МІНІСТЕРСТВО ФІНАНСІВ УКРАЇНИ
ДЕРЖАВНИЙ ПОДАТКОВИЙ УНІВЕРСИТЕТ

ФАКУЛЬТЕТ ФІНАНСІВ ТА ЦИФРОВИХ ТЕХНОЛОГІЙ
КАФЕДРА КОМП'ЮТЕРНИХ ТА ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ І СИСТЕМ

КВАЛІФІКАЦІЙНА РОБОТА

на тему: Розробка веб-додатку управління запасами на підприємстві

Здобувача вищої освіти другого (магістерського) рівня,
ІІ курсу, групи КМІ-24-2, спеціальності 122 «Комп'ютерні науки»,
освітня програма «Інформаційні управляючі системи і технології в економіці»

Луб'яніченко Юрій Сергійович
(підпис) (прізвище, ім'я, по батькові)

Науковий керівник Лаговський Володимир, канд. екон. наук, доцент
(підпис) (прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

Керівник освітньо-професійної програми Омельчук Антон Анатолійович, канд. техн. наук
(підпис) (прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

Допущено до захисту за рішенням кафедри протокол від «02» січня 2026 №10

Завідувач кафедри Філоненко Михайло Миколайович, канд. фіз.-мат.наук, доцент
(підпис) (прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

Ірпінь – 2026

Реєстрація № 26 22.01.2026 _____
(дата) (підпис відповідальної особи)

Результати перевірки до захисту 02.01.2026 _____
(до захисту, на доопрацювання) (дата) (підпис)

Результат захисту 4 (добре) 05.02.2026 _____
(оцінка) (дата) (підпис)

Голова комісії: _____ О.С. Дюбанов
(підпис)

Члени комісії: _____ А.Ю. Горбовий
(підпис)

_____ М.М. Філоненко
(підпис)

_____ І.Д. Погорєловська
(підпис)

БЛАНК ОЦІНЮВАННЯ КВАЛІФІКАЦІЙНОЇ РОБОТИ

Об'єкти контролю	Діапазон оцінки в балах	Фактична кількість балів	Примітки	Дата, П.І.Б.: керівника, голови комісії членів комісії,секретаря, (підписи)
ОЦІНЮВАННЯ КВАЛІФІКАЦІЙНОЇ РОБОТИ КОМІСІЄЮ ПО ЗАХИСТУ				
1. Зміст кваліфікаційної роботи (якість теоретичного матеріалу, його зв'язок з практичним, вміння представляти матеріал в логічній послідовності, наявність висновків та пропозицій)	від 0 до 30	25		05.02.2026 Дюбанов О.С. Горбовий А.Ю Філоненко М.М. Погорєловська І.Д. Суховецька Л.П.
2. Дотримання вимог щодо оформлення	від 0 до 10	8		
3. Вчасне виконання роботи на всіх етапах у встановлені кафедрою терміни	від 0 до 5	4		
4. Рецензент	від 0 до 5	4		
РАЗОМ БАЛІВ	від 0 до 50	41		

Оцінювання кваліфікаційної роботи науковим керівником – 39 балів.

ПЕРЕВЕДЕННЯ ОЦІНКИ ЗА 100-БАЛЬНОЮ ШКАЛОЮ ДО НАЦІОНАЛЬНОЇ ШКАЛИ ТА ШКАЛИ ЄКТС

Оцінка за 100-бальною шкалою	Оцінка за шкалоюЄКТС	Оцінка за національною шкалою	Дата	П.І.Б.: голови, членів комісії, секретаря, (підпис)
80	B	4	05.02.2026	Дюбанов О.С. Горбовий А.Ю Філоненко М.М. Погорєловська І.Д. Суховецька Л.П.

ПРАВИЛА ПЕРЕВЕДЕННЯ ОЦІНОК ДО НАЦІОНАЛЬНОЇ ШКАЛИ ТА ШКАЛИ ЄКТС

Оцінка за 100-бальною шкалою	Оцінка за шкалою ЄКТС	Оцінка за національною шкалою
90-100	A	5 (відмінно)
80-89	B	4 (добре)
70-79	C	
60-69	D	3 (задовільно)
50-59	E	
35-49	FX	2 (незадовільно)

Факультет фінансів та цифрових технологій
Кафедра комп'ютерних та інформаційних технологій і систем
Галузь знань 12 «Інформаційні технології»
Спеціальність 122 «Комп'ютерні науки»
Освітня програма «Інформаційні управляючі системи і технології в економіці»

Затверджую
Завідувач кафедри _____ Філоненко М.М.
«11» грудня 2024

Завдання на кваліфікаційну роботу
здобувачу вищої освіти
Луб'яніченку Юрію Сергійовичу
(прізвище, ім'я, по батькові)

1. Тема кваліфікаційної роботи Розробка веб-додатку управління запасами на підприємстві
Затверджена наказом ректора від «10» грудня 2024 № 1852
2. Строк здачі закінченої роботи «02» січня 2026
3. Мета роботи Розробка веб-додатку для автоматизації управління запасами на підприємстві
Об'єкт дослідження Процес обліку та управління товарними запасами на підприємстві
Предмет дослідження Методи та програмні засоби розробки веб-систем управління запасами.
Орієнтовна база дослідження Інформаційне середовище підприємства з використанням СУБД MySQL та веб-технологій
4. Вихідні дані до роботи Технічне завдання, вимоги до функціоналу системи, структура бази даних, програмні засоби Node.js, MySQL, HTML, CSS, JavaScript.
5. Перелік графічного матеріалу UML-діаграми, інтерфейс користувача
6. Зміст кваліфікаційної роботи (перелік питань за кожним розділом)
Теоретичні основи управління запасами та веб-технологій, аналіз бізнес-процесів і проєктування системи, розробка бази даних, серверної та клієнтської частин веб-додатку, тестування функціональності, оцінка ефективності та практичне впровадження

7. Календарний план виконання роботи

№ п/п	Назва етапів кваліфікаційноїроботи	Строк виконання етапів роботи	
		За планом	Фактично
1	2	3	4
1	Вибір теми, підбір літературних джерел, формування структури кваліфікаційної роботи	До 15.12.2024	До 15.12.2024
2	**І розділ** (розкрити сутність і стан дослідження наукової проблематики роботи, після чого навести її загальну характеристику; огляд літератури за темою і вибір конкретних напрямів досліджень)	30.05.2025	30.05.2025
3	**ІІ розділ** (аналіз експериментальної частини про методики досліджень)	10.09.2025	10.09.2025
4	**ІІІ розділ** (проведені теоретичні і (або) експериментальні дослідження та проектні рішення)	20.11.2025	20.11.2025
5	**Висновки**	11.12.2025	11.12.2025
6	Доопрацювання роботи з урахуванням зауважень наукового керівника	25.12.2025	25.12.2025
7	Отримання відгуку керівника	28.01.2026	28.01.2026
8	Отримання зовнішньої рецензії	30.01.2026	30.01.2026

8. Дата видачі завдання "11" грудня2024

9. Керівник кваліфікаційної роботи _____ Лаговський В.В.
(підпис) (прізвище, ініціали)

10. Керівник освітньо-професійної програми _____ Омельчук А.А.
(підпис) (прізвище, ініціали)

11. Завдання до виконання отримав здобувач вищої освіти _____ Луб'яніченко Ю.С.
(підпис) (прізвище, ініціали)

АНОТАЦІЯ

Розробка веб-додатку управління запасами на підприємстві

Мета роботи полягає у розробці та впровадженні веб-додатку для автоматизації процесів обліку та управління запасами на підприємстві. Завданнями роботи є аналіз предметної області, проєктування структури бази даних, розробка серверної та клієнтської частин, реалізація механізмів авторизації, ролей користувачів і контролю руху товарів, тестування та оцінка ефективності впровадження системи.

Предметом дослідження є інформаційні технології та програмні засоби управління запасами.

Об’єктом дослідження виступає процес обліку товарно-матеріальних цінностей на підприємстві.

Базою дослідження виступає інформаційне середовище підприємства з використанням СУБД MySQL та веб-технологій Node.js, HTML, CSS, JavaScript. За результатами дослідження сформульовані вимоги до веб-системи управління запасами та реалізовано повнофункціональний програмний продукт.

Одержані результати можуть бути використані для автоматизації складського обліку на підприємствах малого та середнього бізнесу.

Кваліфікаційна робота містить 60 сторінок, 4 таблиці, 4 рисунки, список використаних джерел з 40 найменувань.

Ключові слова: веб-додаток, управління запасами, база даних, склад, автоматизація.

Рік виконання кваліфікаційної роботи: 2025

Рік захисту роботи: 2026

ANNOTATION

Development of a Web Application for Inventory Management at an Enterprise

The purpose of the qualification work is to develop and implement a web application for automating inventory management processes at an enterprise. The objectives of the work include analysis of the subject area, database design, development of the server and client parts, implementation of authentication, user roles, and stock movement control, system testing and evaluation of implementation efficiency.

The subject of the research is information technologies and software tools for inventory management.

The object of the research is the process of inventory accounting at an enterprise.

The research base is the enterprise information environment using the MySQL DBMS and web technologies Node.js, HTML, CSS, and JavaScript.

As a result of the study, a full-featured web-based inventory management system was designed and implemented.

The obtained results can be used for automating warehouse accounting at small and medium-sized enterprises.

The qualification work consists of 60 pages, 4 tables, 4 figures, a list of references with 40 sources.

Keywords: web application, inventory management, database, warehouse, automation.

Year of qualification work completion: 2025

Year of defense: 2026

ЗМІСТ

ВСТУП **8**

РОЗДІЛ 1. ТЕОРЕТИЧНІ ОСНОВИ СИСТЕМ УПРАВЛІННЯ ЗАПАСАМИ ТА ВЕБ-ТЕХНОЛОГІЙ **12**

1.1.Поняття та роль управління запасами в інформаційних системах підприємства 12

1.2.Огляд сучасних програмних рішень для управління запасами 14

1.3.Технологічні засади розробки веб-додатків 17

Висновки до розділу 1 20

РОЗДІЛ 2. АНАЛІТИЧНЕ ДОСЛІДЖЕННЯ ТА ПОСТАНОВКА ЗАДАЧІ АВТОМАТИЗАЦІЇ **22**

2.1. Аналіз бізнес-процесів обліку запасів на підприємстві 22

2.2. Формування вимог до веб-додатку 24

2.3. Проєктування системи 27

Висновки до розділу 2 31

РОЗДІЛ 3. РОЗРОБКА ВЕБ-ДОДАТКУ УПРАВЛІННЯ ЗАПАСАМИ **32**

3.1. Розробка бази даних MySQL 32

3.2. Розробка серверної частини 36

3.3. Розробка клієнтської частини 45

3.4. Тестування функціональності системи 49

Висновки до розділу 3 51

РОЗДІЛ 4. ОЦІНКА ЕФЕКТИВНОСТІ ТА ПРАКТИЧНЕ ВПРОВАДЖЕННЯ **53**

4.1. Оцінка підвищення ефективності обліку запасів 53

4.2. Економічне обґрунтування впровадження 54

4.3. Практичне застосування 56

Висновки до розділу 4 57
ЗАГАЛЬНІ ВИСНОВКИ..... 59
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ 61

ВСТУП

Актуальність теми дослідження. В умовах цифрової трансформації бізнесу ефективне управління запасами є одним із основних чинників стабільної роботи підприємства. Неефективний облік товарно-матеріальних цінностей призводить до втрат прибутку, перевитрат ресурсів, дефіциту або надлишку продукції, що негативно впливає на фінансові результати діяльності. Особливо актуальним це питання є для малих та середніх підприємств, які потребують доступних, гнучких та зручних у використанні інформаційних систем.

Сучасні веб-технології дозволяють створювати універсальні інформаційні системи, доступні з будь-якого пристрою та в будь-який час. Веб-додатки для управління запасами забезпечують автоматизацію процесів обліку, контролю, аналізу та прогнозування, що істотно підвищує ефективність управлінських рішень. Саме тому розробка веб-додатку для управління запасами на підприємстві є актуальним науково-практичним завданням у галузі комп’ютерних наук та програмування.

Актуальність теми зумовлена також необхідністю впровадження сучасних інформаційних технологій у підприємницьку діяльність, зростанням вимог до швидкості, точності та безпеки обробки даних, а також потребою в створенні адаптованих програмних рішень для конкретних бізнес-процесів.

Стан і ступінь розробки проблеми в науковій літературі. Проблеми управління запасами досліджуються в працях вітчизняних і зарубіжних науковців у галузях економіки, логістики та інформаційних технологій. У роботах розглянуто методи оптимізації запасів, моделі EOQ, ABC- та XYZ-аналіз, використання ERP-систем та спеціалізованих програмних продуктів. Значна увага приділяється автоматизації складського обліку, інтеграції інформаційних систем і використанню хмарних технологій. У сфері веб-розробки ґрунтовно досліджено використання клієнт-серверної архітектури, REST-API, систем керування базами даних MySQL, фреймворків Laravel та

Node.js, а також сучасних технологій фронтенд-розробки на основі HTML, CSS та JavaScript.

Попри значну кількість наукових досліджень і готових комерційних рішень, проблема створення простих, доступних і кастомізованих веб-додатків для управління запасами малих підприємств залишається актуальною. Більшість існуючих систем є надмірно складними, дорогими у впровадженні або потребують тривалого навчання персоналу. Тому питання розробки власних веб-рішень із базовим, але достатнім функціоналом потребує подальшого дослідження та практичної реалізації.

Метою кваліфікаційної роботи є розробка веб-додатку для управління запасами на підприємстві з використанням сучасних веб-технологій, що забезпечить автоматизацію обліку, контроль руху товарів та підвищення ефективності управління запасами.

Для досягнення поставленої мети необхідно вирішити такі **завдання**:

1) проаналізувати теоретичні основи управління запасами та сучасні інформаційні системи підтримки цього процесу;

2) дослідити існуючі програмні рішення для обліку запасів та визначити їх переваги й недоліки;

3) здійснити аналіз бізнес-процесів обліку запасів на підприємстві;

4) сформувати функціональні та нефункціональні вимоги до веб-додатку;

5) спроєктувати архітектуру системи та структуру бази даних MySQL;

6) розробити серверну частину веб-додатку з використанням PHP (Laravel) або Node.js;

7) реалізувати клієнтську частину засобами HTML, CSS та JavaScript;

8) провести тестування функціональності системи;

9) здійснити оцінку ефективності впровадження веб-додатку.

Об’єктом дослідження є процес управління запасами на підприємстві з використанням інформаційних технологій.

Предметом дослідження є методи та засоби розробки веб-додатку для управління запасами на основі використання баз даних MySQL, серверних технологій PHP (Laravel) або Node.js та клієнтських технологій HTML, CSS і JavaScript.

У ході виконання кваліфікаційної роботи були використані такі **методи дослідження**:

1) метод аналізу та синтезу – для вивчення теоретичних аспектів управління запасами та веб-технологій;
2) порівняльний аналіз – для огляду існуючих програмних рішень;
3) системний підхід – для проєктування архітектури веб-додатку;
4) моделювання – для розробки структури бази даних та логіки роботи системи;
5) метод програмної реалізації – для створення серверної та клієнтської частин додатку;
6) метод тестування – для перевірки працездатності розробленої системи.

База дослідження. Базою дослідження є інформаційні процеси обліку запасів на підприємстві, а також програмно-апаратне середовище для розробки веб-додатку: серверне програмне забезпечення, база даних MySQL, середовища програмування, веб-браузери та інструменти тестування.

Наукова значимість роботи полягає в узагальненні теоретичних положень щодо управління запасами та застосування веб-технологій для автоматизації цього процесу.

Практична значимість роботи полягає в розробці функціонального веб-додатку для управління запасами з можливістю:

1) авторизації користувачів;
2) роботи через адміністративну панель;
3) додавання, редагування та видалення записів;
4) відображення списку товарів і руху запасів.

Результати роботи можуть бути використані в діяльності підприємств для автоматизації складського обліку та оптимізації управлінських процесів.

Особистий внесок здобувача. Усі етапи дослідження, проєктування, програмної реалізації, тестування та оформлення результатів виконані особисто автором. Ідейні, програмні та технологічні рішення розроблені самостійно.

Апробація результатів роботи. Результати кваліфікаційної роботи можуть бути представлені на студентських науково-практичних конференціях, семінарах та використані в навчальному процесі або у практичній діяльності підприємств. За необхідності можливе подальше розширення функціоналу та публікація результатів дослідження.

РОЗДІЛ 1
ТЕОРЕТИЧНІ ОСНОВИ СИСТЕМ УПРАВЛІННЯ ЗАПАСАМИ ТА ВЕБ-ТЕХНОЛОГІЙ

1.1. Поняття та роль управління запасами в інформаційних системах підприємства

Управління запасами є одним із основних елементів логістичної та виробничо-господарської діяльності підприємства, що забезпечує безперервність операційних процесів, своєчасне виконання замовлень та раціональне використання матеріальних ресурсів. Запаси включають сукупність товарно-матеріальних цінностей, що знаходяться на різних стадіях руху – від надходження на склад до кінцевого споживання. До складу запасів відносять сировину, матеріали, напівфабрикати, готову продукцію, комплектуючі вироби та інші матеріальні ресурси.

З економічної точки зору управління запасами розглядається як система заходів, спрямованих на забезпечення оптимального рівня матеріальних ресурсів з метою мінімізації витрат на їх зберігання та обслуговування, запобігання дефіциту і надлишків, а також підвищення ефективності використання оборотних коштів підприємства. Надмірні запаси призводять до заморожування фінансових ресурсів, зростання витрат на зберігання, псування продукції, тоді як їх нестача викликає перебої у виробництві, зниження рівня обслуговування клієнтів і втрату прибутку [12].

З позиції управління підприємством система управління запасами охоплює процеси прогнозування попиту, планування закупівель, формування замовлень постачальникам, обліку надходження та витрачання товарно-матеріальних цінностей, контролю залишків і аналізу їх руху. Ефективність функціонування такої системи безпосередньо впливає на фінансову стабільність підприємства,

його конкурентоспроможність і здатність швидко реагувати на зміни ринкової кон’юнктури.

У сучасних умовах господарювання управління запасами тісно пов’язане з використанням інформаційних систем, які забезпечують автоматизацію процесів обліку, контролю та аналізу. Інформаційні системи управління запасами є частиною загальної корпоративної інформаційної системи підприємства та інтегруються з бухгалтерськими, виробничими, збутовими та управлінськими підсистемами. Завдяки цьому досягається єдине інформаційне середовище, що забезпечує достовірність даних, оперативність прийняття рішень та зменшення людського фактору [9].

Інформаційна система управління запасами виконує низку важливих функцій, серед яких основними є реєстрація руху матеріальних ресурсів, формування електронних складських документів, ведення обліку залишків у режимі реального часу, формування звітності, а також підтримка управлінських рішень щодо поповнення або скорочення запасів. Застосування таких систем дозволяє значно підвищити прозорість складських операцій і спростити контроль за рухом товарів.

Особливого значення набуває інтеграція управління запасами з іншими бізнес-процесами підприємства. Наприклад, зв’язок із системами продажів дає змогу автоматично зменшувати залишки після реалізації продукції, а інтеграція з бухгалтерським обліком забезпечує коректне формування фінансової звітності. Загалом, система управління запасами стає важливим інструментом підтримки стратегічного та оперативного управління [10].

З розвитком цифрових технологій все більшого поширення набувають веб-орієнтовані інформаційні системи управління запасами. Вони забезпечують доступ до даних через мережу Інтернет, підтримують багатокористувацький режим роботи, не потребують складного програмного встановлення та дозволяють керівникам отримувати актуальну інформацію незалежно від місця

перебування. Це особливо актуально для малих та середніх підприємств, які прагнуть оптимізувати витрати на впровадження IT-рішень.

Отже, управління запасами є невід'ємною складовою системи управління підприємством і відіграє важливу роль у забезпеченні його ефективної діяльності. Використання сучасних інформаційних систем дозволяє автоматизувати складські та облікові процеси, підвищити точність даних, зменшити витрати та забезпечити прийняття обґрунтованих управлінських рішень, що підтверджує доцільність і необхідність розробки спеціалізованих веб-додатків для управління запасами.

1.2. Огляд сучасних програмних рішень для управління запасами

У наш час на ринку IT-рішень для бізнесу присутні чисельні програмні продукти, призначені для автоматизації обліку запасів, управління складом, закупівель, продажів, інвентаризації і логістики. Залежно від масштабу підприємства, характеру діяльності (виробництво, торгівля, дистрибуція) та потреб у функціоналі, організації можуть обирати між вузькоспеціалізованими рішеннями (WMS, складські системи) або комплексними ERP-/CRM-системами з модулем управління запасами. Серед популярних підходів – локальні (on-premise), хмарні або гібридні рішення, а також відкриті (open-source) і комерційні продукти [5].

WMS-системи (Warehouse Management System) – спеціалізовані програми, що автоматизують управління складом і запасами. Вони забезпечують прийом, зберігання, переміщення, відвантаження, інвентаризацію, облік залишків у режимі реального часу, маркування (штрих-/RFID-коди), генерацію складських документів і звітів. Такі рішення значно покращують точність обліку, мінімізують помилки, оптимізують використання складських площ.

Комплексні ERP- / бізнес-системи – поєднують управління запасами з іншими бізнес-процесами – закупівлі, продажі, бухгалтерський облік, CRM,

виробництво. Для середніх або великих підприємств такі системи дають змогу інтегрувати усі процеси в єдине інформаційне поле.

Open-source рішення – забезпечують гнучкі можливості кастомізації, відкритий код, часто безкоштовні або з невеликим бюджетом. Вони можуть бути оптимальними для малого або середнього бізнесу, який має власних IT-фахівців та потребує адаптації функціоналу.

Локальні / адаптовані до ринку країни рішення – для підприємств, особливо в Україні, важливими є програми, які враховують специфіку місцевого обліку, мову, законодавство, валюту тощо. Такі рішення часто пропонують підтримку, спрощене впровадження, базові функції обліку запасів і складу.

Водночас, при виборі програмного забезпечення для управління запасами слід враховувати:

1) потенціал бізнесу (масштаб, кількість складів, об’єми товарообігу);
2) необхідний функціонал (лише складський облік, чи інтеграція з бухгалтерією, продажами, виробництвом);
3) бюджет (від економних / open-source рішень до комерційних систем);
4) гнучкість, можливість налаштування або розширення;
5) зручність користування, навчання персоналу, підтримка та оновлення.

Загалом, сучасні програмні рішення надають широкий спектр варіантів – від простих WMS до повноцінних ERP – і дають змогу підприємствам обирати відповідно до потреб, ресурсу та масштабу. Для багатьох малих і середніх підприємств особливо вигідним є створення власного веб-додатку – гнучкого, легкого, з базовим але достатнім функціоналом, що відповідає їхнім специфічним потребам.

Таблиця 1.1

Порівняння деяких популярних програмних рішень для управління запасами / складом / підприємством

Назва / Система	Категорія / Тип	Основні можливості / функції	Переваги	Обмеження / на що звернути увагу
SAP Business One [34]	ERP / комплексна система	Управління запасами, закупівлі, продажі, бухгалтерський облік, CRM, виробництво, інвентаризація	Повний цикл бізнес-процесів, інтеграція, аналітика, масштабованість для SMEs	Вартість, складність впровадження, потреба підготовки персоналу
Odoo [28]	Open-source ERP / бізнес-пакет	Модулі для складу, запасів, продажів, CRM, складу, обліку, e-commerce	Гнучкість, відкритий код, можливість кастомізації, підходить для малого/середнього бізнесу	Потреба технічних навичок для налаштування, можлива нестача «з коробки» готових рішень
Bimp ERP (з WMS-модулем) [1]	ERP + WMS / локальне рішення	Управління закупівлями, запасами, облік складу, інвентаризація, масштабування бізнесу	Адаптоване під локальний бізнес, простота впровадження, підтримка	Обмежений функціонал порівняно з великими ERP, залежність від постачальника
Skynum [35]	Складська / WMS-система	Облік запасів, рух товарів між складами/магазинами, партійний облік, контроль залишків	Простота, фокус на складі, підходить для малого/середнього бізнесу	Можливо, недостатньо модулів для комплексного обліку (фінанси, CRM)
Спеціалізовані WMS-системи (загальні)	WMS / складський облік	Приймання, зберігання, відвантаження, інвентаризація, маркування, звіти, реальний час, переміщення, логістика	Оптимізація складських процесів, зменшення помилок, підвищення точності обліку	Не дають комплексного управління підприємством – лише склад / логістику

Отже, для великих або середніх підприємств з різнопрофільною діяльністю (склад, закупівлі, продажі, бухгалтерія, CRM, виробництво) найзручніше

застосовувати комплексні ERP-системи, такі як SAP Business One – вони покривають більшість бізнес-процесів і забезпечують інтегрованість. Якщо підприємство невелике, або має обмежений бюджет, гнучкістю і кастомізацією відрізняються open-source або локальні ERP/WMS-рішення (наприклад, Odoo, Bimp ERP, Skynum). Якщо ж основне завдання – це складський облік, контроль руху товарів та запасів – спеціалізовані WMS-системи є оптимальними: вони не перевантажують систему непотрібним функціоналом, але добре справляються зі складовою обліку запасів. При цьому, навіть найсучасніші готові рішення мають певні недоліки: високу вартість, залежність від постачальника, складність налаштування, надлишковий функціонал, якого реально не використовують – що аргументує обґрунтованість розробки власного веб-додатку під конкретні потреби компанії.

У контексті дипломної роботи – створення простого веб-додатку з базовим функціоналом (авторизація, CRUD, відображення запасів) – ця альтернатива має хороші перспективи: вона може покрити основні потреби без надмірного ускладнення, бути адаптованою під реальні бізнес-процеси підприємства, легко модифікуватися та розширюватися надалі.

1.3. Технологічні засади розробки веб-додатків

Розробка сучасних веб-додатків ґрунтується на поєднанні клієнтських і серверних технологій, що функціонують у межах клієнт-серверної архітектури. Такий підхід передбачає чітке розмежування відповідальності між користувацьким інтерфейсом (frontend) та логікою обробки даних (backend). У результаті забезпечується масштабованість, гнучкість, безпечність і зручність використання програмного продукту [8].

Frontend (клієнтська частина) відповідає за відображення інформації користувачу та взаємодію з ним. Основними стандартними технологіями фронтенд-розробки є:

1) HTML (HyperText Markup Language) – використовується для створення структури веб-сторінок;

2) CSS (Cascading Style Sheets) – відповідає за стильове оформлення інтерфейсу;

3) JavaScript – забезпечує інтерактивність, динамічну обробку подій, валідацію форм, взаємодію з сервером через API.

Завдяки поєднанню цих технологій створюється адаптивний та інтуїтивно зрозумілий інтерфейс користувача, що є критично важливим для інформаційних систем управління запасами, де значну роль відіграє швидкість доступу до даних та зручність роботи операторів і адміністраторів.

Backend (серверна частина) відповідає за бізнес-логіку, обробку запитів, управління доступом, взаємодію з базою даних та забезпечення безпеки. Для розробки серверної частини веб-додатків широко використовуються сучасні фреймворки та платформи, зокрема Laravel та Node.js. Laravel забезпечує зручну реалізацію MVC-архітектури, механізми роботи з базами даних через ORM, систему аутентифікації та маршрутизацію. Node.js дозволяє будувати високопродуктивні серверні застосунки на основі JavaScript із використанням асинхронної обробки запитів.

Обов’язковим компонентом веб-додатків є система керування базами даних (СКБД). У межах даної роботи використовується MySQL – реляційна СКБД, що забезпечує надійне зберігання, обробку та захист даних. MySQL підтримує транзакції, індексацію, контроль цілісності та забезпечує високу швидкодію навіть при значних обсягах інформації, що особливо важливо для систем управління запасами.

Важливою технологічною засадою є використання MVC-архітектури (Model–View–Controller), яка дозволяє розділити логіку додатку на три основні рівні:

1) Model – відповідає за роботу з даними та бізнес-логікою;

2) View – формує зовнішній вигляд інтерфейсу;

3) Controller – обробляє запити користувачів і координує взаємодію між моделлю та представленням.

Такий підхід значно спрощує розробку, тестування та супровід веб-додатків.

Окреме місце у технологічних засадах розробки веб-додатків посідає забезпечення інформаційної безпеки. Для систем управління запасами це питання є особливо важливим, оскільки вони працюють з критично важливими даними підприємства. Насамперед важливо реалізувати надійні механізми авторизації та автентифікації користувачів, які дозволяють ідентифікувати кожного користувача системи та перевіряти його права доступу. Обов'язковим також є чітке розмежування ролей користувачів, зокрема адміністратора, оператора та звичайного користувача, що забезпечує контроль над виконанням операцій у межах наданих повноважень [3].

Важливою складовою безпеки є захист веб-застосунку від поширених мережевих атак, серед яких найбільш небезпечними є SQL-ін'єкції, міжсайтові скриптові атаки (XSS) та підміна сесій. Для цього застосовуються фільтрація і валідація вхідних даних, використання підготовлених запитів до бази даних, контроль сесій та використання сучасних засобів захисту веб-середовища. Не менш важливим є надійне шифрування паролів користувачів із використанням криптографічних алгоритмів, а також застосування захищених HTTP-з'єднань, що гарантує безпечну передачу даних між клієнтом і сервером. Це в сукупності забезпечує надійний рівень захисту веб-додатку та збереження цілісності й конфіденційності інформації.

Ще одним важливим аспектом є робота з API та асинхронними запитами (AJAX), що дозволяє оновлювати дані на веб-сторінці без її повного перезавантаження. Це підвищує продуктивність і зручність користування системою, особливо при роботі зі списками товарів, фільтрацією та пошуком.

Загалом, технологічні засади розробки веб-додатків базуються на комплексному використанні клієнтських і серверних технологій, СКБД,

архітектурних підходів MVC та сучасних механізмів безпеки. Їх поєднання дозволяє створювати ефективні, надійні та масштабовані веб-системи для управління запасами підприємств, що відповідають сучасним вимогам цифрової економіки.

Висновки до розділу 1

У першому розділі дипломної роботи було розглянуто теоретичні основи управління запасами та технологічні засади розробки сучасних веб-додатків. Визначено, що управління запасами є одним із основних елементів ефективної діяльності підприємства, оскільки воно безпосередньо впливає на безперервність виробничих і торговельних процесів, раціональне використання матеріальних ресурсів та фінансову стабільність господарюючого суб’єкта.

Проаналізовано сутність та роль управління запасами в інформаційних системах підприємства, з’ясовано, що автоматизація процесів обліку, контролю та аналізу руху запасів є необхідною умовою підвищення ефективності управлінських рішень. Встановлено, що застосування інформаційних систем дозволяє забезпечити достовірність даних, оперативність доступу до інформації та зменшити вплив людського фактору.

Виконано огляд сучасних програмних рішень для управління запасами, серед яких виділено ERP-, WMS- та веб-орієнтовані системи. Визначено їхні переваги та недоліки, зокрема високу вартість впровадження, складність адаптації до специфіки малого бізнесу, надлишковість функціоналу та залежність від постачальника програмного забезпечення. Це обґрунтовує доцільність розробки власного веб-додатку, який відповідатиме реальним потребам конкретного підприємства.

Розглянуто технологічні засади розробки веб-застосунків, що включають використання клієнтських і серверних технологій, баз даних, архітектури MVC та засобів інформаційної безпеки. Встановлено, що поєднання HTML, CSS і

JavaScript на стороні клієнта з серверними рішеннями на базі PHP (Laravel) або Node.js та використанням СКБД MySQL дозволяє створювати надійні, масштабовані та безпечні інформаційні системи.

Отримані в розділі теоретичні положення є науково-методичною основою для подальшого аналітичного дослідження бізнес-процесів обліку запасів, формування вимог до веб-додатку, його проєктування та практичної реалізації, що буде здійснено в наступних розділах дипломної роботи.

РОЗДІЛ 2
АНАЛІТИЧНЕ ДОСЛІДЖЕННЯ ТА ПОСТАНОВКА ЗАДАЧІ АВТОМАТИЗАЦІЇ

2.1. Аналіз бізнес-процесів обліку запасів на підприємстві

Облік запасів є однією з найважливіших функціональних складових системи управління підприємством, оскільки саме від його ефективності залежить безперебійність виробничих і торговельних процесів, рівень обслуговування клієнтів, оптимальність використання фінансових ресурсів та достовірність управлінської звітності. Для обґрунтування доцільності автоматизації обліку запасів необхідно провести аналіз існуючих бізнес-процесів, що забезпечують рух товарно-матеріальних цінностей на підприємстві.

Типові бізнес-процеси обліку запасів охоплюють етапи надходження матеріальних ресурсів, їх зберігання, внутрішнього переміщення, відпуску у виробництво або реалізації, а також проведення інвентаризації та формування звітності. На етапі надходження здійснюється приймання товару від постачальників, перевірка кількості та якості продукції, оформлення первинних документів (накладні, акти приймання) і внесення інформації до облікових реєстрів. У процесі зберігання запасів відбувається розміщення товарів на складі, контроль умов зберігання, підтримка оптимального рівня залишків та спостереження за термінами придатності продукції.

Наступним етапом є внутрішнє переміщення та відпуск запасів, коли здійснюється передача матеріалів між підрозділами підприємства, їх списання у виробництво або відвантаження клієнтам. Цей процес супроводжується оформленням відповідних документів, веденням кількісного та вартісного обліку, а також оновленням даних про залишки. Завершальним етапом є інвентаризація, яка полягає у фактичній перевірці наявності запасів і звірці облікових даних із реальним станом складу.

На багатьох малих і середніх підприємствах ці процеси досі здійснюються частково вручну або з використанням фрагментарних програмних засобів, таких як електронні таблиці. Такий підхід має низку суттєвих недоліків: високу ймовірність помилок при ручному введенні даних, дублювання інформації, відсутність оперативного доступу до актуальних залишків, складність формування аналітичної звітності та залежність від людського фактору. У результаті знижується точність обліку, ускладнюється контроль руху товарів та прийняття управлінських рішень.

Аналіз існуючих бізнес-процесів обліку запасів дозволяє виявити також проблеми взаємодії між підрозділами підприємства. Часто інформація про надходження, переміщення та реалізацію товарів передається із запізненням, що призводить до розбіжностей між фактичними та обліковими залишками. Відсутність єдиного інформаційного простору ускладнює планування закупівель, призводить до накопичення надлишкових запасів або, навпаки, до дефіциту необхідних ресурсів.

Крім того, традиційні способи обліку не забезпечують достатньої аналітичної підтримки управлінських рішень. Керівництво підприємства не завжди володіє актуальною інформацією про структуру запасів, швидкість їх обороту, популярність окремих товарних позицій, що ускладнює прогнозування попиту та оптимізацію складських процесів. Це свідчить про необхідність впровадження сучасних автоматизованих систем управління запасами, здатних забезпечити комплексний облік і аналітику в режимі реального часу.

В цілому, проведений аналіз бізнес-процесів обліку запасів на підприємстві свідчить про доцільність їх автоматизації з використанням веб-додатку. Автоматизована система дозволить інтегрувати всі етапи руху товарно-матеріальних цінностей в єдиному інформаційному середовищі, зменшити кількість помилок, підвищити прозорість складських операцій, прискорити формування звітності та забезпечити ефективну підтримку управлінських рішень.

2.2. Формування вимог до веб-додатку

Формування вимог до веб-додатку управління запасами є основним етапом проєктування інформаційної системи, оскільки саме на цьому етапі визначаються її функціональні можливості, технічні характеристики, вимоги до безпеки, зручності використання та продуктивності. Вимоги формуються на основі аналізу бізнес-процесів підприємства (розділ 2.1), потреб кінцевих користувачів, а також з урахуванням сучасних технологічних можливостей.

На основі проведеного аналізу сформовано детальний перелік функціональних та нефункціональних вимог до системи.

2.2.1. Функціональні вимоги

Функціональні вимоги визначають конкретні дії та можливості, які має надавати веб-додаток:

1. Управління системою безпеки та користувачами:

ФВ1.1. Наявність механізму авторизації та аутентифікації користувачів (логін/пароль) з видачею JWT-токена (п. 3.2).

ФВ1.2. Реалізація ролерової моделі доступу з ролями: **Адміністратор** (повний доступ), **Менеджер/Оператор** (облік запасів, CRUD-операції), **Співробітник** (перегляд) (п. 3.1, 3.2).

ФВ1.3. Шифрування паролів користувачів за допомогою алгоритмів хешування (п. 3.1, 3.2).

2. Управління довідниковою інформацією (CRUD-операції):

ФВ2.1. Можливість створення, перегляду, редагування та видалення категорій товарів з перевіркою на наявність пов’язаних товарів перед видаленням (п. 3.2).

ФВ2.2. Можливість створення, перегляду, редагування та видалення даних про постачальників з аналогічною перевіркою (п. 3.2).

3. Управління товарами та запасами:

ФВ3.1. Ведення обліку товарів з атрибутами: назва, опис, категорія, постачальник, ціна, поточний залишок (stock_quantity), мінімальний допустимий рівень запасу (min_stock_level) (п. 3.1, Табл. 3.1).

ФВ3.2. Автоматичне формування списку товарів з низьким рівнем запасів (порівняння stock_quantity з min_stock_level) для попередження дефіциту (п. 3.2).

ФВ3.3. Надання інтерфейсу для перегляду всіх товарів з можливістю сортування та фільтрації (п. 3.3).

4. Облік складських операцій (рух товарів):

ФВ4.1. Фіксація операцій **прийому** товарів на склад (тип in) з автоматичним збільшенням відповідного залишку в реальному часі (п. 3.2).

ФВ4.2. Фіксація операцій **видачі** товарів зі складу (тип out) з автоматичним зменшенням залишку та обов'язковою перевіркою його достатності. У разі нестачі операція має блокуватися (п. 3.2).

ФВ4.3. Ведення повного журналу руху (stock_movements) з записом: товар, користувач, тип операції, кількість, ціна, дата (п. 3.1, 3.2).

ФВ4.4. Забезпечення атомарності та цілісності операцій руху за допомогою механізму транзакцій СУБД (п. 3.2).

2.2.2. Нефункціональні вимоги

Нефункціональні вимоги визначають якісні характеристики системи та технологічні обмеження:

1.Вимоги до продуктивності та надійності:

НФВ1.1. Система має забезпечувати швидку обробку запитів (в межах 2-3 секунд) під час роботи з основними списками (товари, рухи) (п. 2.2).

НФВ1.2. Система має бути стійкою до збоїв, зберігати цілісність даних за допомогою транзакцій та мати можливість відновлення (п. 3.2, 3.4).

НФВ1.3. Архітектура системи (MVC, REST API) має забезпечувати масштабованість для подальшого розширення функціоналу (п. 1.3, 2.3)

Вимоги до безпеки:

НФВ2.1. Захист від поширених веб-загроз: SQL-ін'єкцій (через підготовлені запити), XSS (фільтрація вхідних даних) (п. 1.3).

НФВ2.2. Обов’язкове використання захищених HTTP-з’єднань (HTTPS) для передачі даних (п. 1.3).

НФВ2.3. Фіксація дій користувачів (аудит) у журналі подій для контролю (п. 2.2, реалізовано через stock_movements та users).

2. Вимоги до практичності (Usability) та інтерфейсу:

НФВ3.1. Інтерфейс користувача має бути інтуїтивно зрозумілим, не вимагати тривалого навчання та мати логічну структуру (п. 2.2, 3.3).

НФВ3.2. Система має мати адаптивний веб-інтерфейс, що коректно відображається на різних пристроях (ПК, планшети) (п. 2.2, 3.3).

НФВ3.3. Надання інформативних повідомлень про помилки, підтвердження операцій та підсвічування критичних станів (напр., низький залишок) (п. 3.3).

3. Технологічні вимоги:

НФВ4.1. Реалізація за принципами клієнт-серверної архітектури з використанням REST API (п. 1.3, 2.3).

НФВ4.2. Використання стеку технологій: серверна частина — Node.js + Express, клієнтська — HTML5, CSS3, JavaScript (Vanilla), база даних — MySQL (п. 1.3, 3.2, 3.3).

НФВ4.3. Підтримка кодування utf8mb4 для коректної роботи з українськими символами (п. 3.1).

Сформовані вимоги є основним орієнтиром для подальшого проєктування архітектури системи, структури бази даних та вибору конкретних технологій реалізації, що буде детально розглянуто в підрозділі 2.3.

2.3. Проєктування системи

Проєктування веб-додатку управління запасами ґрунтується на результатах аналізу бізнес-процесів та сформованих вимог до системи. На цьому етапі визначається архітектура програмного продукту, основні компоненти, структура даних, а також взаємодія користувачів із системою. Застосування методів структурного та об’єктно-орієнтованого аналізу дозволяє формалізувати модель предметної області та побудувати UML-діаграми, що описують функціонування веб-додатку [16], [32].

Запропонована система будується за принципами клієнт-серверної архітектури з використанням багаторівневого підходу. На рівні представлення (frontend) працює веб-інтерфейс, реалізований засобами HTML, CSS та JavaScript. Цей рівень відповідає за відображення даних, введення інформації користувачами, виконання запитів до сервера та відображення результатів. Рівень прикладної логіки (backend) реалізує обробку запитів, бізнес-логіку, перевірку прав доступу, валідацію даних та формування відповідей клієнту. Рівень даних представлений системою керування базами даних MySQL, що забезпечує надійне зберігання, пошук і модифікацію інформації про користувачів, товари та рух запасів.

Основними суб’єктами взаємодії із системою є адміністратор та оператор (користувач складу або відповідальний за облік запасів). Адміністратор здійснює керування обліковими записами користувачів, налаштування довідників (категорій товарів, одиниць виміру тощо), а також повний контроль над операціями з товарами. Оператор працює з повсякденними операціями – додає, редагує та списує товари, фіксує надходження, переміщення та відвантаження, переглядає залишки та формує базові звіти. Взаємодія цих ролей із системою може бути відображена за допомогою діаграми прецедентів (рис. 2.1).



Рис. 2.1 Діаграма прецедентів

Для формалізації структури системи використовується діаграма класів (рис. 2.2), яка описує основні сутності предметної області: користувачів, ролі, товари, категорії, рух запасів тощо. Кожен клас має відповідний набір атрибутів і методів, що відображають його призначення. Зв’язки між класами (асоціація, композиція, агрегація) демонструють логіку взаємодії об’єктів у системі. Наприклад, товар може належати до певної категорії, мати множину записів руху (операцій надходження або списання), а користувач є ініціатором відповідних операцій.

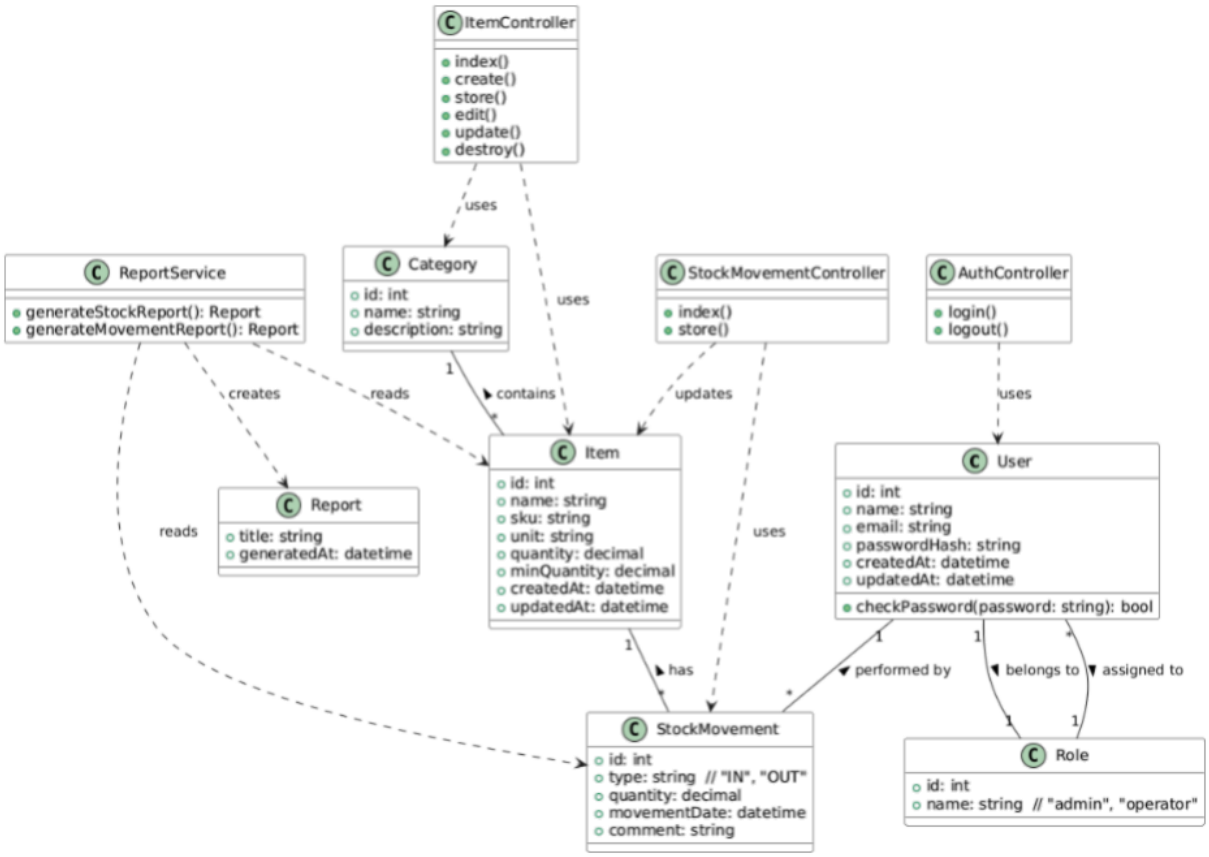


Рис. 2.2 Діаграма класів

Процеси взаємодії клієнта із сервером доцільно описати за допомогою діаграми послідовності (рис. 2.3). Вони дозволяють простежити життєвий цикл типових операцій: авторизації користувача, додавання нового товару, перегляду списку запасів, редагування чи видалення запису. У межах такого опису відображаються об’єкти (користувач, веб-браузер, контролер, сервіс доступу до даних, база даних) та послідовність запитів і відповідей між ними.

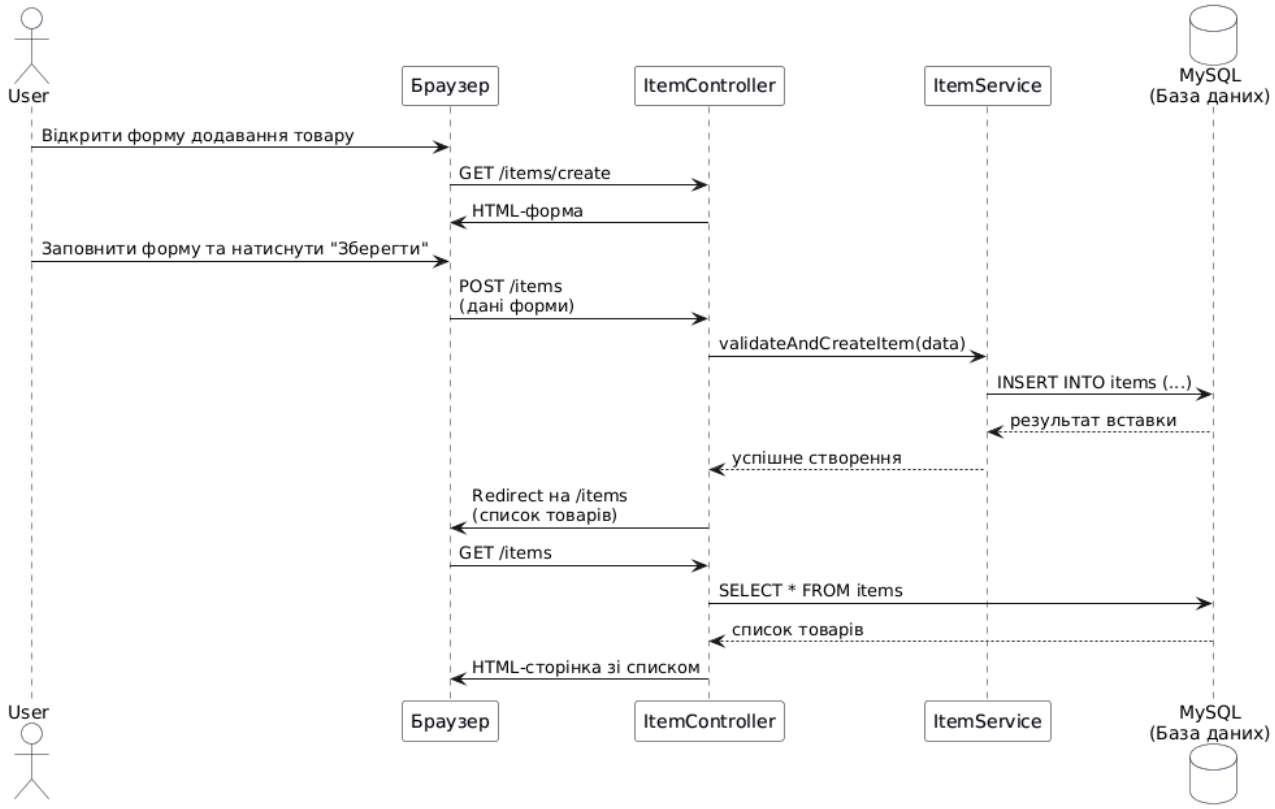


Рис. 2.3 Діаграма послідовності

Загальна архітектура програмної системи базується на застосуванні патерну MVC (Model–View–Controller). Модель відповідає за роботу з даними та бізнес-логікою (класи користувачів, товарів, рухів запасів), представлення реалізує інтерфейс користувача у вигляді веб-сторінок, а контролери забезпечують маршрутизацію запитів і координацію взаємодії між моделлю та представленням. Такий підхід забезпечує розділення відповідальностей, полегшує супровід і тестування системи, а також дає змогу гнучко розширювати функціонал у майбутньому.

Отже, проєктування системи включає визначення її архітектури, структури даних та сценаріїв взаємодії користувачів із веб-додатком. UML-діаграми, побудовані на цьому етапі, слугують основою для подальшої реалізації бази даних, серверної частини та клієнтської частини (розділ 3.3).

Висновки до розділу 2

У другому розділі дипломної роботи було виконано аналітичне дослідження процесів обліку запасів на підприємстві та здійснено постановку задачі автоматизації. Проведений аналіз бізнес-процесів показав, що традиційні способи ведення складського обліку, засновані на ручному введенні даних або використанні розрізнених програмних засобів, не забезпечують належного рівня оперативності, точності та контролю за рухом товарно-матеріальних цінностей. Виявлено основні проблеми, серед яких дублювання інформації, висока ймовірність помилок, складність формування звітності та відсутність єдиного інформаційного середовища.

На основі проведеного аналізу сформовано вимоги до веб-додатку управління запасами, які включають функціональні, нефункціональні, технічні та вимоги до безпеки. Визначено необхідність реалізації механізмів авторизації, розмежування прав доступу, обліку руху товарів, формування звітності, а також забезпечення надійного захисту даних. Зазначені вимоги окреслюють цільовий функціонал майбутньої інформаційної системи та виступають основою для її подальшої реалізації.

У межах підрозділу 2.3 виконано проєктування системи, визначено архітектуру веб-додатку, основні програмні компоненти та їх взаємодію. Розроблено UML-моделі, які відображають сценарії використання системи, структуру її сутностей та логіку основних операцій. Запропонована модель базується на клієнт-серверній архітектурі та застосуванні патерну MVC, що забезпечує гнучкість, масштабованість і зручність супроводу програмного продукту.

Отже, результати другого розділу створюють ґрунтовну аналітичну та проєктну основу для практичної реалізації веб-додатку управління запасами, що буде здійснено у третьому розділі дипломної роботи.

РОЗДІЛ 3
РОЗРОБКА ВЕБ-ДОДАТКУ УПРАВЛІННЯ ЗАПАСАМИ

3.1. Розробка бази даних MySQL

Однією з основних складових веб-додатку управління запасами є база даних, яка забезпечує збереження, обробку та цілісність інформації про користувачів, товари, постачальників, категорії та рухи товарів на складі. У межах даної кваліфікаційної роботи було спроєктовано та реалізовано реляційну базу даних warehouse_db у середовищі MySQL, що є однією з найпоширеніших систем управління базами даних для веб-застосунків [27].

Для забезпечення коректного збереження українських символів і міжнародної сумісності використано кодування utf8mb4 з відповідною колацією utf8mb4_unicode_ci. Це дозволяє уникнути втрат даних при введенні текстових значень та забезпечує стабільну роботу системи з багатомовним контентом.

База даних складається з п’яти взаємопов’язаних таблиць, які відображають основні інформаційні об’єкти предметної області:

1) користувачі системи;
2) категорії товарів;
3) постачальники;
4) товари;
5) рух товарів на складі.

Між таблицями реалізовано логічні зв’язки за допомогою зовнішніх ключів (FOREIGN KEY), що гарантує референційну цілісність та виключає появу неконсистентних даних.

Таблиця users призначена для збереження інформації про всіх зареєстрованих користувачів веб-застосунку. Вона містить такі основні поля: унікальний ідентифікатор користувача, ім’я, електронну пошту, пароль, роль та дату створення облікового запису. Поле role реалізоване у вигляді переліку

значень admin, manager, employee, що дозволяє впровадити рольову модель доступу. Адміністратор має повні права керування системою, менеджер – обмежений доступ до редагування даних, а співробітник – лише до перегляду. Паролі користувачів зберігаються у хешованому вигляді, що відповідає сучасним вимогам інформаційної безпеки та унеможливлює несанкціонований доступ до персональних даних.

Таблиця categories використовується для класифікації товарів за логічними групами. Вона містить назву категорії, її опис та дату створення запису. Зв’язок між таблицями categories і products дозволяє групувати товари, здійснювати фільтрацію та формувати аналітичні звіти за категоріями.

Таблиця suppliers призначена для збереження інформації про постачальників товарів. До її складу входять такі поля: назва компанії, контактна особа, номер телефону, електронна пошта, адреса та дата створення запису. Таблиця зв’язана з таблицею products, що дозволяє відстежувати постачальника кожного товару та аналізувати ефективність співпраці.

Таблиця products є центральною у структурі бази даних, оскільки містить основну інформацію про всі товари, що зберігаються на складі. Вона включає назву товару, опис, категорію, постачальника, ціну, поточний залишок, мінімально допустимий рівень запасу, а також службові поля дат створення та оновлення. Наявність поля min_stock_level дозволяє автоматично визначати товари з критично низьким рівнем запасів, що використовується у веб-інтерфейсі для формування попереджень та своєчасного поповнення складу. Зовнішні ключі до таблиць categories та suppliers забезпечують цілісність взаємопов’язаних даних.

Таблиця stock_movements використовується для реєстрації всіх операцій прийому та видачі товарів. Вона містить посилання на товар, користувача, тип операції (прийом або видача), кількість, ціну, примітки та дату виконання операції. Дана таблиця виконує функцію журналу складських операцій, що дозволяє:

1) здійснювати аудит руху товарів;
2) контролювати зміну залишків;
3) аналізувати історію операцій за періодами;
4) фіксувати відповідальних осіб за кожну операцію.

Зведена структура бази даних представлена в таблиці 3.1.

Таблиця 3.1

Зведена структура бази даних веб-додатку управління запасами

Таблиця	Назва поля	Тип даних	Опис
users	id	INT, PK, AUTO_INCREMENT	Унікальний ідентифікатор користувача
	name	VARCHAR(255)	Ім’я користувача
	email	VARCHAR(255), UNIQUE	Електронна пошта
	password	VARCHAR(255)	Хешований пароль
	role	ENUM('admin','manager','employee')	Роль користувача
	created_at	TIMESTAMP	Дата створення облікового запису
categories	id	INT, PK, AUTO_INCREMENT	Ідентифікатор категорії
	name	VARCHAR(255)	Назва категорії
	description	TEXT	Опис категорії
	created_at	TIMESTAMP	Дата створення
suppliers	id	INT, PK, AUTO_INCREMENT	Ідентифікатор постачальника
	name	VARCHAR(255)	Назва компанії
	contact_person	VARCHAR(255)	Контактна особа
	phone	VARCHAR(50)	Телефон
	email	VARCHAR(255)	Електронна пошта
	address	TEXT	Адреса
	created_at	TIMESTAMP	Дата створення
products	id	INT, PK, AUTO_INCREMENT	Ідентифікатор товару
	name	VARCHAR(255)	Назва товару
	description	TEXT	Опис товару
	category_id	INT, FK	Посилання на категорію
	supplier_id	INT, FK	Посилання на постачальника
	price	DECIMAL(10,2)	Ціна товару
	stock_quantity	INT	Поточний залишок
	min_stock_level	INT	Мінімальний допустимий залишок
	created_at	TIMESTAMP	Дата створення
	updated_at	TIMESTAMP	Дата оновлення

Продовження таблиці 3.1

Таблиця	Назва поля	Тип даних	Опис
stock_movements	Id	INT, PK, AUTO_INCREMENT	Ідентифікатор операції
	product_id	INT, FK	Посилання на товар
	user_id	INT, FK	Посилання на користувача
	type	ENUM('in','out')	Тип операції (прийом/видача)
	quantity	INT	Кількість
	price	DECIMAL(10,2)	Ціна
	note	TEXT	Примітка
	created_at	TIMESTAMP	Дата операції

Для підвищення швидкодії системи у базі даних було створено додаткові індекси за найбільш використовуваними полями: категоріями товарів, постачальниками, товарами та датами руху. Це дозволяє значно зменшити час виконання запитів під час формування звітів, аналітики та вибірок у реальному часі.

Для перевірки працездатності бази даних та демонстрації роботи веб-додатку було наповнено її тестовими даними, які включають користувачів з різними ролями, категорії товарів, постачальників, товари та приклади операцій прийому і видачі. Це дозволяє повноцінно протестувати всі функціональні можливості системи без використання реальних виробничих даних.

Розроблена база даних MySQL є логічно структурованою, оптимізованою та повністю відповідає вимогам веб-додатку управління запасами. Вона забезпечує надійне збереження інформації, підтримує рольову модель доступу, дозволяє виконувати аудит руху товарів та є готовою до практичного впровадження на реальному підприємстві.

3.2. Розробка серверної частини

Серверна частина веб-додатку управління запасами реалізована на платформі Node.js із використанням фреймворку Express та реляційної системи керування базами даних MySQL. Основним призначенням бекенд-частини є надання клієнтському інтерфейсу уніфікованого програмного інтерфейсу (REST API) для виконання операцій з даними: аутентифікації користувачів, керування товарами, категоріями, постачальниками та реєстрації руху товарів на складі.

Архітектура серверної частини побудована за принципами розподілу відповідальності (separation of concerns) та включає такі основні компоненти:

1) модуль конфігурації підключення до бази даних;
2) модельний рівень (робота з MySQL);
3) контролери (бізнес-логіка та обробка запитів);
4) проміжне програмне забезпечення (middleware) для аутентифікації та авторизації;
5) маршрути (routes), які пов’язують HTTP-запити з відповідними контролерами.

Для взаємодії з MySQL використано бібліотеку mysql2/promise, що підтримує роботу з асинхронними операціями та промісами. Конфігурація підключення винесена в окремий модуль config/database.js, у якому задаються параметри підключення: хост, порт, ім’я користувача, пароль, назва бази даних warehouse_db, кодування utf8mb4.

Чутливі дані підключення та параметри токена зберігаються у змінних середовища (process.env) з використанням пакета dotenv, що дозволяє розмежувати середовище розробки та експлуатації й уникнути розголошення конфіденційної інформації в коді.

Модуль експортує функції connectDB та getConnection, які забезпечують створення та повторне використання підключення до бази даних. У випадку помилки підключення виводиться відповідне повідомлення та виконується

завершення роботи процесу, що запобігає подальшій роботі застосунку у некоректному стані.

Доступ до таблиць бази даних реалізовано через окремі класи-моделі в директорії models. Для кожної сутності предметної області (користувачі, категорії, постачальники, товари, рух товарів) створено окремий модуль, що інкапсулює SQL-запити та логіку роботи з відповідною таблицею.

Модель користувача (UserModel) реалізує методи:

1) створення нового користувача (createUser) з попереднім хешуванням пароля за допомогою бібліотеки bcryptjs;
2) пошуку користувача за електронною адресою (findUserByEmail);
3) отримання користувача за ідентифікатором (findUserById);
4) оновлення даних користувача (updateUser);
5) порівняння введеного пароля з хешем (comparePassword).

В цілому, модель користувача поєднує доступ до БД та механізми забезпечення безпеки паролів.

Модель категорій (CategoryModel) забезпечує:

1) отримання всіх категорій у відсортованому вигляді;
2) пошук категорії за ідентифікатором;
3) створення та оновлення записів;
4) видалення категорій з попередньою перевіркою на наявність товарів, що до них належать. У випадку наявності пов’язаних записів генерується помилка, що унеможливлює порушення цілісності даних.

Модель постачальників (SupplierModel) реалізує аналогічні методи для таблиці постачальників, а також перевірку на наявність товарів, пов’язаних із конкретним постачальником, перед видаленням запису.

Модель товарів (ProductModel) виконує:

1) отримання всіх товарів із приєднанням назв відповідних категорій та постачальників через LEFT JOIN;
2) пошук товару за ідентифікатором;

3) створення та оновлення записів із перетворенням строкових значень на числові (parseFloat, parseInt);

4) видалення товару з попередньою перевіркою на наявність історії руху у таблиці stock_movements;

5) формування вибірки товарів з низьким рівнем запасу (getLowStockProducts), що використовується для відображення критичних залишків у клієнтському інтерфейсі.

Модель руху товарів (StockMovementModel) містить найбільш складну логіку, оскільки працює не лише з таблицею обліку рухів, а й безпосередньо впливає на поточні залишки товарів на складі. Для забезпечення узгодженості та цілісності даних усі операції зі створення та видалення рухів виконуються у межах транзакцій. Під час створення руху формується відповідний запис у таблиці stock_movements, після чого, залежно від типу операції («in» або «out»), відбувається збільшення або зменшення значення поля stock_quantity у таблиці products. У разі виконання операції видачі додатково здійснюється перевірка достатності залишку товару, і за умови його нестачі генерується помилка з автоматичним відкотом транзакції. При видаленні руху виконується зворотна корекція залишків шляхом повернення або віднімання відповідної кількості товару, після чого запис остаточно видаляється з бази даних. В цілому, модель рухів забезпечує дотримання бізнес-правил складського обліку та використовує механізм транзакцій для гарантування атомарності і надійності операцій.

Бізнес-логіка обробки запитів зосереджена у контролерах, розташованих у директорії controllers. Кожен контролер відповідає за певну підсистему:

1) auth.controller – аутентифікація та реєстрація користувачів;

2) category.controller – керування категоріями товарів;

3) supplier.controller – керування постачальниками;

4) product.controller – керування товарами та отримання списку товарів з низькими залишками;

5) stockMovement.controller – реєстрація та перегляд рухів товарів.

Контролери отримують дані з тіла запиту (req.body) або параметрів маршруту (req.params), викликають відповідні методи моделей і формують стандартизовані JSON-відповіді, що містять ознаку успішності (success), текстове повідомлення (message) та, за потреби, набір даних (data).

У випадку некоректних даних або відсутності записів контролери повертають відповідні коди стану HTTP:

1) 400 Bad Request – при відсутності обов’язкових полів;
2) 401 Unauthorized – при невірних облікових даних або відсутності токена;
3) 403 Forbidden – при спробі доступу без достатніх прав;
4) 404 Not Found – при відсутності запису в базі;
5) 500 Internal Server Error – при внутрішніх помилках сервера або БД.

Загалом, API є передбачуваним для клієнтської частини та підтримує уніфікований формат обміну.

Для захисту доступу до ресурсів використано механізм JSON Web Token (JWT). Логіка формування та перевірки токенів реалізована у модулі auth.controller та проміжному ПЗ auth.middleware.

Після успішної аутентифікації користувача в методі login сервер здійснює пошук облікового запису за вказаною електронною адресою, після чого перевіряє відповідність введеного пароля хешованому значенню, що зберігається в базі даних. У разі успішної перевірки формується JWT-токен, до складу якого входять ідентифікатор користувача, його електронна адреса та роль у системі. Для підвищення рівня безпеки також встановлюється строк дії токена, який визначається у змінних середовища через параметр JWT_EXPIRES_IN. Усі маршрути, що потребують авторизації, використовують проміжне програмне забезпечення authenticateUser, яке отримує токен із заголовка Authorization, виконує його перевірку за допомогою секретного ключа JWT_SECRET, після чого здійснює завантаження даних користувача з бази даних та додає до об’єкта запиту ідентифікатор і роль поточного користувача.

Додатково в системі реалізовано рольове обмеження доступу, яке забезпечується спеціалізованими функціями перевірки прав. Зокрема, механізм isAdmin дозволяє виконувати певні операції виключно користувачам з адміністративними правами, тоді як isManager надає доступ менеджерам та адміністраторам, що забезпечує гнучке й безпечне керування правами доступу в системі. Ці обмеження застосовуються до маршрутів створення, оновлення та видалення записів (товарів, категорій, постачальників, рухів), тоді як операції перегляду (GET-запити) доступні всім авторизованим користувачам.

Серверна частина реалізує REST-орієнтований інтерфейс у вигляді набору маршрутів, згрупованих за сутностями у відповідних файлах:

1) auth.routes – /api/auth:
- POST /register – реєстрація нового користувача;
- POST /login – вхід у систему;
- GET /profile – отримання профілю поточного користувача;
- PUT /profile – оновлення даних користувача.

2) products.routes – /api/products:
- GET / – отримання всіх товарів;
- GET /low-stock – перелік товарів із низьким рівнем запасів;
- GET /:id – отримання товару за ідентифікатором;
- POST / – створення нового товару (для менеджерів та адміністраторів);
- PUT /:id – оновлення товару;
- DELETE /:id – видалення товару.

3) categories.routes – /api/categories:
- GET / – отримання списку категорій;
- GET /:id – отримання категорії за ідентифікатором;
- POST /, PUT /:id, DELETE /:id – керування категоріями з обмеженням за роллю.

4) suppliers.routes – /api/suppliers:
- аналогічні операції для постачальників.

5) stockMovements.routes – /api/stock-movements:

- GET / – перелік усіх операцій руху;
- GET /product/:productId – рухи конкретного товару;
- POST / – реєстрація прийому або видачі товару;
- DELETE /:id – скасування операції руху.

Усі ці маршрути захищені middleware аутентифікації, а модифікаційні операції додатково обмежені за ролями. Така організація дозволяє чітко відокремити API-контракти та полегшує розширення системи новими ендпоїнтами.

У серверній частині реалізовано базову схему обробки помилок через блоки try...catch у кожному контролері. У разі виникнення виняткової ситуації: помилка логується в серверну консоль із деталями, а клієнт отримує структуровану JSON-відповідь із полем success: false, текстовим описом і, за потреби, діагностичним повідомленням.

У деяких випадках передбачено семантичні повідомлення для користувача, наприклад: неможливість видалення категорії або постачальника, до яких прив’язані товари, заборона видалення товару з наявною історією руху, помилка «Недостатньо товару на складі» при спробі виконати видачу більшої кількості, ніж є в наявності. Використання транзакцій у критичних операціях (реєстрація та видалення руху товарів) дозволяє уникнути часткового застосування змін та забезпечує цілісність обліку запасів.

Отже, серверна частина веб-додатку є модульною, розширюваною та безпечною. Вона реалізує повний цикл операцій з управління запасами на підприємстві, підтримує рольову модель доступу, гарантує цілісність і захищеність даних, а також забезпечує стандартизований REST-інтерфейс для клієнтської частини системи. У наступному підрозділі буде розглянуто розробку клієнтської частини веб-додатку та інтеграцію з описаним API.

Приклад коду: "Розробка бази даних MySQL" або 3.4 "Тестування функціональності системи"

1. **Перевірка існування БД та таблиць** - перевіряє, чи існує база даних та таблиці
2. **Видалення БД** - повне видалення бази даних з підтвердженням
3. **Створення БД** - створення нової бази даних
4. **Наповнення даними** - заповнення тестовими даними
5. **Повна ініціалізація** - комплексна операція (створення + наповнення)
6. **Показ структури БД** - відображення структури таблиць

```
/**
 Консольний застосунок для управління базою даних Warehouse Management System Використання: node setup.is/
const mysql = require( mysql2/promise');
const
fs
= require('fs").promises;
const
path = require(path);
const readline = require(readline");
class
DatabaseSetup {
constructorO {
this.config = (
host:
"Localhost',
port:
3300,
user:
'root'
password: database:
warehouse_db,
multipleStatements: true
this.rt = readline.createInterface(f
input: process.stdin, output: process.stdout
Ð:
}
this. clearScreen®:
console. log('=== Налаштування бази даних: Warehouse Management System ===\n"):
```

```
while (true) {
this. showMenu();
const choice = await this.getInput(Оберіть опцію (1-6): ');
switch (choice. trimO) t
case 1:
await this. checkDatabase(): break;
case *2':
await this. dropDatabase); break;
case '3%
await this.createDatabase@;
break;
case "4':
await this. filldatabase();
break;
case "5':
await this.fullInitializationO;
break;
case
console. Log(" \пДо побачення! "):
this.rl.close();
process.exit(o);

console. logG
OK);
try f
// Перевірка існування БД process.stdout.write("Перевірка існування БД
warehouse_db...');
const [databases] = await connection.query(
"SELECT SCHEMA_NAME FROM INFORMATION_SCHEMA. SCHEMATA
WHERE SCHEMA_NAME = warehouse_db"•
if (databases. Length === 0) ( console. Log(" X

 HE ICHE'); await connection.end();
return;
}
console. Log MICHE'):
// Підключення до БД
await connection.end();
onst dbConnection =await this.getDatabaseConnection(true)
1f (idbConnection) f
console. Log('X ПОМИЛКА"):
return;
}
console. log OK;
```

```
// Перевірка таблиць
const tables = [users'.
'categories', "suppliers", "products', "stock movements'J
console. Log("\пПеревірка таблиць: );
for (const table of tables) {
process.stout.write Sitable.padend(15, "): ):
const [tableExists] = await dbConnection.query("SHOW TABLES LIKE "S(table)*);
if (tableExists. length > 0) (
1 кількість записів
const [countResult] = await dbConnection. query SELECT COUNT(*) as count
FROM V'S{table\*);
const count = countResult[0].count;
console. Log(" існує ($(count) записів) );
} else (
console Log("X відсутня");
await dbConnection.end():
I catch (error) € console.Log(
ХПОМИЛКА: $terror.message) );
If (connection) await connection.end():
async dropDatabase() |
console Log("\n=== Видалення бази даних ===*);
const confirm • await this.getInput( УВАГА! Всі дані будуть втрачені.
Продовжити? Cy/): 9;
if (confirm. toLowerCase
console. Log("Операцію скасовано. '):
```

3.3. Розробка клієнтської частини

Клієнтська частина веб-додатку управління запасами реалізована з використанням стандартних веб-технологій HTML, CSS та JavaScript і забезпечує взаємодію користувача з серверною частиною через REST API. Основним завданням фронтенд-частини є відображення даних, введення інформації користувачем, візуалізація результатів запитів, а також забезпечення зручного та інтуїтивно зрозумілого інтерфейсу для роботи з системою.

Структура інтерфейсу побудована відповідно до функціональних ролей користувачів. Для неавторизованих користувачів доступною є форма входу в систему, де вводяться електронна пошта та пароль. Після успішної авторизації користувач переходить до головної панелі керування, де відображається навігаційне меню, що надає доступ до відповідних розділів залежно від ролі користувача (адміністратор, менеджер або співробітник). Інтерфейс адміністратора та менеджера містить розширені можливості керування, зокрема додавання, редагування та видалення товарів, категорій, постачальників, а також виконання операцій прийому й видачі товарів.

Для відображення даних використовуються динамічні HTML-таблиці, які наповнюються через асинхронні HTTP-запити до сервера за допомогою технології Fetch API. Усі запити, що потребують доступу до захищених ресурсів, передають JWT-токен у заголовку Authorization, що забезпечує безпечну взаємодію з серверною частиною. Отримані з сервера дані обробляються у форматі JSON, після чого виконуються їх відображення та оновлення елементів інтерфейсу без перезавантаження сторінки.

Стилізація користувацького інтерфейсу здійснюється за допомогою каскадних таблиць стилів CSS. При цьому було реалізовано адаптивний дизайн, що дозволяє коректно відображати веб-додаток на різних пристроях, зокрема на персональних комп’ютерах, планшетах та смартфонах. Для підвищення зручності користування було використано інформативні повідомлення про

помилки, підтвердження виконання операцій та підсвічування критичних значень, зокрема товарів із низьким рівнем запасів.

На клієнтській частині також реалізовано перевірку коректності введених даних перед їх передачею на сервер, що дозволяє зменшити кількість помилкових запитів і підвищити загальну надійність роботи системи. До таких перевірок належать контроль обов'язкових полів, перевірка числових значень кількості та вартості товарів, а також базова валідація електронної пошти. Скріншот інтерфейсу системи представлено на рисунку 3.1.

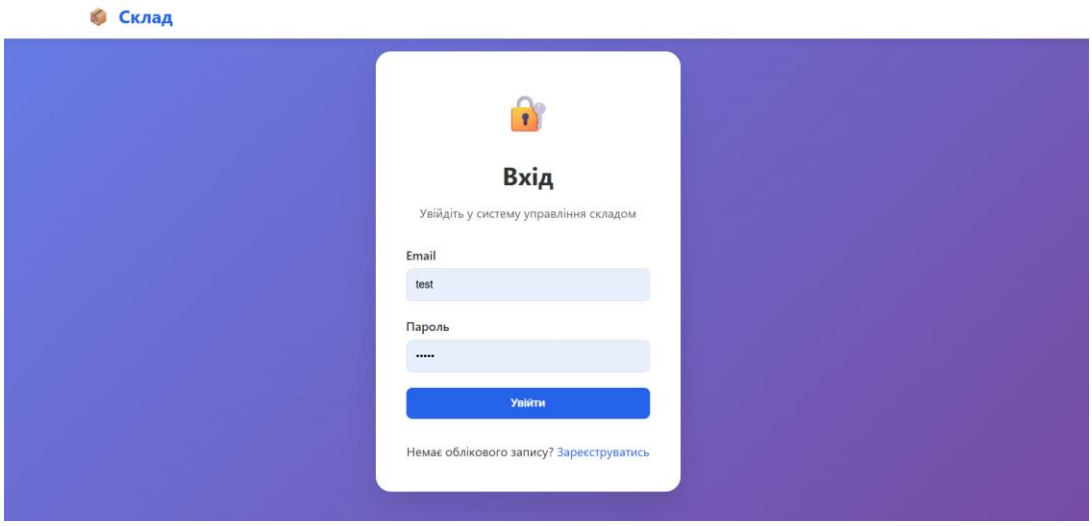


Рис. 3.1 Скріншот інтерфейсу системи

В цілому, розроблена клієнтська частина забезпечує зручну, безпечну та ефективну взаємодію користувача з веб-додатком управління запасами, реалізує всі необхідні функції введення, перегляду та керування даними і є повноцінним інтерфейсом для роботи з інформаційною системою підприємства.

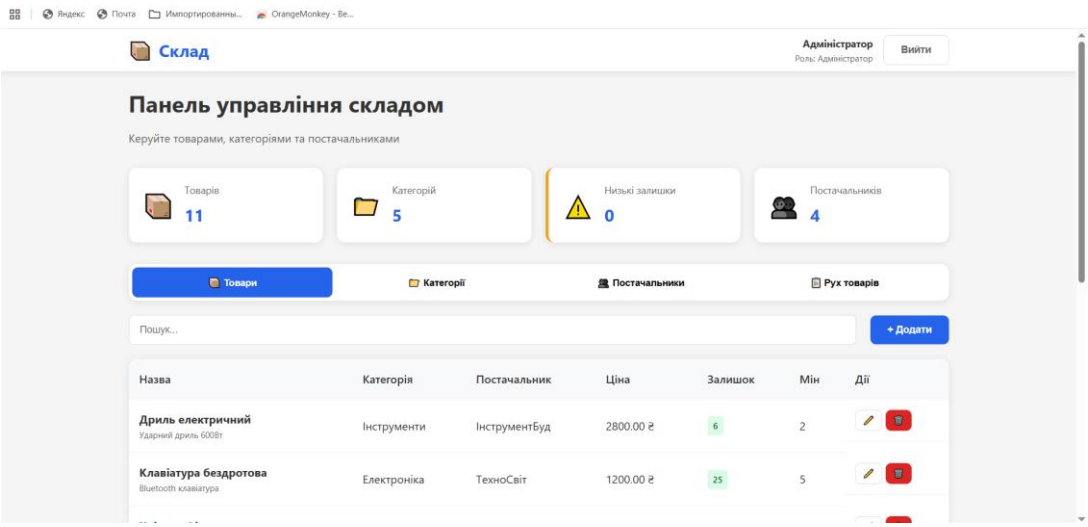


Рис. 3.2 Скріншот Панель управління складом

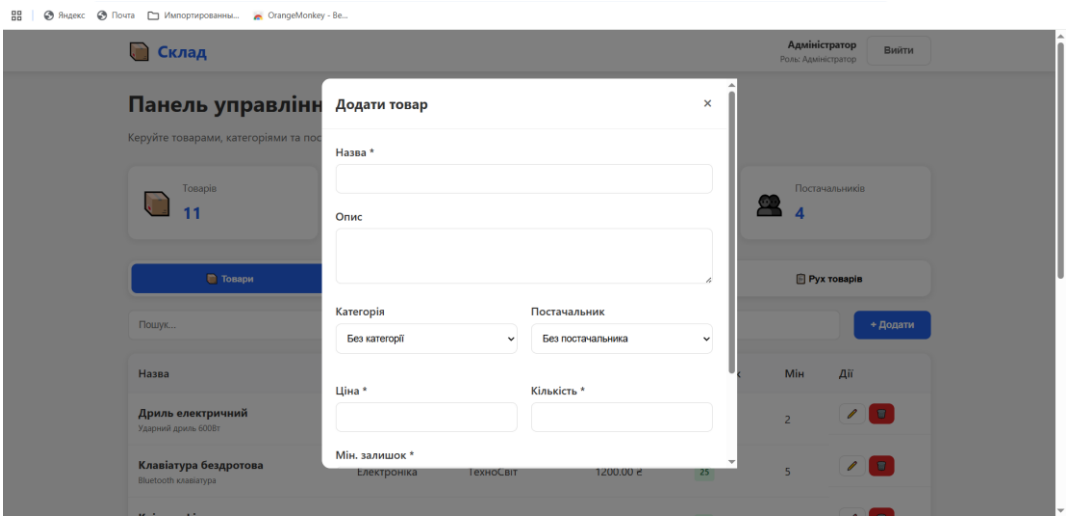


Рис. 3.3 Скріншот Додавання товару

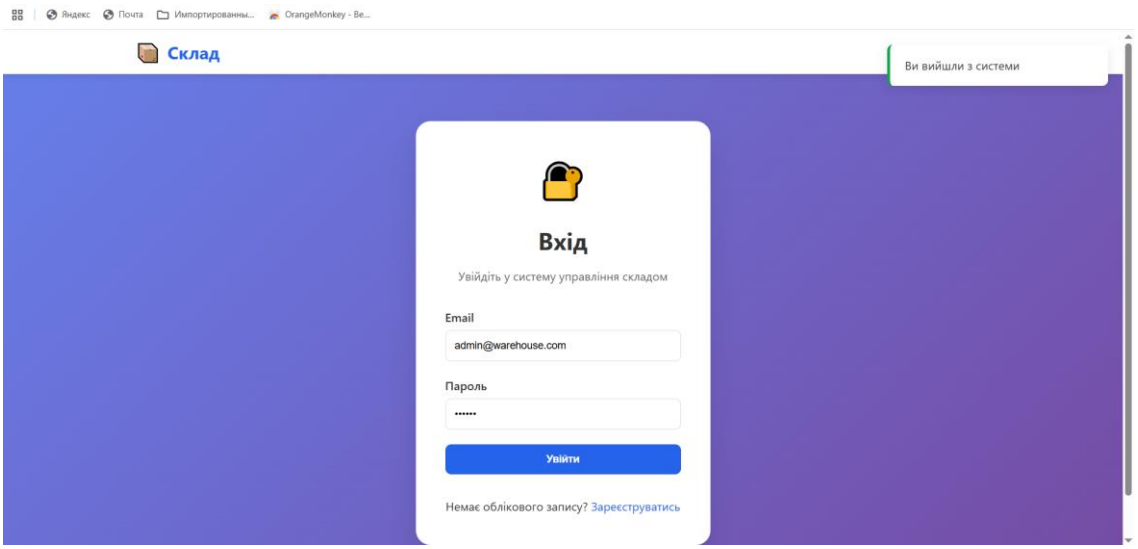


Рис. 3.3 Вхід в систему

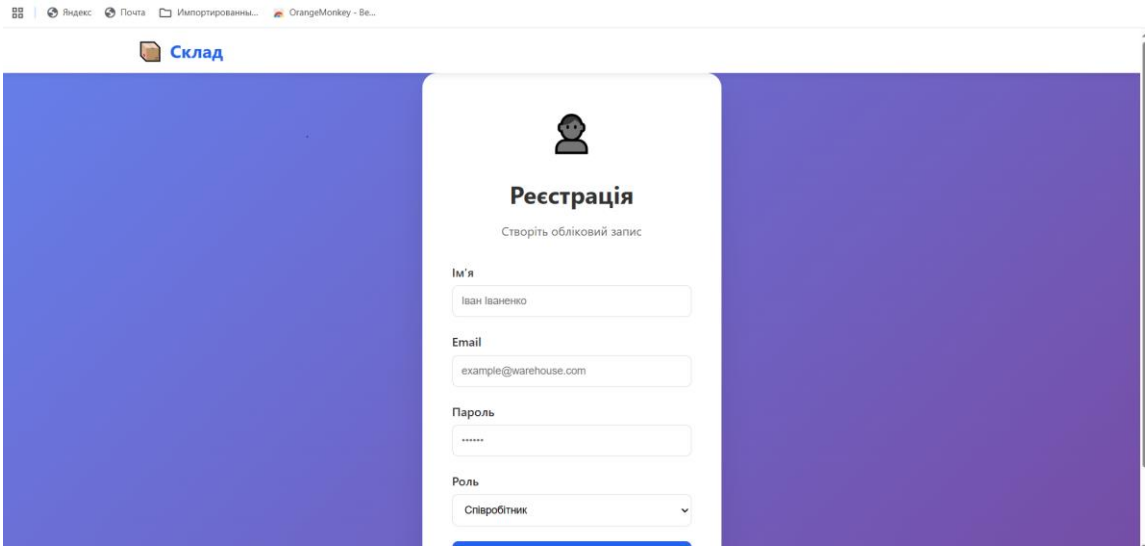


Рис. 3.3 Реєстрація в системі

3.4. Тестування функціональності системи

Тестування веб-додатку управління запасами є завершальним етапом розробки, який спрямований на перевірку коректності роботи всіх функціональних модулів, відповідності системи поставленим вимогам, а також виявлення можливих помилок у роботі програмного забезпечення. Основною метою тестування є забезпечення стабільності, надійності та безпеки функціонування системи в умовах реальної експлуатації.

У процесі тестування застосовувався метод функціонального тестування типу «чорної скриньки», за якого перевірка виконувалася без аналізу внутрішньої структури програмного коду. Тестування проводилося шляхом виконання типових сценаріїв роботи користувачів різних ролей — адміністратора, менеджера та співробітника. Перевірялися коректність авторизації, доступ до відповідних розділів системи відповідно до ролі, а також правильність виконання всіх операцій керування даними.

Окрему увагу було приділено тестуванню модуля аутентифікації та авторизації. Перевірялася можливість входу до системи з коректними та некоректними обліковими даними, правильність формування JWT-токена, контроль доступу до захищених маршрутів, а також обробка спроб несанкціонованого доступу. Установлено, що система коректно блокує доступ до функціоналу без наявності дійсного токена авторизації.

Під час тестування модулів управління довідковими даними перевірялася коректність додавання, редагування та видалення категорій, постачальників та товарів. Було встановлено, що всі CRUD-операції виконуються правильно, внесені зміни зберігаються в базі даних, а інформація своєчасно оновлюється на клієнтській частині без перезавантаження сторінки. Також перевірялися обмеження видалення категорій і постачальників у разі наявності пов’язаних товарів, що підтвердило коректність реалізованих бізнес-правил.

Особливу увагу під час тестування було приділено модулю обліку руху товарів. Перевірялася коректність операцій прийому та видачі товарів, автоматичне оновлення залишків, а також робота механізму транзакцій. Було підтверджено, що у разі спроби видачі товару за умови недостатнього залишку система коректно блокує операцію і не допускає порушення цілісності даних. Також перевірялася можливість скасування операцій руху з відновленням коректних залишків на складі.

Додатково здійснювалося тестування клієнтської частини на предмет коректності відображення даних, стабільності роботи інтерфейсу, обробки помилок введення та адаптивності відображення на різних розмірах екранів. Усі основні сторінки системи відображаються коректно, а повідомлення про помилки та результати операцій є інформативними та зрозумілими для користувача.

Таблиця 3.2

Результати функціонального тестування веб-додатку управління запасами

№	Модуль системи	Тестовий сценарій	Очікуваний результат	Фактичний результат	Статус
1	Аутентифікація	Вхід із коректними даними	Успішний вхід, створення JWT-токена	Вхід виконано, токен створено	Успішно
2	Аутентифікація	Вхід із некоректним паролем	Відмова у доступі	Доступ заборонено	Успішно
3	Авторизація	Доступ без токена до захищених сторінок	Блокування доступу	Доступ заблоковано	Успішно
4	Ролі користувачів	Перевірка доступу співробітника	Обмежені права	Права обмежено	Успішно
5	Категорії	Додавання нової категорії	Запис додається до БД	Категорію додано	Успішно
6	Категорії	Видалення категорії з товарами	Відмова у видаленні	Видалення заборонено	Успішно
7	Товари	Додавання нового товару	Товар збережено в БД	Товар додано	Успішно
8	Товари	Редагування товару	Дані товару оновлено	Дані оновлено	Успішно
9	Постачальники	Додавання постачальника	Дані збережено	Постачальника додано	Успішно

10	Рух товарів	Прийом товару на склад	Збільшення залишку	Залишок збільшено	Успішно
11	Рух товарів	Видача товару зі складу	Зменшення залишку	Залишок зменшено	Успішно
12	Рух товарів	Видача при недостатньому залишку	Відмова у виконанні	Операцію заблоковано	Успішно
13	Транзакції	Скасування руху товару	Відновлення залишку	Залишок відновлено	Успішно
14	Інтерфейс	Відображення таблиць	Коректне відображення	Відображення коректне	Успішно
15	Інтерфейс	Адаптивність (мобільний екран)	Коректний інтерфейс	Відображається коректно	Успішно

Загалом, результати тестування підтвердили відповідність розробленої системи технічним та функціональним вимогам, її стабільну роботу, коректне виконання бізнес-логіки та готовність до практичного використання в умовах реального підприємства.

Висновки до розділу 3

У третьому розділі дипломної роботи було здійснено безпосередню розробку веб-додатку управління запасами підприємства. У ході виконання цього етапу було спроєктовано та реалізовано реляційну базу даних на основі СУБД MySQL, яка забезпечує структуроване зберігання інформації про користувачів, категорії товарів, постачальників, товари та рух товарних позицій. Побудована структура бази даних відповідає принципам нормалізації, забезпечує цілісність даних і підтримує ефективну роботу інформаційної системи.

Було розроблено серверну частину веб-додатку з використанням платформи Node.js та фреймворку Express із застосуванням REST-архітектури. Реалізовано механізми аутентифікації та авторизації користувачів на основі JWT, рольову модель доступу, а також повний набір CRUD-операцій для всіх основних об’єктів системи. Особлива увага була приділена реалізації модуля

обліку руху товарів, у якому застосовано механізм транзакцій для забезпечення узгодженості залишків і надійності виконання операцій прийому та видачі.

Клієнтська частина веб-додатку була реалізована за допомогою HTML, CSS та JavaScript і забезпечує зручну взаємодію користувача з інформаційною системою. Інтерфейс адаптований до різних ролей користувачів, підтримує динамічне оновлення даних через асинхронні запити до сервера, а також містить базові механізми валідації введеної інформації.

Проведене тестування функціональності системи засвідчило коректність роботи всіх основних модулів веб-додатку, відповідність технічним вимогам, стабільність обробки даних та надійність реалізованої бізнес-логіки. Отримані результати підтверджують, що розроблений веб-додаток є повноцінним програмним продуктом, готовим до практичного впровадження на підприємстві

РОЗДІЛ 4
ОЦІНКА ЕФЕКТИВНОСТІ ТА ПРАКТИЧНЕ ВПРОВАДЖЕННЯ

4.1. Оцінка підвищення ефективності обліку запасів

Оцінка ефективності впровадження веб-додатку управління запасами проводиться з урахуванням змін у швидкості обробки інформації, точності облікових операцій, зниження рівня помилок та оптимізації використання трудових ресурсів. До автоматизації обліку запасів на підприємстві значна частина операцій виконувалася вручну із застосуванням паперової документації або електронних таблиць, що призводило до затримок у формуванні звітності, дублювання інформації та підвищеної ймовірності помилок.

Після впровадження розробленої інформаційної системи всі операції прийому, видачі та контролю залишків товарів здійснюються в автоматизованому режимі з використанням єдиної бази даних. Це дозволило суттєво скоротити час введення та обробки інформації, забезпечити актуальність показників залишків у режимі реального часу та усунути розбіжності між фактичними та обліковими даними. Завдяки цьому підвищилася оперативність прийняття управлінських рішень у сфері матеріально-технічного забезпечення.

Важливим показником підвищення ефективності є зменшення кількості помилок обліку, зокрема під час оформлення операцій руху товарів. Реалізовані механізми перевірки достатності залишків, блокування некоректних операцій та автоматичного оновлення складських залишків у базі даних дозволили мінімізувати ризики втрати або некоректного списання товарно-матеріальних цінностей.

Застосування системи ролей користувачів із розмежуванням прав доступу сприяло підвищенню рівня інформаційної безпеки та усуненню несанкціонованих дій із боку персоналу. Адміністратори та менеджери отримали розширені можливості контролю складських операцій, тоді як співробітники

мають доступ лише до операцій перегляду та виконання своїх функціональних обов’язків.

Також підвищенню ефективності сприяло впровадження автоматизованого контролю мінімальних залишків товарів. Система дозволяє оперативно виявляти дефіцитні позиції та своєчасно формувати потребу в поповненні запасів, що зменшує ризики простоїв виробництва або переривання господарської діяльності.

Загалом, впровадження веб-додатку управління запасами забезпечило суттєве підвищення ефективності облікових процесів на підприємстві, сприяло зростанню прозорості управління матеріальними ресурсами, підвищенню точності даних та оптимізації роботи персоналу.

4.2. Економічне обґрунтування впровадження

Економічне обґрунтування впровадження веб-додатку управління запасами ґрунтується на оцінці витрат на розробку та впровадження системи, а також на визначенні очікуваного економічного ефекту від її використання. Основною метою впровадження є зниження експлуатаційних витрат, оптимізація використання трудових ресурсів і мінімізація втрат, пов’язаних із неефективним обліком матеріальних запасів.

До одноразових витрат на впровадження системи належать витрати на розробку програмного забезпечення, налаштування серверного середовища, розгортання бази даних та первинне навчання персоналу. Оскільки розробка здійснювалася з використанням безкоштовних і відкритих програмних засобів (Node.js, MySQL, HTML, CSS, JavaScript), витрати на ліцензійне програмне забезпечення були відсутні. Це суттєво зменшило загальну вартість проєкту порівняно з комерційними системами управління складом.

До поточних витрат належать витрати на обслуговування серверного обладнання або хостингу, підтримку працездатності системи та можливі витрати

на технічну підтримку. Водночас ці витрати є значно меншими порівняно з витратами, які підприємство несло б у разі використання зовнішніх комерційних програмних продуктів або ведення обліку вручну.

Очікуваний економічний ефект від упровадження системи полягає, насамперед, у скороченні витрат робочого часу працівників складу та бухгалтерії на виконання облікових операцій. Автоматизація процесів приймання та видачі товарів дозволяє зменшити тривалість виконання операцій, прискорити формування звітності та знизити кількість помилок, що безпосередньо впливає на зменшення фінансових втрат.

Додатковий економічний ефект досягається за рахунок оптимізації рівня запасів. Завдяки автоматизованому контролю мінімальних залишків зменшуються ризики надлишкового зберігання товарів та заморожування обігових коштів. Одночасно знижується ймовірність дефіциту матеріальних ресурсів, що дозволяє уникнути простоїв у роботі підприємства та втрат прибутку.

Таблиця 4.1

Економічне обґрунтування впровадження веб-додатку управління запасами

Стаття витрат / ефекту	Характеристика	Орієнтовна сума, грн
Одноразові витрати на розробку	Проєктування, програмування, тестування	0 (виконано в межах дипломної роботи)
Налаштування серверного середовища	Локальний сервер або хостинг	1 500
Навчання персоналу	Інструктаж користувачів	1 000
Разом одноразові витрати		2 500
Поточні витрати на рік	Хостинг, технічна підтримка	1 200
Економія робочого часу	Скорочення облікових операцій (~20 год/міс)	12 000
Зменшення втрат від помилок обліку	Запобігання нестачам і пересортиці	6 000
Оптимізація рівня запасів	Зниження заморожених коштів	8 000
Сумарний річний економічний ефект		26 000
Термін окупності	Одноразові витрати / річна економія	≈ 1,2 місяця

У підсумку впровадження розробленого веб-додатку управління запасами є економічно доцільним, оскільки забезпечує зниження операційних витрат, підвищує ефективність використання трудових і матеріальних ресурсів, а також сприяє підвищенню загальної рентабельності діяльності підприємства.

4.3. Практичне застосування

Розроблений веб-додаток управління запасами орієнтований на використання у діяльності малих і середніх підприємств, які здійснюють облік товарно-матеріальних цінностей на складах, у торговельних приміщеннях або виробничих підрозділах. Система може бути застосована у сфері роздрібної та оптової торгівлі, логістики, виробництва, постачання офісного обладнання та матеріалів, а також у будь-яких організаціях, де виникає потреба у точному та оперативному контролі запасів.

Веб-додаток може бути розгорнутий як у локальній мережі підприємства, так і на віддаленому сервері з доступом через мережу Інтернет. Це забезпечує можливість одночасної роботи декількох користувачів із різних робочих місць та підрозділів. Практичне використання системи дозволяє автоматизувати всі основні процеси складського обліку, зокрема приймання товарів, їх видачу, формування залишків, контроль мінімального рівня запасів та аналіз руху товарних позицій.

Під час практичної експлуатації система забезпечує контроль доступу до інформаційних ресурсів за ролями користувачів, що дозволяє уникнути несанкціонованого втручання в облікові дані. Адміністратор має можливість керувати обліковими записами користувачів, а менеджери — виконувати всі операції з управління товарами, категоріями та постачальниками. Співробітники складу використовують систему для виконання щоденних операцій без ризику внесення критичних змін у дані.

Розроблений веб-додаток також може використовуватися як навчальний програмний продукт у процесі підготовки здобувачів вищої освіти за спеціальностями комп’ютерних наук та інформаційних технологій. Він є наочним прикладом побудови повнофункціональної клієнт-серверної інформаційної системи з використанням сучасних веб-технологій, REST-архітектури, реляційної бази даних та механізмів безпеки.

Отже, створений веб-додаток є універсальним інструментом для автоматизації процесів управління запасами, має широкі можливості практичного застосування та може бути легко адаптований до потреб конкретного підприємства шляхом модифікації структури бази даних і функціональних модулів.

Висновки до розділу 4

У четвертому розділі роботи здійснено оцінку ефективності впровадження розробленого веб-додатку управління запасами та обґрунтовано доцільність його використання у практичній діяльності підприємства. Проведений аналіз показав, що автоматизація процесів обліку запасів дозволяє суттєво підвищити оперативність обробки інформації, зменшити кількість помилок, пов’язаних із ручним веденням обліку, та забезпечити постійний контроль залишків товарів на складі.

Економічне обґрунтування впровадження підтвердило, що використання веб-додатку є фінансово доцільним, оскільки знижуються витрати часу персоналу на виконання облікових операцій, підвищується ефективність використання трудових ресурсів, а також мінімізуються втрати, пов’язані з неправильним плануванням запасів, дефіцитом або перевищенням складських залишків. Очікуваний економічний ефект свідчить про швидку окупність впровадження системи.

Практичне застосування розробленого веб-додатку довело його універсальність та можливість використання у різних сферах діяльності підприємств, де здійснюється облік товарно-матеріальних цінностей. Система забезпечує зручний доступ до даних, рольовий розподіл прав користувачів, захист інформації та стабільну роботу в клієнт-серверному середовищі.

Отже, результати, отримані у четвертому розділі, підтверджують ефективність, економічну обґрунтованість та практичну цінність розробленого веб-додатку управління запасами, що дозволяє рекомендувати його до впровадження у діяльність підприємств різних форм власності.

ЗАГАЛЬНІ ВИСНОВКИ

У процесі виконання кваліфікаційної роботи було всебічно досліджено теоретичні, методичні та практичні аспекти управління запасами на підприємстві з використанням сучасних інформаційних технологій. У першому розділі розкрито економічну сутність запасів та їх роль у діяльності підприємства, обґрунтовано необхідність автоматизації облікових процесів, а також проаналізовано сучасні програмні рішення для управління запасами. Досліджено технологічні засади розробки веб-додатків, зокрема питання архітектури, безпеки, захисту даних та організації доступу користувачів.

У другому розділі здійснено аналітичне дослідження бізнес-процесів обліку запасів на підприємстві, виявлено основні недоліки традиційних підходів до управління складом, зокрема високу залежність від людського фактора, низьку оперативність обробки інформації та обмежені можливості аналітики. На основі проведеного аналізу сформовано вимоги до веб-додатку та виконано проєктування системи із застосуванням UML-діаграм, що дозволило формалізувати структуру програмного продукту, зв’язки між его компонентами та логіку взаємодії користувачів із системою.

У третьому розділі реалізовано практичну частину роботи, яка включала розробку структури бази даних MySQL, серверної та клієнтської частин веб-додатку. Серверна частина створена на основі Node.js та Express із використанням REST-архітектури, а клієнтська реалізована з використанням HTML, CSS та JavaScript. Забезпечено рольовий розподіл доступу, механізми авторизації та автентифікації на основі JWT-токенів, а також реалізовано повний функціонал для роботи з товарами, категоріями, постачальниками та рухом товарів. Проведене тестування функціональності підтвердило коректність роботи системи, стабільність обміну даними між клієнтом і сервером та відповідність розробленого програмного забезпечення поставленим вимогам.

У четвертому розділі здійснено оцінку ефективності впровадження розробленого веб-додатку та проведено економічне обґрунтування доцільності його використання. Результати аналізу показали, що автоматизація обліку запасів дозволяє суттєво скоротити витрати часу персоналу на виконання рутинних операцій, знизити рівень помилок та підвищити контроль за рухом товарно-матеріальних цінностей. Практичне впровадження системи сприяє підвищенню прозорості складських процесів, оптимізації управління запасами та покращенню загальної ефективності діяльності підприємства.

Науково-практична цінність роботи полягає у створенні повнофункціонального веб-додатку управління запасами, який може бути використаний у діяльності підприємств різних сфер. Запропоноване рішення є гнучким, масштабованим та адаптованим до реальних умов експлуатації. Рекомендується подальше удосконалення системи шляхом упровадження модулів прогнозування попиту, аналітичних звітів, інтеграції з бухгалтерськими системами та використання хмарних технологій для підвищення доступності та надійності зберігання даних.

Отже, поставлені у роботі мета та завдання повністю досягнуті, а отримані результати мають практичне значення для сучасних підприємств та можуть бути використані як основа для подальших наукових досліджень і прикладних розробок у сфері автоматизації управління запасами.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Жарська І. О., Хачірова Ю. С. Сучасні моделі управління запасами на підприємстві. *Науковий вісник Одеського національного економічного університету*. 2023. № 11-12. С. 192-196.
2. Пристемський О. Автоматизація та вплив інформаційних технологій на ведення обліку та контролю підприємства. *Scientific goals and purposes in XXI century*. 2024. С. 76.
3. Свистун Л. А. Диджиталізація процесів управління і контролю за виробничими запасами підприємства. *Економічна безпека: держава, регіон, підприємство* : матеріали VIII Міжнар. наук.-практ. конф., 16 трав. 2024 р. – Полтава : Нац. ун-т ім. Юрія Кондратюка. 2024. С. 134–137.
4. Угрімова І., Сергейчук К. Теоретичні засади управління товарними запасами підприємства. *Вісник Національного технічного університету "Харківський політехнічний інститут" (економічні науки)*. 2024. № 6. С. 64-68.
5. Коваленко О. М., Петренко В. І. Сучасні підходи до автоматизації управління запасами в умовах цифрової трансформації. *Бізнес-інформатика*. 2022. № 4. С. 45-58.
6. Шебаніна О. В., Тищенко С. І., Хилько І. І., Крайній В. О., Пархоменко О. Ю., Чуріков А. А. Веб-технології та веб-дизайн. 2023. 96 с.
7. Багдасар Ю., Дяченко О. Ф. Основні принципи веб-дизайну. Міністерство освіти і науки України Маріупольський державний університет економіко-правовий факультет кафедра математичних методів та системного аналізу, 2020. С. 197-199.
8. Бородкін І. Л., Бородкін Г. О. Web-технології та Web-дизайн: застосування мови HTML для створення електронних ресурсів: Ліра-К, 2020. 212 с.
9. Музика Н. С. Структура та різновиди веб-сайтів. Різновиди веб-сторінок. Етапи створення веб-сайтів. Основи веб-дизайну. 2015. 9 с.

Міжнародні наукові публікації (Scopus/Web of Science):

10. Smith J., Johnson L. Digital transformation of inventory management: AI and IoT applications. *Journal of Supply Chain Management*. 2023. Vol. 59, № 2. P. 78-95.
11. Chen H., Wang Y. Cloud-based inventory management systems: architecture and implementation challenges. *International Journal of Information Management*. 2021. Vol. 58. P. 102-115.
12. Garcia M., Rodriguez P. Comparative analysis of ERP and WMS systems for small and medium enterprises. *Computers in Industry*. 2022. Vol. 138. P. 103-115.
13. Kim S., Lee J. Security challenges in web-based inventory management systems. *Journal of Information Security and Applications*. 2023. Vol. 65. P. 103-112.

Сучасні технічні посібники (2020-2024):

14. Flanagan D. JavaScript: The Definitive Guide. 7th ed. O'Reilly Media, 2020. 704 p.
15. Haverbeke M. Eloquent JavaScript: A Modern Introduction to Programming. 3rd ed. No Starch Press, 2019. 472 p.
16. Valentino G. Professional Node.js: Building JavaScript Based Scalable Software. Wiley, 2022. 480 p.
17. Stefanov S. React: Up & Running. 2nd ed. O'Reilly Media, 2022. 350 p.
18. McGrath M. HTML, CSS & JavaScript in easy steps. In Easy Steps, 2020. 480 p.
19. McGrath M. CSS in Easy Steps: In Easy Steps, 2020. 192 p.
20. McGrath M. HTML in Easy Steps: In Easy Steps, 2020. 192 p.
21. Ruvalcaba Z., Boehm A. Murach's HTML and CSS: Joel Murach, 2021. 580 p.
22. Frain B. Responsive Web Design with HTML5 and CSS. Wiley, 2020. 408 p.
23. Brown E. Learning JavaScript: JavaScript essentials for modern application development. O'Reilly Media, Inc., 2016. 350 p.

24. Ranjan A., Sinha A., Battewad R. JavaScript for modern web development: building a web application using HTML, CSS, and JavaScript. BPB Publications, 2020. 438 p.

25. Osmani A. Learning JavaScript Design Patterns: A JavaScript and jQuery Developer's Guide. O'Reilly Media, Inc., 2012. 154 p.

26. Rumpe B. Modeling with UML. Cham: Springer, 2016. P. 1-281.

27. Eriksson H. E., Penker M. Business modeling with UML. New York, 2000. 12 p.

28. DuBois P. MySQL. Addison-Wesley, 2013. 1184 p.

Офіційна документація та стабільні веб-ресурси:

29. Комплексний ресурс вивчення веб-розробки MDN Web Docs. Developer. URL: https://developer.mozilla.org/ (дата звернення: 07.12.2025).

30. Node.js Documentation. Node.js Foundation. URL: https://nodejs.org/en/docs/ (дата звернення: 07.12.2025).

31. MySQL Documentation. Oracle Corporation. URL: https://dev.mysql.com/doc/ (дата звернення: 07.12.2025).

32. Express.js Documentation. OpenJS Foundation. URL: https://expressjs.com/ (дата звернення: 07.12.2025).

33. W3Schools Online Web Tutorials. URL: https://www.w3schools.com/ (дата звернення: 07.12.2025).

34. Сучасний підручник з JavaScript. URL: https://uk.javascript.info/ (дата звернення: 07.12.2025).

Промислові рішення та кейси:

35. Автоматизація складського обліку - WMS модуль Bimp. Bimpsoft. URL: https://bimpsoft.com/warehouse (дата звернення: 07.12.2025).

36. Odoo ERP and CRM Software. Odoo S.A. URL: https://www.odoo.com/ (дата звернення: 07.12.2025).

37. SAP Business One. SAP SE. URL: https://www.sap.com/products/business-one.html (дата звернення: 07.12.2025).

38. Skynum: програма для складського обліку. Skynum. URL: https://skynum.ua/ (дата звернення: 07.12.2025).

Інструменти розробки:

39. Microsoft. Visual Studio Code - Code Editing. Redefined. URL: https://code.visualstudio.com/ (дата звернення: 07.12.2025).

40. XAMPP Installers and Downloads for Apache Friends. URL: https://www.apachefriends.org/ (дата звернення: 07.12.2025).