Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет Автоматики і комп'ютерних технологій
(повна назва)
Кафедра Комп'ютерно-інтегрованих технологій, автоматизації та робототехніки
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

Перший (бакалаврський)
(рівень вищої освіти)

Розробка інформаційно-пошукової системи обліку продукції для автоматизованої багаторівневої внутрішньо-складської підсистеми зберігання
(тема)

Виконав:
студент 4 курсу, групи АКТСІ-20-2
Трохін В.В.
(прізвище, ініціали)
Спеціальності 151 Автоматизація та комп'ютерно інтегровані технології
(код і повна назва спеціальності)
Тип програми Освітньо-професійна
(освітньо-професійна або освітньо-наукова)
Освітня програма Системна інженерія
(повна назва освітньої програми)
Керівник доц. Чала О.О.
(посада, прізвище, ініціали)

Допускається до захисту
Зав. кафедри КІТАР _____ Невлюдов І.Ш.
(підпис) (прізвище, ініціали)

2024 р.

ХАРКІВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ РАДІОЕЛЕКТРОНІКИ

Факультет АКТ
Кафедра КІТАР
Рівень вищої освіти перший (бакалаврський)
Спеціальність 151 Автоматизація та комп'ютерно інтегровані технології
Тип програми Освітньо-професійна
Освітня програма Автоматизація та комп'ютерно-інтегровані технології

ЗАТВЕРЖДУЮ:
Зав. кафедри _____
(підпис)
«25» червня 2024 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

студентові Трохіну Владиславу Віталійовичу
(прізвище, ім'я, по батькові)
1. Тема роботи Розробка інформаційно-пошукової системи обліку продукції для автоматизованої багаторівневої внутрішньо-складської підсистеми зберігання
Затверджена наказом по університету від 03.06.2024 №545 Ст.
2. Термін подання студентом роботи до екзаменаційної комісії 25.06.2024
3. Вхідні дані до роботи
3.1 Ноутбук для розробки ІПС
3.2 Програмне середовище SQL

4. Перелік питань, що потрібно розглянути у роботі
4.1 Вступ
4.2 Огляд теми та її актуальності
4.3 Технології для моніторингу параметрів навколишнього середовища
4.4 Розробка апаратної та програмної частини
4.5 Питання пов'язані з охороною праці
4.6 Висновки та перелік джерел посилань

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій Демонстраційний матеріал, представлений у форматі презентації PowerPoint (*.ppt) – 13 с. формату А4.

6. Консультанти розділів роботи

Найменування розділу	Консультант (посада, прізвище, ім'я, по батькові)	Позначка консультанта про виконання розділу	
		підпис	дата

КАЛЕНДАРНИЙ ПЛАН

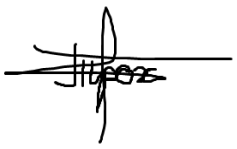
№	Назва етапів роботи	Термін виконання етапів роботи	Приміт-ка
1	Дослідження процесів обліку продукції на складі	29.01.2024	Виконано
2	Аналіз вимог і потреб користувачів	14.02.2024	Виконано
4	Дослідження наявних баз даних	26.02.2024	Виконано
5	Вибір середовища програмування	04.03.2024	Виконано
6	Розробка бази даних	10.04.2024	Виконано
7	Розробка додатку	19.04.2024	Виконано
8	Тестування роботи інформаційно-пошукової системи	15.05.2024	Виконано
9	Оформлення звіту з виконаної роботи	22.05.2024	Виконано

Дата видачі завдання 20.01.2024

Студент _____ Трохін В.В.
(підпис) (прізвище, ініціали)

Керівник роботи _____ доц. Чала О. О.
(підпис) (посада, прізвище, ініціал

Я, як студент ХНУРЕ, розумію і підтримую політику закладу із академічної доброчесності. Я не надавав і не одержував недозволену допомогу під час підготовки кваліфікаційної роботи. Використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело.

"03" червня 2024 р  Трохін В.В.

РЕФЕРАТ

Пояснювальна записка: 90 с., 14 табл., 47 рис., 2 дод., 30 джерел.

АВТОМАТИЗАЦІЯ, БАЗА ДАНИХ, ІНФОРМАЦІЙНО-ПОШУКОВА СИСТЕМА, ОБЛІК ПРОДУКЦІЇ, УПРАВЛІННЯ ПРОДУКЦІЄЮ.

Об’єкт розробки: процес управління складським обліком продукції, який включає приймання, зберігання, переміщення та відвантаження товарів.

Предмет розробки: методи та засоби автоматизації обліку продукції на складі за допомогою інформаційно-пошукової системи.

Мета кваліфікаційної роботи: розробка ефективної інформаційно-пошукової системи обліку продукції, яка дозволить покращити процеси управління складськими операціями та забезпечити зручний інтерфейс для користувачів різних ролей.

Методи дослідження: аналіз теоретичних основ обліку продукції на складі, порівняння аналогічних програмних рішень, моделювання процесів обліку продукції за допомогою діаграм послідовності та активності, проектування логічної та фізичної моделей бази даних, реалізація програмного забезпечення на платформі .NET Framework із використанням мови програмування C# та системи управління базами даних MySQL.

У кваліфікаційній роботі розглянуто актуальні питання автоматизації обліку продукції на складі. Запропоновано рішення, що дозволяють оптимізувати процеси приймання, зберігання, переміщення та відвантаження продукції. Спроектовано та реалізовано інформаційно-пошукову систему обліку продукції, яка забезпечує функціональні можливості для користувачів різних ролей. Розроблена система може використовуватись на підприємствах для автоматизації складських операцій, що сприяє підвищенню ефективності управління продукцією та покращенню обслуговування клієнтів.

ABSTRACT

Explanatory note: 90 pp., 14 tables, 47 figures, 2 appendices, 30 sources.

AUTOMATION, DATA BASE, INFORMATION SEARCH SYSTEM, PRODUCT ACCOUNTING, PRODUCT MANAGEMENT.

Object of development: the process of managing the warehouse accounting of products.

The subject of development: methods and means of automating the accounting of products in the warehouse with the help of an information and search system.

The goal of the qualification work: the development of an effective information and search system for product accounting, which will improve warehouse operations management processes and provide a convenient interface for users of various roles.

Research methods: analysis of the theoretical foundations of inventory accounting, comparison of similar software solutions, modeling of product accounting processes using sequence and activity diagrams, design of logical and physical database models, software implementation on the .NET Framework platform using the C# programming language and system MySQL database management.

In the qualification work, the actual issues of automation of accounting of products in the warehouse are considered. Solutions have been proposed that allow optimizing the processes of receiving, storing, moving and shipping products. An information and search system for product accounting was designed and implemented. The developed system can be used at enterprises to automate warehouse operations, which helps to increase the efficiency of product management and improve customer service.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ ТА СКОРОЧЕНЬ

БД – база даних;

СУБД – система управління базами даних;

.NET – платформа розробки програмного забезпечення від Microsoft;

API (Application Programming Interface) – інтерфейс програмування застосунків ;

C# – мова програмування, розроблена компанією Microsoft;

CRUD (Create, Read, Update, Delete) – операції створення, читання, оновлення та видалення даних;

ERP (Enterprise Resource Planning) – система планування ресурсів підприємства;

GUI (Graphical User Interface) – графічний інтерфейс користувача;

IDE (Integrated Development Environment) – інтегроване середовище розробки;

SQL (Structured Query Language) – мова структурованих запитів;

WMS (Warehouse Management System) – система управління складом.

ВСТУП

В умовах стрімкого розвитку інформаційних технологій та їх інтеграції у всі сфери життєдіяльності суспільства, автоматизація внутрішньо-складських процесів стає надзвичайно актуальною. Особливо це стосується управління обліком продукції, що є важливою складовою ефективного функціонування будь-якої організації, яка займається виробництвом, зберіганням та реалізацією товарів. Відсутність надійної системи обліку продукції може призвести до значних фінансових втрат, низької ефективності роботи складу та незадоволення клієнтів.

На сьогоднішній день існує значна кількість різних програмних рішень, які пропонують автоматизацію складських процесів. Проте, більшість з них мають низку обмежень, таких як складність у налаштуванні, відсутність інтеграції з іншими інформаційними системами підприємства, висока вартість впровадження та обслуговування [1]. Водночас, потреби підприємств у функціональних та гнучких системах управління складськими процесами постійно зростають.

Проблема ефективного управління складським обліком потребує розробки інформаційно-пошукової системи, яка забезпечить автоматизацію всіх рівнів складських процесів, від приймання та зберігання товарів до їх відвантаження. Така система повинна враховувати специфіку внутрішньо-складських процесів, забезпечувати швидкий доступ до необхідної інформації, бути зручною у використанні та легкою у налаштуванні [2].

Вихідні дані для розробки системи включають аналіз існуючих програмних рішень, вивчення сучасних підходів до автоматизації складських процесів, а також потреби та вимоги кінцевих користувачів. Необхідність проведення даного дослідження обумовлена потребою в створенні універсального та доступного рішення для управління складським обліком,

яке зможе задовольнити вимоги сучасного ринку та забезпечити ефективну роботу підприємств.

Створити ефективну, функціональну та зручну в користуванні інформаційно-пошукову систему обліку продукції для автоматизованої багаторівневої внутрішньо-складської підсистеми зберігання, що забезпечить підвищення ефективності управління складськими процесами та зниження витрат.

Основні задачі дослідження включають:

– вивчити теоретичні основи обліку продукції на складі та проаналізувати існуючі програмні рішення. Це включає розгляд понять обліку продукції, основних процесів обліку на складі та детальний аналіз аналогічних систем, щоб визначити їх переваги та недоліки;

– провести аналіз вимог і потреб користувачів системи обліку продукції. Збір та систематизація функціональних та технічних вимог до системи, а також визначення ключових очікувань кінцевих користувачів;

– здійснити огляд та вибір інструментів розробки системи. Це включає вибір платформи для розробки, середовища розробки та системи управління базами даних, що найкраще відповідатимуть технічним та функціональним вимогам системи;

– спроектувати та реалізувати систему обліку продукції та її базу даних. Проектування бази даних, поведінки системи, створення логічної та фізичної моделей бази даних, реалізація коду системи, а також проведення функціонального та модульного тестування для забезпечення надійності та ефективності системи.

– оформити пояснювальну роботу згідно з ДСТУ 3008:2015 [3], також методичними вказівками з підготовки й оформлення кваліфікаційної роботи здобувачами першого (бакалаврського) рівня вищої освіти спеціальності 151 Автоматизація та комп’ютерно-інтегровані технології[4-5].

1 ОСНОВИ ОБЛІКУ ПРОДУКЦІЇ НА СКЛАДІ ТА АНАЛІЗ СИСТЕМ

1.1 Поняття обліку продукції

Облік продукції є важливим складником діяльності будь-якого підприємства, що займається виробництвом, зберіганням та реалізацією товарів. Він включає систематичний процес реєстрації, класифікації, аналізу та узагальнення інформації про рух і залишки продукції на складі. Метою обліку продукції є забезпечення точності даних про наявність товарів, що дозволяє оптимізувати складські операції, покращити управління запасами та підвищити ефективність роботи підприємства.

Облік продукції має велике значення для ефективного управління складськими операціями та забезпечення безперебійного функціонування підприємства. Він дозволяє:

– отримувати точну та своєчасну інформацію про наявність товарів на складі;

– планувати закупівлі та виробництво на основі актуальних даних про запаси;

– знижувати витрати на утримання надлишкових запасів та мінімізувати ризики втрат через псування або крадіжки;

– підвищувати ефективність роботи складу шляхом оптимізації внутрішніх складських процесів;

– забезпечувати високий рівень обслуговування клієнтів шляхом своєчасного та точного виконання замовлень [6].

Від точності та своєчасності обліку залежить оптимізація запасів, зниження витрат та підвищення рівня задоволення клієнтів. Тому впровадження автоматизованих систем обліку продукції є актуальним

завданням для сучасних підприємств, яке сприяє підвищенню їх конкурентоспроможності та ефективності.

1.2 Основні процеси обліку продукції на складі

Облік продукції на складі охоплює комплекс взаємопов'язаних процесів, які забезпечують ефективне управління запасами, оптимізацію складських операцій та точність даних про наявність товарів. Основні процеси обліку продукції на складі включають приймання, зберігання, переміщення, інвентаризацію та відвантаження продукції. Кожен з цих процесів відіграє ключову роль у забезпеченні безперебійного функціонування складської системи.

На рис. 1.1 представлено діаграму основних аспектів обліку продукції на складі.

Рисунок 1.1 – Основні аспекти обліку продукції [7]

Основні аспекти обліку продукції включають:

– приймання продукції. Процес приймання продукції включає перевірку відповідності отриманих товарів супровідним документам,

кількісний та якісний контроль продукції, реєстрацію отримання в обліковій системі та розміщення товарів на складі. Важливим аспектом приймання є точність і своєчасність внесення даних до системи обліку для подальшого контролю та аналізу;

– зберігання продукції. Зберігання продукції передбачає розміщення товарів у відповідних зонах складу з урахуванням умов зберігання, таких як температура, вологість, вимоги до безпеки тощо. Ефективне управління простором складу та дотримання умов зберігання є ключовими для збереження якості продукції та запобігання втратам;

– переміщення продукції. Переміщення продукції включає внутрішньоскладські операції з переміщення товарів між різними зонами складу, а також підготовку до відвантаження. Цей процес вимагає точного обліку та контролю, щоб уникнути помилок та забезпечити своєчасне виконання замовлень клієнтів;

– відвантаження продукції. Відвантаження продукції є завершальним етапом складських операцій і включає підготовку товарів до відправлення, комплектування замовлень, перевірку відповідності замовлення та супровідних документів, а також реєстрацію відвантаження в обліковій системі. Важливим аспектом є забезпечення точності та своєчасності виконання замовлень, що впливає на рівень задоволення клієнтів [8].

1.3 Аналіз аналогічних програмних рішень

Аналіз аналогічних програмних рішень є важливим етапом у процесі розробки інформаційно-пошукової системи обліку продукції. Дослідження існуючих рішень на ринку дозволяє не лише оцінити сучасний стан технологій у даній сфері, але й виявити їхні переваги та недоліки. Такий аналіз надає можливість зрозуміти, які функціональні можливості є найбільш затребуваними користувачами, а також визначити області, в яких існують прогалини чи недоліки, що потребують удосконалення. Це допомагає

розробникам уникнути повторення помилок попередників та створити конкурентоспроможний продукт, який відповідатиме сучасним вимогам і потребам користувачів. Вивчення аналогічних програмних рішень дозволяє також оцінити їхню сумісність з іншими інформаційними системами, які використовуються на підприємствах, та можливість інтеграції з ними, що є важливим аспектом для забезпечення комплексного підходу до автоматизації складських процесів. Завдяки такому підходу можна досягти більш ефективного та раціонального використання ресурсів, підвищити продуктивність роботи складу та забезпечити високу якість обслуговування клієнтів.

Fishbowl Warehouse є комплексним програмним рішенням для управління складськими операціями та обліку продукції (рис. 1.2). Основним призначенням цього застосунку є забезпечення точного та ефективного обліку товарів на складі, оптимізація складських процесів та підвищення продуктивності роботи складу. Fishbowl Warehouse дозволяє автоматизувати процеси приймання, зберігання, переміщення та відвантаження продукції, а також забезпечує інтеграцію з іншими системами управління підприємством, такими як QuickBooks [9].

Архітектура Fishbowl Warehouse базується на модульному підході, що забезпечує гнучкість та масштабованість системи. Основні компоненти включають серверну частину, яка відповідає за зберігання даних та виконання бізнес-логіки, клієнтські додатки для користувачів складу, а також інтеграційні модулі для зв'язку з іншими інформаційними системами. Серверна частина використовує реляційну базу даних для зберігання інформації про товари, замовлення, операції та користувачів. Клієнтські додатки надають інтерфейси для взаємодії користувачів зі складською системою, забезпечуючи інтуїтивно зрозумілий доступ до всіх необхідних функцій. Інтеграційні модулі дозволяють синхронізувати дані з іншими системами, що забезпечує цілісність та актуальність інформації по всьому підприємству [10].

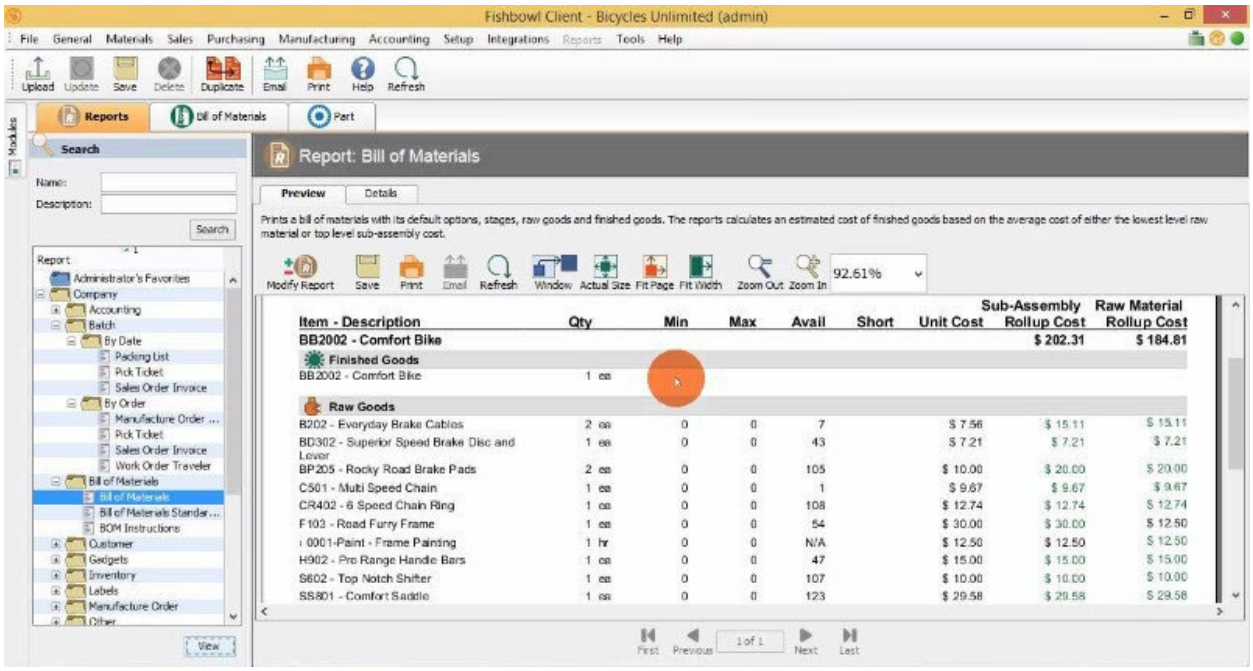


Рисунок 1.2 – Інтерфейс системи Fishbowl Warehouse

Переваги Fishbowl Warehouse:

– інтеграція з QuickBooks. Забезпечує безшовну інтеграцію з популярною бухгалтерською програмою, що полегшує фінансовий облік та управління;

– модульність і масштабованість. Система легко налаштовується під потреби різних підприємств і може масштабуватися разом з їхнім зростанням;

– автоматизація процесів. Автоматизує основні складські операції, що знижує людський фактор та підвищує точність обліку;

– інтуїтивний інтерфейс. Зручний та зрозумілий інтерфейс, який спрощує навчання персоналу та знижує ймовірність помилок.

Недоліки Fishbowl Warehouse:

висока вартість впровадження. Первісні витрати на впровадження та налаштування можуть бути значними для малих підприємств;

– складність інтеграції з іншими системами. Інтеграція з деякими специфічними системами управління може бути складною та вимагати додаткових ресурсів;

– потреба в технічній підтримці. Для ефективного використання системи може знадобитися постійна технічна підтримка, що додає додаткові витрати;

– обмеження функціоналу для специфічних галузей. Може не відповідати специфічним потребам деяких галузей без додаткових налаштувань або модифікацій [11].

Можна сказати, що Fishbowl Warehouse є потужним інструментом для автоматизації складських операцій, що надає користувачам можливість ефективно управляти запасами, інтегруючись з бухгалтерськими системами на зразок QuickBooks. Основні переваги включають модульність, автоматизацію процесів та інтуїтивний інтерфейс. Однак, високі витрати на впровадження, складність інтеграції з іншими системами, потреба в технічній підтримці та можливі обмеження функціоналу можуть бути значними перешкодами для деяких підприємств. В цілому, Fishbowl Warehouse є надійним вибором для тих, хто шукає комплексне рішення для управління складом з можливістю інтеграції та масштабування.

NetSuite WMS комплексним програмним рішенням управління складськими операціями, що призначений для оптимізації обліку продукції, покращення логістики та досягнення підвищення ефективності складських процесів (рис. 1.3). Основною метою цього застосунку є автоматизація управління запасами, зменшення помилок при виконанні складських операцій та покращення обслуговування клієнтів завдяки точному та своєчасному виконанню замовлень [12].

Архітектура NetSuite WMS базується на хмарній платформі, що забезпечує гнучкість, масштабованість та доступність з будь-якої точки світу. Вона включає серверну частину, яка розміщується в хмарі і відповідає за зберігання даних, обробку запитів та виконання бізнес-логіки. Клієнтські

додатки доступні через веб-браузери та мобільні пристрої, що дозволяє користувачам отримувати доступ до системи в режимі реального часу. Інтеграція з іншими модулями NetSuite, такими як фінансовий облік, управління замовленнями та CRM, забезпечує цілісність даних та покращує координацію між різними підрозділами підприємства. NetSuite WMS підтримує автоматизовані процеси приймання, зберігання, переміщення та відвантаження товарів, що сприяє підвищенню продуктивності та зниженню витрат на управління складом [13].

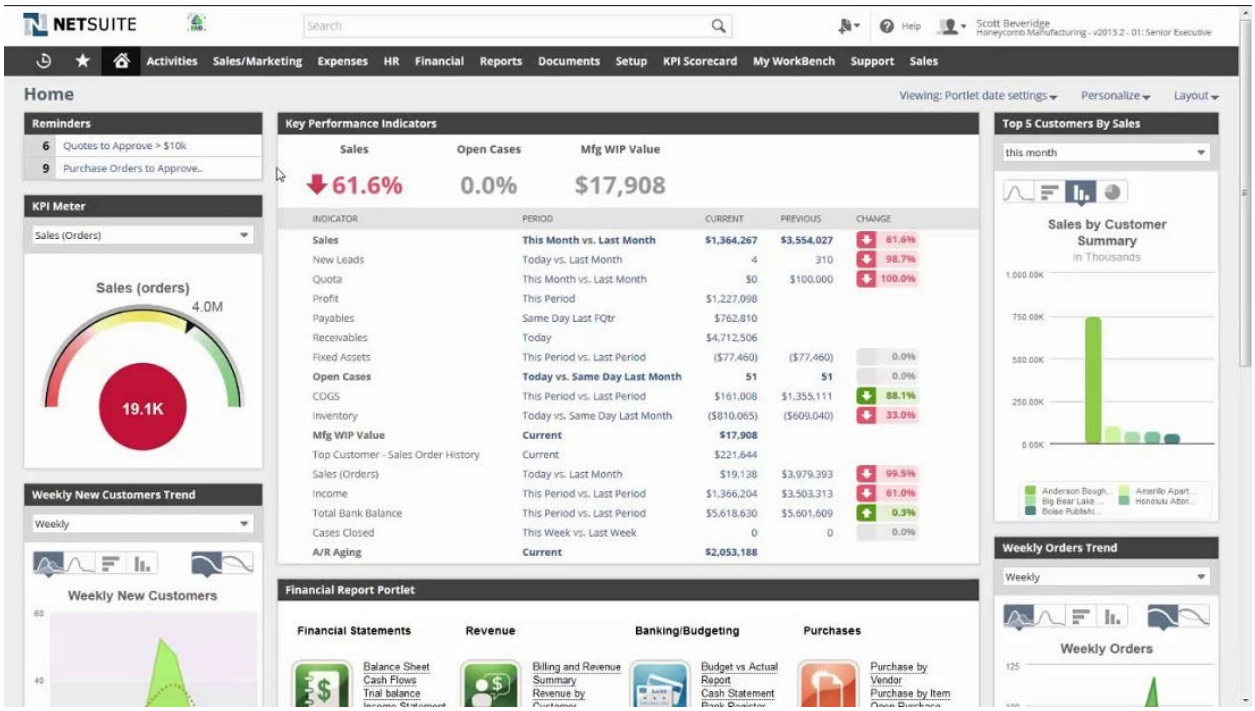


Рисунок 1.3 – Інтерфейс системи NetSuite WMS

Переваги NetSuite WMS:

– хмарна платформа. Забезпечує доступність системи з будь-якої точки світу, гнучкість та масштабованість без потреби у великих початкових інвестиціях в інфраструктуру;

– інтеграція з іншими модулями NetSuite. Плавна інтеграція з фінансовим обліком, управлінням замовленнями та CRM забезпечує

цілісність даних та покращує координацію між різними підрозділами підприємства;

– автоматизація складських процесів. Підтримує автоматизовані процеси приймання, зберігання, переміщення та відвантаження товарів, що знижує людський фактор та підвищує ефективність;

– мобільний доступ. Можливість роботи з мобільних пристроїв дозволяє співробітникам складу виконувати операції в режимі реального часу, що підвищує продуктивність і швидкість обробки замовлень.

Недоліки NetSuite WMS:

– висока вартість підписки. Постійні витрати на підписку можуть бути значними для малих та середніх підприємств, що може обмежити їх можливість впровадження;

– залежність від інтернет-з'єднання. Хмарна архітектура вимагає надійного інтернет-з'єднання, що може бути проблемою у віддалених або слаборозвинених районах;

– складність налаштування. Може вимагати значних зусиль для налаштування та інтеграції з існуючими системами, що потребує додаткових ресурсів та часу;

– обмежена гнучкість. Для деяких специфічних потреб бізнесу може знадобитися додаткова кастомізація, яка може бути дорогою і складною у реалізації [14].

Отже, NetSuite WMS є потужним інструментом для автоматизації складських операцій, що забезпечує гнучкість, масштабованість та доступність завдяки хмарній архітектурі. Основні переваги включають інтеграцію з іншими модулями NetSuite, автоматизацію процесів та можливість мобільного доступу, що підвищує продуктивність і точність обліку продукції. Однак, висока вартість підписки, залежність від інтернет-з'єднання, складність налаштування та обмежена гнучкість можуть бути значними перешкодами для впровадження, особливо для малих та середніх

підприємств. В цілому, NetSuite WMS є надійним вибором для великих компаній, які шукають комплексне рішення для управління складом з інтеграцією в загальну систему управління підприємством.

3PL Central WMS є спеціалізованим програмним рішенням для управління складськими операціями, розробленим для третьої сторони логістичних провайдерів (рис. 1.4). Основним призначенням цього застосунку є забезпечення ефективного обліку продукції, оптимізація логістичних процесів та надання високоякісних послуг клієнтам у сфері складування та логістики. 3PL Central WMS дозволяє автоматизувати процеси приймання, зберігання, переміщення та відвантаження товарів, а також забезпечує точний облік і контроль над запасами [15].

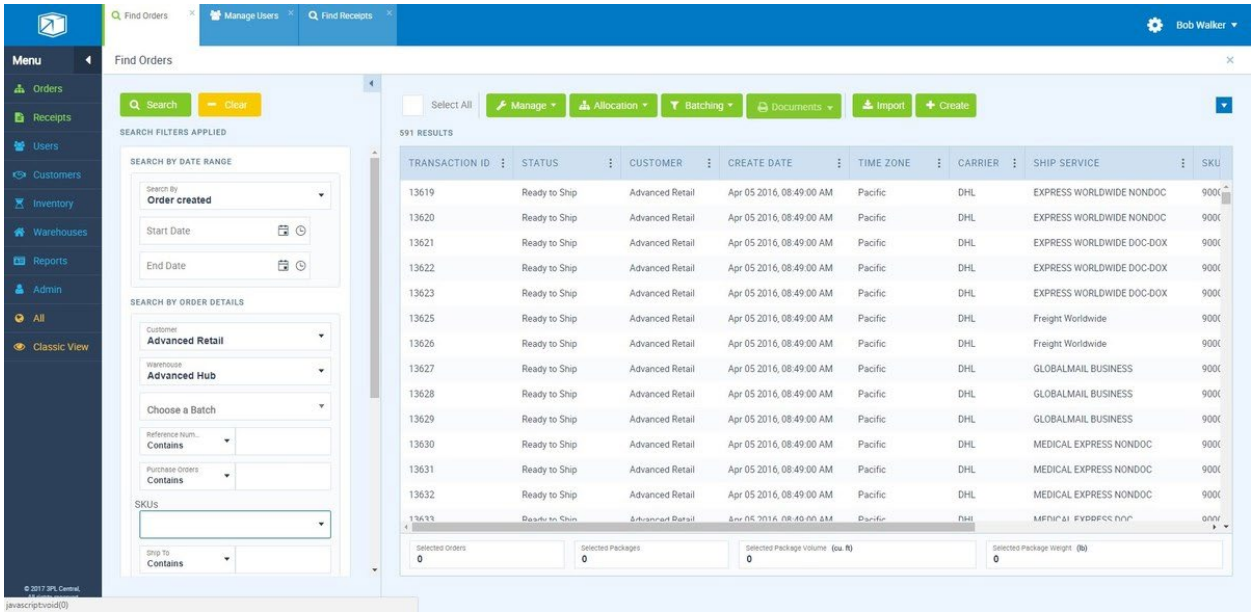


Рисунок 1.4 – Інтерфейс системи 3PL Central WMS

Архітектура 3PL Central WMS базується на хмарній платформі, що забезпечує гнучкість та доступність системи з будь-якої точки світу. Вона включає серверну частину, яка розміщується в хмарі і відповідає за зберігання даних, обробку запитів та виконання бізнес-логіки. Клієнтські додатки доступні через веб-браузери та мобільні пристрої, що дозволяє користувачам отримувати доступ до системи в режимі реального часу.

Інтеграція з іншими інформаційними системами, такими як ERP і TMS, забезпечує цілісність даних та покращує координацію між різними логістичними процесами. Завдяки модульному підходу, 3PL Central WMS може легко налаштовуватися та масштабуватися під потреби різних клієнтів, що дозволяє забезпечити індивідуальний підхід до кожного користувача та ефективне управління складськими операціями [16].

Переваги 3PL Central WMS:

– хмарна платформа. Забезпечує доступ до системи з будь-якої точки світу, гнучкість та масштабованість без великих початкових інвестицій в інфраструктуру;

– спеціалізація для 3PL провайдерів. Оптимізований для потреб логістичних провайдерів третьої сторони, що дозволяє ефективно керувати різноманітними складськими операціями та обслуговувати різних клієнтів;

– інтеграція з іншими системами. Легко інтегрується з ERP, TMS та іншими інформаційними системами, забезпечуючи цілісність даних та покращену координацію логістичних процесів;

– мобільний доступ. Можливість роботи з мобільних пристроїв дозволяє співробітникам складу виконувати операції в режимі реального часу, підвищуючи продуктивність і швидкість обробки замовлень.

Недоліки 3PL Central WMS:

– висока вартість підписки. Постійні витрати на підписку можуть бути значними для деяких провайдерів, що може обмежити їх можливість впровадження;

– залежність від інтернет-з'єднання. Хмарна архітектура вимагає надійного інтернет-з'єднання, що може бути проблемою у віддалених або слаборозвинених районах;

– складність налаштування. Може вимагати значних зусиль для налаштування та інтеграції з існуючими системами, що потребує додаткових ресурсів та часу;

– обмежена гнучкість для нестандартних потреб. Може не повністю відповідати специфічним потребам деяких клієнтів без додаткових кастомізацій, що може бути дорого і складно у реалізації [17].

Можна сказати, що 3PL Central WMS є потужним інструментом для управління складськими операціями, спеціально розробленим для логістичних провайдерів третьої сторони. Його основні переваги включають хмарну платформу, оптимізацію для 3PL провайдерів, інтеграцію з іншими системами та мобільний доступ. Однак, висока вартість підписки, залежність від інтернет-з'єднання, складність налаштування та обмежена гнучкість для специфічних потреб можуть бути значними перешкодами для впровадження. В цілому, 3PL Central WMS є надійним вибором для логістичних провайдерів, які шукають спеціалізоване рішення для ефективного управління складськими операціями та обслуговування клієнтів [18].

Виходячи з проведеного аналізу програмних рішень, виявлено, що більшість існуючих систем мають високу вартість підписки, складність налаштування та інтеграції, а також вимагають надійного інтернет-з'єднання. Це створює значні перешкоди для малих та середніх підприємств, які потребують простішого та доступнішого рішення. Розробка нового рішення з простішим функціоналом, легкістю впровадження та наявністю безкоштовної версії дозволить значно знизити бар'єри для впровадження автоматизованих систем обліку продукції. Це сприятиме підвищенню ефективності складських операцій та забезпеченню конкурентних переваг для підприємств різних масштабів.

2 АНАЛІЗ ВИМОГ І ПОТРЕБ КОРИСТУВАЧІВ СИСТЕМИ ОБЛІКУ ПРОДУКЦІЇ

2.1 Функціональні вимоги до системи обліку продукції

Для забезпечення ефективного управління складськими операціями та автоматизації обліку продукції, необхідно визначити чіткі функціональні вимоги до системи. Ці вимоги мають враховувати всі аспекти процесів приймання, зберігання, переміщення та відвантаження товарів, а також забезпечити інтеграцію з іншими інформаційними системами підприємства. У табл. 2.1 представлено основні функціональні вимоги, що висуваються до системи обліку продукції, які забезпечать її надійну та ефективну роботу.

Таблиця 2.1 – Функціональні вимоги до застосунку

Вимоги	Опис
FR1	Система має надавати функціонал для реєстрації нових користувачів, що дозволить їм отримувати доступ до всіх необхідних функцій програми.
FR-2	Потрібно розробити інтерфейс пошуку, який дозволить користувачам ефективно знаходити продукцію на різних складах за допомогою критеріїв пошуку, таких як назва, склад, модель, виробник, тощо.
FR -3	Система має дозволяти користувачам вносити зміни в свої персональні профілі, включаючи оновлення контактної інформації.
FR-4	Передбачається розробка функціоналу для додавання та оновлення інформації про категорії та продукцію в системі, що включає інтерфейс для керування даними і взаємодію з базою даних для зберігання змін.
FR-5	Система повинна підтримувати можливість додавання і оновлення даних про накладні на продукцію.
FR-6	Необхідно реалізувати інструменти для генерації звітів за певні періоди, які включають дані про надходження та продаж продукції, дозволяючи аналізувати обіг товарів та ефективність продажів.

Нефункціональні вимоги до системи визначають критерії, які спрямовані на забезпечення якості, ефективності, безпеки та інших аспектів системи, які не визначають конкретні функції, але впливають на загальне користування та виконання. У табл. 2.2 приведено основні нефункціональні вимоги до системи.

Таблиця 2.2 – Нефункціональні вимоги до застосунку

Вимоги	Опис
NFR-1	Безпека. Система повинна забезпечувати високий рівень безпеки даних, включаючи шифрування персональних даних користувачів.
NFR-2	Масштабованість. Система повинна бути спроєктована таким чином, щоб вона могла ефективно масштабуватися відповідно до збільшення кількості користувачів та обсягу даних без втрати продуктивності.
NFR-3	Надійність. Система повинна забезпечувати стабільну та неперервну роботу. Це включає здатність до швидкого відновлення після збоїв або помилок, а також резервне копіювання даних для відновлення в критичних ситуаціях.
NFR-4	Продуктивність. Система повинна відповідати вимогам швидкодії, зокрема, час відгуку на запити користувачів не повинен перевищувати встановлених порогів. Продуктивність повинна бути оптимізована для обробки великих обсягів даних.
NFR-5	Інтерфейс користувача. Інтерфейс системи повинен бути інтуїтивно зрозумілим та зручним для користувачів, що включає адаптивний дизайн, що підтримує візуальну привабливість та легкість навігації. Користувачі повинні мати можливість легко знаходити необхідну інформацію та виконувати потрібні дії з мінімальним навчанням.

Актори системи обліку продукції відіграють ключову роль у забезпеченні її функціональності та ефективності. Основними акторами є директор підприємства, складські працівники, менеджери та клієнти. Директор відповідає за управління користувачами, заключення умов на постачання та контроль за виконанням складських операцій. Складські працівники здійснюють операції з приймання, зберігання, переміщення та

відвантаження продукції. Менеджери контролюють рівень запасів, планують закупівлі та аналізують ефективність роботи складу. Клієнти мають доступ до інформації про стан їхніх замовлень та запасів на складі. У табл. 2.3 представлено детальний опис цілей кожного з цих акторів.

Таблиця 2.3 – Ролі та цілі учасників застосунку

Актор	Основні цілі
Директор підприємства	Основною відповідальністю є керування всіма аспектами діяльності підприємства, забезпечення його стабільної роботи та визначення стратегічних напрямків розвитку для підвищення ефективності та доходів.
Робітник складу	Займається організацією та контролем товарних запасів, відповідає за приймання, зберігання та видачу товарів зі складу, а також за відстеження руху товарів для уникнення розбіжностей у запасах.
Менеджер	Відповідає за обробку замовлень, перевірку наявності товарів у наявності та їх резервування для клієнтів. Також координує процеси продажу, щоб забезпечити високий рівень задоволеності клієнтів.
Покупець	Взаємодіє з інтерфейсом системи для ознайомлення з асортиментом продукції, вибору та оформлення замовлень для покупки товарів згідно зі своїми потребами і перевагами.

Варіанти використання системи обліку продукції описують типові сценарії взаємодії користувачів із системою для виконання різних складських операцій. Ці варіанти включають процеси приймання товарів, їх розміщення на складі, переміщення між різними зонами складу, інвентаризацію та відвантаження продукції. Кожен варіант використання допомагає визначити конкретні дії, які необхідно виконати для досягнення поставленої мети, а також забезпечує ясність щодо взаємодії між користувачами та системою. Детальний опис варіантів використання системи наведено у табл. 2.4, де розкрито основні сценарії та їхні ключові етапи.

Таблиця 2.4 – Опис варіантів використання системи

Варіант	Ім’я	Опис
1	2	3
UC1	Реєстрація користувача	Дозволяє створювати нові облікові записи
UC2	Вхід у систему	Забезпечує вхід та доступ до системи
UC3	Перегляд користувачів	Надає адміністраторам можливість перевіряти список зареєстрованих користувачів
UC4	Додавання користувачів	Дозволяє адміністратору вносити нових користувачів до системи
UC5	Оновлення даних користувача	Оновлює персональну інформацію користувачів
UC6	Перегляд категорій продукції	Надає інформацію про наявні категорії
UC7	Реєстрація категорії	Додає нові категорії продукції до системи
UC8	Оновлення даних категорії	Оновлює існуючу інформацію про категорії
UC9	Перегляд продукції	Забезпечує доступ до перегляду наявних товарів
UC10	Додавання продукції	Додає нові товари до асортименту системи
UC11	Редагування продукції	Уможливлює оновлення інформації про продукти в системі
UC12	Перегляд списку складів	Дає змогу користувачам оглядати всі наявні складські приміщення
UC13	Додавання складу	Дозволяє адміністратору вводити нові складські приміщення до системи
UC14	Редагування складу	Надає можливість модифікації даних існуючих складів
UC15	Формування накладної	Дозволяє створювати документи для оформлення покупок товарів
UC16	Підтвердження видачі продукції	Забезпечує авторизацію комірника для підтвердження видачі товарів
UC17	Накладна на постачання	Відповідає за створення документації на поставку товарів
UC18	Перевірка надходження продукції	Підтверджує реєстрацію приходу товарів на склад

Продовження таблиці 2.4

1	2	3
UC19	Огляд постачальників	Забезпечує доступ до даних про всіх постачальників
UC20	Введення даних постачальника	Уможливлює додавання інформації про нових постачальників в базу даних системи
UC21	Оновлення інформації постачальника	Редагування наявних даних про постачальників
UC22	Огляд умов доставки	Дає доступ до даних про умови доставки товарів
UC23	Реєстрація умови доставки	Внесення нових умов доставки в систему
UC24	Модифікація умови доставки	Оновлення існуючої інформації про умови доставки
UC25	Складання звіту про продажі	Генерація аналітичних звітів з продаж за визначений період
UC26	Звітність по прибуттю товарів	Ведення звітів про прибуття товарів у визначений час
UC27	Пошук продукції	Пошук і вибір продукції за параметрами на складах

На основі вимог до системи розроблено use-case діаграми прецедентів, які представлені на рис. 2.1- 2.4 відповідно.

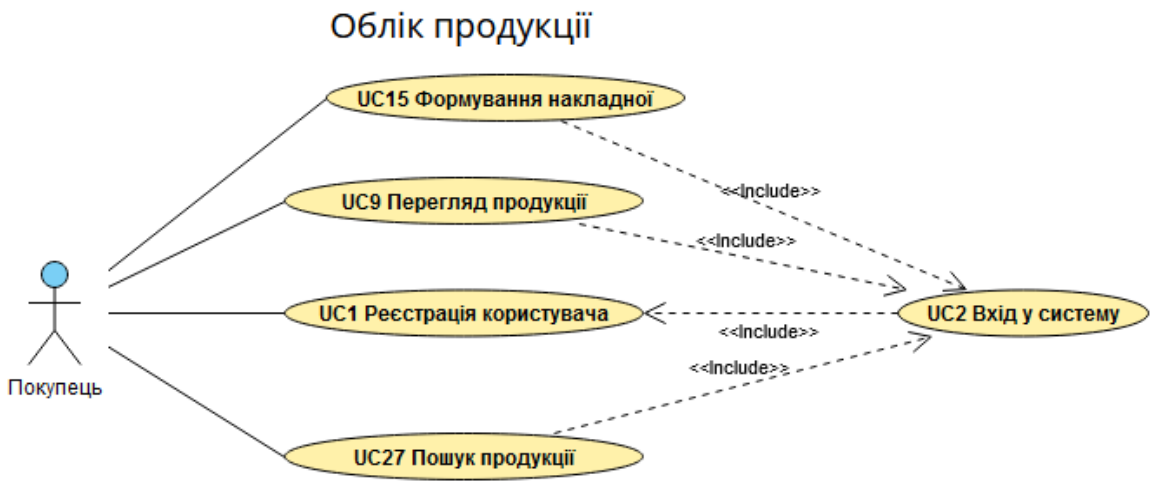


Рисунок 2.1 – Діаграма use-case для ролі покупця

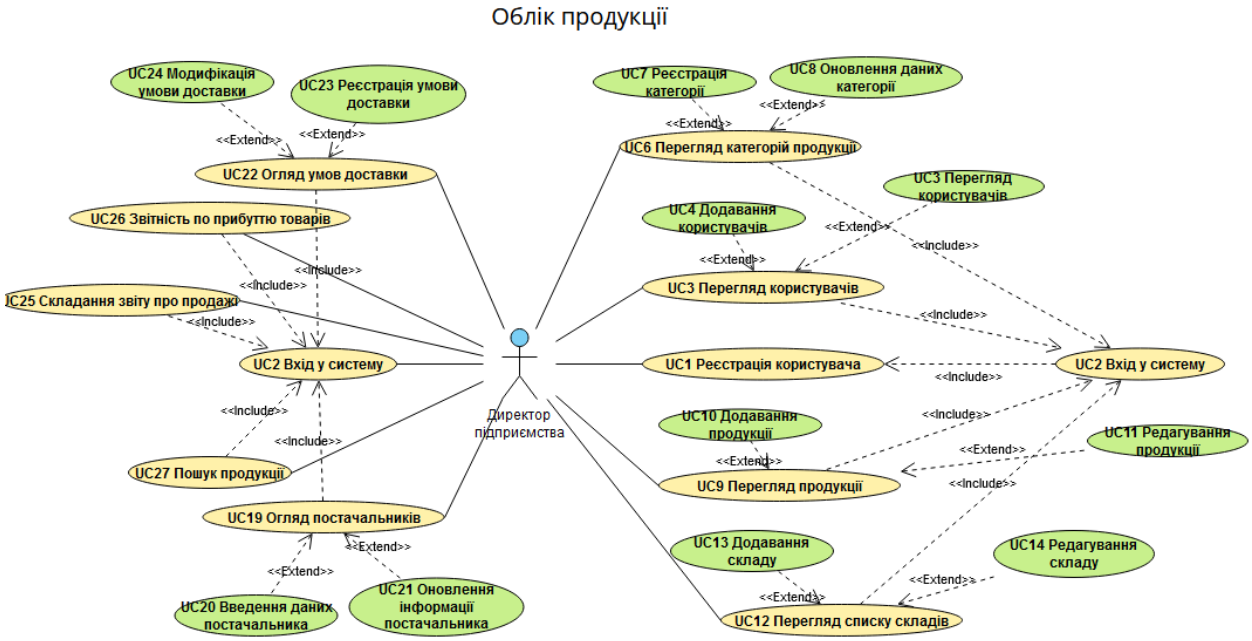


Рисунок 2.2 – Діаграма use-case для ролі директора підприємства

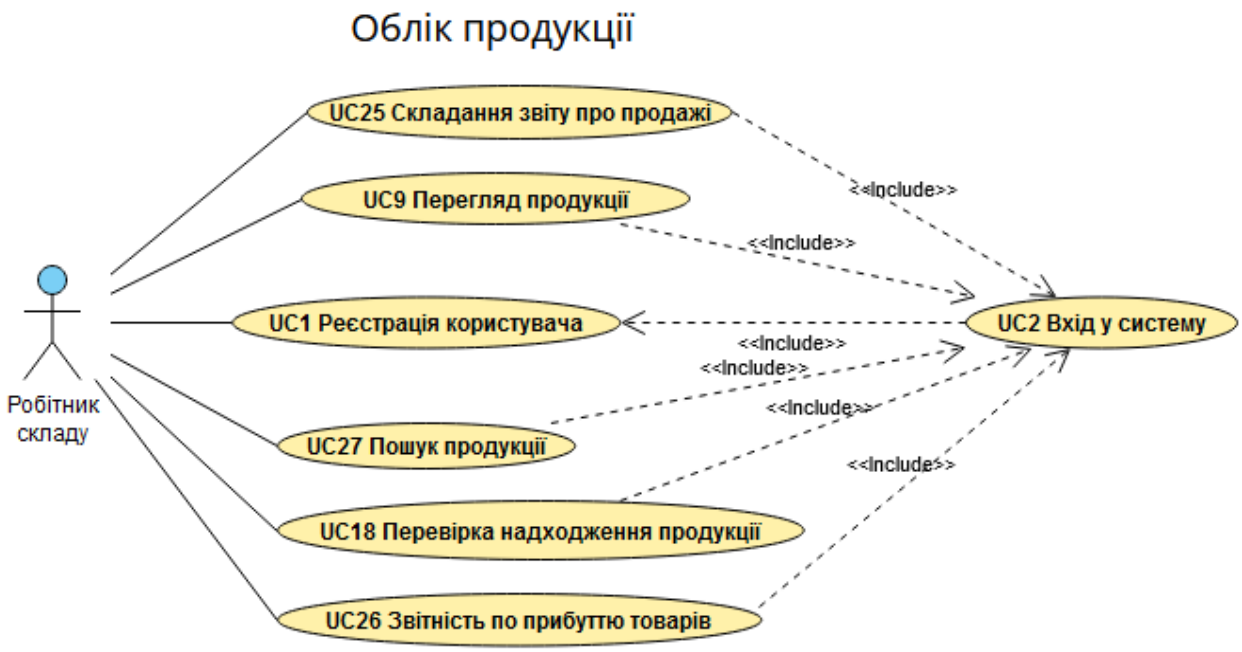


Рисунок 2.3 – Діаграма use-case для ролі робітника складу

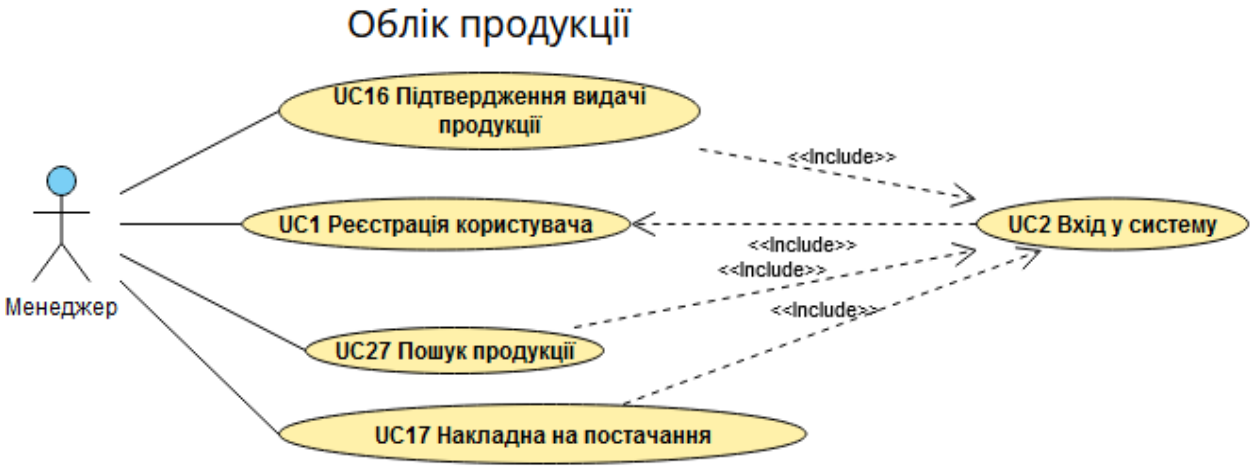


Рисунок 2.4 – Діаграма use-case для ролі менеджера

Завершення побудови чотирьох діаграм варіантів використання дозволяє детально визначити взаємодію ключових акторів із системою обліку продукції. Директор підприємства зосереджується на стратегічному керуванні та контролі ефективності роботи системи, робітник складу виконує операції з управління товарними запасами, менеджер відповідає за обробку замовлень та забезпечення високого рівня обслуговування клієнтів, а покупець використовує систему для ознайомлення з продукцією та оформлення замовлень. Завдяки цим діаграмам можна чітко уявити, як різні користувачі взаємодіють із системою для досягнення своїх цілей та забезпечення ефективної роботи підприємства.

2.2 Технічні вимоги до системи обліку продукції

Технічне забезпечення є одним із основних ресурсів у системах автоматизованої обробки інформації. Воно включає набір обчислювальних пристроїв і технічних засобів, що використовуються для виконання різноманітних інформаційних операцій. Комплекс технічних засобів (КТЗ) охоплює апаратуру для збору, передачі, реєстрації інформації, обладнання для обробки даних і пристрої для зберігання та виведення результатів.

При виборі технічного обладнання важливо дотримуватись декількох принципів: відповідність технічних засобів до специфічної сфери використання; забезпечення необхідної обчислювальної потужності; придатність для спеціалізованого та універсального використання; зручність для користувача; якісне технічне обслуговування; відповідність високим ергономічним стандартам; потрібні технічні характеристики обчислювальних систем; оцінка економічної ефективності використання обладнання; можливість розширення комплексу шляхом додавання периферійних пристроїв; оптимізація асортименту технічних засобів для підвищення ефективності роботи та зменшення вартості обробки інформації; високий рівень надійності обладнання [18].

Особливо популярною стала децентралізована система обробки даних із використанням персональних електронно-обчислювальних машин (ПЕОМ), орієнтована на створення автоматизованих робочих місць (АРМ). АРМ є спеціалізованими обчислювальними системами або наборами периферійних пристроїв, керованих ПЕОМ, і призначені для спрощення робочих процесів професіоналів у конкретних галузях.

Комплектація технічних засобів АРМ визначається завданнями та умовами праці, але зазвичай включає стандартні компоненти: відеомонітор, накопичувачі на жорстких і оптичних дисках, клавіатуру, компактний принтер, модем і мережевий адаптер. Кожен із цих пристроїв виконує важливі функції. Наприклад, принтер потрібен для друку звітів, монітор відображає інформацію, що вводиться з клавіатури та у формі електронних звітів, жорсткі диски зберігають програмне забезпечення та бази даних, а модем і мережеві адаптери забезпечують обмін даними через інтернет та локальну мережу. Для ефективної роботи системи необхідно достатньо місця на диску для програм, архівів даних та обміну інформацією між різними комплексами.

У цій роботі пропонується використовувати обчислювальний комплекс на основі ПЕОМ з процесорами Intel Core i3 або аналогічними, які є широко

поширеними на сучасному ринку. ПЕОМ, сумісні з моделями IBM, є універсальними системами, що використовуються в різних галузях. Зазвичай ці системи працюють на мікропроцесорах INTEL. Серед відомих виробників, що випускають сумісні з IBM моделі, можна виділити такі компанії, як Compaq Computer, HPQ, Asus, MSI тощо.

На рис. 2.5 показано стандартну конфігурацію ПЕОМ, призначену для використання в інформаційно-пошукової системи обліку продукції.

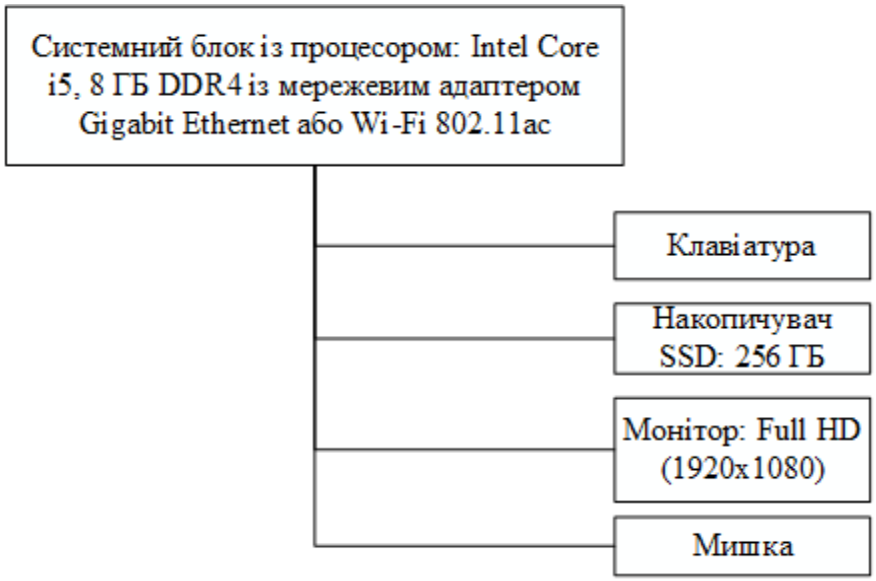


Рисунок 2.5 – Конфігурація ПЕОМ

Конфігурація ПЕОМ, показана на рис. 2.5, включає системний блок із процесором Intel Core i5 і 8 ГБ оперативної пам'яті DDR4. Він обладнаний мережевим адаптером Gigabit Ethernet або Wi-Fi 802.11ac для високошвидкісного підключення до мережі. Для зберігання даних використовується твердотільний накопичувач (SSD) об'ємом 256 ГБ або більше, який призначений для операційної системи та програмного забезпечення.

Монітор має діагональ від 21 до 24 дюймів і підтримує роздільну здатність Full HD (1920x1080), що забезпечує високу якість зображення.

Системний блок оснащений інтерфейсами USB 3.0 або новішими, а також портами HDMI або DisplayPort для підключення монітора.

Інформаційна система аналізу обсягів продажу працює як мережева структура з серверами, тому важливо правильно вибрати сервер для оптимізації її роботи. Вже існуюча в компанії локальна мережа інтегрує робочі станції співробітників і спрощує виконання завдань, пов'язаних з роботою інформаційної системи [19].

Локальна обчислювальна мережа (ЛОМ) об'єднує кілька комп'ютерів і периферійних пристроїв у межах певної території. Ця мережа пропонує численні переваги, зокрема зниження витрат за рахунок спільного використання ресурсів, уніфікацію програмного забезпечення, швидкий доступ до інформації та зручність у взаємодії між співробітниками, що дозволяє ефективно організувати робочий процес.

Мережа працює за допомогою спеціалізованого сервера, налаштованого для ефективної обробки запитів з клієнтських станцій і забезпечення високого рівня безпеки даних та директорій. Сервер виконує різні завдання, які потребують високої спеціалізації, особливо в масштабних мережевих середовищах.

Для ефективної роботи веб-серверів і серверів баз даних рекомендується використовувати сервери, такі як Dell PowerEdge R740. Цей сервер оснащений двома процесорами Intel Xeon з підтримкою до 24 ядер, що забезпечує значну обчислювальну потужність. Оперативна пам'ять обсягом до 3 ТБ дозволяє обробляти великі обсяги даних, а технології віртуалізації підвищують гнучкість управління ресурсами. Місткість зберігання до 48 ТБ забезпечує достатньо місця для даних і резервних копій, а вбудовані засоби безпеки захищають від зовнішніх загроз і шкідливих програм.

При створенні мережі важливо також правильно вибрати кабель. Основні параметри, які слід враховувати, включають легкість монтажу, рівень екранування та захисту, швидкість передачі даних, а також затухання

сигналу і вартість [20]. Враховуючи ці характеристики, а також відстань між сервером і клієнтськими комп'ютерами та вимоги до мережевої інфраструктури, оптимальним вибором є кабель типу вита пара TPI. Цей тип кабелю є універсальним рішенням, яке забезпечує надійне підключення і відповідає вимогам сучасних мереж.

3 ОГЛЯД ТА ВИБІР ІНСТРУМЕНТІВ РОЗРОБКИ СИСТЕМИ

3.1 Вибір платформи для розробки

Вибір платформи для розробки є одним з ключових етапів створення системи обліку продукції, оскільки саме від цього залежить ефективність, продуктивність та масштабованість майбутнього програмного забезпечення. Правильно обрана платформа повинна забезпечувати зручність розробки, підтримку сучасних технологій, високу продуктивність та надійність. У даному підрозділі буде розглянуто кілька популярних платформ для розробки, зокрема .NET Framework, Java EE та Node.js, для визначення найбільш підходящої для розробки системи обліку продукції.

.NET Framework є потужною та гнучкою платформою для розробки різноманітних додатків, розробленою компанією Microsoft. Вона забезпечує широкий спектр інструментів та бібліотек, що дозволяють створювати як десктопні, так і веб-додатки, підтримуючи високу продуктивність та безпеку. .NET Framework добре інтегрується з іншими продуктами Microsoft, що забезпечує зручність у розробці та підтримці програмного забезпечення [21].

Java EE (Enterprise Edition) є платформою для створення корпоративних додатків на базі мови програмування Java. Вона пропонує набір специфікацій та API для розробки надійних, масштабованих та безпечних додатків, що працюють у розподіленому середовищі. Java EE підтримує багатоплатформність, що дозволяє розгортати додатки на різних операційних системах та серверних середовищах. Крім того, вона забезпечує високу продуктивність та надійність, що робить її популярним вибором для великих корпоративних проектів [22].

Node.js є середовищем виконання для JavaScript, яке дозволяє розробляти серверні додатки з високою продуктивністю та масштабованістю. Воно забезпечує асинхронну обробку запитів, що дозволяє ефективно

використовувати ресурси сервера та забезпечувати швидку відповідь на запити користувачів. Node.js має активну спільноту розробників та багатий набір бібліотек, що полегшує створення різноманітних додатків, зокрема реальних чатів, потокових серверів та інших інтерактивних сервісів [23].

Нижче у табл. 3.1 представлено порівняльну таблицю основних характеристик трьох платформ для розробки: .NET Framework, Java EE та Node.js, з акцентом на переваги .NET Framework.

Таблиця 3.1 – Основні характеристики розглянутих платформ

Характеристика	.NET Framework	Java EE	Node.js
Мова програмування	C#, VB.NET	Java	JavaScript
Підтримка типів додатків	Веб-додатки, десктопні додатки, служби	Веб-додатки, корпоративні додатки, служби	Серверні додатки, реальні часи, потокові
Інтеграція	Відмінна інтеграція з продуктами Microsoft	Добра інтеграція з різними середовищами	Широкий набір бібліотек через npm
Безпека	Високий рівень безпеки, вбудовані механізми	Високий рівень безпеки, стандартні API	Добра безпека, але залежить від бібліотек
Продуктивність	Висока продуктивність для різних типів додатків	Висока продуктивність для корпоративних рішень	Висока продуктивність для I/O інтенсивних задач
Масштабованість	Висока, підтримка розподілених систем	Висока, підтримка кластерів та розподілених систем	Висока, але залежить від налаштувань
Інструменти розробки	Visual Studio, потужні інструменти та відлагодження	Eclipse, NetBeans, інші IDE	Visual Studio Code, легкі та гнучкі інструменти

Вибір .NET Framework для розробки системи обліку продукції обґрунтований кількома ключовими перевагами цієї платформи. По-перше, .NET Framework забезпечує відмінну інтеграцію з продуктами Microsoft, що є важливим для підприємств, які вже використовують Microsoft-технології. По-друге, висока продуктивність та надійність платформи дозволяють розробляти масштабовані та ефективні рішення для обліку продукції. Крім того, вбудовані механізми безпеки гарантують захист конфіденційної інформації, а потужні інструменти для розробки та відлагодження, такі як Visual Studio, значно спрощують процес розробки та підтримки системи.

3.2 Вибір середовища розробки

Вибір середовища розробки є критично важливим етапом у процесі створення системи обліку продукції. Від цього вибору залежить зручність та ефективність роботи розробників, а також можливості для тестування та відлагодження програмного забезпечення. У даному підрозділі буде розглянуто кілька популярних середовищ розробки, зокрема Visual Studio, JetBrains Rider та Visual Studio Code, для визначення найбільш підходящого для розробки системи на платформі .NET Framework.

Visual Studio, розроблена компанією Microsoft, є потужним та всебічним середовищем розробки, спеціально оптимізованим для роботи з платформою .NET Framework. Воно надає широкий спектр інструментів для розробки, тестування та відлагодження, що значно спрощує процес створення високоякісного програмного забезпечення. Завдяки інтеграції з іншими продуктами Microsoft, Visual Studio забезпечує зручність у роботі та підвищену продуктивність для розробників [24].

JetBrains Rider є потужним середовищем розробки для .NET, що поєднує в собі функціональність IntelliJ IDEA та ReSharper. Вона забезпечує високу продуктивність, інтелектуальне автозаповнення та розширені можливості відлагодження, що робить її привабливою для розробників .NET.

Rider також підтримує кросплатформенність, дозволяючи розробникам працювати на Windows, macOS та Linux, що робить його гнучким інструментом для різних середовищ [25].

Visual Studio Code є легким та гнучким редактором коду, розробленим компанією Microsoft, що підтримує багато мов програмування, включаючи C# через розширення. Він забезпечує інтеграцію з Git, інтелектуальне автозаповнення та базові можливості відлагодження, що робить його зручним для швидкої розробки та тестування. Visual Studio Code має широкий спектр плагінів, які дозволяють розширювати його функціональність, але його можливості для .NET менш потужні в порівнянні з Visual Studio та Rider [26].

У табл. 3.2 наведено порівняння середовищ розробки, що підтримують .NET Framework: Visual Studio, JetBrains Rider та Visual Studio Code.

Таблиця 3.2 – Порівняння середовищ розробки

Характеристика	Visual Studio	JetBrains Rider	Visual Studio Code
Підтримка .NET Framework	Повна підтримка	Повна підтримка	Обмежена підтримка через плагіни
Інтеграція з іншими продуктами	Відмінна інтеграція з продуктами Microsoft	Добра інтеграція з різними інструментами	Висока інтеграція через розширення
Інтелектуальне автозаповнення	Високий рівень	Високий рівень	Середній рівень через розширення
Інструменти для відлагодження	Потужні інструменти	Потужні інструменти	Обмежені інструменти через плагіни
Продуктивність	Висока продуктивність для великих проектів	Висока продуктивність	Легка та швидка, але менш потужна
Інтерфейс користувача	Інтуїтивний, багатий на функції	Інтуїтивний, багатий на функції	Легкий, мінімалістичний

Вибір Visual Studio для розробки системи обліку продукції обґрунтований її повною підтримкою платформи .NET Framework, що забезпечує максимальну сумісність і ефективність роботи. Visual Studio надає потужні інструменти для розробки, відлагодження та тестування, що значно спрощує процес створення високоякісного програмного забезпечення. Відмінна інтеграція з іншими продуктами Microsoft, такими як Azure і SQL Server, дозволяє легко розгортати та керувати додатками в масштабованому середовищі. Крім того, інтуїтивний та багатий на функції інтерфейс робить Visual Studio зручним та ефективним інструментом для розробників, що підвищує їхню продуктивність та якість кінцевого продукту.

3.3 Вибір СУБД

Вибір системи управління базами даних (СУБД) є критично важливим етапом у процесі розробки системи обліку продукції. СУБД забезпечує надійне зберігання, швидкий доступ та ефективне управління даними, що є основою для функціонування будь-якого інформаційного системного рішення. У цьому підрозділі будуть розглянуті три популярні СУБД – MS SQL Server, MySQL та Oracle – щоб визначити найбільш підходящу для розробки системи обліку продукції.

MS SQL Server, розроблений компанією Microsoft, є потужною реляційною СУБД, яка пропонує високу продуктивність, надійність та масштабованість. Вона має тісну інтеграцію з іншими продуктами Microsoft, такими як .NET Framework та Azure, що спрощує розробку та розгортання додатків. MS SQL Server також забезпечує розширені можливості безпеки та управління даними, включаючи підтримку транзакцій, збережених процедур та складних запитів [27].

MySQL є популярною відкритою реляційною СУБД, відомою своєю високою продуктивністю та гнучкістю. Вона широко використовується для веб-додатків та інтернет-проектів завдяки своїй простоті використання та

великій спільноті підтримки. MySQL підтримує основні функції реляційних баз даних, включаючи транзакції, реплікацію та кластеризацію, що дозволяє розробникам створювати масштабовані та надійні системи [28].

Oracle Database є однією з найпотужніших і найбільш функціонально багатих реляційних СУБД на ринку, що забезпечує високу продуктивність, надійність та безпеку. Вона пропонує широкий спектр можливостей для управління великими обсягами даних, включаючи підтримку складних транзакцій, резервне копіювання та відновлення, а також розширені засоби аналітики. Oracle часто використовується в корпоративних середовищах, де необхідні висока масштабованість та надійність для критично важливих додатків [29].

Нижче представлено порівняльну табл. 3.3 основних характеристик трьох СУБД: MS SQL Server, MySQL та Oracle, з акцентом на переваги MySQL.

Таблиця 3.3 – Порівняння СУБД

Характеристика	MS SQL Server	MySQL	Oracle
Ліцензія	Пропрієтарна, платна	Відкрита, GPL	Пропрієтарна, платна
Вартість	Висока	Низька або безкоштовна	Дуже висока
Продуктивність	Висока для великих додатків	Висока для веб-додатків та середніх проектів	Висока для корпоративних додатків
Масштабованість	Висока	Висока, особливо для веб-додатків	Дуже висока
Підтримка платформ	Windows, Linux, Docker	Windows, Linux, macOS, Docker	Windows, Linux, Solaris, Docker
Безпека	Високий рівень, розширені функції	Високий рівень, але залежить від налаштувань	Високий рівень, розширені функції

Вибір MySQL для розробки системи обліку продукції обґрунтований кількома ключовими перевагами цієї СУБД. По-перше, MySQL є відкритим та безкоштовним рішенням, що значно знижує витрати на розробку та впровадження системи. По-друге, MySQL відомий своєю легкістю у налаштуванні та використанні, що дозволяє швидко розпочати роботу та забезпечує високу продуктивність для веб-додатків та середніх проектів. Крім того, велика та активна спільнота користувачів і розробників забезпечує швидку підтримку та доступ до широкого спектру ресурсів, що сприяє ефективному вирішенню проблем та розвитку системи.

4 ПРОЕКТУВАННЯ СИСТЕМИ ОБЛІКУ ПРОДУКЦІЇ ТА ЇЇ БАЗИ ДАНИХ

4.1 Моделювання поведінки системи

Моделювання поведінки системи є ключовим етапом у розробці програмного забезпечення, оскільки дозволяє зрозуміти, як система буде взаємодіяти з користувачами та іншими компонентами. Воно включає в себе опис динамічних аспектів роботи системи, що відображають зміну станів і обмін повідомленнями між об'єктами. Одним із найбільш ефективних методів моделювання поведінки є діаграми активності та послідовності, які наочно ілюструють порядок виконання операцій та взаємодії між об'єктами.

На рис. 4.1 представлена діаграма послідовності процесу формування накладної.

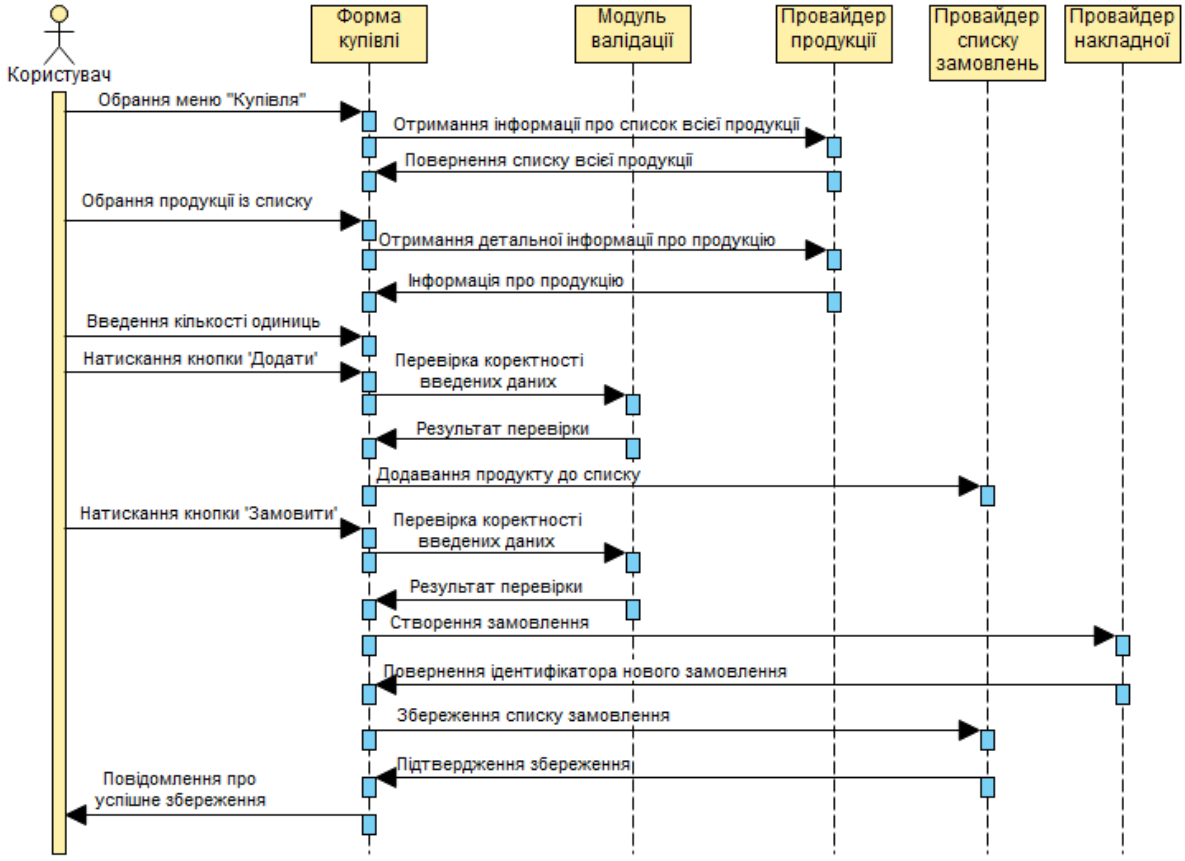


Рисунок 4.1 – Процес формування накладної

Процес формування накладної починається з обрання користувачем меню Купівля. Після цього користувач обирає продукцію зі списку, і система отримує інформацію про весь перелік продукції від провайдера продукції та повертає список користувачу. Коли користувач обирає конкретну продукцію, система отримує детальну інформацію про цю продукцію і відображає її у формі купівлі. Наступним кроком користувач вводить кількість одиниць продукції, яку він бажає придбати, та натискає кнопку Додати. Система перевіряє коректність введених даних через модуль валідації і повертає результат перевірки. У випадку успішної перевірки, продукція додається до списку замовлень.

Коли користувач натискає кнопку Замовити, система ще раз перевіряє коректність введених даних і, за умови успішної перевірки, створює нове замовлення. Провайдер списку замовлень повертає ідентифікатор нового замовлення, і система зберігає список замовлень. Після успішного збереження, користувач отримує повідомлення про це.

Також важливим є представити процес пошуку продукції, який починається з обрання користувачем меню Пошук продукції. Користувач вводить інформацію для пошуку і натискає кнопку Пошук за назвою. Система перевіряє коректність введених даних через модуль валідації і, за умови успішної перевірки, надсилає запит на отримання списку складів підприємства до провайдера пошуку. Після отримання списку складів, система повертає результати пошуку продукції за назвою і відображає їх користувачу. Далі користувач може здійснити пошук за моделлю, натиснувши відповідну кнопку. Система знову перевіряє коректність введених даних і, після успішної перевірки, надсилає запит на отримання продукції за моделлю. Отримані результати пошуку відображаються користувачу.

Окрім того, користувач може здійснити пошук продукції за складом. Після натискання кнопки Пошук за складом, система перевіряє введені дані і надсилає запит на отримання продукції за складом. Результати пошуку

відображаються користувачу, що дозволяє знайти потрібну продукцію на конкретному складі.

Діаграма послідовності на рис. 4.2 представляє процес пошуку продукції в системі.

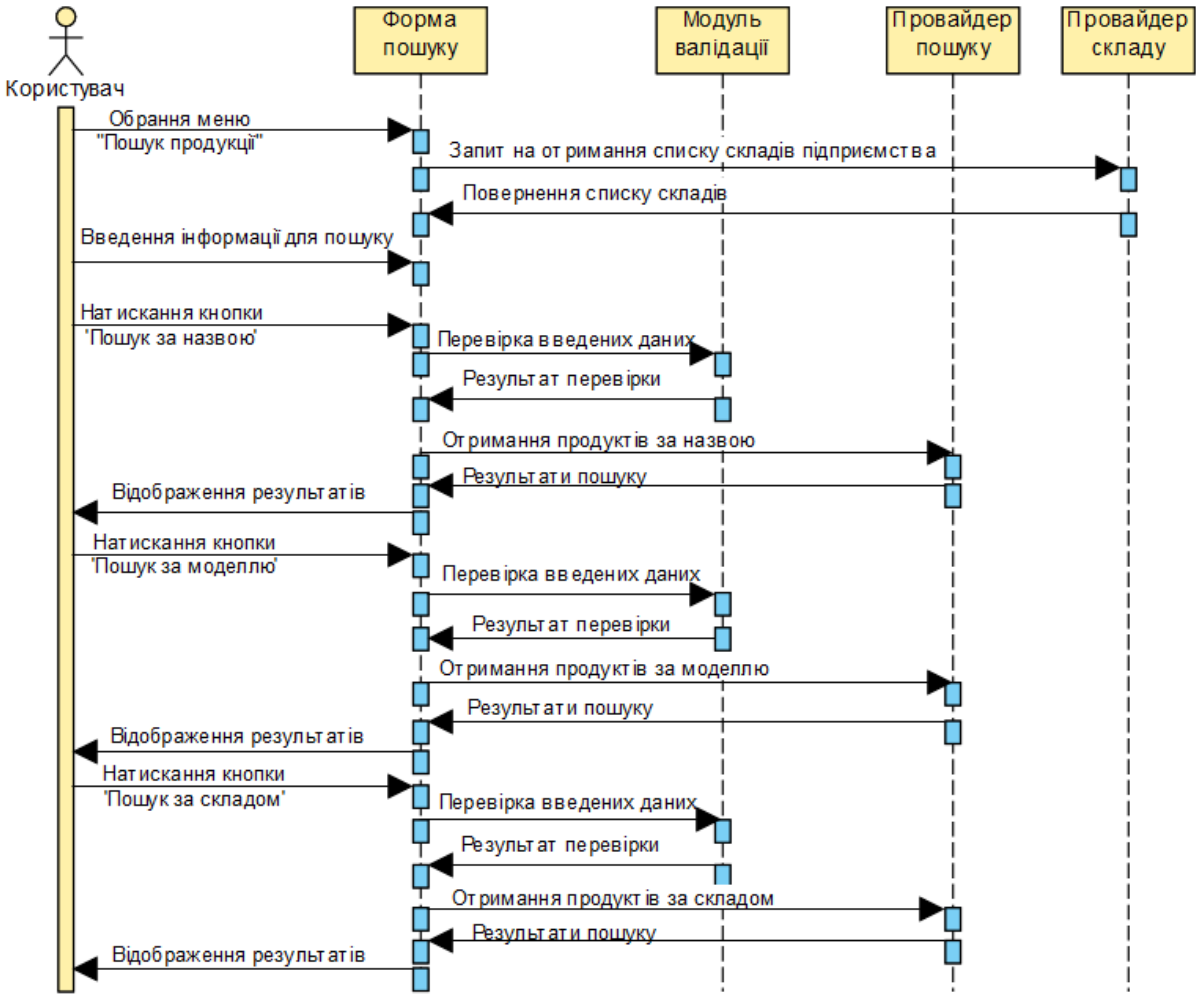


Рисунок 4.2 – Процес пошуку інформації

Діаграми активності є важливим інструментом для моделювання динамічної поведінки системи, оскільки вони відображають послідовність дій і потоки управління між ними. Вони допомагають зрозуміти логіку виконання процесів і виявити можливі точки оптимізації. Діаграми активності зазвичай використовуються для моделювання бізнес-процесів, алгоритмів і робочих процесів у системах.

На рис. 4.3 зображено процес додавання інформації про продукцію. Ця діаграма відображає всі кроки, починаючи від введення даних про новий продукт, перевірки їх коректності, до успішного збереження цієї інформації в базі даних.

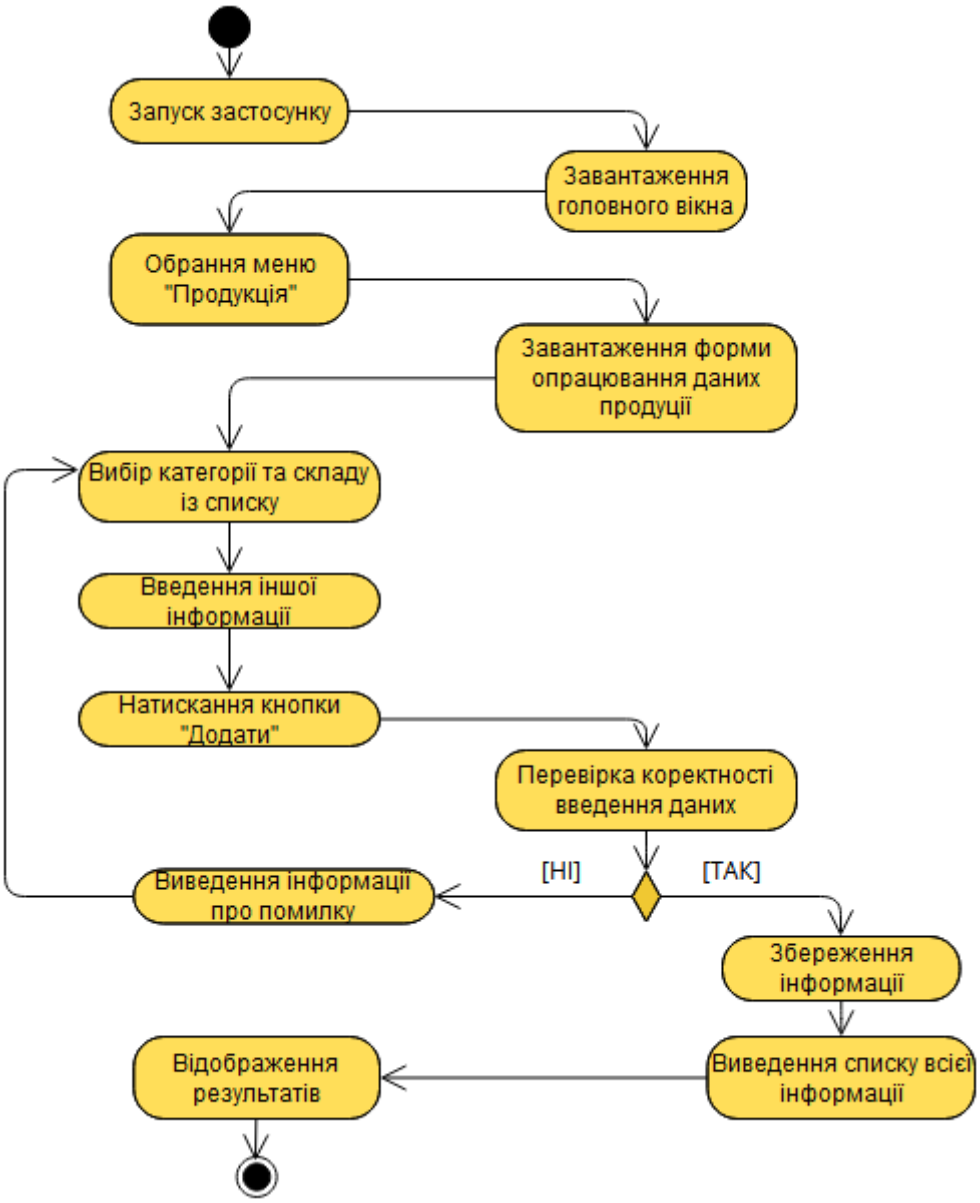


Рисунок 4.3 – Процес додавання інформації про нову продукцію

Процес додавання інформації про продукцію починається з запуску застосунку. Після запуску завантажується головне вікно, де користувач обирає меню Продукція. Далі завантажується форма опрацювання даних про продукцію. Користувач обирає категорію та склад із списку, вводить іншу

необхідну інформацію і натискає кнопку Додати. Система перевіряє коректність введених даних. Якщо дані введені некоректно, користувач отримує відповідне повідомлення і має можливість виправити помилки. У разі успішної перевірки даних, інформація зберігається в базі даних, після чого система виводить список всієї інформації про продукцію та відображає результати додавання.

Рис. 4.4 відображає діаграму послідовності для процесу редагування даних обраної із списку продукції.

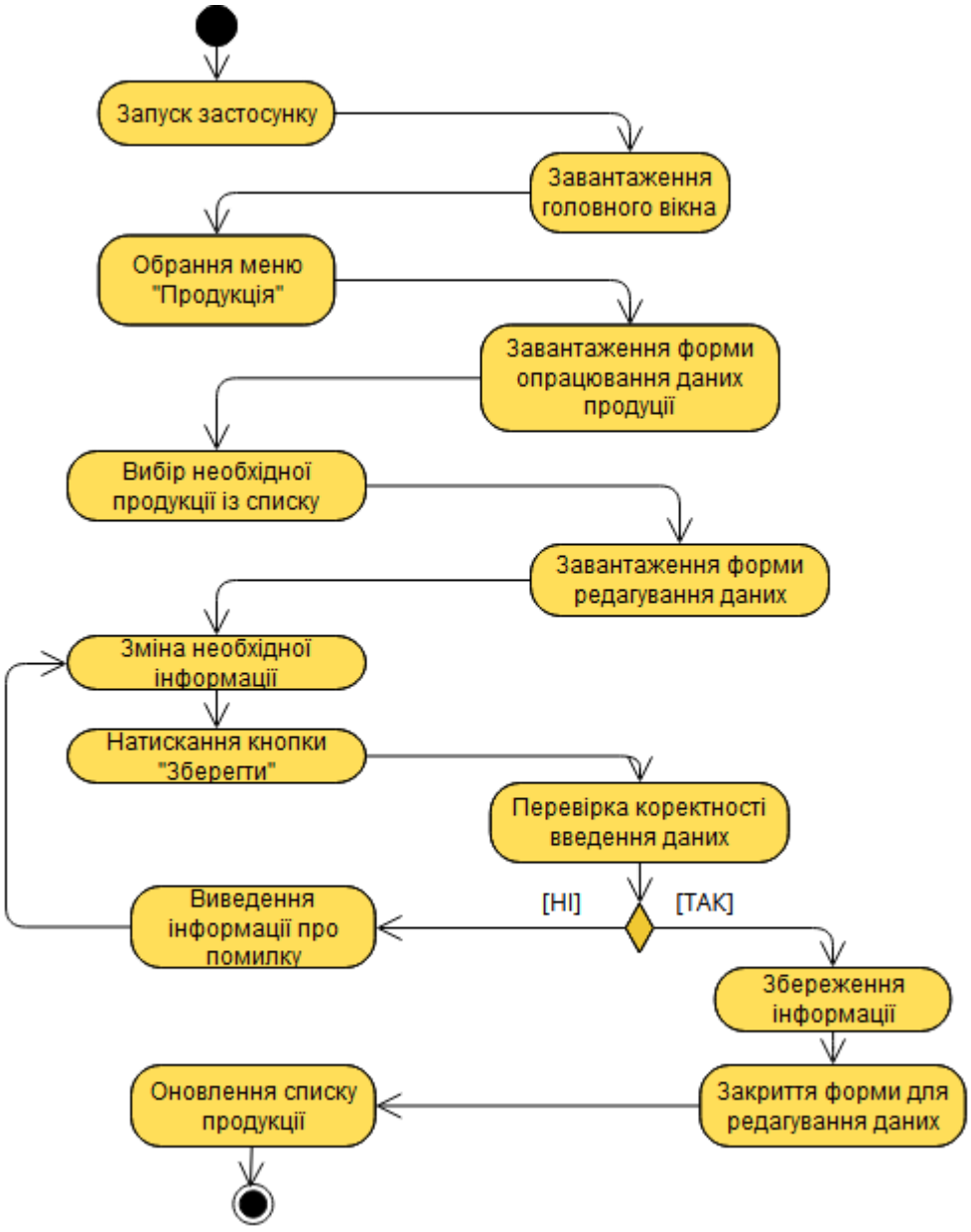


Рисунок 4.4 – Процес редагування даних обраної продукції

Процес редагування інформації про продукцію починається з запуску застосунку, після чого завантажується головне вікно. Користувач обирає меню Продукція, і завантажується форма опрацювання даних про продукцію. Далі користувач обирає необхідну продукцію із списку, що призводить до завантаження форми редагування даних. Користувач вносить зміни до необхідної інформації та натискає кнопку Зберегти. Система перевіряє коректність введених даних. Якщо дані введені некоректно, система виводить інформацію про помилку, і користувач має можливість виправити дані. У разі успішної перевірки даних інформація зберігається, список продукції оновлюється, і форма редагування закривається.

4.2 Створення логічної моделі бази даних

Однією з ключових складових розробки інформаційно-пошукової системи обліку продукції є визначення та моделювання бізнес-правил. Бізнес-правила визначають логіку та обмеження, які керують процесами в системі, забезпечуючи коректність і цілісність даних. Вони є основою для розробки бази даних, оскільки регламентують взаємодію між сутностями, а також правила створення, збереження, оновлення та видалення інформації.

При створенні такої системи важливо врахувати всі аспекти бізнес-процесів, що відбуваються у межах складів, зокрема управління категоріями продукції, облік замовлень, роботу з постачальниками та користувачами. Це дозволяє забезпечити узгодженість даних та оптимізувати роботу системи, роблячи її більш ефективною та надійною.

Нижче розглянуто бізнес-правила, які регулюють роботу з різними сутностями бази даних:

а) Категорії продукції:

- кожна категорія продукції повинна мати унікальний ідентифікатор;

- назва категорії повинна бути унікальною та не порожньою;
- категорії можуть містити опис, але він є необов'язковим.

б) Продукція:

- кожен продукт повинен мати унікальний ідентифікатор;
- назва продукції повинна бути не порожньою;
- кожен продукт має бути прив'язаний до однієї категорії;
- кожен продукт має зберігатися на певному складі;
- кількість продукції повинна бути позитивною цілою числом;
- ціна продажу не може бути від'ємною.

в) Склади:

- кожен склад повинен мати унікальний ідентифікатор;
- назва складу та місце знаходження не можуть бути порожніми;
- склади можуть містити опис, але він є необов'язковим.

г) Замовлення:

- кожне замовлення повинно мати унікальний ідентифікатор;
- замовлення створюється користувачем;
- дата замовлення не може бути в майбутньому;
- загальна сума замовлення повинна бути обчислена як сума цін проданих товарів;
- статус замовлення може мати значення нове, виконане або скасоване.

ґ) Список замовлень:

- кожен елемент списку замовлення повинен мати унікальний ідентифікатор;
- кожен елемент пов'язаний з конкретним замовленням та продуктом;
- кількість продукції повинна бути позитивною цілою числом;
- ціна продажу не може бути від'ємною.

д) Рахунки-фактури:

- кожен рахунок-фактура повинен мати унікальний ідентифікатор;
- рахунок-фактура створюється користувачем;
- дата рахунка-фактури не може бути в майбутньому;
- загальна сума рахунка-фактури повинна бути обчислена як сума цін закуплених товарів;
- статус рахунка-фактури може мати значення новий або оплачений.

е) Постачальники:

- кожен постачальник повинен мати унікальний ідентифікатор;
- назва постачальника не може бути порожньою;
- інші атрибути постачальника можуть бути порожніми.

є) Користувачі:

- кожен користувач повинен мати унікальний ідентифікатор;
- ім'я користувача повинно бути унікальним і не порожнім;
- пароль користувача не може бути порожнім;
- кожен користувач повинен мати прив'язку до певної ролі.

Ці бізнес-правила забезпечать коректну роботу системи, уникнення помилок при введенні даних та підтримання цілісності бази даних.

Після аналізу предметної області виділено ряд сутностей та їх атрибутів, які представлено у табл. 4.1.

Таблиця 4.1 – Сукупна таблиця сутностей та їх атрибутів

Сутності	Атрибути
1	2
Категорії	ідентифікатор (CategoryId), назва (CategoryName), опис (Description)
Журнали	ідентифікатор (LogsId), ідентифікатор користувача (UsersId), опис події (EventNameShow), дата події (EventDate)
Список замовлень	ідентифікатор (OrderListId), ідентифікатор замовлення (OrderId), ідентифікатор продукції (ProductId), кількість (Quantity), ціна продажу (SalePrice)

Продовження таблиці 4.1

1	2
Замовлення	ідентифікатор (OrderId), ідентифікатор користувача (UsersId), дата замовлення (OrderDate), загальна сума (TotalPrice), статус (Status)
Продукція	ідентифікатор (ProductId), назва (ProductName), модель (Model), виробник (Manufacturer), ціна продажу (SalePrice), опис (Description), кількість (Quantity), ідентифікатор категорії (CategoryId), ідентифікатор складу (WarehousesId)
Список продукції	ідентифікатор (ProductsListId), ідентифікатор рахунка-фактури (SalesInvoicesId), ідентифікатор продукції (ProductId), кількість (Quantity), ціна закупівлі (PurchasePrice)
Рахунки-фактури	ідентифікатор (SalesInvoiceId), ідентифікатор користувача (UsersId), дата рахунка-фактури (InvoiceDate), загальна сума (TotalPrice), ідентифікатор постачальника (SupplierId), статус (Status)
Постачальники	ідентифікатор (SupplierId), назва (SupplierName), адреса (Address), електронна пошта (Email), телефон (Phone)
Умови поставок	ідентифікатор (SupplyConditionId), ідентифікатор постачальника (SupplierId), ідентифікатор продукції (ProductId), умови доставки (DeliveryTerms), мінімальний обсяг (MinVolume), максимальний обсяг (MaxVolume), ціна за одиницю (UnitPrice)
Користувачі	ідентифікатор (UsersId), ім'я (FirstName), прізвище (LastName), ім'я користувача (UsersName), пароль (UsersPassword), ідентифікатор ролі (RoleId), опис (Description)
Склади	ідентифікатор (WarehousesId), назва (WarehousesName), місцезнаходження (Location), опис (Description)

Для забезпечення коректної роботи інформаційно-пошукової системи обліку продукції необхідно чітко визначити характеристики доменів атрибутів, що використовуються в базі даних. Це включає в себе визначення типів даних, довжини рядків, форматів дат та числових значень, які можуть приймати атрибути. Такий підхід дозволяє зменшити кількість помилок при введенні даних і забезпечити їх відповідність встановленим стандартам.

Нижче наведено табл. 4.2 з характеристиками доменів атрибутів, що використовуються в базі даних.

Таблиця 4.2 – Відомості про домени та їх атрибути

Ім’я домену	Характеристика домену	Приклад допустимих значень
1	2	3
CategoryName	Рядок змінної довжиною до 100 символів	Електроніка, Одяг
Description	Текст змінної довжини	Продукти електроніки, Взуття для спорту
EventNameShow	Текст змінної довжини	Вхід в систему, Створення замовлення
EventDate	Дата та час	2023-06-22 14:30:00, 2024-01-01 00:00:00
OrderDate	Дата	2023-06-22, 2024-01-01
TotalPrice	Десяткове число (12,2)	1200.50, 599.99
Status	Ціле число (tinyint)	0, 1
ProductName	Рядок змінної довжиною до 255 символів	iPhone 12, Samsung Galaxy
Model	Рядок змінної довжиною до 255 символів	XPS 13, MacBook Pro
Manufacturer	Рядок змінної довжиною до 255 символів	Apple, Dell
SalePrice	Десяткове число (12,2)	999.99, 1499.50
Quantity	Ціле число	100, 500
PurchasePrice	Десяткове число (10,2)	800.00, 1200.50
InvoiceDate	Дата	2023-06-22, 2024-01-01
SupplierName	Рядок змінної довжиною до 255 символів	ТОВ "Постачальник", ТОВ "Вендор"
Address	Рядок змінної довжиною до 255 символів	Україна, Київ, вул. Хрещатик 25
Email	Рядок змінної довжиною до 150 символів	example@example.com

Продовження таблиці 4.2

1	2	3
Phone	Рядок змінної довжиною до 45 символів	+380979200095
DeliveryTerms	Рядок змінної довжиною до 1255 символів	Доставка протягом 3 днів, Самовивіз
MinVolume	Ціле число	10, 50
MaxVolume	Ціле число	1000, 5000
UnitPrice	Десяткове число (10,2)	10.50, 20.00
FirstName	Рядок змінної довжиною до 45 символів	Олександр, Дарія
LastName	Рядок змінної довжиною до 45 символів	Антонів, Яцків
UsersName	Рядок змінної довжиною до 45 символів	user123, admin
UsersPassword	Рядок змінної довжиною до 150 символів	P@ssw0rd, 123456
RoleId	Ціле число	1, 2
WarehousesName	Рядок змінної довжиною до 45 символів	Склад №1, Центральний склад
Location	Рядок змінної довжиною до 245 символів	Україна, Київ, вул. Промислова 1

Для забезпечення ефективного функціонування інформаційно-пошукової системи обліку продукції, важливо чітко визначити взаємозв'язки між сутностями в базі даних. Це дозволяє підтримувати цілісність даних та забезпечувати правильну логіку бізнес-процесів. У табл. 4.3 наведені основні зв'язки між сутностями, які відображають, як різні елементи системи взаємодіють між собою та забезпечують узгодженість інформації.

Таблиця 4.3 – Зв'язки між сутностями в базі даних

Тип сутності	Тип зв’язку	Тип сутності	Тип зв’язку
1	2	3	4
category (категорії)	мають	products (продукція)	1: M
products (продукція)	зберігаються	warehouses (склади)	M: 1
orders (замовлення)	мають	order_list (список замовлень)	1: M
order_list (список замовлень)	містить	products (продукція)	M: 1
orders (замовлення)	створені	users (користувачі)	M: 1
salesinvoices (рахунки-фактури)	мають	products_list (список продукції)	1: M
products_list (список продукції)	містить	products (продукція)	M: 1
salesinvoices (рахунки-фактури)	створені	users (користувачі)	M: 1
salesinvoices (рахунки-фактури)	мають постачальника	suppliers (постачальники)	M: 1
supplyconditions (умови поставок)	мають	suppliers (постачальники)	M: 1
supplyconditions (умови поставок)	містять	products (продукція)	M: 1
logs (журнали)	мають	users (користувачі)	M: 1
users (користувачі)	мають	roles (ролі)	M: 1

Побудова логічної моделі бази даних є важливим етапом у розробці інформаційно-пошукової системи обліку продукції. Вона дозволяє детально описати структуру даних та взаємозв'язки між сутностями, що забезпечують ефективне зберігання і обробку інформації. На рис. 4.5 представлена логічна модель бази даних, яка відображає основні сутності та їх взаємозв'язки, визначені відповідно до бізнес-правил системи.

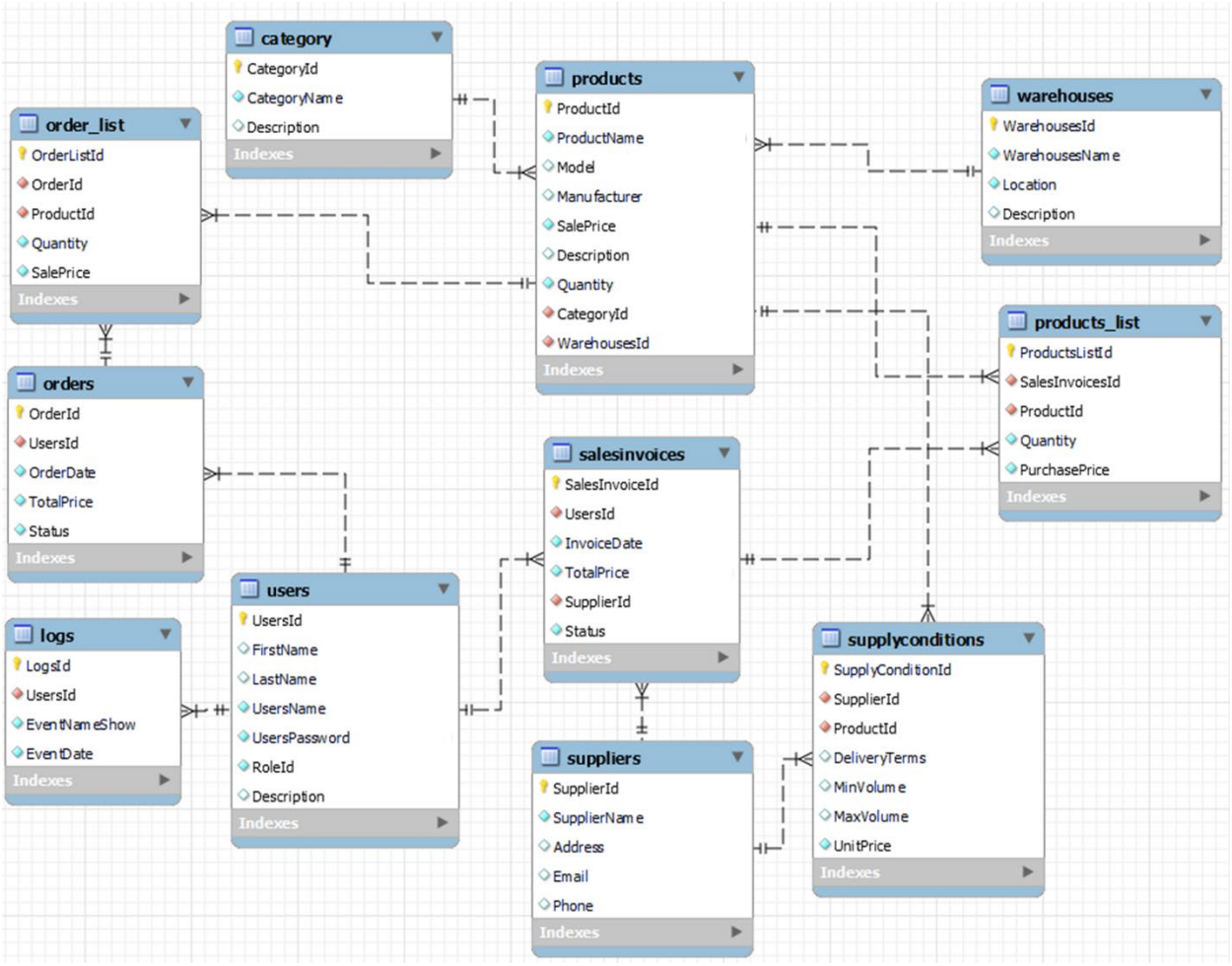


Рисунок 4.5 – Логічна модель бази даних

Логічна модель забезпечує основу для створення фізичної структури бази даних та дозволяє оптимізувати процеси обробки даних. Вона також сприяє підтримці цілісності та узгодженості інформації в системі.

4.3 Створення фізичної моделі

Наступним кроком після розробки логічної моделі розробимо фізичну. Вона включає конкретні деталі реалізації, такі як визначення таблиць, атрибутів, типів даних, індексів та обмежень. Вона описує, як дані зберігатимуться на фізичному рівні та забезпечує оптимальну продуктивність і цілісність бази даних.

Для забезпечення ефективної роботи бази даних та зменшення дублювання даних, необхідно переконатися, що структура бази даних відповідає всім нормальним формам. Наша база даних відповідає першій нормальній формі (1НФ), оскільки всі атрибути містять лише атомарні (неподільні) значення, наприклад, ProductName, OrderDate, TotalPrice, тощо. Крім того, всі записи у таблицях є унікальними завдяки наявності первинних ключів у кожній таблиці.

База даних також відповідає другій нормальній формі (2НФ), оскільки всі неключові атрибути залежать від первинного ключа. Наприклад, у таблиці products атрибути, такі як ProductName, Model, SalePrice залежать від ProductId, що є первинним ключем цієї таблиці. Це гарантує, що всі неключові атрибути мають однозначну залежність від первинного ключа.

Також, структура бази даних відповідає третій нормальній формі (3НФ), оскільки в ній відсутні транзитивні залежності. Це означає, що жоден неключовий атрибут не залежить транзитивно від первинного ключа через інший неключовий атрибут. Наприклад, у таблиці orders атрибути OrderDate та TotalPrice залежать безпосередньо від OrderId і не мають залежності один від одного.

Таким чином, структура нашої бази даних відповідає всім трьом нормальним формам, що забезпечує оптимальне зберігання та цілісність даних. Це дозволяє уникнути аномалій при вставці, оновленні та видаленні даних, зменшує надлишковість та покращує продуктивність бази даних.

У даній базі даних важливу роль відіграють обмеження цілісності, такі як NOT NULL, які гарантують, що певні атрибути не можуть мати відсутні значення. Це особливо важливо для ключових полів та атрибутів, що мають критичне значення для коректного функціонування системи. Наприклад, атрибути ProductName в таблиці products, OrderDate в таблиці orders та CategoryName в таблиці category визначені з обмеженням NOT NULL, що забезпечує обов'язкову наявність значень у цих полях. Це допомагає

запобігти помилкам, пов'язаним з відсутністю даних, та підтримує загальну цілісність і надійність інформаційної системи.

Побудова фізичної моделі бази даних є критичним етапом, що слідує після створення логічної моделі. Фізична модель визначає конкретні деталі реалізації, включаючи структуру таблиць, типи даних для кожного атрибута, індекси та обмеження. На рис. 4.6 представлена фізична модель бази даних, яка ілюструє, як дані будуть зберігатися на фізичному рівні та як забезпечується цілісність і ефективність обробки даних.

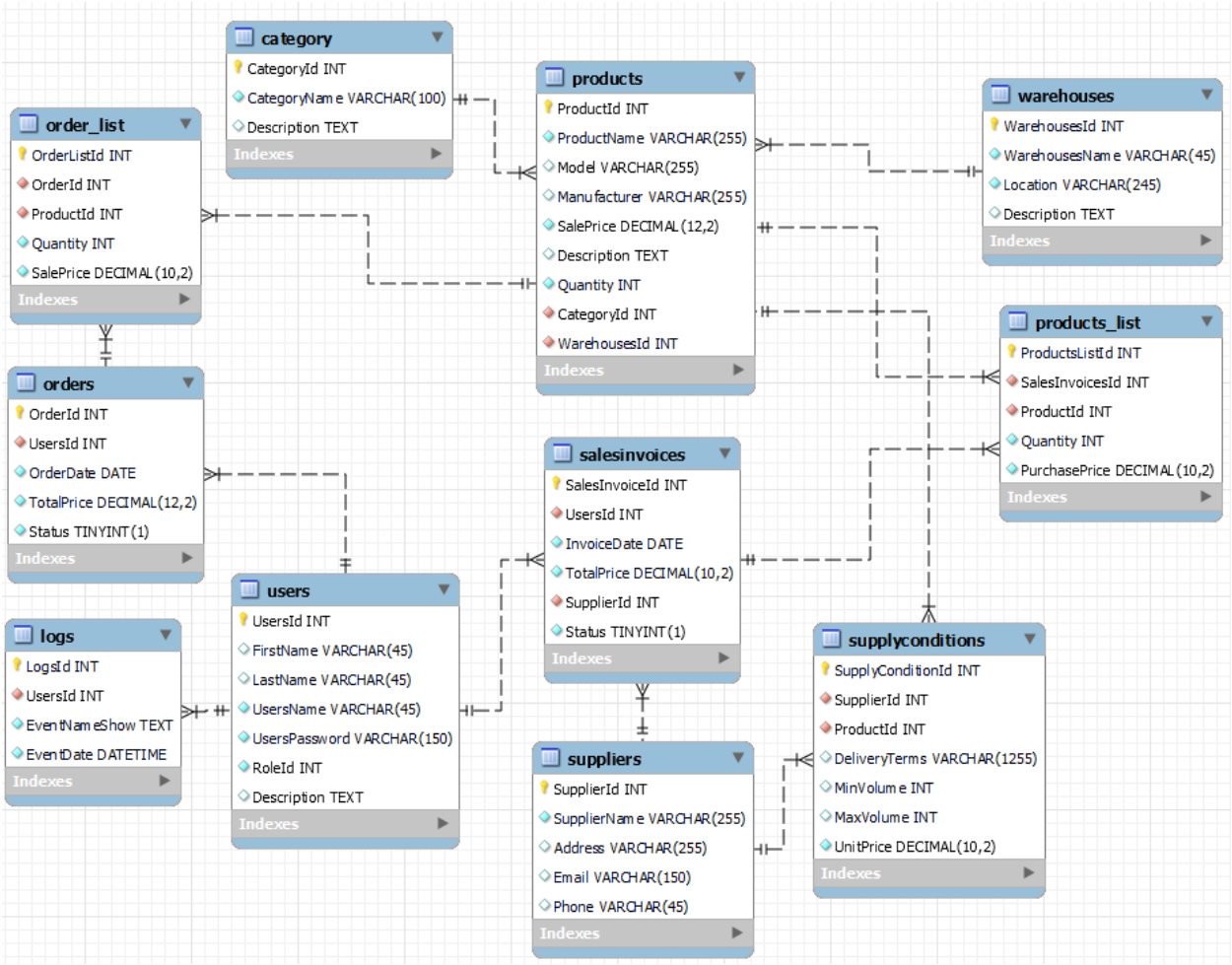


Рисунок 4.6 – Фізична модель бази даних

Ця модель забезпечує основу реалізації бази даних у реальному середовищі та підтримує всі необхідні бізнес-правила. Вона також гарантує,

що всі аспекти логічної моделі будуть втілені у фізичній структурі з урахуванням продуктивності та надійності системи.

5 РЕАЛІЗАЦІЯ, ТЕСТУВАННЯ ТА КОРИСТУВАННЯ СИСТЕМОЮ ОБЛІКУ ПРОДУКЦІЇ

5.1 Реалізація коду системи

Реалізація коду системи обліку продукції включає розробку програмного забезпечення, яке забезпечує взаємодію з базою даних, обробку запитів користувачів та виконання бізнес-логіки. Основним завданням цього етапу є створення функціональних модулів, які реалізують усі визначені раніше процеси, такі як формування накладних, пошук продукції, додавання та редагування даних. Особливу увагу приділено конфігурації системи та налаштуванню з'єднання з базою даних, оскільки це є критично важливим для стабільної та ефективної роботи програми.

На рис. 5.1 зображені конфігураційні змінні та процес підключення застосунку до бази даних, що забезпечують коректну взаємодію з інформаційною системою та підтримку цілісності даних. Це включає налаштування параметрів з'єднання, таких як адреса сервера бази даних, ім'я користувача, пароль та інші необхідні конфігураційні параметри.

```
<appSettings>
  <!-- Підключення до бази даних MySQL -->
  <add key="CONNECTSQL"
       value="server=localhost;user=root;database=warehouse;password=12345;" />
</appSettings>
```

Рисунок 5.1 – Конфігурація з’єднання із базою даних

У конфігураційному файлі застосунку в секції <appSettings> визначено параметри підключення до бази даних MySQL. Ключ CONNECTSQL містить всі необхідні деталі для встановлення з'єднання з базою даних. Значення параметра value включає адресу сервера (server=localhost), що вказує на те, що база даних розміщена на локальному комп'ютері. Далі вказується ім'я користувача (user=root), який має права доступу до бази даних. Параметр

database=warehouse вказує на назву бази даних, до якої буде здійснено підключення. Пароль для доступу до бази даних заданий значенням password=12345. Таке налаштування забезпечує безперервне та безпечне підключення застосунку до бази даних MySQL, дозволяючи виконувати всі необхідні операції з даними.

На початку реалізації системи були розроблені класи та методи для роботи з базою даних, що забезпечують основні CRUD-операції (створення, читання, оновлення, видалення). Ці класи та методи дозволяють ефективно взаємодіяти з базою даних, виконуючи всі необхідні запити та обробляючи отримані результати.

На рис. 5.2 зображено код методу, який відповідає за додавання нової продукції до бази даних. Цей метод включає в себе логіку підключення до бази, формування SQL-запиту та обробки можливих помилок.

```
public void InsertProduct(string ProductName, string Model, string Manufacturer,
  double SalePrice, string Description, int Quantity, int CategoryId, int WarehousesId) {
  MySqlConnection connection = new MySqlConnection(_ConnString);
  string query = "INSERT INTO products (ProductName, Model, Manufacturer, " +
    "SalePrice, Description, Quantity, CategoryId, WarehousesId) " +
    "VALUES(@ProductName, @Model, @Manufacturer, @SalePrice," +
    " @Description, @Quantity, @CategoryId, @WarehousesId)";
  MySqlCommand cmd = new MySqlCommand(query, connection);
  cmd.Parameters.AddWithValue("@ProductName", ProductName);
  cmd.Parameters.AddWithValue("@Model", Model);
  cmd.Parameters.AddWithValue("@Manufacturer", Manufacturer);
  cmd.Parameters.AddWithValue("@SalePrice", SalePrice);
  cmd.Parameters.AddWithValue("@Description", Description);
  cmd.Parameters.AddWithValue("@Quantity", Quantity);
  cmd.Parameters.AddWithValue("@CategoryId", CategoryId);
  cmd.Parameters.AddWithValue("@WarehousesId", WarehousesId);
  connection.Open();
  cmd.ExecuteNonQuery();
  connection.Close();
}
```

Рисунок 5.2 – Код методу для додавання інформації про продукцію

На початку методу створюється об'єкт MySqlConnection, який ініціалізується рядком підключення _ConnString. Далі формується SQL-запит для вставки нового запису в таблицю products. Запит використовує параметризовані значення для атрибутів ProductName, Model, Manufacturer,

SalePrice, Description, Quantity, CategoryId і WarehousesId. Об'єкт MySqlCommand створюється для виконання цього запиту, і параметри додаються до команди за допомогою методу AddWithValue.

Метод InsertOrderAndGetId призначений для додавання нового замовлення до бази даних і отримання ідентифікатора щойно доданого запису (рис. 5.3). На початку методу створюється змінна orderId, яка буде зберігати ідентифікатор замовлення, та об'єкт MySqlConnection, ініціалізований рядком підключення _ConnString. У блоці try формується SQL-запит на вставку нового запису в таблицю orders. Запит включає параметризовані значення для UsersId, OrderDate, TotalPrice і Status. Об'єкт MySqlCommand використовується для виконання цього запиту з параметрами, які додаються за допомогою методу AddWithValue.

```
public int InsertOrderAndGetId(int UsersId, DateTime OrderDate, double TotalPrice, int Status) {
  int orderId = 0;
  MySqlConnection connection = new MySqlConnection(_ConnString);
  try {
    string query = "INSERT INTO orders (UsersId, OrderDate, TotalPrice, Status) " +
      "VALUES(@UsersId, @OrderDate, @TotalPrice, @Status); SELECT LAST_INSERT_ID();";
    MySqlCommand command = new MySqlCommand(query, connection);
    command.Parameters.AddWithValue("@UsersId", UsersId);
    command.Parameters.AddWithValue("@OrderDate", OrderDate);
    command.Parameters.AddWithValue("@TotalPrice", TotalPrice);
    command.Parameters.AddWithValue("@Status", Status);

    connection.Open();
    // Використовуємо ExecuteScalar для отримання останнього ідентифікатора доданого запису
    orderId = Convert.ToInt32(command.ExecuteScalar());
  } catch (Exception ex) {
    // Обробка помилок
    Console.WriteLine("An error occurred: " + ex.Message);
  } finally {
    connection.Close();
  }
  return orderId;
}
```

Рисунок 5.3 – Код методу InsertOrderAndGetId

Після підготовки запиту встановлюється з'єднання з базою даних через метод Open об'єкта MySqlConnection. Запит виконується за допомогою методу ExecuteScalar, який повертає значення першого стовпця першого рядка результату запиту. Це значення конвертується до типу int та присвоюється змінній orderId. У випадку виникнення винятків під час

виконання запиту, вони обробляються у блоці catch, де виводиться повідомлення про помилку. З'єднання з базою даних закривається в блоці finally, забезпечуючи таким чином, що воно буде закрите незалежно від успішності виконання запиту. Метод повертає ідентифікатор доданого замовлення, що дозволяє використовувати його у подальших операціях.

На рис. 5.4 представлено код методу, який призначений для отримання списку продуктів, які не мають пов'язаних умов поставок. Спочатку створюється порожній список productsWithoutConditions для зберігання знайдених продуктів. Потім створюється об'єкт MySqlConnection, ініціалізований рядком підключення _ConnString, і відкривається з'єднання за допомогою блоку using, що забезпечує автоматичне закриття з'єднання після завершення операцій.

```
public List<Product> GetProductsWithoutSupplyConditions() {
  List<Product> productsWithoutConditions = new List<Product>();
  using (MySqlConnection connection = new MySqlConnection(_ConnString)) {
    string query = @"
          SELECT p.*
          FROM products p
          LEFT JOIN supplyconditions sc ON p.ProductId = sc.ProductId
          WHERE sc.SupplyConditionId IS NULL";

    MySqlCommand command = new MySqlCommand(query, connection);
    connection.Open();
    using (MySqlDataReader reader = command.ExecuteReader()) {
      while (reader.Read()) {
        Product product = new Product() {
          ProductId = Convert.ToInt32(reader["ProductId"]),
          ProductName = reader["ProductName"].ToString(),
          Model = reader["Model"].ToString(),
          Manufacturer = reader["Manufacturer"].ToString(),
          SalePrice = reader.IsDBNull(reader.GetOrdinal("SalePrice")) ? 0 : Convert.ToDouble(reader["SalePrice"]),
          Description = reader["Description"].ToString(),
          Quantity = Convert.ToInt32(reader["Quantity"]),
          CategoryId = Convert.ToInt32(reader["CategoryId"]),
          WarehousesId = Convert.ToInt32(reader["WarehousesId"])
        };
        productsWithoutConditions.Add(product);
      }
    }
  }
  return productsWithoutConditions;
}
```

Рисунок 5.4 – Код методу «GetProductsWithoutSupplyConditions»

SQL-запит використовує LEFT JOIN для об'єднання таблиці products з таблицею supplyconditions, і фільтрує результати, щоб залишити тільки ті продукти, які не мають відповідних записів в таблиці supplyconditions

(WHERE sc.SupplyConditionId IS NULL). Команда MySqlCommand виконує цей запит, а результати зчитуються за допомогою MySqlDataReader всередині іншого блоку using.

Читач послідовно перебирає кожен запис, створюючи новий об'єкт Product з даними зчитаними з поточного рядка. У випадку, якщо значення поля SalePrice є відсутнім (DBNull), воно встановлюється в 0. Кожен об'єкт Product заповнюється даними з відповідних колонок і додається до списку productsWithoutConditions. Після завершення читання всіх записів і закриття з'єднання метод повертає заповнений список продуктів без умов поставок.

Розробка форм застосунку є важливим етапом у створенні зручного та функціонального інтерфейсу користувача. Форми дозволяють користувачам взаємодіяти із системою, вводити та отримувати необхідну інформацію, виконуючи різноманітні операції. На рис. 5.5 показано форму для здійснення пошуку інформації по продукції за різними критеріями.

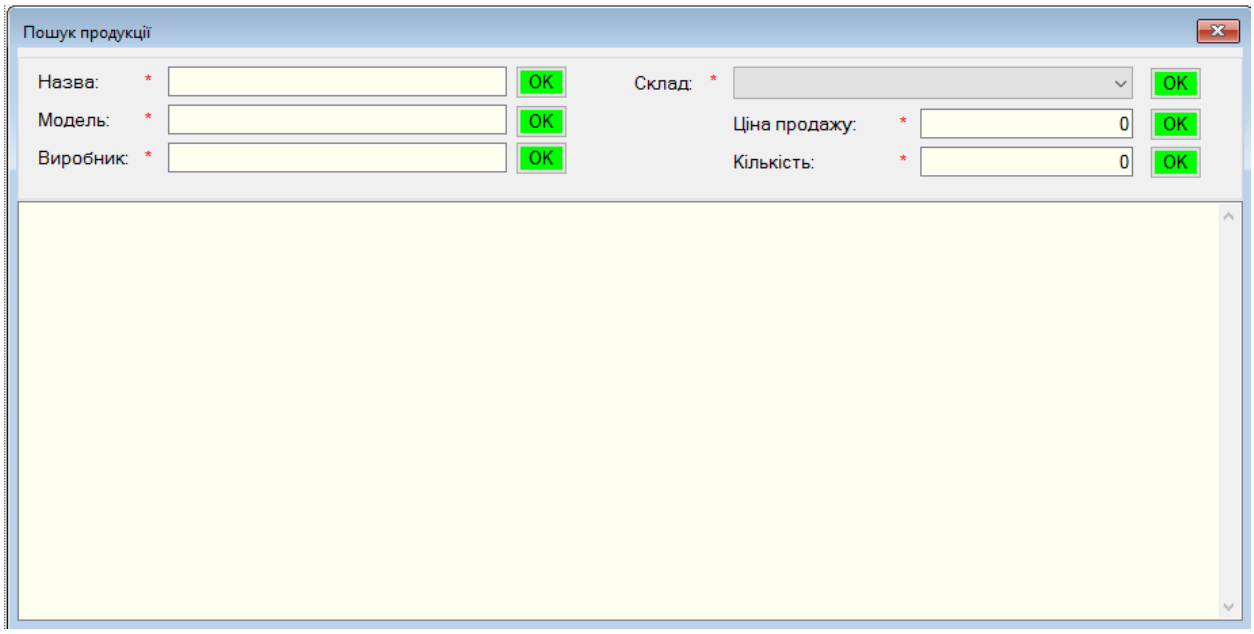


Рисунок 5.5 – Реалізація форми видачі інструментів

Ця форма включає поля для введення назви, моделі, виробника, ціни продажу, кількості та вибору складу. Кожне поле має мітку та поле для введення даних або вибору з випадаючого списку, а також кнопку OK для

підтвердження введених даних. Обов'язкові поля позначені червоними зірочками. У нижній частині форми передбачено місце для відображення результатів пошуку. Такий підхід до розробки форми забезпечує зручний та ефективний пошук продукції за різними критеріями.

При завантаженні форми викликається метод LoadWarehouses, код якого представлено на рис. 5.6.

```
1 reference
private void LoadWarehouses() {
    _WarehousesL = _WarehousesProvider.GetAllWarehouses();
    WarehousesCBox.DataSource = _WarehousesL;
    WarehousesCBox.ValueMember = "WarehousesId";
    WarehousesCBox.DisplayMember = "WarehousesName";
}
```

Рисунок 5.6 – Код методу LoadWarehouses

Метод LoadWarehouses призначений для завантаження списку складів і відображення їх у випадаючому списку на формі. Спочатку метод викликає функцію GetAllWarehouses, яка отримує повний список складів і зберігає його в змінну _WarehousesL. Потім цей список призначається як джерело даних для елемента керування WarehousesCBox, що представляє собою випадаючий список. Параметр ValueMember встановлюється на WarehousesId, що означає, що значенням кожного елемента у списку буде ідентифікатор складу. Параметр DisplayMember встановлюється на WarehousesName, що визначає відображуваний текст для кожного елемента у випадаючому списку. Таким чином, користувачі можуть бачити назви складів у випадаючому списку, але фактично вибирати відповідні ідентифікатори складів для подальшої обробки.

Для пошуку інформації про продукції за назвою реалізовано метод ProductNameSearchBtn_Click обробляє подію натискання кнопки пошуку (рис. 5.7).

```
private void ProductNameSearchBtn_Click(object sender, EventArgs e) {
    if (_validation.IsDataEntering(ProductNameTBox.Text)) {
        ProductNameValiadtionLbl.Text = NamesMy.ProgramButtons.RequiredValidation;
        _ProductSearch = _SearchBLL.GetProductsByName(ProductNameTBox.Text);
        DisplayProductSearchResults(_ProductSearch);
    } else {
        ProductNameValiadtionLbl.Text = NamesMy.ProgramButtons.ErrorValidation;
    }
}
```

Рисунок 5.7 – Код методу ProductNameSearchBtn_Click

Спочатку метод перевіряє, чи введені дані в текстове поле ProductNameTBox за допомогою методу IsDataEntering. Якщо дані введені коректно, мітка ProductNameValiadtionLbl встановлюється на текст, що вказує на успішну валідацію, визначений у RequiredValidation. Потім викликається метод GetProductsByName, який повертає список продуктів, що відповідають введеній назві, і цей список зберігається у змінну _ProductSearch. Далі, метод DisplayProductSearchResults викликається для відображення результатів пошуку на формі. Якщо дані введені некоректно, мітка ProductNameValiadtionLbl встановлюється на текст, що вказує на помилку валідації, визначений у ErrorValidation.

На рис. 5.8 представлено реалізацію методу для пошуку інформації за назвою.

```
//За вказаною назвою або частиною назви ProductName:
1 reference
public List<ProductSearch> GetProductsByName(string productName) {
    List<ProductSearch> products = new List<ProductSearch>();
    string query = $@"
        SELECT p.*, c.CategoryName, w.WarehousesName
        FROM products p
        JOIN category c ON p.CategoryId = c.CategoryId
        JOIN warehouses w ON p.WarehousesId = w.WarehousesId
        WHERE p.ProductName LIKE @ProductName";
    using (MySqlConnection connection = GetConnection()) {
        MySqlCommand command = new MySqlCommand(query, connection);
        command.Parameters.AddWithValue("@ProductName", "%" + productName + "%");
        connection.Open();
        using (MySqlDataReader reader = command.ExecuteReader()) {
            while (reader.Read()) {
                products.Add(MapProduct(reader));
            }
        }
    }
    return products;
}
```

Рисунок 5.8 – Код методу GetProductsByName

Метод GetProductsByName виконує пошук продуктів за вказаною назвою або частиною назви. Спочатку створюється порожній список products, який буде зберігати результати пошуку. SQL-запит будується з використанням операторів JOIN, щоб отримати дані з таблиць products, category, і warehouses, забезпечуючи повернення додаткової інформації про категорію та склад кожного продукту. Запит використовує оператор LIKE для порівняння назви продукту з введеним значенням, дозволяючи знайти продукти, що містять вказану назву або її частину.

З'єднання з базою даних встановлюється за допомогою методу GetConnection у блоці using, що забезпечує автоматичне закриття з'єднання після завершення операцій. Об'єкт MySqlCommand використовується для виконання запиту, причому параметр @ProductName заповнюється значенням, що містить шаблон для пошуку (%productName%). Після відкриття з'єднання виконується запит за допомогою методу ExecuteReader.

Результати запиту зчитуються за допомогою об'єкта MySqlDataReader всередині іншого блоку using. Всі знайдені продукти додаються до списку products за допомогою методу MapProduct, який відображає дані з кожного рядка результату на об'єкт ProductSearch. Після завершення зчитування результатів з'єднання автоматично закривається, і метод повертає список знайдених продуктів.

Також необхідно згадати і про важливі реалізовані запити у системі. Наприклад на рис. 5.9 зображено запит, який призначений для отримання списку продуктів, яких не вистачає для виконання конкретного замовлення.

```
string query = @"
        SELECT p.ProductId, p.ProductName, ol.Quantity AS QuantityOrdered, p.Quantity AS QuantityAvailable
        FROM order_list ol
        JOIN products p ON ol.ProductId = p.ProductId
        WHERE ol.OrderId = @OrderId AND ol.Quantity > p.Quantity;
    ";
```

Рисунок 5.9 – Списку продуктів, яких не вистачає для виконання замовлення

SQL-запит використовується для вибору даних про продукти з таблиці order_list і приєднання їх до таблиці products, щоб отримати інформацію про доступну кількість продуктів.

Запит вибирає ідентифікатор продукту (ProductId), назву продукту (ProductName), кількість замовленого продукту (QuantityOrdered), і кількість доступного продукту (QuantityAvailable). Він містить умову для вибірки тільки тих продуктів, кількість яких у замовленні (ol.Quantity) перевищує кількість доступних на складі (p.Quantity). Умова WHERE ol.OrderId = @OrderId забезпечує фільтрацію результатів за конкретним ідентифікатором замовлення.

Для отримання списку продуктів, які не мають умов постачання від конкретного постачальника реалізовано запит, що представлено на рис. 5.10.

```
string query = $@"
    SELECT p.ProductId, p.ProductName
    FROM products p
    LEFT JOIN supplyconditions sc ON p.ProductId = sc.ProductId AND sc.SupplierId = @SupplierId
    WHERE p.ProductId IN ({inParams}) AND sc.SupplyConditionId IS NULL";
```

Рисунок 5.10 – Список продуктів, які не мають умов постачання

На початку формується SQL-запит, що вибирає ідентифікатор продукту (ProductId) і назву продукту (ProductName) з таблиці products. Запит використовує LEFT JOIN для об'єднання таблиці products з таблицею supplyconditions за умовою, що ідентифікатори продукту збігаються, і додатково перевіряє, чи постачальник відповідає вказаному supplierId. Умова WHERE використовується для фільтрації продуктів, які входять до списку продуктів, переданого як аргумент методу (productList), і для яких немає записів в таблиці supplyconditions (sc.SupplyConditionId IS NULL). У кінці запит повертає тільки ті продукти, які присутні в списку і не мають умов постачання від зазначеного постачальника, що дозволяє легко ідентифікувати продукти без відповідних умов постачання.

На рис. 5.11 показано запит, який призначений для отримання інформації про продану продукцію за вказаний період.

```
string query = @"
        SELECT o.OrderId, o.OrderDate, p.ProductName, ol.Quantity,
               ol.SalePrice, (ol.Quantity * ol.SalePrice) AS TotalSale
        FROM orders o
        JOIN order_list ol ON o.OrderId = ol.OrderId
        JOIN products p ON ol.ProductId = p.ProductId
        WHERE o.Status = 1 AND o.OrderDate BETWEEN @StartDate AND @EndDate
        ORDER BY o.OrderDate ASC";
```

Рисунок 5.11 – Продаж продукції за період часу

Спочатку формується SQL-запит, який вибирає дані з трьох таблиць: orders, order_list, та products. Запит вибирає ідентифікатор замовлення (OrderId), дату замовлення (OrderDate), назву продукту (ProductName), кількість проданих одиниць (Quantity), ціну продажу (SalePrice), а також обчислює загальну суму продажу (TotalSale) для кожного продукту як добуток кількості та ціни продажу. Таблиця orders з'єднується з таблицею order_list по полю OrderId, а таблиця order_list з'єднується з таблицею products по полю ProductId.

Умова WHERE використовується для вибору замовлень, які мають статус виконано (o.Status = 1) і дата яких знаходиться в межах вказаного періоду (o.OrderDate BETWEEN @StartDate AND @EndDate). Результати сортуються за датою замовлення у зростаючому порядку (ORDER BY o.OrderDate ASC), що дозволяє відображати інформацію про продані продукти в хронологічному порядку.

5.2 Функціональне та модульне тестування системи

Функціональне та модульне тестування системи є критично важливими етапами в процесі розробки програмного забезпечення, що забезпечують якість та надійність роботи системи. Функціональне тестування спрямоване

на перевірку відповідності роботи системи її функціональним вимогам, тоді як модульне тестування дозволяє виявити помилки на рівні окремих модулів або компонентів системи. Ці типи тестування допомагають виявити та усунути помилки на ранніх етапах розробки, що сприяє підвищенню стабільності та продуктивності системи.

У табл. 5.1 наведено тестові сценарії для форми Продукція, які включають перевірку основних функцій, таких як додавання, редагування та видалення інформації про продукцію, а також перевірку коректності відображення даних.

Таблиця 5.1 – Тестові сценарії для форми Продукція

№	Тестовий сценарій	Вхідні дані	Очікуваний результат
1	Додавання нової продукції	Назва: "Test Product", Модель: "Model 123", Виробник: "Test Manufacturer", Ціна продажу: "100", Кількість: "10".	Продукція успішно додана до списку, новий запис з'являється у таблиці
2	Додавання продукції з порожніми обов'язковими полями	Назва: "", Модель: "", Виробник: "", Ціна продажу: "", Кількість: ""	Повідомлення про помилку, продукція не додається
3	Редагування існуючої продукції	Вибір продукції зі списку, зміна назви на "Updated Product"	Назва продукції у списку змінюється на "Updated Product"
4	Видалення продукції	Вибір продукції зі списку, натискання кнопки "Видалити"	Продукція успішно видалена зі списку, запис зникає з таблиці
5	Перевірка відображення кількості продукції	Вибір продукції зі списку	Відображається правильна кількість обраної продукції
6	Очищення форми	Натискання кнопки "Очистити"	Всі поля очищуються

Тестові сценарії охоплюють основні функції форми Продукція, включаючи додавання, редагування та видалення, а також перевірку коректності введених даних і очищення форми. Результат виконання тестових сценарії представлено на рис. 5.12.

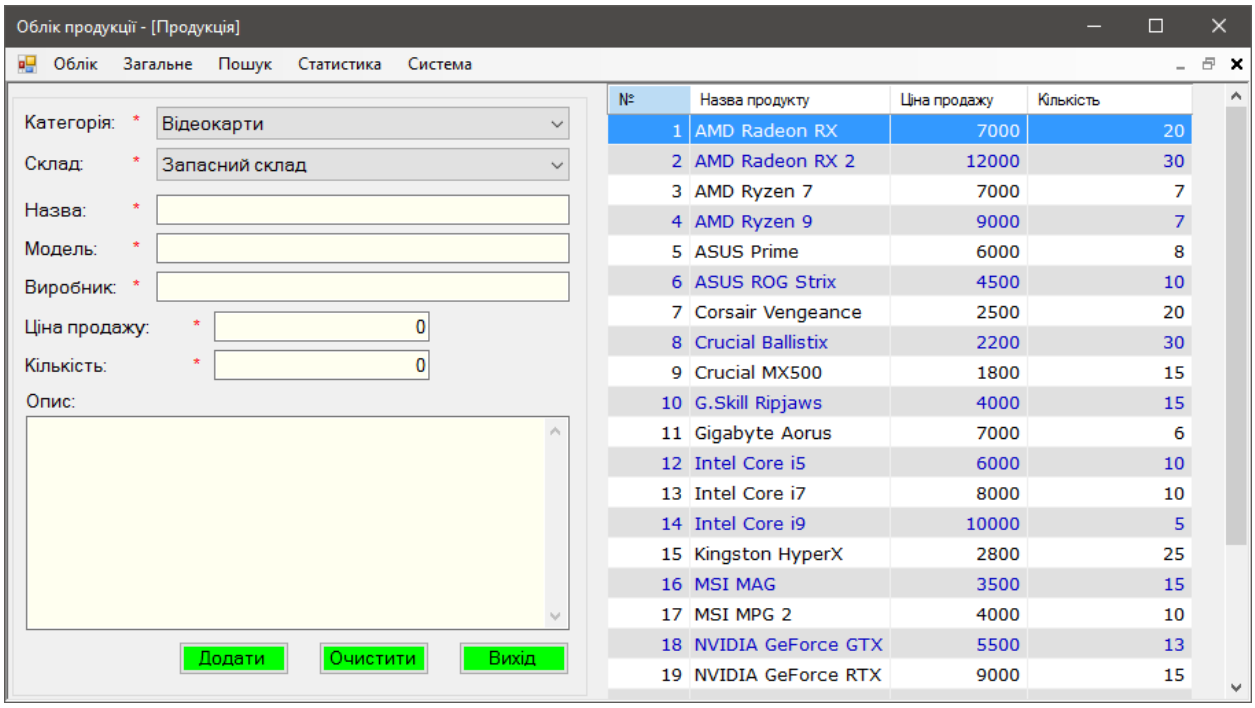


Рисунок 5.12 – Результат тестування форми Продукція

У табл. 5.2 представлено тестові сценарії, які охоплюють основні функції форми.

Таблиця 5.2 – Тестові сценарії для форми Умови постачання

№	Тестовий сценарій	Вхідні дані	Очікуваний результат
1	2	3	4
1	Додавання нової умови постачання	Постачальник: "ТОВ Комплекс", Продукція: "AMD Radeon RX", Мінімальний об'єм: "10", Максимальний об'єм: "50", Ціна поставки: "6000", Умови доставки: "Доставка протягом 3 днів"	Умова постачання успішно додана до списку, новий запис з'являється у таблиці

Продовження табл. 5.2

1	2	3	4
2	Додавання умови постачання з порожніми обов'язковими полями	Постачальник: "", Продукція: "", Мінімальний об'єм: "", Максимальний об'єм: "", Ціна поставки: ""	Повідомлення про помилку, умова постачання не додається
3	Редагування існуючої умови постачання	Вибір умови постачання зі списку, зміна ціни поставки на "7000"	Ціна поставки у списку змінюється на "7000"
4	Видалення умови постачання	Вибір умови постачання зі списку, натискання кнопки "Видалити"	Умова постачання успішно видалена зі списку, запис зникає з таблиці
5	Перевірка відображення коректної кількості продукції	Вибір умови постачання зі списку	Відображаються правильні дані обраної умови постачання
6	Очищення форми	Натискання кнопки "Очистити"	Всі поля форми очищуються

На рис. 5.13 представлено результат виконання тестових сценаріїв.

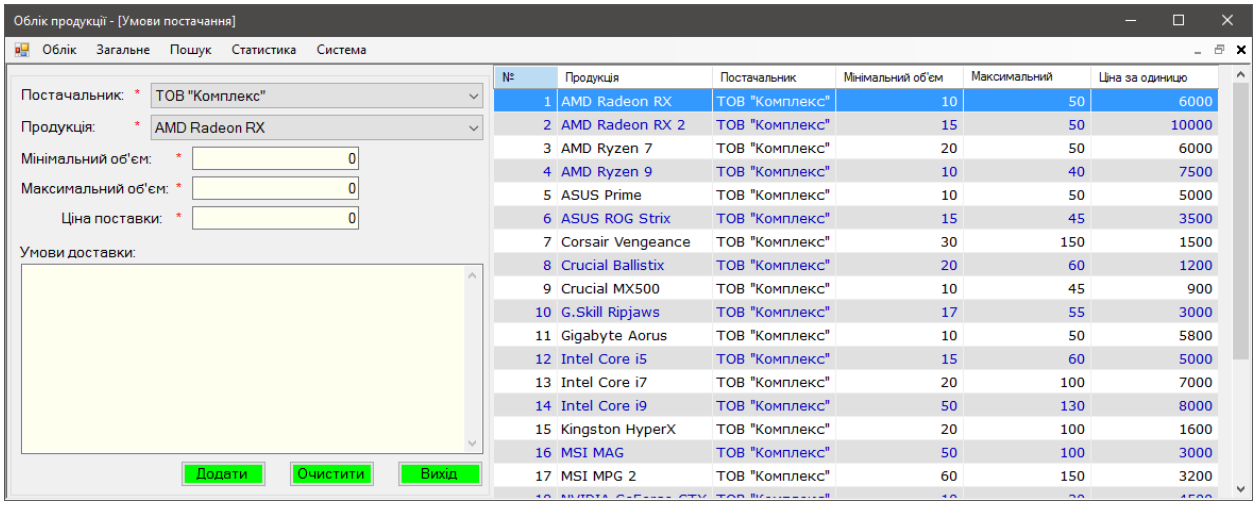


Рисунок 5.13 – Результат тестування форми Умови постачання

У табл. 5.3 представлені тестові сценарії, які охоплюють основні функції форми Купівля, включаючи додавання, видалення та пошук товарів, перевірку коректності відображення деталей товарів та загальної суми, очищення форми і завершення замовлення.

Таблиця 5.3 – Тестові сценарії для форми Купівля

№	Тестовий сценарій	Вхідні дані	Очікуваний результат
1	Додавання товару до замовлення	Категорія: "Жорсткі диски", Товар: "Toshiba P300", Кількість: "1"	Товар успішно додано до списку замовлень, новий запис з'являється у таблиці
2	Видалення товару із замовлення	Вибір товару зі списку, натискання кнопки "Видалити"	Товар успішно видалений зі списку замовлень, запис зникає з таблиці
3	Перевірка відображення деталей обраного товару	Вибір товару зі списку	Відображаються коректні деталі обраного товару у відповідних полях
4	Перевірка відображення правильної загальної суми	Додавання декількох товарів до замовлення	Загальна сума коректно обчислюється і відображається в полі "Загальна сума"
5	Очищення форми	Натискання кнопки "Очистити"	Всі поля форми очищуються
6	Завершення замовлення	Натискання кнопки "Замовити"	Замовлення успішно завершено, відображається повідомлення про успіх та форма закривається
7	Додавання товару з негативною кількістю або ціною	Категорія: "Жорсткі диски", Товар: "Toshiba P300", Кількість: "-1", Ціна продажу: "-1100"	Повідомлення про помилку «Вкажіть адекватну кількість», товар не додається

На рис. 5.14 представлено результат виконання тестових сценаріїв для форми Купівля.

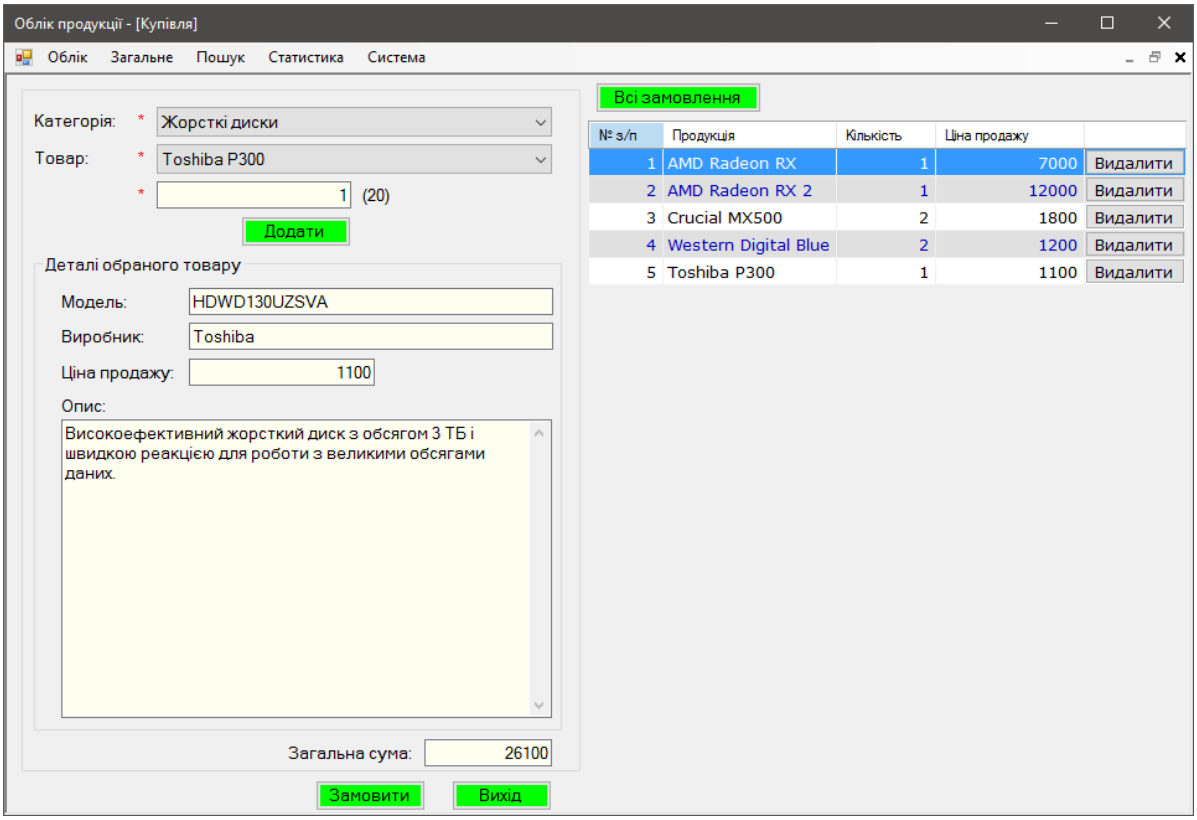


Рисунок 5.14 – Результат тестування форми Купівля

Тестові сценарії у табл. 5.4 охоплюють основні функції форми Пошук продукції, включаючи пошук за різними критеріями, перевірку відображення коректної інформації та обробку помилок.

Таблиця 5.4 – Тестові сценарії для форми Пошук продукції

№	Тестовий сценарій	Вхідні дані	Очікуваний результат
1	Пошук продукції за назвою	Назва: "Intel"	Відображаються всі продукти, що містять "Intel" у назві
2	Пошук продукції за моделлю	Модель: "i7"	Відображаються всі продукти, що містять "i7" у назві моделі
3	Пошук продукції за виробником	Виробник: "Intel"	Відображаються всі продукти, що мають виробника "Intel"

Продовження табл 5.4

4	Пошук продукції за назвою, що не існує	Назва: "NonExistingProduct"	Повідомлення про відсутність результатів пошуку
5	Пошук продукції за частиною назви	Назва: "Core"	Відображаються всі продукти, що містять "Core" у назві
6	Пошук продукції за складом	Склад: "Запасний склад"	Відображаються всі продукти, що зберігаються на "Запасний склад"
7	Пошук продукції за ціною продажу	Ціна продажу: "6000"	Відображаються всі продукти з ціною продажу "6000"
8	Пошук продукції за кількістю	Кількість: "10"	Відображаються всі продукти, кількість яких дорівнює "10"
9	Перевірка відображення коректної інформації про продукцію	Назва: "Intel"	Відображаються деталі кожного продукту у списку результатів пошуку
10	Пошук продукції без введення жодних критеріїв	Усі поля порожні	Повідомлення про необхідність вказати хоча б один критерій для пошуку

На рис. 5.15 представлено результат виконання тестових сценаріїв.

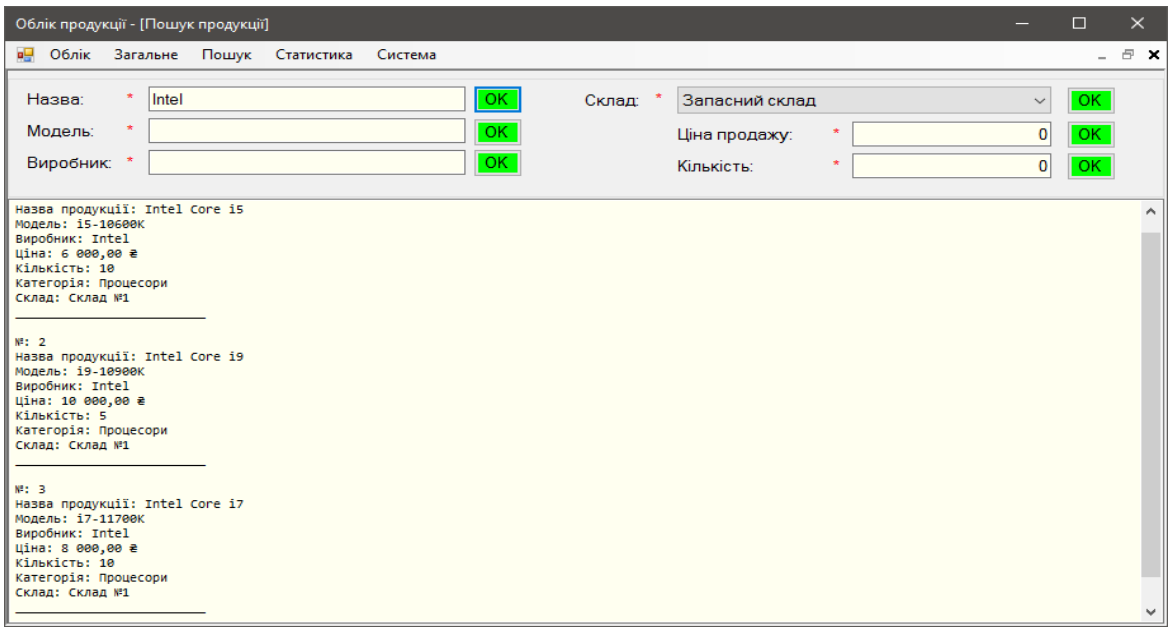


Рисунок 5.15 – Результат тестування форми Пошук продукції

Модульне тестування є критично важливим етапом розробки програмного забезпечення, що забезпечує перевірку правильності роботи окремих модулів або компонентів системи. Воно дозволяє виявити та виправити помилки на ранніх стадіях розробки, що значно знижує витрати на подальше виправлення дефектів. Модульне тестування також сприяє підтримці високої якості коду та забезпечує, що кожен окремий модуль працює коректно відповідно до специфікацій. Використання автоматизованих тестів, таких як MS Test, дозволяє автоматично виконувати тести під час змін у коді, що сприяє швидкому виявленню помилок.

На рис. 5.16 представлено результати модульного тестування (MS Test), які демонструють успішне проходження тестів для різних модулів системи.

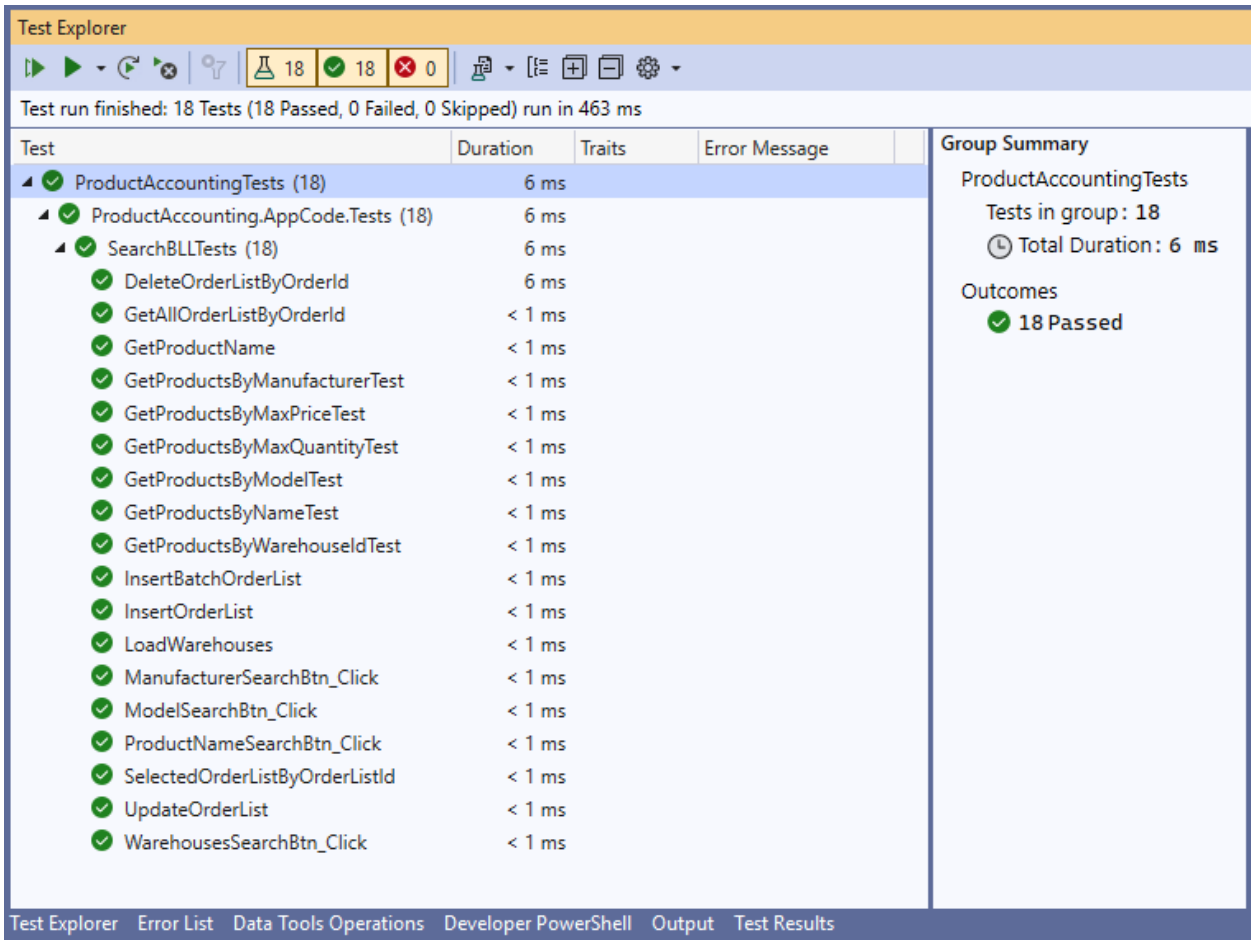


Рисунок 5.16 – Результат тестування

Проведене функціональне та модульне тестування показало, що система працює відповідно до визначених вимог і специфікацій. Тести підтвердили коректність роботи основних функцій, що забезпечує надійність та стабільність програмного забезпечення.

5.3 Інструкція використання системи

Інструкція використання системи надає детальні вказівки щодо роботи з програмним забезпеченням, допомагаючи користувачам ефективно взаємодіяти з усіма його функціями. Цей розділ містить покрокові рекомендації щодо виконання основних операцій, таких як авторизація, пошук продукції, додавання замовлень та управління складськими запасами. Метою є забезпечення зрозумілого і зручного користування системою для всіх категорій користувачів.

На рис. 5.17 представлено форму авторизації користувача, яка є першою взаємодією користувача із системою. Ця форма дозволяє здійснити вхід до системи, надаючи доступ до функціоналу відповідно до ролі користувача: директор магазину, робітник складу, менеджер або покупець. Кожна роль має свої специфічні права та можливості, що забезпечує належний рівень безпеки та доступу до даних.

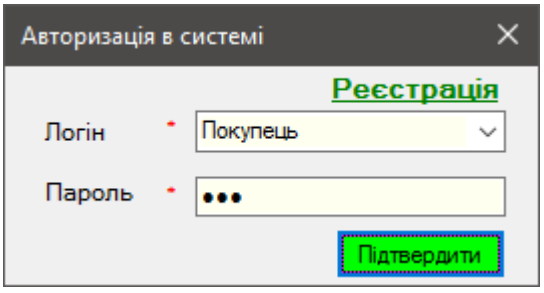


Рисунок 5.17 – Авторизація користувача

Форма, представлена на рис. 5.18, призначена для створення накладних користувачем з роллю Покупець та забезпечує введення, оновлення та відображення інформації про замовлення. У лівій частині екрану знаходиться форма для введення даних, що включає поля для вибору категорії, товару та кількості. Користувач обирає категорію зі спадного списку, потім конкретний товар та вводить кількість. Для додавання товару до замовлення необхідно натиснути кнопку Додати.

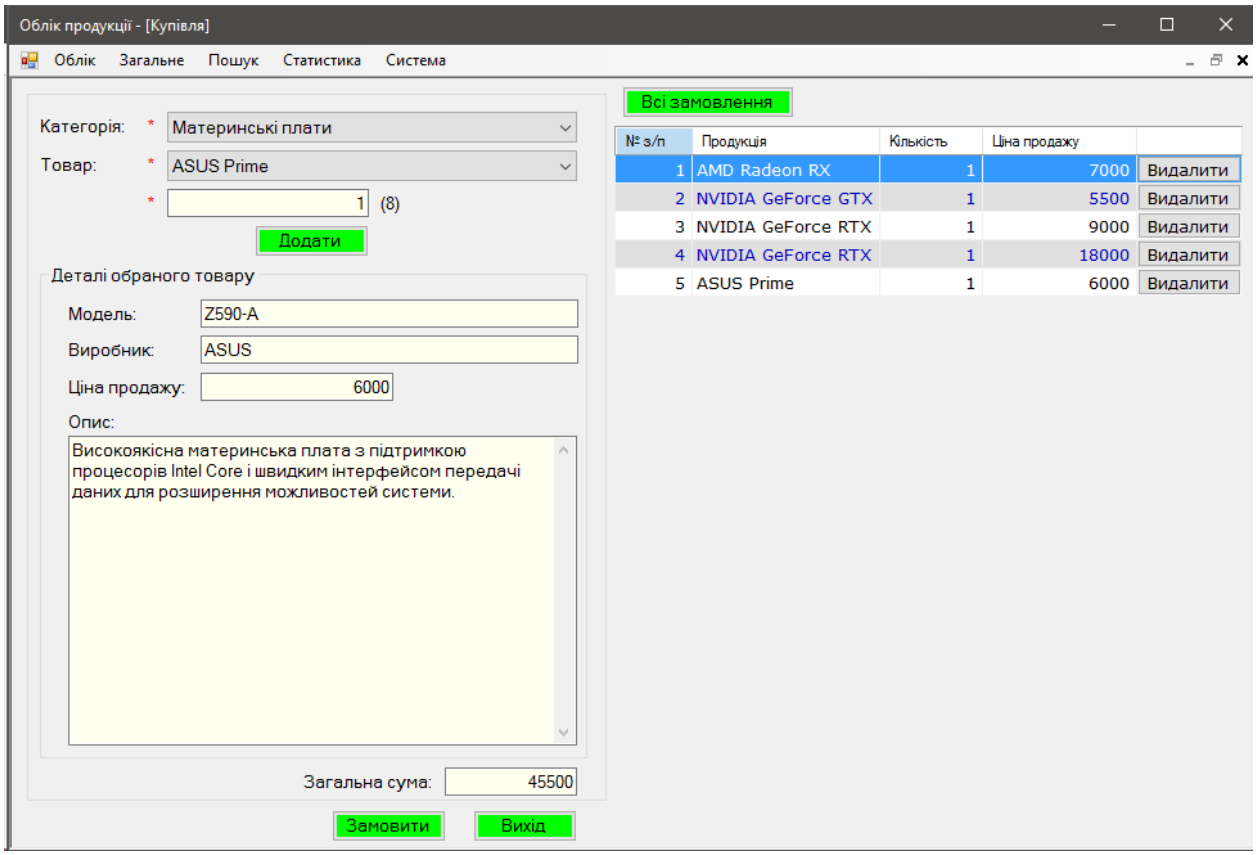


Рисунок 5.18 – Формування накладних

Під цими полями автоматично заповнюються деталі обраного товару, такі як модель, виробник, ціна продажу та опис. Права частина екрану містить таблицю з поточними замовленнями, де відображаються номер, назва продукції, кількість та ціна продажу. Користувач може переглядати всі додані до замовлення товари. Для видалення товару зі списку замовлень необхідно натиснути кнопку Видалити поруч із відповідним записом. У

нижній частині екрану відображається загальна сума замовлення. Після введення всіх товарів користувач натискає кнопку Замовити для підтвердження замовлення.

При натисканні кнопки Всі замовлення відкривається форма для перегляду всіх замовлень користувача (рис. 5.19). У лівій частині екрану розташована таблиця з основною інформацією про замовлення, включаючи номер замовлення, дату замовлення та статус виконання (наприклад, Виконано або Не виконано).

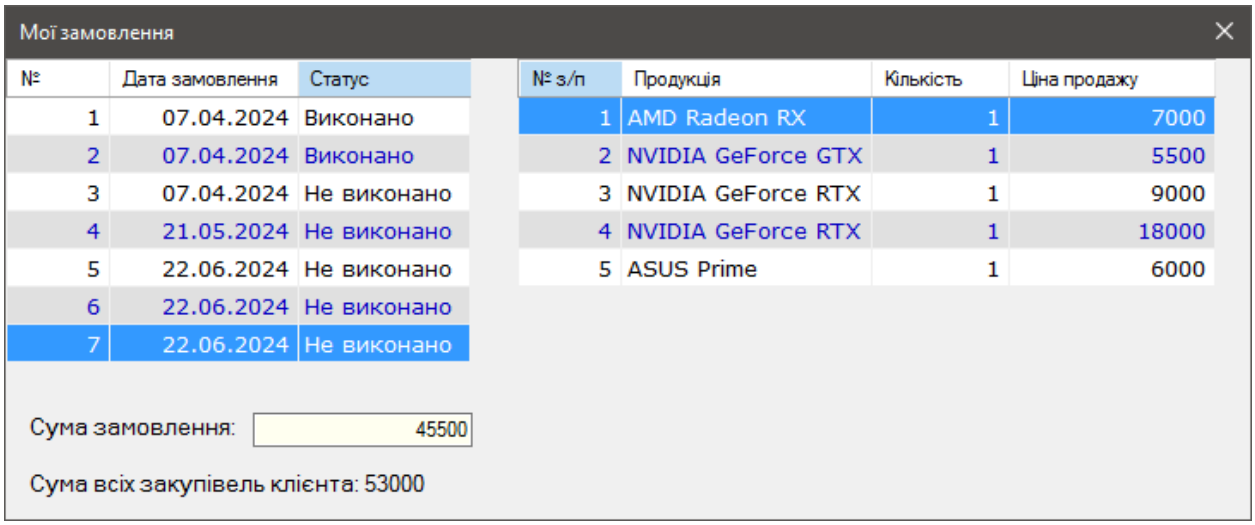


Рисунок 5.19 – Перегляд всіх замовлень

Права частина екрану містить таблицю з деталями обраного замовлення. Тут відображаються номер позиції, назва продукції, кількість та ціна продажу кожного товару, включеного до замовлення.

Під таблицями розташовані два поля, які відображають фінансову інформацію: сума обраного замовлення та сума всіх закупівель клієнта. Це дозволяє користувачу швидко отримати інформацію про вартість конкретного замовлення та загальну суму витрат на закупівлі.

Форма для підтвердження купівлі, представлена на рис. 5.20, призначена для користувача з роллю Продавець і дозволяє обробляти замовлення клієнтів. У лівій частині екрану розташована таблиця з основною

інформацією про замовлення, включаючи номер замовлення, дату замовлення та ім'я клієнта.

Права частина екрану містить таблицю з деталями обраного замовлення, де відображаються номер позиції, назва продукції, кількість та ціна продажу кожного товару, включеного до замовлення.

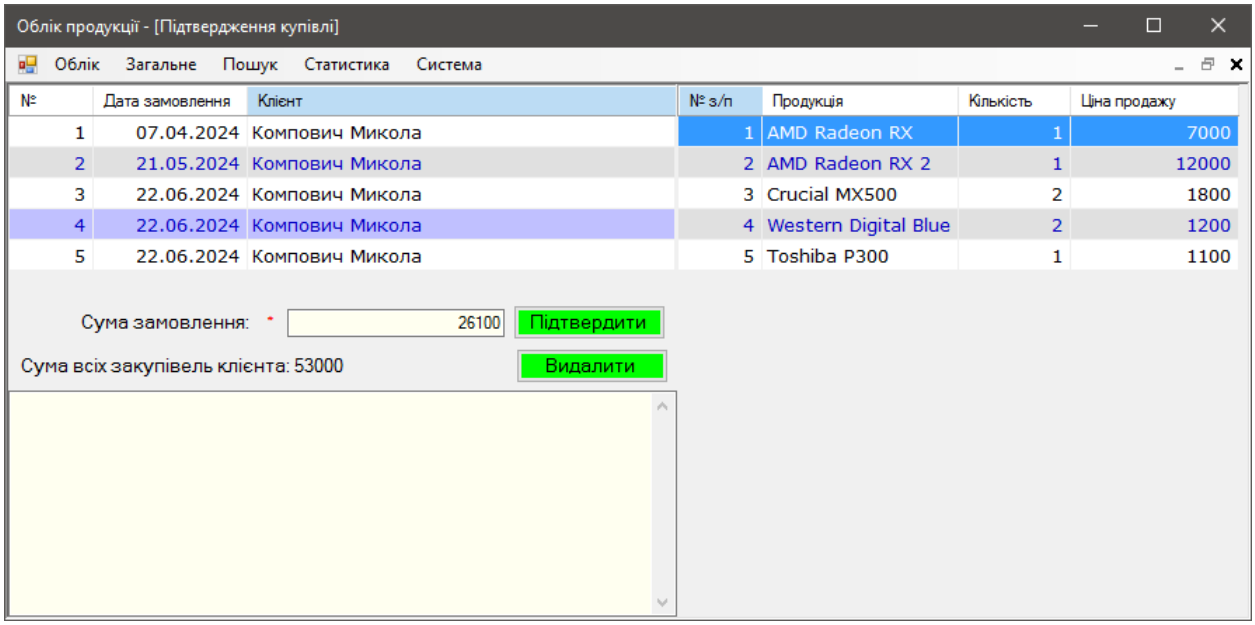


Рисунок 5.20 – Підтвердження купівлі

Під таблицями знаходяться поля для фінансової інформації: сума обраного замовлення та сума всіх закупівель клієнта. Це дозволяє продавцю швидко отримати інформацію про вартість конкретного замовлення та загальну суму витрат клієнта. Для підтвердження замовлення продавець натискає кнопку Підтвердити, що змінює статус замовлення на Виконано. Якщо потрібно видалити замовлення, натискається кнопка Видалити, і відповідний запис зникає з таблиці. Таким чином, ця форма забезпечує зручний інтерфейс для підтвердження та управління замовленнями клієнтів, забезпечуючи належний рівень обслуговування.

Форма для замовлення товару, представлена на рис. 5.21, призначена для користувача з роллю Продавець і дозволяє створювати накладні на замовлення продукції від постачальників. У лівій частині екрану розташована

форма для введення даних, що включає поля для вибору категорії, товару та кількості. Користувач обирає категорію зі спадного списку, потім конкретний товар та вводить кількість. Для додавання товару до замовлення необхідно натиснути кнопку Додати.

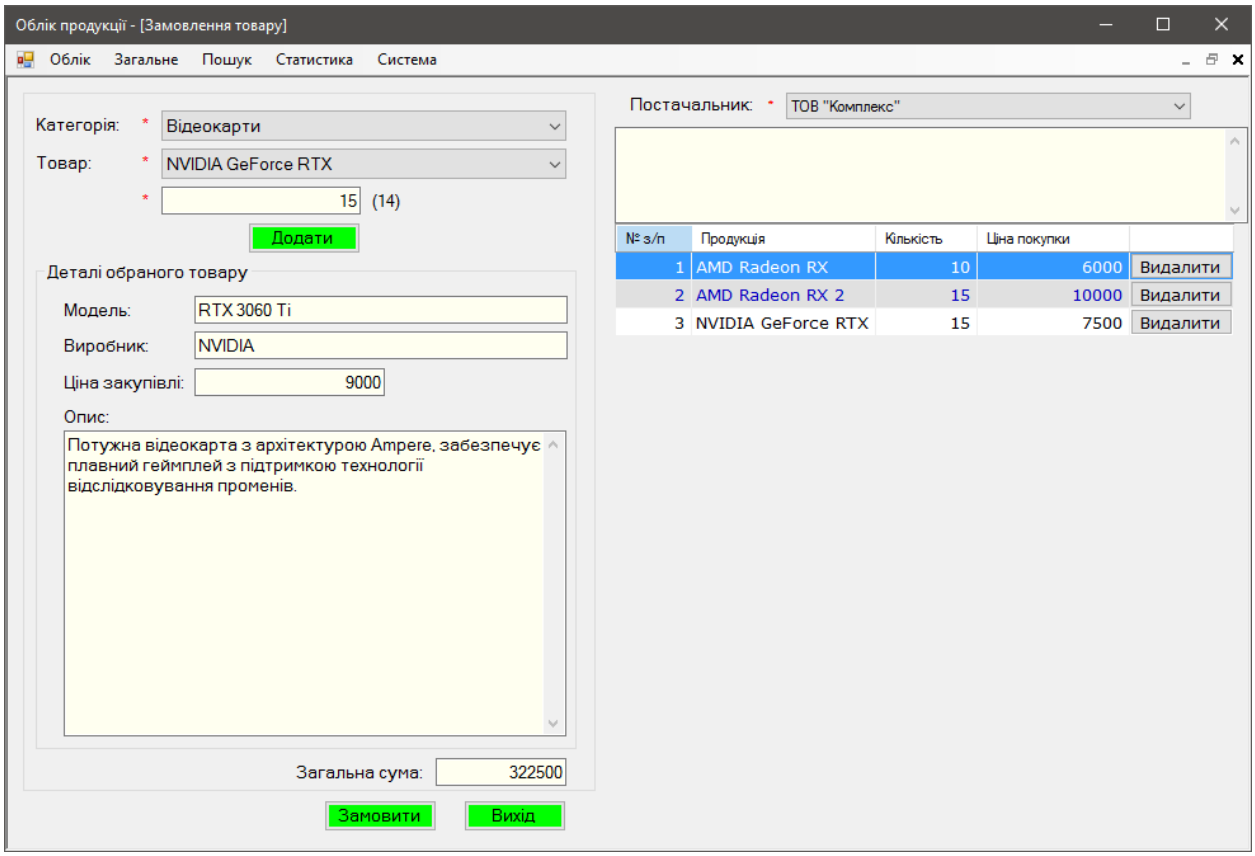


Рисунок 5.21 – Замовлення нової продукції

Під цими полями автоматично заповнюються деталі обраного товару, такі як модель, виробник, ціна закупівлі та опис. Права частина екрану містить таблицю з поточними замовленнями, де відображаються номер позиції, назва продукції, кількість та ціна покупки кожного товару, включеного до замовлення. Це дозволяє продавцю переглядати всі товари, додані до замовлення, та контролювати їх кількість і вартість.

Під таблицею розташовані поля для фінансової інформації: загальна сума замовлення. Це дозволяє продавцю швидко отримати інформацію про загальну вартість замовлення. Для створення накладної на замовлення

продукції продавець натискає кнопку Замовити, що завершує процес замовлення. Якщо потрібно видалити товар зі списку замовлень, натискається кнопка Видалити поруч із відповідним записом.

Форма для перевірки надходження продукції на склад, представлена на рис. 5.22, призначена для користувача з роллю Робітник складу і дозволяє обробляти замовлення продукції, що надійшла від постачальників. У лівій частині екрану розташована таблиця з основною інформацією про замовлення, включаючи номер замовлення, дату замовлення та назву постачальника.

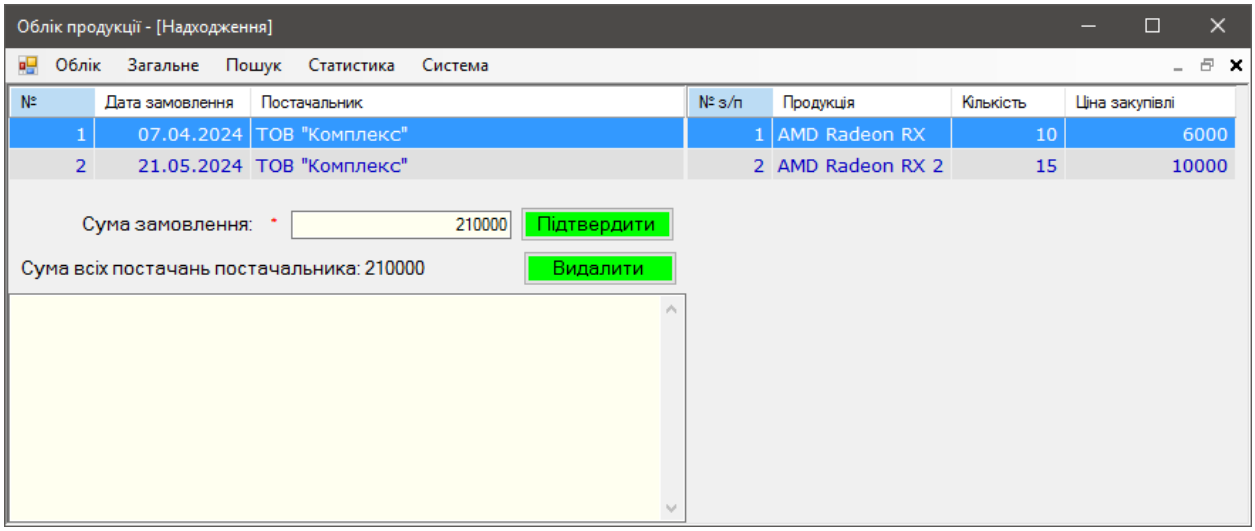


Рисунок 5.22 – Надходження продукції

Права частина екрану містить таблицю з деталями обраного замовлення, де відображаються номер позиції, назва продукції, кількість та ціна закупівлі кожного товару, включеного до замовлення. Це дозволяє робітнику складу переглядати всі товари, що надійшли, та перевіряти їх кількість і вартість. Під таблицями розташовані поля для фінансової інформації: сума обраного замовлення та сума всіх постачань від конкретного постачальника. Це дозволяє робітнику складу швидко отримати інформацію про загальну вартість поточного замовлення та суму всіх постачань від даного постачальника.

Для підтвердження надходження продукції на склад робітник натискає кнопку Підтвердити, що змінює статус замовлення на Прийнято. Якщо потрібно видалити запис про надходження, натискається кнопка Видалити, і відповідний запис зникає з таблиці.

Форма для управління складською інформацією, яка представлена на рис. 5.23, призначена для користувачів з ролями Робітник складу та Директор. Вона дозволяє додавати, редагувати та видаляти записи про склади.

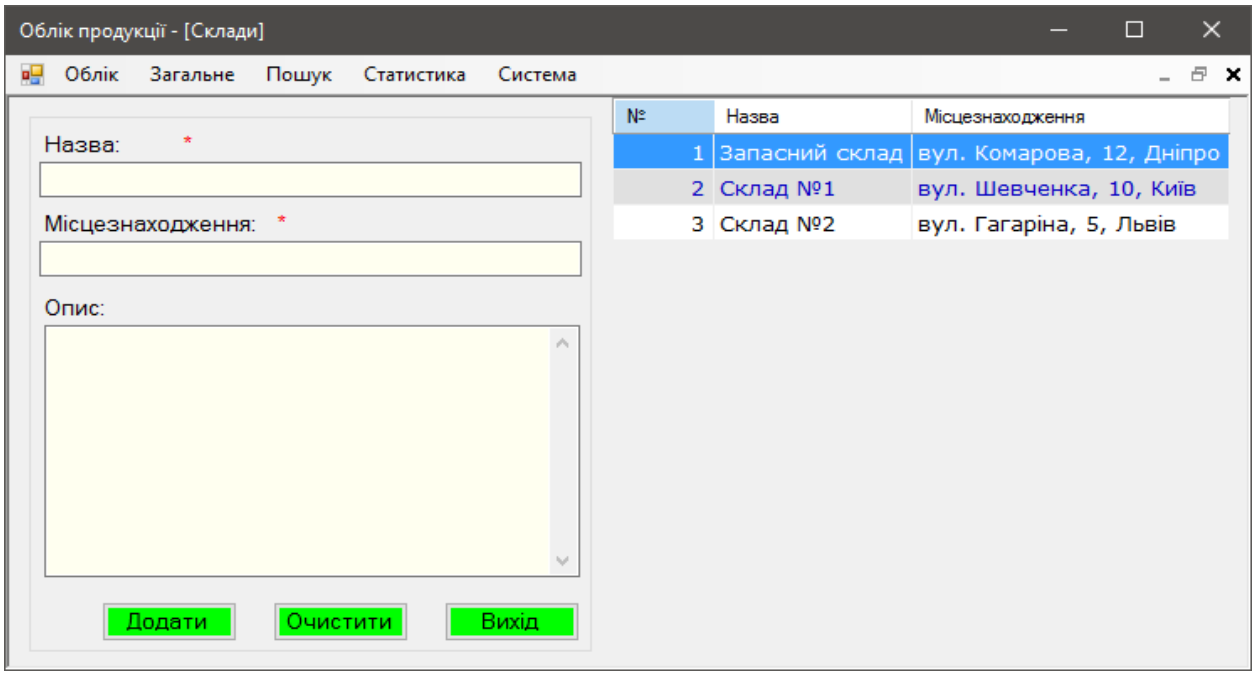


Рисунок 5.23 – Керування інформацією про склади

У лівій частині екрану розташована форма для введення даних, що включає поля для назви складу, місцезнаходження та опису. Користувач повинен ввести назву складу у відповідне поле, вказати місцезнаходження та за потреби додати опис. Після введення даних користувач натискає кнопку Додати для збереження інформації про новий склад. Права частина екрану містить таблицю зі списком складів, де відображаються їх номер, назва та місцезнаходження. Користувач може переглядати всі наявні записи про склади та за необхідності редагувати або видаляти їх.

Форма для керування інформацією про продукцію, представлена на рис. 5.24, призначена для користувача з роллю Директор і дозволяє додавати, редагувати та видаляти записи про продукцію.

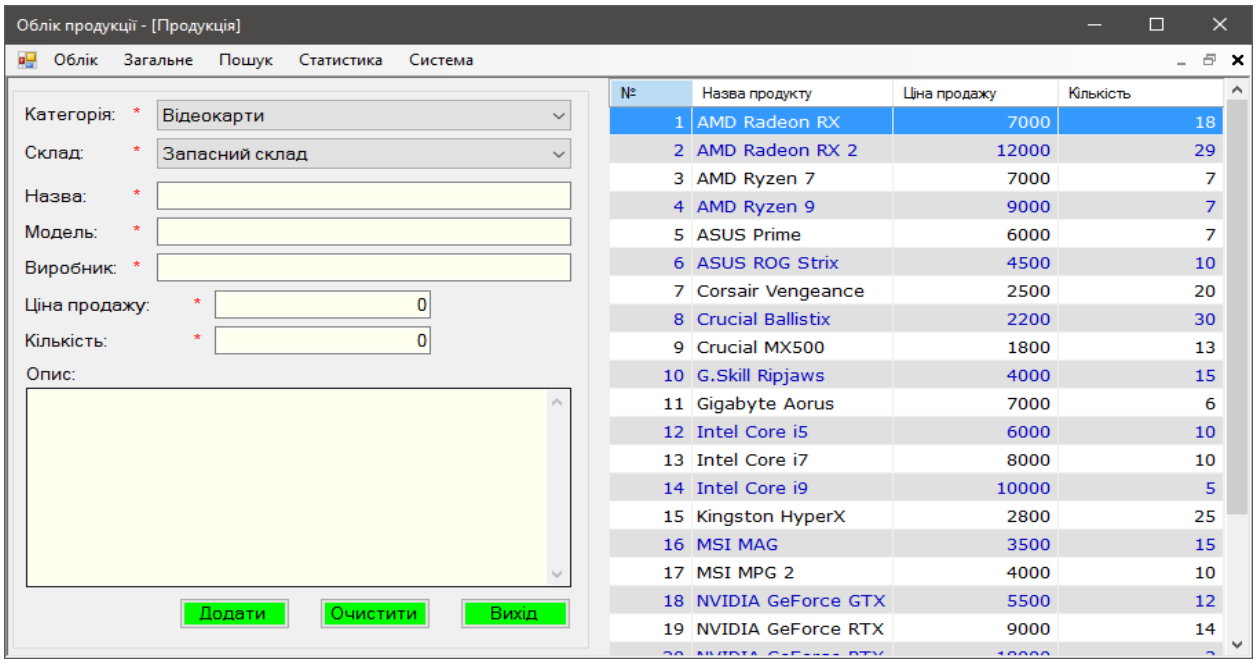


Рисунок 5.24 – Керування інформацією про продукцію

У лівій частині екрану розташована форма для введення даних, яка включає такі поля: категорія, склад, назва, модель, виробник, ціна продажу, кількість та опис. Користувач повинен вибрати категорію продукції та склад зі спадного списку, після чого заповнити інші обов'язкові поля, такі як назва, модель, виробник, ціна продажу та кількість. Поле для опису є необов'язковим, але може містити додаткову інформацію про товар. Після введення всіх необхідних даних користувач натискає кнопку Додати для збереження нової продукції у списку.

Форма для керування інформацією про умови постачання продукції, представлена на рис. 5.25, призначена для користувача з роллю Директор і дозволяє додавати, редагувати та видаляти записи про умови постачання. У лівій частині екрану розташована форма для введення даних, яка включає такі поля: постачальник, продукція, мінімальний об'єм, максимальний об'єм,

ціна поставки та умови доставки. Користувач повинен вибрати постачальника та продукцію зі спадних списків, після чого ввести мінімальний та максимальний об'єм поставки, ціну та за потреби додати умови доставки. Після введення всіх необхідних даних користувач натискає кнопку Додати для збереження нової умови постачання у списку.

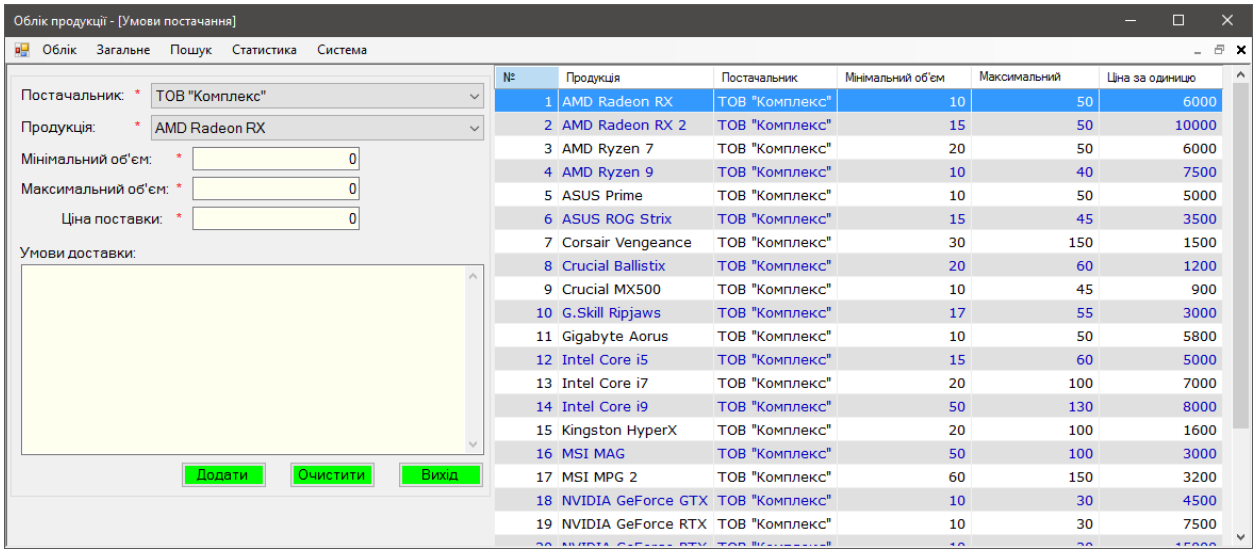


Рисунок 5.25 – Керування інформацією про умови постачання продукції

Права частина екрану містить таблицю зі списком умов постачання, де відображаються номер, назва продукції, постачальник, мінімальний та максимальний об'єм, а також ціна за одиницю. Користувач може переглядати всі наявні записи про умови постачання, редагувати або видаляти їх за необхідності.

Форма для пошуку продукції, представлена на рис. 5.26, доступна для всіх ролей користувачів і дозволяє здійснювати пошук продукції за різними критеріями. У верхній частині екрану розташована форма для введення даних, яка включає поля для введення назви, моделі, виробника, складу, ціни продажу та кількості. Користувачі можуть ввести будь-яке поєднання цих критеріїв для пошуку необхідної продукції.

Поля для введення назви, моделі та виробника дозволяють користувачу здійснювати пошук за текстовими параметрами. Поле для вибору складу зі

спадного списку дозволяє обмежити пошук продукції до певного складу. Поля для введення ціни продажу та кількості дозволяють здійснювати пошук за цими числовими параметрами.

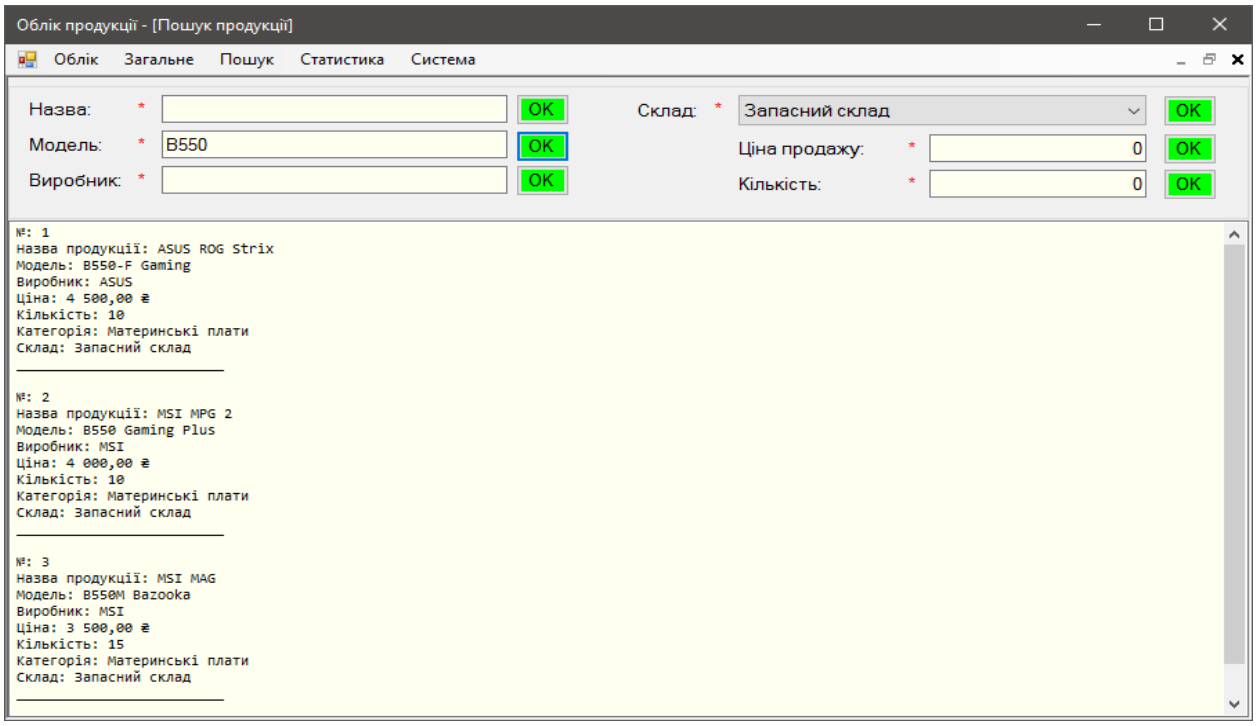


Рисунок 5.26 – Пошук продукції за різними критеріями

Після введення критеріїв пошуку користувач натискає кнопку OK поруч із кожним полем для підтвердження введених даних. Результати пошуку відображаються в нижній частині екрану у вигляді списку, де для кожного знайденого товару показується його номер, назва, модель, виробник, ціна продажу, кількість, категорія та склад.

Користувачі можуть переглядати детальну інформацію про кожний знайдений товар, що дозволяє їм швидко знайти потрібну продукцію відповідно до заданих критеріїв. Ця форма забезпечує зручний та ефективний інтерфейс для пошуку продукції, який можна використовувати для різноманітних потреб усіх ролей користувачів системи.

Форма для формування звітності по проданій продукції, представлена на рис. 5.27, призначена для користувача з роллю Директор і дозволяє

створювати звіти про продажі за певний період. У верхній частині екрану розташовані поля для введення початку та кінця періоду звітності. Користувач може вибрати відповідні дати зі спадного календаря для визначення часових рамок звіту.

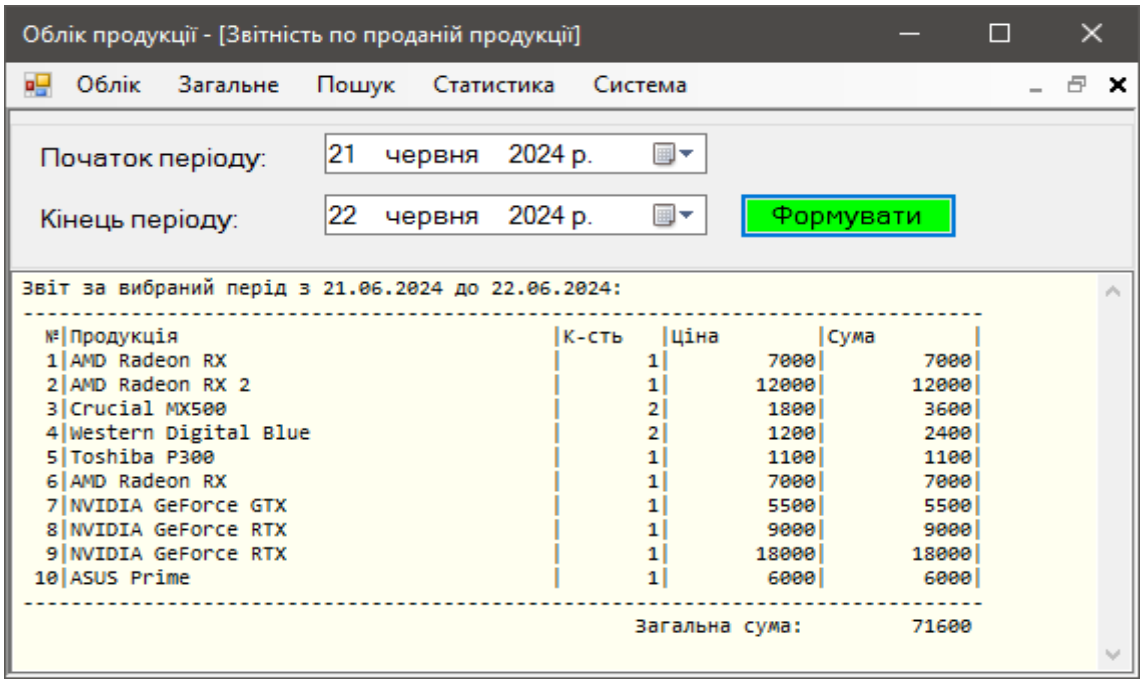


Рисунок 5.27 – Формування звітності по проданій продукції

Після вибору дат користувач натискає кнопку Формувати, щоб згенерувати звіт. У нижній частині екрану відображається детальний звіт, що містить інформацію про продану продукцію за вибраний період. Звіт включає такі дані, як номер позиції, назва продукції, кількість проданих одиниць, ціна за одиницю та загальна сума для кожної позиції. Наприкінці звіту відображається загальна сума всіх продажів за вказаний період, що дозволяє директору отримати повну картину фінансових результатів діяльності.

Ця форма забезпечує зручний та ефективний інтерфейс для генерації звітів по проданій продукції, що сприяє прийняттю обґрунтованих управлінських рішень на основі точних даних про продажі.

Форма для формування звітності по надходженню продукції, представлена на рис. 5.28, призначена для користувача з роллю Директор і

дозволяє створювати звіти про надходження продукції за певний період. У верхній частині екрану розташовані поля для введення початку та кінця періоду звітності. Користувач може вибрати відповідні дати зі спадного календаря для визначення часових рамок звіту.

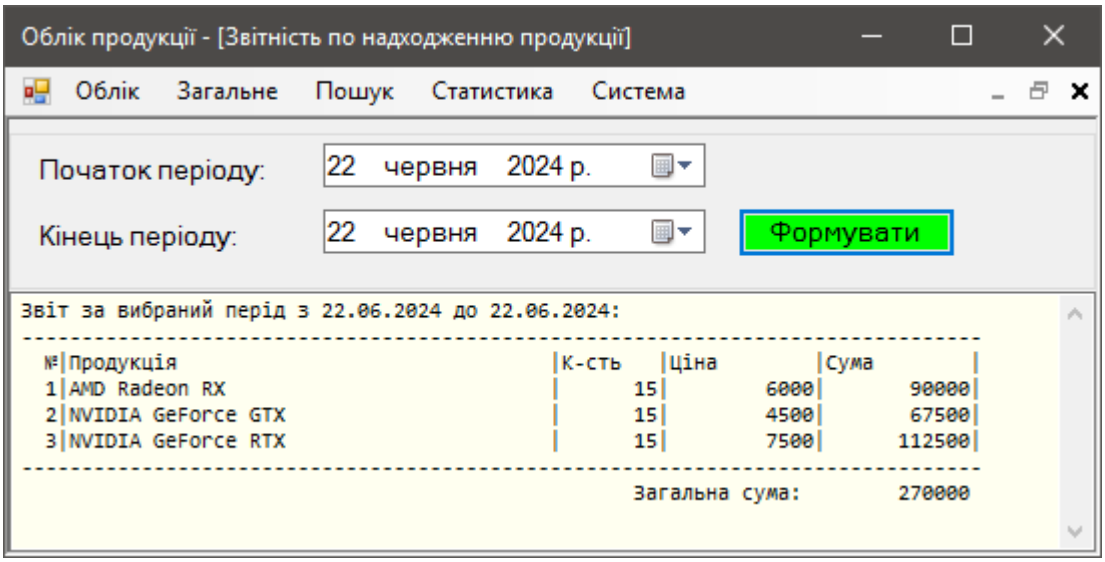


Рисунок 5.28 – Формування звітності по надходженню продукції

Для виходу із програми користувачу системи необхідно перейти по меню Управління та обрати опцію Вихід.

5.4 Охорона праці

Під час виконання реалізації, тестування та реалізації інформаційно пошукової системи було дотримано вимог охорони праці та безпеки, недопущено травмувань та нещастних випадків під час написання.

Робота з даною системою обліку продукції (СОП) не була пов'язана з безпосереднім впливом шкідливих та небезпечних факторів виробничого середовища. Проте, при роботі з комп'ютером та іншими електронними пристроями, які використовуються для доступу до СОП, було дотримано правил охорони праці.

Під час написання та тестування даної роботи, необхідно було перевіряти справність обладнання та усіх підключених до нього компонентів.

Було дотримано санітарно- гігієнічних норм – підтримування робочого місця в чистому вигляді, організація зручного місця для роботи та перерівів кожні декілька годин для відпочинку очей.

Також використовувалося лише якісне і сертифіковане обладнання, яке перевірялося перед кожним етапом праці аби унеможливити отримання травм.

Дотримання всіх наведених вимог охорони праці під час розробки, та користування інформаційно пошуковою системою дозволить забезпечити безпечні умови праці та уникнути можливих ризиків [30].

ВИСНОВКИ

Дана кваліфікаційна робота присвячена розробці інформаційно-пошукової системи обліку продукції, яка забезпечує ефективне управління складськими операціями та покращує процеси обліку продукції. Система дозволяє здійснювати комплексне управління інформацією про продукцію, включаючи приймання, зберігання, переміщення та відвантаження товарів. Ролі користувачів системи, такі як директор підприємства, робітник складу, менеджер та покупець, мають доступ до різного функціоналу, що відповідає їхнім завданням та потребам. Розроблена система забезпечує зручний та ефективний інтерфейс для виконання всіх необхідних операцій з обліку продукції.

У першому розділі була детально розглянута теоретична база обліку продукції на складі, включаючи основні процеси такі як приймання, зберігання, переміщення та відвантаження продукції. Також було проведено аналіз аналогічних програмних рішень (Fishbowl Warehouse, NetSuite WMS, 3PL Central WMS), що дозволило виявити їхні переваги та недоліки, обґрунтувавши необхідність розробки нового рішення.

Другий розділ розкриває вимоги та потреби користувачів системи обліку продукції, визначаючи функціональні та нефункціональні вимоги, а також ролі та цілі учасників системи, таких як директор підприємства, робітник складу, менеджер та покупець. Було описано варіанти використання системи та побудовано діаграми прецедентів для кожної ролі.

Третій розділ присвячений огляду та вибору інструментів розробки системи. Після аналізу різних платформ (.NET Framework, Java EE, Node.js), середовищ розробки (Visual Studio, JetBrains Rider, Visual Studio Code) та систем управління базами даних (MS SQL Server, MySQL, Oracle) було обрано оптимальний набір інструментів для реалізації системи – .NET Framework, Visual Studio та MySQL.

У четвертому розділі проведено проектування системи обліку продукції та її бази даних. Було створено діаграми послідовності для процесу формування накладної та процесу пошуку інформації, а також діаграми активності для процесу додавання нової продукції та редагування існуючих даних. Створено логічну модель бази даних, яка включає таблиці сутностей та їх атрибути, а також фізичну модель бази даних, яка відповідає нормалізаційним формам.

П’ятий розділ описує реалізацію коду системи, функціональне та модульне тестування, а також інструкцію з використання системи. Було реалізовано необхідні класи і методи для роботи з базою даних, проведено тестування функціональних можливостей та модулів системи, що забезпечило високу надійність та стабільність програмного забезпечення. Інструкція користувача надає детальні вказівки щодо роботи з основними функціями системи, такими як авторизація, пошук продукції, формування замовлень та генерація звітів.

Проведена робота демонструє застосування сучасних підходів до розробки інформаційних систем та успішну інтеграцію різних технологій для створення ефективної системи обліку продукції. Отримані результати дозволяють значно підвищити ефективність управління складськими операціями та забезпечити зручний інтерфейс для користувачів різних ролей.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАНЬ

1. Огоновська Ірина, Нестор Думанський. Автоматизація складських процесів на підприємстві. Інформація, комунікація, суспільство 2018, – 86 с.

2. Філь Н. Ю.; Гавріш А. В. Автоматизація бізнес процесів складського обліку. 2020, – 32 с.

3. ДСТУ 3008-15. Документація. Звіти у сфері науки та техніки. структура та правила оформлення. Введ. 2015-06-22. К. Держстандарт України, 2017. - 29 с.

4. Методичні вказівки з підготовки кваліфікаційної роботи бакалавра для здобувачів першого (бакалаврського) рівня вищої освіти спеціальності 151 Автоматизація та комп’ютерно-інтегровані технології освітньої програми «Системна інженерія» / Упоряд.: І.Ш. Невлюдов, О.М. Цимбал, О.В. Токарєва, А.І. Бронніков. Харків: ХНУРЕ, 2022. - 66 с.

5. Положення про академічну доброчесність [Електронний ресурс]: Наказ ХНУРЕ від 02 лютого 2021 р. No 50. – Режим доступу: https://nure.ua/wpcontent/uploads/Main_Docs_NURE/polozhennja-pro-akademi chnu-dobrochesnist.pdf.

6. Баланюк І. Ф. Григорів О. О. Іванюк Т.Л. Організація обліку і контролю готової продукції підприємства. Інноваційна економіка, 2020, 1-2: 163 с.

7. Осадча Г.Г. Темчишина, Ю. Л. Насідзе Д. Облік готової продукції та його науково-методичні аспекти на сучасному промисловому підприємстві. 2017. – 34 с.

8. Мардус Н. Ю. Основні аспекти обліку і аудиту готової продукції та аналізу її реалізації. Стратегічно-інноваційний розвиток суб'єктів економічної системи в умовах глобалізації : зб. тез наук. робіт 4-ї Міжнар. наук.-практ. інтернет-конф., 5-7 листопада 2019 р. – 88 с.

9. Fishbowl Warehouse URL: https://www.explorewms.com/fishbowl-warehouse-wms-software-profile.html (дата звернення 21.06.2024).

10. Fishbowl Inventory - #1 Inventory Management Software URL:https://www.fishbowlinventory.com/ (дата звернення 21.06.2024).

11. Benefits of using inventory programs URL:https://www.fishbowlinventory.com/blog/benefits-of-using-inventory-programs (дата звернення 21.06.2024).

12. NetSuite Warehouse Management System (WMS) URL: https://www.netsuite.com/portal/products/erp/warehouse-fulfillment/wms.shtml (дата звернення 21.06.2024).

13. NetSuite Warehouse Management System URL: https://www.netsuite.com/portal/assets/pdf/ds-netsuite-wms.pdf (дата звернення 21.06.2024).

14. The benefits of netsuite wms for distributors - Blog URL: https://blog.worldsynergy.com/benefits-netsuite-wms/ (дата звернення 21.06.2024).

15. 3PL WMS https: URL: //www.explorewms.com/3pl-wms.html (дата звернення 21.06.2024).

16. 3PL WMS software: Why Choose a Custom SaaS WMS URL: https://acropolium.com/blog/3pl-warehouse-management-system-why-choose-a-custom-saas-wms-for-your-warehouse/ (дата звернення 21.06.2024).

17. Top Benefits of a 3PL Warehouse Management System - GoBolt URL: https://www.gobolt.com/blog/3pl-warehouse-management-system/ (дата звернення 21.06.2024).

18. Christner Carl Henning; Strömsten Torkel. Scientists, venture capitalists and the stock exchange: The mediating role of accounting in product innovation. Management Accounting Research, 2015. – 59 c.

19. Roberts Chris. Personal computers. In: Communication Technology Update and Fundamentals. Routledge, 2016. – 156 p.

20. Полотай Орест Іванович; БРАЙКО, О. П. Проектування локальної обчислювальної мережі та організація її захисту. 2017. – 215 с.

21. THAI, Thuan L.; LAM, Hoang. . NET framework essentials. " O'Reilly Media, Inc.", 2003. – 94 p.

22. GUPTA, Arun. Java EE 7 Essentials: Enterprise Developer Handbook. " O'Reilly Media, Inc.", 2013. – 318 p.

23. Shah Hezbullah; Soomro Tariq Rahim. Node. js challenges in implementation. Global Journal of Computer Science and Technology, 2017. –83p.

24. Visual Studio; GO, Surface Laptop. Visual Studio. Microsoft. , 2019. URL: https://visualstudio.microsoft.com/ (дата звернення 21.06.2024).

25. JetBrains Rider URL : https://ua.softlist.com.ua/catalog/product-jetbrains-rider/ (дата звернення 21.06.2024).

26. Visual Studio Code. URL: https://code.visualstudio.com/ (дата звернення 21.06.2024).

27. Mistry Ross; Misner Stacia. Introducing Microsoft SQL Server 2014. Microsoft Press, 2014. 512 p.

28. Vaswani Vikram. MySQL database usage & administration. McGraw Hill Professional, 2009. – 369 p.

29. Greenwald Rick; Stackowiak Robert; Stern Jonathan. Oracle essentials: Oracle database 12c. " O'Reilly Media, Inc.", 2013. – 91 p.

30. Методичні вказівки до виконання розділу "Охорона праці" у випускних роботах ОКР "бакалавр" усіх форм навчання / Упоряд.: Б.В. Дзюндзюк, В.А. Айвазов, Т.Є. Стиценко. – Харків: ХНУРЕ, 2012. – 26с.