовище виконання .NET 5/6/7 або .NET Core на сервері для запуску серверної частини додатка.

При використанні хмарного хостингу або платформи як сервісу (PaaS - Platform as a Service), відповідальність за апаратні ресурси лягає на провайдера, і вимоги можуть відрізнятися.

Для якісної роботи клієнтської частини програми рекомендується персональний комп'ютер до складу якого входять наступні компоненти та технічні засоби:

- процесор Intel Core i3-8100 (3.6 ГГц)(Intel UHD Graphics 630) з тактовою частотою, ГГц – 3,6;
- материнська плата Asus Prime H310M-R R2.0 (s1151 v2, Intel H310, PCI-Ex16);
- оперативна пам'ять DDR3 з об'ємом 4ГБ;
- блок живлення 450 Вт;
- об'єм логічної пам'яті SSD – 120 ГБ;
- інструменти вводу інформації – клавіатура та миша;

– засоби виводу інформації – монітор з роздільною здатністю 1920x1080.

Для запуску програми достатньо мати встановлену на комп'ютері будь-яку версію ОС та один із встановлених браузерів (Google Chrome, Mozilla Firefox, Opera, Microsoft Edge).

A.5 Вимоги до програмної документації

Для розробленого програмного забезпечення має бути оформлена наступна документація: технічне завдання, керівництво програміста, керівництво оператора, опис програми, тексти основних функціональних алгоритмів програми.

A.6 Стадії та етапи розробки

A.6.1 Стадії розробки

Розробка складається з таких стадій:

- аналіз предметної області;
- проєктування;
- розробка;
- тестування.

A.6.2 Етапи розробки

Аналіз предметної області – збір інформації, визначення функціоналу.

Проєктування – розробка архітектури системи, визначення компонентів та БД.

Розробка – написання коду: розробка функціональності програмного забезпечення згідно з визначеними вимогами та проєктуванням.

Тестування – проведення необхідних тестів для перевірки функціональності, надійності та безпеки системи, перевірка системи на відповідність вимогам.

A.6.3 Зміст робіт по етапах

Аналіз предметної області:

- збір вимог: визначення основних потреб і функціональних вимог до програмного забезпечення;
- аналіз аналогів ПЗ: вивчення аналогічних рішень на ринку, виявлення їхніх сильних і слабких сторін.

Проектування:

- архітектурне проектування: визначення загальної архітектури програмного рішення, вибір технологій та інструментів розробки;
- проектування бази даних: створення структури бази даних, визначення таблиць, їхніх зв'язків і атрибутів.

Розробка:

- розробка БД та запитів до неї;
- розробка GUI;
- розробка модулів, що пов'язують БД та GUI;
- обробка збійних ситуацій.

Тестування:

- модульне тестування : перевірка окремих модулів програми на відповідність специфікації та очікуваній поведінці;
- інтеграційне тестування: тестування взаємодії між різними компонентами програмного забезпечення для забезпечення їхньої правильної роботи в сукупності;
- системне тестування: перевірка програмного забезпечення в цілому на відповідність вимогам і його готовність до впровадження;

- тестування верстки за допомогою вбудованих у браузер інструментів розробника та шляхом практичного використання сайту;
- виправлення виявлених помилок.

ДОДАТОК Б
Опис програми

Б.1 Загальні відомості

Назва вебсайту – High-Quality Wooden Furniture. Назва проєкту при розробці - WebsiteOfTheManufacturer.

Розроблений вебсайт призначений для використання у сфері обліку та продажу товарів, де має допомогти мікро та малим підприємствам з обмеженими ресурсами спростити процес ведення обліку товарів та реалізовувати власну продукцію при підтримці інших підприємств за допомогою простих і сучасних рішень, починаючи від використання БД і закінчуючи впровадженням автоматизації для керування процесами оформлення і виконання замовлень, додавання та зміни товарів тощо.

Ціль вебзастосунку:

- зменшити час, необхідний для ведення обліку товарів;
- спростити ведення обліку товарів на складі;
- спростити керування списками користувачів, замовлень, каталогом товарів;
- забезпечити збереження усіх необхідних для роботи даних;
- надати малим підприємствам доступне і водночас функціональне рішення для організації електронної комерції;
- спростити та вдосконалити якість аналізу роботи підприємства.

Б.2 Функціональне призначення

Програмне забезпечення має надавати можливість авторизації на сайті та розподіляти користувачів на три ролі: адміністратор, партнери та гості.

Гості: користувачі, що не мають власного акаунта на сайті. Мають можливість переглядати каталог доступної продукції та детальні характеристики обраного товару.

Партнери: користувачі-підприємства, які мають зареєстрований адміністратором акаунт на сайті. Мають можливість переглядати каталог

доступної продукції та детальні характеристики обраного товару, а також оформлювати замовлення на товар в декілька кліків. Список замовлень та детальну інформацію про них можна буде передивитись на відповідній сторінці сайту. За бажанням – партнер зможе скасувати замовлення.

Адміністратор — користувач від підприємства-власника сайту, що має права на повне керування сайтом, а саме:

- реєстрація акаунтів для підприємств-партнерів;
- можливість переглядати каталог доступної продукції та детальні характеристики, а також редагувати інформацію про обраний товар;
- можливість реєструвати новий товар у каталозі;
- можливість переглядати інформацію про наповненість складу, та додавати нові товари на склад. Максимальна кількість товару для додавання обмежується вільним місцем на складі та об'ємом однієї одиниці обраного товару;
- переглядати історію керування товарами, яка формується при виконанні таких дій, як: реєстрація, редагування і додавання товару на склад;
- передивлятися усі дійсні акаунти, детальну інформацію про них та виконувати з ними наступні дії: редагувати інформацію акаунту, змінювати пароль від акаунту та деактивувати акаунт;
- можливість продивитись список всіх замовлень та детальну інформацію про них. Актуальні замовлення можна позначити як виконані.

При взаємодії адміністратора з акаунтом партнера слід забезпечити надсилання email-повідомлень відповідному підприємству про виконання конкретних дій адміністратором.

Б.3 Опис логічної структури вебсайту

На рисунку Б.1 наведено структурну схему програми, на якій видно розподіл функціоналу по модулях та по ролях користувачів. Жовтими стрілками наведено обмін даними між користувачем та БД, червоними – завантаження даних до БД, синіми – завантаження даних з БД.

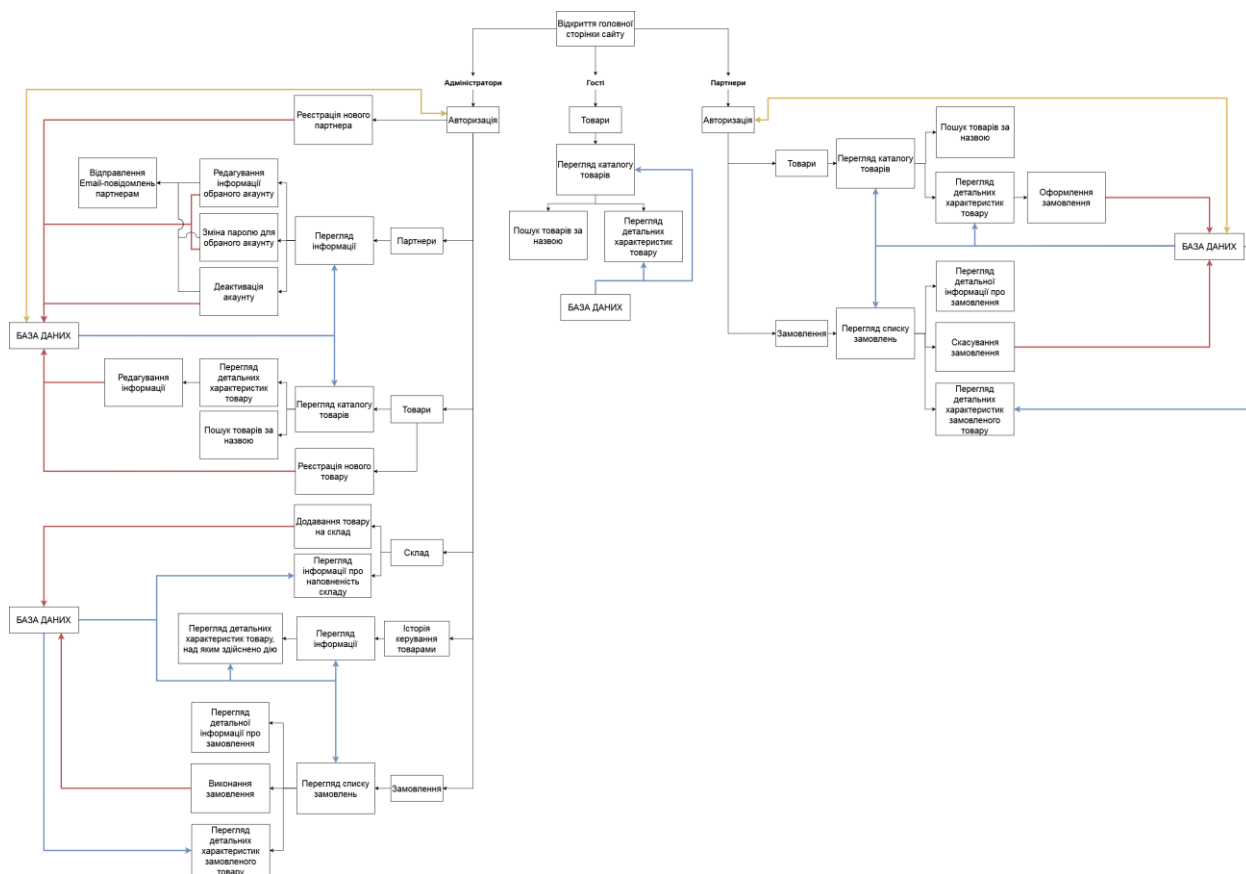


Рисунок Б.1 – Структурна схема програми

Б.3.1 Опис роботи вебсайту

Запуск вебсайту відбувається за посиланням [18]. При відкритті вебсайту користувач потрапляє на головну сторінку. Тут можна ознайомитись із призначенням і тематикою сайту, побачити контакти для зв'язку з адміністрацією (у футері). В шапці знаходиться кнопка для переходу до меню авторизації, а в лівій частині сторінки розташована навігаційна панель. З її

допомогою користувачі можуть використовувати функціонал сайту в залежності від своїх прав. При переході до авторизації, можна зайти на свій акаунт, якщо користувач є адміністратором сайту або представником підприємства-партнера, для якого заздалегідь було зареєстровано акаунт адміністратором. Звичайні відвідувачі сайту не мають можливості увійти до акаунту.

Неавторизований користувач має можливість перейти лише до каталогу товарів. Такий каталог є спільним для усіх типів користувачів. При переході на сторінку товарів можна побачити рядок для пошуку товарів за назвою та кнопки для здійснення пошуку та показу усього списку. Нижче пошукового рядку розташовані картки існуючих товарів. При натисканні на кнопку «Деталі» користувач переходить до перегляду детальних характеристик товару. Гості сайту мають можливість тільки переглянути подану інформацію.

Користувачі від підприємств-партнерів мають розширені права при роботі на сайті. При перегляді детальних характеристик товару вони можуть побачити додаткову інформацію про кількість товару на складі та мають можливість оформити тут замовлення. Для цього варто обрати бажану кількість товару та натиснути кнопку для замовлення. Після цього замовлення буде автоматично зформовано і додано до списку замовлень активного користувача. Додатково, при оформленні клієнт може побачити інформацію про вартість замовлення, спосіб оплати та спосіб доставки.

Оформлені замовлення користувач може продивитись за допомогою кнопки «Замовлення» у навігаційній панелі. При переході на дану сторінку відвідувач може побачити список своїх замовлень у вигляді таблиці та їх статус. Білим позначаються актуальні замовлення, сірим – скасовані, зеленим – виконані. При натисканні на рядок у списку в правій частині сторінки відображається детальна інформація про обране замовлення. Ці дані доступні тільки для читання. Якщо клієнт має бажання скасувати актуальне замовлення, він може зробити це за допомогою кнопки «Скасувати», що знаходиться у

третьому стовпці таблиці. Виконання даної операції потребує підтвердження дії у спливаючому повідомленні. Якщо користувач дав згоду – замовлення змінює свій статус та колір на відповідний. На цьому можливості клієнтів-партнерів закінчуються.

Користувачі-адміністратори мають найбільші права при взаємодії з вебсайтом. Через навігаційну панель вони можуть перейти до реєстрації нового користувача. Тут варто ввести унікальні дані підприємства-партнера та натиснути кнопку «Реєстрація».

Наявні у базі даних акаунти можна продивитись за допомогою сторінки «Список партнерів». Тут у вигляді таблиці відображаються усі зареєстровані користувачі. Якщо є необхідність у видаленні акаунта, адміністратор має деактивувати його за допомогою кнопки «Деактивувати», що знаходиться у третьому стовпці таблиці із списком користувачів. Виконання даної операції потребує підтвердження дії у спливаючому повідомленні. Якщо користувач дав згоду – акаунт змінює свій статус і не відображається у списку. При натисканні на рядок у списку в правій частині сторінки відображається детальна інформація про підприємства, до якого відноситься обраний акаунт. Ці дані можна відредагувати, проте вони мають залишатись унікальними. За необхідністю, адміністратор може змінити пароль від обраного акаунта за допомогою відповідної форми, що розташована в нижній правій частині сторінки.

При перегляді детальних характеристик обраного з каталогу товару, адміністратор має доступ до усієї інформації, що включає в себе об'єм товару, кількість на складі та кількість товару, що заброньовано для замовлень. Для редагування характеристик товару адміністратор може перейти на відповідну сторінку за допомогою кнопки «Редагувати товар», що знаходиться у правій нижній частині сторінки. Процес редагування поділяється на дві частини: можна змінити або зображення товару (форма для цього знаходиться у верхній частині сторінки, нове зображення має бути унікальним) або інші характеристики товару (форма для цього знаходиться у нижній частині

сторінки, назва товару має бути унікальною). Виключенням є категорія, об'єм та розміри товару – ці характеристики змінювати не можна. При успішному редагуванні або виникненні помилок адміністратору відображаються відповідні повідомлення у верхній частині сторінки. При внесенні змін на даній сторінці зберігається запис про редагування товару.

За допомогою кнопки у навігаційному меню «Реєстрація товару» відбувається додавання до каталогу нового товару. При додаванні адміністратору варто заповнити усі наявні поля, обрати унікальне зображення та назву товару, обрати категорію та заповнити унікальні атрибути, пов'язані з нею. При успішній реєстрації товару або виникненні помилок адміністратору відображаються відповідні повідомлення у верхній частині сторінки. При виконанні операцій на даній сторінці зберігається запис про реєстрацію товару.

На сторінці «Додавання виготовленої продукції» адміністратор має змогу додавати товар на склад. Сторінка за своєю структурою повторює сторінку з каталогом товарів, тут наявний рядок пошуку та картки з товарами, проте в нижній частині ще додається блок для додавання товарів. Дана операція відбувається таким чином: адміністратор обирає картку товару, який хоче додати і в нижній частині сторінки обирає кількість товару для додавання на склад. Тут також відображається об'єм товарів, що додається та наповненість складу. Максимальна кількість для додавання продукції обмежується вільним місцем на складі та об'ємом однієї одиниці товару. При успішному наповненні складу або виникненні помилок адміністратору відображаються відповідні повідомлення у верхній частині сторінки. При виконанні операцій на даній сторінці зберігається запис про додавання товару.

Сторінка «Історія керування товарами» дозволяє адміністратору продивитись усі операції, які були здійснені над товарами. Сюди входить реєстрація, редагування та додавання товару на склад. Дана інформація доступна тільки для читання. При редагуванні зберігається запис лише про сам

факт редагування, без зазначення детальних змін. При додаванні товару зазначається кількість товару, яку було додано до складу.

При відкритті адміністратором сторінки «Замовлення» він може побачити список замовлень від усіх користувачів та їх статус. Білим позначаються актуальні замовлення, сірим – скасовані, зеленим – виконані. При натисканні на рядок у списку в правій частині сторінки відображається детальна інформація про обране замовлення. Ці дані доступні тільки для читання. Якщо адміністратору необхідно позначити виконаним актуальне замовлення, він може зробити це за допомогою кнопки «Виконати», що знаходиться у третьому стовпці таблиці. На цьому можливості адміністраторів закінчуються.

Вихід користувача з акаунту здійснюється тільки при натисканні відповідної кнопки у шапці сайту. При звичайному закритті вкладки, дані про активну сесію залишаються.

Б.4 Використані технічні засоби

Вебсайт було розроблено на мові програмування C# (для back-end частини) та JavaScript (для front-end частини) з використанням фреймворку ASP.NET Core MVC. У якості СКБД обрано MS SQL Server, для доступу до бази даних використано ORM-фреймворк Entity Framework Core. Для розробки використовувалося IDE Microsoft Visual Studio 2022.

Нижче наведено вимоги до технічних та програмних засобів.

У разі використання власного сервера для роботи вебсайту, до серверної частини висуваються наступні вимоги:

- операційна система: Windows Server 2022;
- процесор: рекомендується процесор Intel Xeon E5-4620@8x2,2GHz або краще;

- пам'ять: оперативна пам'ять має бути достатньою для запуску та обробки запитів, залежить від обсягу даних і навантаження на сервер. Рекомендується не менше 16 ГБ оперативної пам'яті;
- сховище даних: у разі можливості, встановлення бази даних на SSD може значно збільшити продуктивність роботи програми. Рекомендований обсяг накопичувача становить 500ГБ;
- середовище виконання: необхідне встановлене середовище виконання .NET 5/6/7 або .NET Core на сервері для запуску серверної частини додатка.

При використанні хмарного хостингу або платформи як сервісу (PaaS - Platform as a Service), відповідальність за апаратні ресурси лягає на провайдера, і вимоги можуть відрізнятися.

Для якісної роботи клієнтської частини програми рекомендується персональний комп'ютер до складу якого входять наступні компоненти та технічні засоби:

- процесор Intel Core i3-8100 (3.6 ГГц)(Intel UHD Graphics 630) з тактовою частотою, ГГц – 3,6;
- материнська плата Asus Prime H310M-R R2.0 (s1151 v2, Intel H310, PCI-Ex16);
- оперативна пам'ять DDR3 з об'ємом 4ГБ;
- блок живлення 450 Вт;
- об'єм логічної пам'яті SSD – 120 ГБ;
- інструменти вводу інформації – клавіатура та миша;
- засоби виводу інформації – монітор з роздільною здатністю 1920x1080.

Для запуску програми достатньо мати встановлену на комп'ютері будь-яку версію ОС та один із встановлених браузерів (Google Chrome, Mozilla Firefox, Opera, Microsoft Edge).

Б.5 Виклик і завантаження

Запуск вебсайту відбувається за посиланням [18].

Б.6 Початкові дані

Початковими даними при розгортанні вебсайту будуть наявні в БД зареєстрований адміністратор (Логін: HQWFAdministrator, пароль: PasswordForSite138), визначені категорії (столи, стільці, тумби), визначені унікальні атрибути для характеристик:

- столи (призначення стола, форма стола, механізм трансформації);
- стільці (призначення стільця, висота стільця, каркас);
- тумби (призначення тумби, кількість ящиків, тип відкриття ящиків).

ДОДАТОК В
Текст програми

B.1 Файл EditingItemController.cs

```

namespace WebsiteOfTheManufacturer.Controllers
{
    [Authorize(Roles = "Admin")]
    public class EditingItemController : Controller
    {
        private readonly IMemoryCache _memoryCache;

        private readonly IWebHostEnvironment _appEnvironment;

        private readonly DataBaseContext _db;

        public EditingItemController(IMemoryCache memoryCache,
        IWebHostEnvironment appEnvironment, DataBaseContext db)
        {
            _memoryCache = memoryCache;
            _appEnvironment = appEnvironment;
            _db = db;
        }

        public async Task<IActionResult> Index(int ID, bool editItem = false, bool
        editImage = false, bool errorItem = false, bool errorImage = false)
        {
            try
            {
                if (!_memoryCache.TryGetValue("Items", out List<ItemModel> items))
                {
                    items = await _db.GetItems();

                    _memoryCache.Set("Items", items, TimeSpan.FromHours(1));
                }

                var item = items.First(p => p.ItemID == ID);

                if (!_memoryCache.TryGetValue("AttributeValues", out
                List<AttributeValueModel> attributes))
                {
                    attributes = await _db.GetAttributeValues();
                }
            }
        }
    }
}

```

```

        _memoryCache.Set("AttributeValues",
        TimeSpan.FromHours(1));
    }

    EditableItem model = new EditableItem()
    {
        ItemID = item.ItemID,
        Name = item.Name,
        CategoryName = item.CategoryName,
        WoodType = item.WoodType,
        Color = item.Color,
        Size = item.Size,
        Weight = item.Weight,
        Volume = item.Volume,
        Description = item.Description,
        SalePrice = item.SalePrice,
        PhotoPath = item.Photo
    };

    model.Attributes = attributes.Where(a => a.ItemID == ID).Select(a =>
new ItemAttribute
    {
        AttributeID = a.AttributeID,
        Name = a.AttributeName,
        Value = a.Value
    }).ToList();

    if (editItem) ViewData["SuccessMessage"] = "Дані товару змінено!";
    else if (editImage) ViewData["SuccessMessage"] = "Фото товару
змінено!";

    if (errorImage) ViewData["Error"] = "Фото товару має бути
унікальним!";
    else if (errorItem) ViewData["Error"] = "Назва товару має бути
унікальною!";

    return View("~/Views/Website/EditingItem.cshtml", model);
}
catch (Exception)
{
    return RedirectToAction("Index", "CriticalError");
}
}

```

```

[HttpPost]
public async Task<IActionResult> EditImage(EditableItem model)
{
    try
    {
        if (model.Photo == null || model.Photo.FileName.Length > 35)
        {
            ViewData["Error"] = "Оберіть фото товару з назвою не більше 35
символів!";

            return RedirectToAction("Index", new { ID = model.ItemID });
        }

        if (!_memoryCache.TryGetValue("Items", out List<ItemModel>
fullItems))
        {
            fullItems = await _db.GetItems();

            _memoryCache.Set("Items", fullItems, TimeSpan.FromHours(1));
        }

        if (!_memoryCache.TryGetValue("ItemsValidation", out
List<ItemValidationModel> itemsValidation))
        {
            itemsValidation = fullItems.Select(i => new ItemValidationModel
            {
                ItemID = i.ItemID,
                ItemName = i.Name,
                ImagePath = i.Photo
            }).ToList();

            _memoryCache.Set("ItemsValidation", itemsValidation,
TimeSpan.FromMinutes(10));
        }

        var fileName = model.Photo.FileName;
        var filePath = Path.Combine(@"\Images", fileName);

        filePath = filePath.Replace(@"\", "/");

        bool result = itemsValidation.Any(i => i.ItemID != model.ItemID &&
i.ImagePath == filePath);

        if (!result)

```

```

        {
            var filePathFull = Path.Combine(_appEnvironment.WebRootPath,
"Images", fileName);

            if (System.IO.File.Exists(filePathFull))
                System.IO.File.Delete(filePathFull);

            using (var stream = new FileStream(filePathFull, FileMode.Create))
            {
                await model.Photo.CopyToAsync(stream);
            }

            model.PhotoPath = filePath;

            var tmpItem = fullItems.FirstOrDefault(i => i.ItemID ==
model.ItemID);
            if (tmpItem != null) tmpItem.Photo = model.PhotoPath;

            var tmpItem2 = itemsValidation.FirstOrDefault(i => i.ItemID ==
model.ItemID);
            if (tmpItem2 != null) tmpItem2.ImagePath = model.PhotoPath;

            await _db.EditItemImage(model.ItemID, model.PhotoPath);

            return RedirectToAction("Index", new { ID = model.ItemID, editImage
= true });
        }

        else
        {
            // ViewData["Error"] = "Фото товару має бути унікальним!";

            return RedirectToAction("Index", new { ID = model.ItemID,
errorImage = true });
        }
    }
    catch (Exception)
    {
        return RedirectToAction("Index", "CriticalError");
    }
}

```

[HttpPost]

```

public async Task<IActionResult> EditItem(EditableItem model)
{
    try
    {
        if (!ModelState.IsValid) return RedirectToAction("Index", new { ID =
model.ItemID });

        if (!_memoryCache.TryGetValue("Items", out List<ItemModel>
fullItems))
        {
            fullItems = await _db.GetItems();

            _memoryCache.Set("Items", fullItems, TimeSpan.FromHours(1));
        }

        if (!_memoryCache.TryGetValue("ItemsValidation", out
List<ItemValidationModel> itemsValidation))
        {
            itemsValidation = fullItems.Select(i => new ItemValidationModel
            {
                ItemID = i.ItemID,
                ItemName = i.Name,
                ImagePath = i.Photo
            }).ToList();

            _memoryCache.Set("ItemsValidation", itemsValidation,
TimeSpan.FromMinutes(10));
        }

        bool result = itemsValidation.Any(i => i.ItemID != model.ItemID &&
i.ItemName == model.Name);

        if (!result)
        {
            var tmpItem = fullItems.FirstOrDefault(i => i.ItemID ==
model.ItemID);
            if (tmpItem != null)
            {
                tmpItem.Name = model.Name;
                tmpItem.WoodType = model.WoodType;
                tmpItem.Color = model.Color;
                tmpItem.Size = model.Size;
                tmpItem.Weight = model.Weight;
                tmpItem.Volume = model.Volume;
            }
        }
    }
}

```

```

        tmpItem.SalePrice = model.SalePrice;
        tmpItem.Description = model.Description;
    }

    var tmpItem2 = itemsValidation.FirstOrDefault(i => i.ItemID ==
model.ItemID);
    if (tmpItem2 != null) tmpItem2.ItemName = model.Name;

    if (!_memoryCache.TryGetValue("AttributeValues", out
List<AttributeValueModel> attributes))
    {
        attributes = await _db.GetAttributeValues();

        _memoryCache.Set("AttributeValues", attributes,
        TimeSpan.FromHours(1));
    }

    var tmpAttributes = attributes.Where(a => a.ItemID ==
model.ItemID).ToList();

    foreach (var item in model.Attributes)
    {
        var cachedAttribute = tmpAttributes.FirstOrDefault(a =>
a.AttributeID == item.AttributeID);

        if (cachedAttribute != null) cachedAttribute.Value = item.Value;
    }

    await _db.EditItem(model);

    return RedirectToAction("Index", new { ID = model.ItemID, editItem
= true });
    }
    else
    {
        // ViewData["Error"] = "Назва товару має бути унікальною!";

        return RedirectToAction("Index", new { ID = model.ItemID, errorItem
= true });
    }
    }
    catch (Exception)
    {
        return RedirectToAction("Index", "CriticalError");
    }

```

```

    }
  }
}

```

В.2 Файл AddingItemController.cs

```

namespace WebsiteOfTheManufacturer.Controllers
{
    [Authorize(Roles = "Admin")]
    public class AddingItemsController : Controller
    {
        private readonly IMemoryCache _memoryCache;

        private readonly DataBaseContext _db;

        public AddingItemsController(IMemoryCache memoryCache,
        DataBaseContext db)
        {
            _memoryCache = memoryCache;
            _db = db;
        }

        public async Task<IActionResult> Index(bool success = false, bool fail = false,
        string searchQuery = "")
        {
            try
            {
                var model = new List<CatalogModel>();

                model = await WarehouseInfo(searchQuery);

                if (success) TempData["SuccessMessage"] = "Виготовлений товар  
додано до складу!";
                if (fail) TempData["ErrorMessage"] = "На складі недостатньо місця для  
нового товару!";
                return View("~/Views/Website/AddingItems.cshtml", model);
            }
            catch (Exception)
            {
                return RedirectToAction("Index", "CriticalError");
            }
        }
    }
}

```

```

    }

    [HttpPost]
    public async Task<IActionResult> AddItemInWarehouse()
    {
        var item = new ItemModel();

        var data = Request.Form;
        int ID = Convert.ToInt32(data["ID"]);
        int Quantity = Convert.ToInt32(data["Quantity"]);
        decimal Volume = Convert.ToDecimal(data["Volume"],
CultureInfo.InvariantCulture);
        string SearchQuery_ = data["SearchQuery_"];

        var warehouse = new Warehouse();

        try
        {
            if (!_memoryCache.TryGetValue("Warehouse", out warehouse))
            {
                warehouse = await _db.GetWarehouseInfo();

                _memoryCache.Set("Warehouse", warehouse,
TimeSpan.FromHours(1));
            }

            if (!_memoryCache.TryGetValue("Items", out List<ItemModel> items))
            {
                items = await _db.GetItems();

                _memoryCache.Set("Items", items, TimeSpan.FromHours(1));
            }

            if (warehouse.MaxCapacity >= warehouse.OccupiedSpace + Volume)
            {
                item = items.FirstOrDefault(i => i.ItemID == ID);

                if (item != null) item.QuantityInStock += Quantity;

                warehouse.OccupiedSpace += Volume;

                await _db.AddItemInWarehouse(ID, Volume, Quantity);
            }
        }
    }

```

```

        return RedirectToAction("Index", new { success = true, searchQuery =
SearchQuery_ });
    }

    else return RedirectToAction("Index", new { fail = true });
}
catch (DbUpdateConcurrencyException)
{
    if (item != null) item.QuantityInStock -= Quantity;

    if (warehouse != null) warehouse.OccupiedSpace -= Volume;

    return RedirectToAction("Index", "CriticalError", new {
DbUpdateConcurrencyException = true });
}
catch (Exception)
{
    return RedirectToAction("Index", "CriticalError");
}
}

```

[HttpGet]

```

public async Task<IActionResult> Search(string SearchQuery)
{

```

```

    try
    {
        var model = new List<CatalogModel>();

        model = await WarehouseInfo(SearchQuery);

        return View("~/Views/Website/AddingItems.cshtml", model);
    }
    catch (Exception)
    {
        return RedirectToAction("Index", "CriticalError");
    }
}

```

```

public async Task<List<CatalogModel>> WarehouseInfo(string SearchQuery)
{
    try
    {
        if (!_memoryCache.TryGetValue("Items", out List<ItemModel> items))
        {
            items = await _db.GetItems();

```

```

        _memoryCache.Set("Items", items, TimeSpan.FromHours(1));
    }

    var model = new List<CatalogModel>();

    ViewData["SearchQuery"] = SearchQuery;

    if (String.IsNullOrEmpty(SearchQuery))
        model = items.Select(i => new CatalogModel { ItemID = i.ItemID,
Name = i.Name, Price = i.SalePrice, Photo = i.Photo, Volume = i.Volume
}).ToList();
    else
    {
        model = items.Select(i => new CatalogModel { ItemID = i.ItemID,
Name = i.Name, Price = i.SalePrice, Photo = i.Photo, Volume = i.Volume }).
        Where(i
i.Name.ToLower().Contains(SearchQuery.ToLower())).ToList();
    }
    if (!_memoryCache.TryGetValue("Warehouse", out Warehouse
warehouse))
    {
        warehouse = await _db.GetWarehouseInfo();

        _memoryCache.Set("Warehouse", warehouse,
TimeSpan.FromHours(1));
    }

    ViewData["OccupiedSpace"] = warehouse.OccupiedSpace;
    ViewData["MaxCapacity"] = warehouse.MaxCapacity;

    return model;
}
catch (Exception)
{
    throw;
}
}
}
}

```

ДОДАТОК Г
Слайди презентації



Рисунок Г.1 – Слайд №1

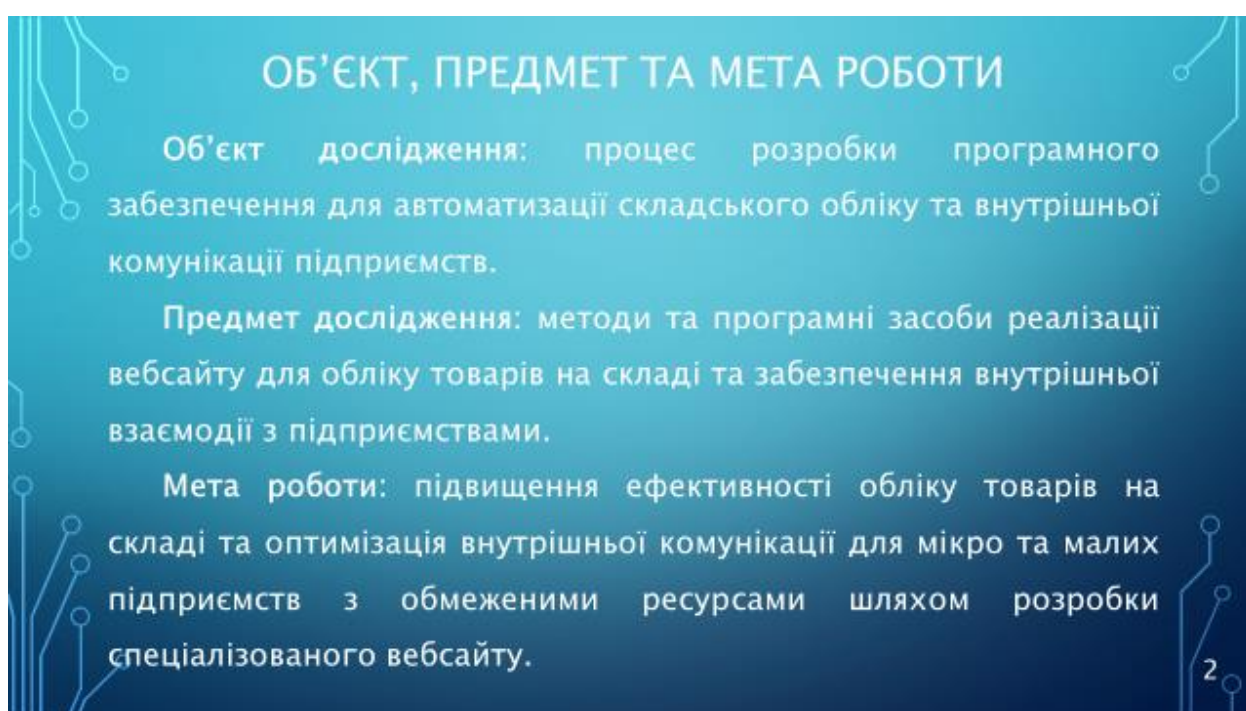


Рисунок Г.2 – Слайд №2

АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

Актуальність створення універсального вебсайту полягає в тому, що більшість мікропідприємств не мають можливості використовувати складні та коштовні ІТ-рішення, а тому потребують простих, доступних, але водночас функціональних інструментів для організації електронної комерції.

Тематикою вебсайту обрано меблі з деревини, тому що деревообробка є традиційною галуззю в Україні, яка активно розвивається. А продукція з натуральної деревини є затребуваною на ринку завдяки екологічності, довговічності та естетичній привабливості.

3

Рисунок Г.3 – Слайд №3

ПОРІВНЯННЯ ІСНУЮЧИХ ПРОГРАМНИХ ЗАСОБІВ

	Сервіс USAP-Online	Сервіс RemOnline	Вебзастосунок, що розроблюється
Врахування об'єму товарів	+	—	+
Врахування наповненості складу	—	—	+
Інтуїтивно зрозумілий інтерфейс	— (дуже важкий для сприйняття)	+	+
Автоматичне формування замовлення	—	—	+
Автоматичне внесення змін при оформленні замовлення	—	—	+
			(За рахунок поєднання з Інтернет-магазином)
			(За рахунок поєднання з Інтернет-магазином)

4

Рисунок Г.4 – Слайд №4

7

Рисунок Г.8 – Слайд №8





Рисунок Г.9 – Слайд №9

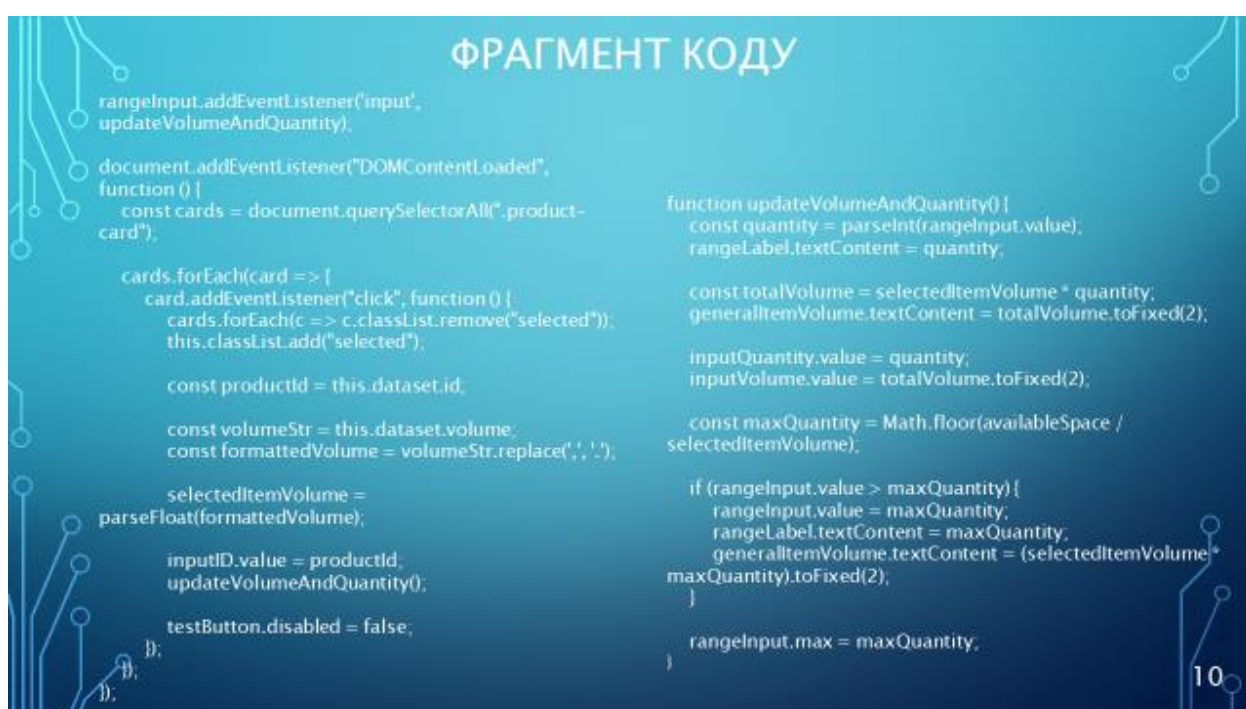


Рисунок Г.10 – Слайд №10

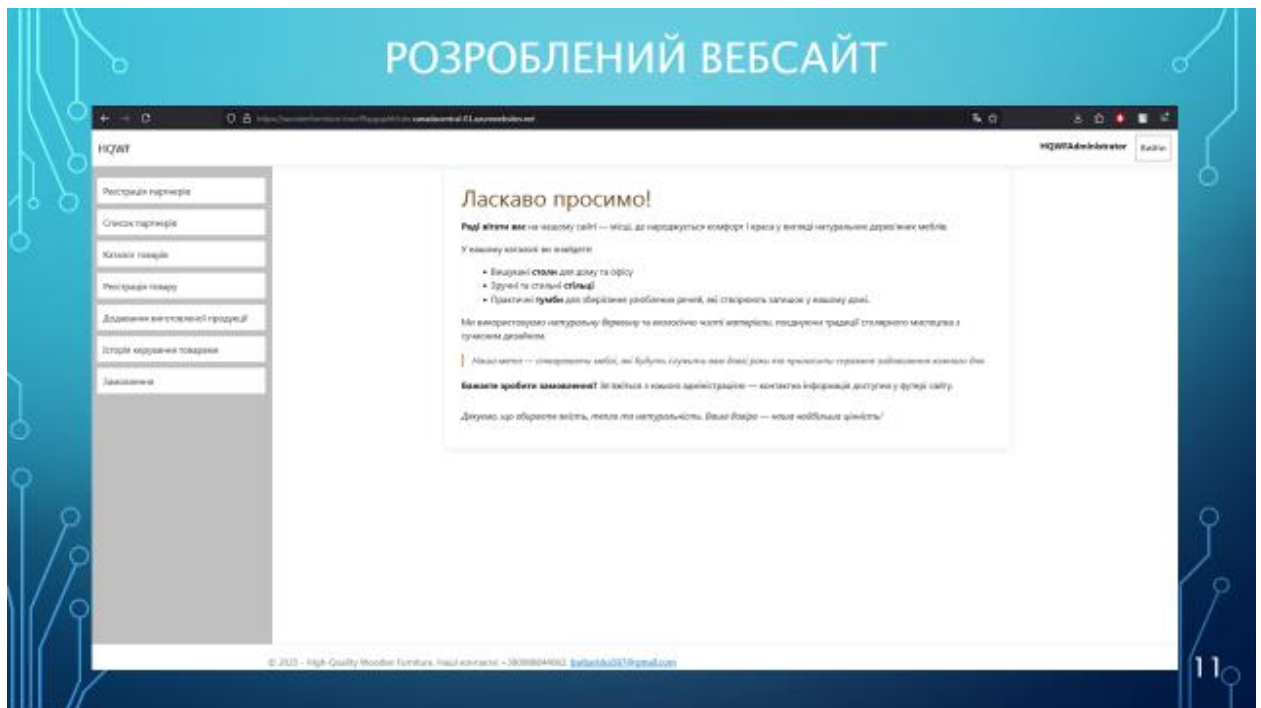


Рисунок Г.11 – Слайд №11

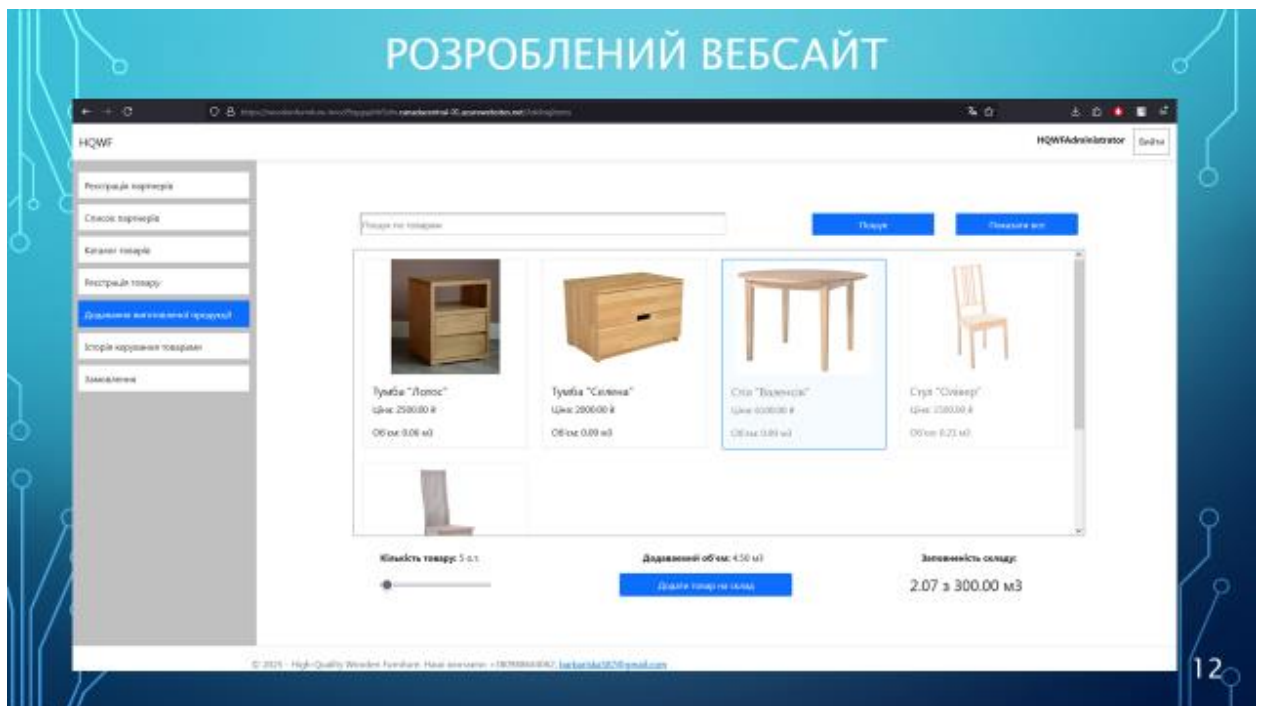


Рисунок Г.12 – Слайд №12

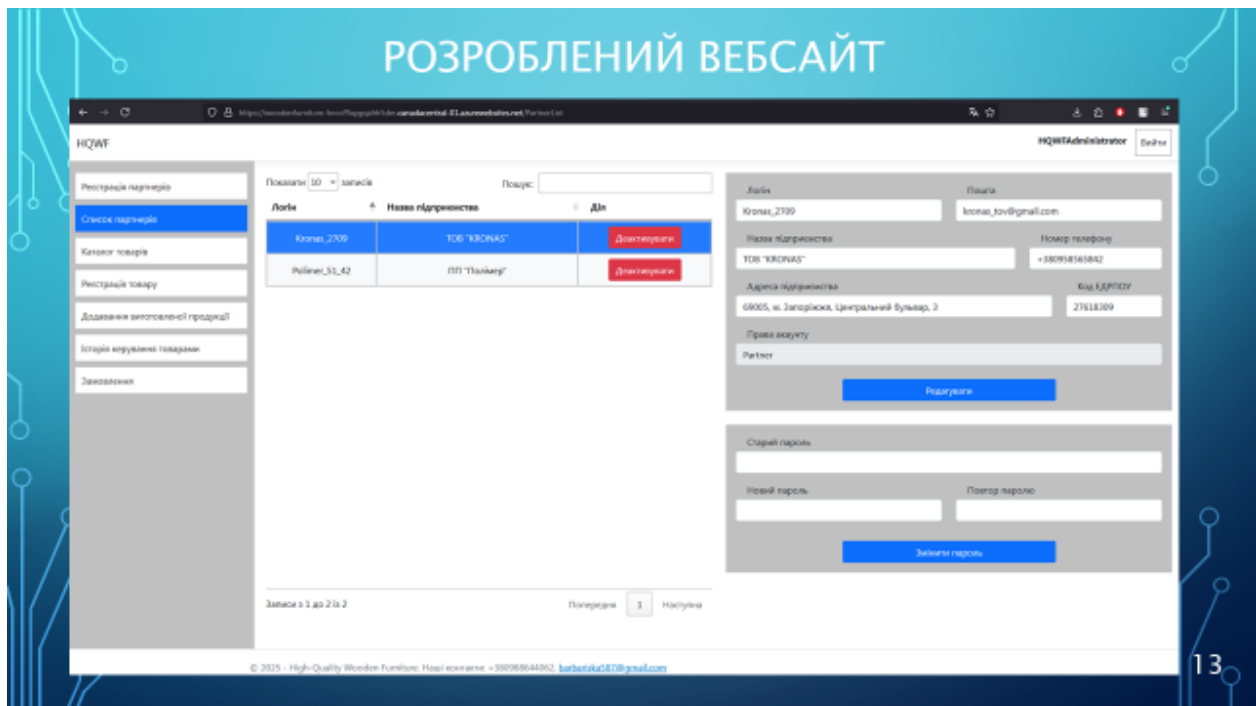
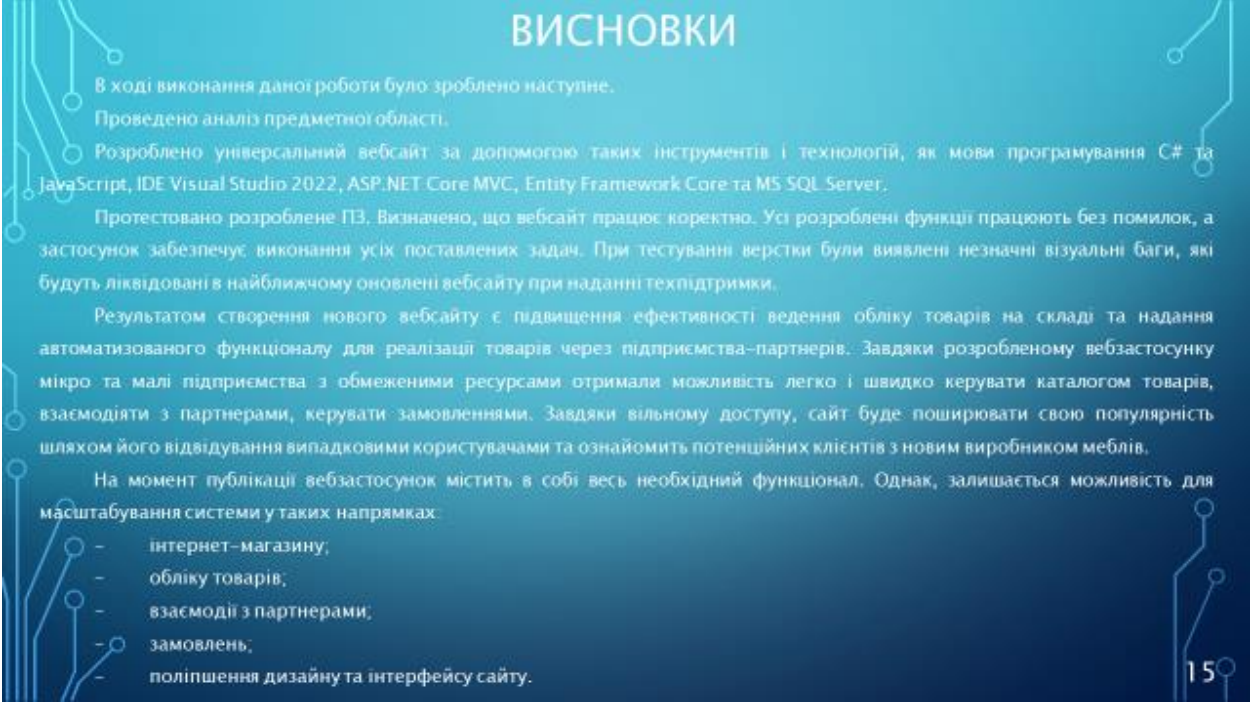


Рисунок Г.13 – Слайд №13

ТЕСТУВАННЯ РОЗРОБЛЕНОГО ВЕБСАЙТУ			
Тестування	Результат	Коментарі	Користувач, для якого проводилося тестування
Автоматичний розрахунок вартості замовлення при виборі кількості товару	ОК	Вартість розраховується коректно	Партнер
Коректне відображення замовлень згідно з їх статусом	ОК	Актуальні замовлення виділені білим кольором, скасовані – сірим, а виконані – зеленим	Партнер, Адміністратор
Відправка email-повідомлень при роботі з акаунтом	ОК	Коли адміністратор редагує акаунт, відповідний партнер отримує повідомлення на вказану пошту	Партнер, Адміністратор

Рисунок Г.14 – Слайд №14



ВИСНОВКИ

В ході виконання даної роботи було зроблено наступне.

Проведено аналіз предметної області.

Розроблено універсальний вебсайт за допомогою таких інструментів і технологій, як мови програмування C# та JavaScript, IDE Visual Studio 2022, ASP.NET Core MVC, Entity Framework Core та MS SQL Server.

Протестовано розроблене ПЗ. Визначено, що вебсайт працює коректно. Усі розроблені функції працюють без помилок, а застосунок забезпечує виконання усіх поставлених задач. При тестуванні верстки були виявлені незначні візуальні баги, які будуть ліквідовані в найближчому оновленні вебсайту при наданні техпідтримки.

Результатом створення нового вебсайту є підвищення ефективності ведення обліку товарів на складі та надання автоматизованого функціоналу для реалізації товарів через підприємства-партнерів. Завдяки розробленому вебзастосунку мікро та малі підприємства з обмеженими ресурсами отримали можливість легко і швидко керувати каталогом товарів, взаємодіяти з партнерами, керувати замовленнями. Завдяки вільному доступу, сайт буде поширювати свою популярність шляхом його відвідування випадковими користувачами та ознайомить потенційних клієнтів з новим виробником меблів.

На момент публікації вебзастосунок містить в собі весь необхідний функціонал. Однак, залишається можливість для масштабування системи у таких напрямках:

- інтернет-магазину;
- обліку товарів;
- взаємодії з партнерами;
- замовлень;
- поліпшення дизайну та інтерфейсу сайту.

15

Рисунок Г.15 – Слайд №15