ХЕРСОНСЬКИЙ НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ

ФАКУЛЬТЕТ ІНЖЕНЕРІЇ ТА ТРАНСПОРТУ

КАФЕДРА АВТОМАТИЗАЦІЇ, РОБОТОТЕХНІКИ І МЕХАТРОНІКИ

Пояснювальна записка

до кваліфікаційної роботи бакалавра

на тему: Розробка програмно-апаратної моделі складського обліку на приватному підприємств «Анжіо»»

« Development of a software and hardware model of warehouse accounting at the private enterprise 'Anzhio'»

Виконав: студент 4 курсу, групи 4А
спеціальність 151 – «Автоматизація та комп’ютерно-інтегровані технології»
Окуневич Б.О.

Керівник: к. т. н. доцент Поліщук В. М.

Рецензент_____
(прізвище та ініціали)

Херсон – 2025 рік

Форма № Н-9.01

Херсонський національний технічний університет

Факультет Інженерії та транспорту

Кафедра Автоматизації, робототехніки і мехатроніки

Ступінь вищої освіти бакалавр

Спеціальність 151 – «Автоматизація та комп'ютерно-інтегровані технології»

ЗАТВЕРДЖУЮ:
Завідувач кафедри автоматизації,
робототехніки і мехатроніки
Сєліверстов І. А.
«___» _____ 2025 р.

З А В Д А Н Н Я

на дипломну роботу студенту

Окуевичу Богдану Олеговичу

1. Тема проекту: Розробка програмно-апаратної моделі складського обліку на приватному підприємств «Анжіо

керівник проекту: к. т. н. доцент Поліщук В. М.

затверджена наказом вищого навчального закладу від 27.01.2025 р. № 132-с

2. Строк подання студентом проекту «16» червня 2025 р.

3. Вихідні дані до проекту: Розробити автоматизовану програмно-апаратну модель складського обліку на приватному підприємств «Анжіо» згідно технічного завдання

4. Зміст розрахунково-пояснювальної записки: 1) аналіз теоретичих основ автоматизації складського обліку 2) аналіз діяльності пп «анжіо» та постановка задачі автоматизації 3) технічна реалізація системи управління складом; Висновки

5. Перелік графічного матеріалу

Додаток А лістинг програми

Додаток Б архітектура бд

Додаток В модуль «управління складами та слотами»

Додаток Г модуль «управління операціями» частина 1

Додаток Д модуль «управління операціями» частина 2

Додаток Е модуль «інвентаризація та залишки»

Додаток Ж тестування системи

6. Консультанти розділів проекту

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Основна частина	к. т. н. доцент Поліщук В. М.		

7. Дата видачі завдання «27» січня 2025 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломного проекту	Строк виконання етапів проекту	Примітка
1	Підбір і огляд літератури	12.03.2025	
2	Аналіз теоретичних основ автоматизації складського обліку	20.04.2025	
3	Аналіз діяльності ПП «Анжіо» та постановка задачі автоматизації	28.04.2025	
4	Технічна реалізація системи управління складом	19.05.2025	
5	Тестування	23.05.2025	
6	Оформлення пояснювальної записки	10.06.2025	
7	Підготовка роботи до здачі	11.06.2025	

Студент _____ Окуневич Б. О.
(підпис)

Керівник проекту _____ Поліщук В. М.
(підпис)

Номер рядка	Формат	Позначення	Найменування	Кіль-кість	Шифр до-кумента	Приміт-ка
1	А4		Завдання на проектування	2		
2	А4	ХНТУ 151.КРБ.25.А04 РФ	Реферат	2	РФ	
3	А4	ХНТУ 151.КРБ.25.А04 ТЗ	Технічне завдання	2	ТЗ	
4	А4	ХНТУ 151.КРБ.25.А04 ПЗ	Пояснювальна записка	66	ПЗ	
5	А1	Демонстраційні плакат	Лістинг програми	1		
6	А1	Демонстраційні плакат	Архітектура БД	1		
7	А1	Демонстраційні плакат	Модуль «управління складами	1		
			та слотами»			
8	А1	Демонстраційні плакат	Модуль «управління операціями»	1		
			частина 1			
9	А1	Демонстраційні плакат	Модуль «управління операціями»	1		
			частина 2			
10	А1	Демонстраційні плакат	Модуль «Інвентаризація та залишки»	1		
11	А1	Демонстраційні плакат	Модуль «Тестування системи»	1		

Зм.	Арк.	№ докум	Підпис	Дата	ХНТУ 151.КРБ.25.А04 ВР		
Розроб.		Окуневич Б.О.			Відомість об'єму роботи	Літ.	Аркуш 4 / Аркушів 172
Перевір.		Поліщук В.М.					
Реценз.						ХНТУ, гр.4А	
Н. Контр.		Поліщук В.М.					
Затверд.		Сєліверстров.					

РЕФЕРАТ

Дипломна робота: 172 стор., 39 рис., 5 табл., 19 джерел, графічна частина: форма А1 6 аркушів

Об'єкт дослідження: бізнес-процеси складської логістики на підприємстві оптової та роздрібної торгівлі будівельними матеріалами ПП «Анжіо».

Предмет дослідження: методи, програмні та апаратні засоби для автоматизації процесів управління складом.

Мета роботи: розробка та тестування програмної системи (WMS) для комплексної автоматизації складського обліку, що дозволяє підвищити точність даних, прискорити виконання операцій та посилити контроль за рухом товарно-матеріальних цінностей.

Результати: У ході виконання дипломної роботи було проаналізовано теоретичні основи складської логістики та діяльність підприємства ПП «Анжіо». На основі аналізу було спроєктовано реляційну базу даних та розроблено веб-додаток з використанням стеку PHP та MariaDB. Розроблена система включає модулі автентифікації , управління номенклатурою , виконання складських операцій (прихід, продаж, списання, переміщення) , а також проведення інвентаризації. Було проведено функціональне тестування ключових модулів, яке підтвердило їх працездатність. Запропоновано архітектуру мережевої інфраструктури для розгортання системи на підприємстві.

Практичне значення: Впровадження розробленої системи на ПП «Анжіо» дозволить значно підвищити точність обліку залишків, зменшити вплив людського фактора та пов'язані з ним втрати, а також прискорити процеси приймання, комплектації та відвантаження товарів.

СКЛАДСЬКА ЛОГІСТИКА, СИСТЕМА УПРАВЛІННЯ СКЛАДОМ, WMS, АВТОМАТИЗАЦІЯ, БАЗА ДАНИХ, PHP, ОБЛІК ЗАЛИШКІВ, ІНВЕНТАРИЗАЦІЯ, ШТРИХ-КОДУВАННЯ

ABSTRACT

Thesis: 63 pages, 39 figures, 5 tables, 19 sources.

Research object: business processes of warehouse logistics at the wholesale and retail trade enterprise of construction materials "Anzhio".

Research subject: methods, software and hardware tools for automating warehouse management processes.

Aim of the work: development and testing of a software system (WMS) for comprehensive automation of warehouse accounting, allowing to increase data accuracy, accelerate operations, and enhance control over the movement of goods and materials.

Results: During the completion of the thesis, the theoretical foundations of warehouse logistics and the activities of the enterprise "Anzhio" were analyzed. Based on the analysis, a relational database was designed and a web application was developed using the PHP and MariaDB stack. The developed system includes modules for authentication, inventory management, execution of warehouse operations (receipt, sale, write-off, transfer), as well as conducting inventory. Functional testing of key modules was conducted, which confirmed their operability. An architecture for the network infrastructure has been proposed for the deployment of the system at the enterprise.

Practical significance: The implementation of the developed system at ‘Anzhi’ LLC will significantly improve the accuracy of inventory accounting, reduce the impact of the human factor and related losses, and accelerate the processes of receiving, picking, and shipping goods.

WAREHOUSE LOGISTICS, WAREHOUSE MANAGEMENT SYSTEM, WMS, AUTOMATION, DATABASE, PHP, INVENTORY ACCOUNTING, INVENTORY, BARCODING

ТЕХНІЧНЕ ЗАВДАННЯ

1. Найменування проєкту

Повне найменування – автоматизована система управління складом (АСУС) для обліку товарно-матеріальних цінностей на торговельному підприємстві ПП «Анжіо».

2. Призначення та цілі створення системи

2.1. Призначення системи

Автоматизована система управління складом (АСУС) є інтегрованою інформаційною системою, що виконує такі функції:

- Централізований облік товарно-матеріальних цінностей у розрізі складів та місць зберігання (слотів).
- Реєстрація та документування всіх складських операцій (прихід, продаж, списання, внутрішнє переміщення, інвентаризація) в режимі реального часу.
- Надання актуальної інформації про залишки товарів, їхні ціни та фізичне розташування.
- Забезпечення розмежованого доступу персоналу до функціоналу системи відповідно до їхніх ролей та повноважень.
- Формування базової звітності для аналізу руху товарів.

2.2. Мети створення системи

Метою створення АСУС є:

- Зменшення впливу людського фактора та мінімізація операційних помилок (пересорт, недостача, некоректне списання).
- Підвищення точності та актуальності даних про товарні залишки до 99% і вище.
- Прискорення процесів приймання, комплектації, інвентаризації та відвантаження товарів.

• Забезпечення керівництва достовірною інформацією для прийняття управлінських рішень.

• Зниження втрат, пов'язаних із несвоєчасним виявленням надлишків або дефіциту продукції.

3. Вимоги до системи

АСУС має бути структурованою за класичною трирівневою архітектурою веб-додатків:

• Рівень представлення (Frontend): Реалізується за допомогою HTML, CSS та JavaScript. Відповідає за формування користувацького інтерфейсу, відображення даних та взаємодію з користувачем через веб-браузер. Інтерфейс має бути адаптивним та інтуїтивно зрозумілим.

• Рівень логіки (Backend): Реалізується мовою програмування PHP. Цей рівень забезпечує виконання всієї бізнес-логіки системи: обробку запитів від клієнта, взаємодію з базою даних, виконання розрахунків, управління сесіями та автентифікацію. • Рівень даних (Database): Реалізується за допомогою СУБД MariaDB. Відповідає за надійне зберігання, цілісність та швидкий доступ до всієї інформації системи (довідників, документів, залишків).

Система повинна бути реалізована за клієнт-серверною архітектурою та мати модульну структуру, що дозволить у майбутньому розширювати її функціонал без повної перебудови ядра. У разі відсутності зв'язку з сервером, робота в системі неможлива.

Центральним елементом системи виступає веб-сервер (Nginx або Apache), що обробляє запити, та сервер бази даних, який веде архів усіх операцій та даних.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ ТА СКОРОЧЕНЬ

WMS (Warehouse Management System) – Система управління складом; програмний комплекс, що забезпечує автоматизацію та цифровий контроль над усіма етапами складських логістичних операцій

АСУС – Автоматизована система управління складом.

БД – База даних; структурована сукупність даних, що зберігається в комп'ютерній системі.

СУБД – Система управління базами даних; програмне забезпечення для створення, редагування та контролю баз даних

PHP (Hypertext Preprocessor) – Скриптова мова програмування, що виконується на стороні сервера та використовується для розробки веб-додатків

SQL (Structured Query Language) – Мова структурованих запитів, що використовується для взаємодії з базами даних.

HTML (Hypertext Markup Language) – Стандартна мова розмітки для створення веб-сторінок.

CSS (Cascading Style Sheets) – Мова таблиць стилів, що використовується для опису зовнішнього вигляду веб-сторінок

JS (JavaScript) – Мова програмування, що використовується для створення інтерактивних ефектів та динамічної поведінки на стороні клієнта (у браузері)

AJAX (Asynchronous JavaScript and XML) – Технологія, що дозволяє веб-сторінці оновлювати дані асинхронно, без повного перезавантаження.

XAMPP – Безкоштовна, кросплатформна збірка веб-сервера, що включає Apache, MariaDB, PHP та Perl

LEMP – Стек серверного програмного забезпечення (Linux, Nginx, MariaDB/MySQL, PHP) для розгортання веб-додатків у робочому середовищі

PHP-FPM (FastCGI Process Manager) – Альтернативна реалізація PHP FastCGI, що використовується для обробки PHP-скриптів у зв'язці з веб-сервером Nginx

UI (User Interface) – Інтерфейс користувача; сукупність засобів, за допомогою яких користувач взаємодіє з системою

КВЕД – Класифікація видів економічної діяльності

ПП – Приватне підприємство

ТМЦ – Товарно-матеріальні цінності.

ТЗД – Термінал збору даних; мобільний пристрій, що поєднує функції сканера та комп'ютера

VLAN (Virtual Local Area Network) – Віртуальна локальна мережа; технологія, що дозволяє логічно розділити фізичну мережу на декілька незалежних сегментів

ВСТУП

Актуальність теми. В умовах сучасної ринкової економіки, що характеризується високим рівнем конкуренції та зростаючими вимогами клієнтів до швидкості обслуговування, ефективність складської логістики стає одним із ключових факторів успіху торговельного підприємства. Класичні методи ведення складського обліку, що базуються на паперовому документообігу та ручному введенні даних, не здатні забезпечити необхідну точність, оперативність та прозорість операцій. Це призводить до виникнення помилок, таких як пересорт, недостача, втрата актуальності даних про залишки, що, в свою чергу, веде до фінансових втрат та зниження рівня клієнтського сервісу.

Автоматизація складських процесів за допомогою спеціалізованих програмних систем (WMS) дозволяє вирішити ці проблеми, забезпечуючи цифровий контроль над усіма етапами руху товарів — від приймання до відвантаження. Для підприємств малого та середнього бізнесу, як-от ПП «Анжіо», що спеціалізується на торгівлі будівельними матеріалами , розробка власної, адаптованої до унікальних бізнес-процесів системи є економічно обґрунтованим та стратегічно важливим кроком. Таким чином, розробка програмно-апаратної моделі для автоматизації складу є надзвичайно актуальним завданням, спрямованим на підвищення конкурентоспроможності підприємства.

Головною метою даної дипломної роботи є розробка та тестування веб-орієнтованої системи управління складом для автоматизації облікових та логістичних процесів на прикладі філії ПП «Анжіо».

Для досягнення поставленої мети були визначені наступні завдання:

проаналізувати теоретичні основи та сучасні тенденції в галузі складської логістики;

дослідити діяльність та бізнес-процеси ПП «Анжіо» для виявлення потреб в автоматизації;

спроєктувати реляційну базу даних, що відповідає вимогам системи;

розробити програмні модулі для реалізації основного функціоналу: управління структурою складу, товарами, операціями (прихід, продаж, списання, переміщення) та інвентаризацією;

розробити інтуїтивно зрозумілий інтерфейс користувача;

провести тестування розробленого продукту для підтвердження його працездатності;

запропонувати архітектуру мережевої інфраструктури для впровадження системи на підприємстві.

Об'єкт дослідження — процеси складської логістики на підприємстві ПП «Анжіо».

Предмет дослідження — методи, алгоритми та програмно-апаратні засоби автоматизації складського обліку.

Для вирішення поставлених завдань було використано наступні методи: системний аналіз (для вивчення діяльності підприємства), моделювання (при проєктуванні бази даних та мережевої архітектури), об'єктно-орієнтоване програмування (при написанні коду на PHP), модульне проєктування та функціональне тестування.

Практичне значення отриманих результатів. Результатом роботи є готовий до впровадження програмний продукт, який дозволяє ПП «Анжіо» автоматизувати облік товарних залишків, підвищити точність та швидкість проведення складських операцій, посилити контроль за діями персоналу та мінімізувати втрати, пов'язані з людським фактором.

РОЗДІЛ 1
ТЕОРЕТИЧНІ ОСНОВИ АВТОМАТИЗАЦІЇ СКЛАДСЬКОГО ОБЛІКУ

1.1 Особливості сучасної складської логістики

Складська логістика є ключовим компонентом логістичного ланцюга постачання, який забезпечує приймання, зберігання, облік, комплектацію та відвантаження продукції. У сучасних умовах жорсткої конкуренції, високої вартості логістичних помилок та необхідності оперативного обслуговування клієнтів, роль ефективно організованої складської логістики зросла в рази.

Основні виклики для складської логістики сьогодення:

–Зростання обсягів товарообігу;

–Необхідність обслуговування мультиканальних продажів (онлайн + офлайн);

–Вимоги до оперативності обробки замовлень (same-day delivery);

–Зростаюча кількість SKU (асортиментних позицій);

–Контроль строків придатності, партій, серійних номерів;

–Потреба в наскрізній інтеграції з ERP, CRM та TMS-системами.

Ключові особливості сучасної складської логістики:

1. Автоматизація та цифровізація процесів

Сучасні склади інтегрують WMS-системи, сканери штрихкодів, RFID-мітки, мобільні термінали збору даних, безпровідні мережі, автоматичні стелажі, стрічкові транспортери, сенсорні датчики. Це дозволяє мінімізувати людський фактор і зменшити втрати, пов’язані з неточностями обліку, недовкладенням чи пересортом.

2. Гнучкість та адаптивність

Завдяки модульній архітектурі WMS-систем, підприємства можуть налаштовувати складські алгоритми під свої бізнес-процеси: різні типи зберігання (стелажне, піддонне, динамічне), правила відбору товару (FIFO,

LIFO, FEFO), спеціалізовані зони (холодильники, зони фліт-пакування, карантину, браку).

3. Інтегрованість у загальну систему управління підприємством

Склади не функціонують автономно — вони є частиною цілісної цифрової екосистеми підприємства, де дані автоматично передаються між модулями закупівель, виробництва, бухгалтерії, продажів. Це забезпечує єдину базу даних, прозорість операцій, унеможливлення дублювання або розбіжностей в обліку.

4. Інтелектуалізація

В сучасних логістичних центрах все частіше впроваджуються елементи штучного інтелекту та машинного навчання, які дозволяють:

–Прогнозувати залишки та потреби у поповненні;

–Оптимізувати розташування товарів у складі (ABC/XYZ аналіз, теплові карти руху);

–Визначати оптимальний маршрут збору товарів для замовлень;

–Мінімізувати непродуктивні переміщення і простій.

5. Швидкість та точність обробки замовлень

В умовах сучасної конкуренції, перевагою є не лише наявність товару, але й швидкість його підготовки до відвантаження. Завдяки автоматизації, сучасні склади досягають точності понад 99%, швидкості обробки до декількох тисяч замовлень на годину та здатні працювати в режимі 24/7.

6. Використання принципу "goods-to-person"

На зміну класичному методу, коли працівник ходить по складу, приходить підхід, за яким товар автоматично доставляється до оператора — це прискорює відбір, зменшує втому персоналу і знижує відстані переміщення.

7. Застосування IoT, хмарних рішень та мобільних технологій

Завдяки Internet of Things на складі збираються дані з ваг, температурних датчиків, трекерів переміщення; ці дані аналізуються в реальному часі. Хмарні

платформи дають можливість керувати складом з будь-якої точки, а мобільні додатки — мати контроль з планшетів або смартфонів.

8. Фокус на екологічність та стійкість

Сучасні складські центри зменшують споживання енергії, запроваджують переробку пакування, використовують автоматичні системи освітлення і вентиляції. Зменшення паперового документообігу також є внеском у green-логістику.

1.2 Загальні задачі складського обліку

Складський облік — це фундаментальна система ведення обліку матеріальних ресурсів підприємства. Від правильності його організації залежить не лише точність інвентаризації товарів, але й ефективність логістичних процесів, своєчасність поставок, мінімізація втрат і загальний рівень сервісу клієнтів.

Сучасні складські облікові системи повинні забезпечувати реєстрацію всіх логістичних операцій у реальному часі, повну простежуваність товару, автоматичну генерацію звітів та інтеграцію з бухгалтерськими або управлінськими системами.

Класифікація основних задач складського обліку

1. Приймання товарів

Процес починається із перевірки наявності документів, відповідності номенклатури, кількості та якості продукції. Під час приймання відбувається:

–реєстрація постачання (накладна, партія, дата);

–зчитування штрихкоду або RFID-мітки;

–присвоєння унікального ідентифікатора (ID);

–фіксація умов зберігання (температура, вологість, строк придатності).

Приклад:

Підприємство отримує піддон із товаром. Працівник сканує етикетку, система автоматично створює запис у базі, призначає адресу зберігання та проводить перехресну перевірку за заявкою.

2. Зберігання товарів

Після приймання товар повинен бути правильно розміщений у зоні зберігання. Для цього застосовуються:

–адресне зберігання — кожна одиниця зберігається за фіксованим місцем (ряд, секція, рівень);

–динамічне зберігання — місце визначається автоматично системою, з урахуванням об’єму, ваги, черговості реалізації;

–зони спеціального призначення — карантин, контроль якості, брак, фрозен.

Основна задача — забезпечити логічну, безпечну та оптимальну організацію простору складу.

3. Внутрішнє переміщення

Це всі зміни місця розташування продукції в межах складу:

–переміщення з приймання в зону зберігання;

–поповнення зон відбору (picking zones);

–переукладання товару для оптимізації простору;

–переміщення з метою контролю або переупаковки.

Контроль переміщень запобігає втратам і дозволяє в будь-який момент визначити фактичне місце перебування товару.

4. Відбір, комплектація і пакування

Комплектація — один з найбільш ресурсоємних процесів на складі. Система повинна:

–формувати замовлення на основі заявки;

–генерувати оптимальний маршрут збору;

–забезпечувати точність і контроль (поштучний, за масою, штрихкодом);

–формувати пакувальні листи, накладні, етикетки.

Залежно від типу товару застосовують методи:

–Pick-to-order — відбір під кожне замовлення окремо;

–Batch picking — збір кількох замовлень одночасно;

–Zone picking — поділ складу на зони та паралельна робота кількох працівників.

5. Відвантаження

Після комплектації товар реєструється як відвантажений:

–створюється запис у системі обліку;

–фіксується час і відповідальна особа;

–створюється логістична документація;

–відбувається зняття товару з обліку.

Важливо забезпечити, щоб на момент відвантаження товар не мав статусу "брак", "заблоковано", "прострочено" тощо.

6. Інвентаризація

Періодичне або вибіркове звіряння фактичної наявності товарів із записами в обліковій системі. Може бути:

–повною — охоплює всі запаси;

–вибірковою — лише окремі комірки або партії;

–перманентною — постійно діюча перевірка певних зон.

Застосування мобільних терміналів збору даних дає змогу зменшити тривалість інвентаризації з декількох днів до годин.

7. Облік специфічних параметрів

Сучасні системи повинні підтримувати:

–облік за партіями, серіями, постачальниками;

–управління строками придатності (FEFO — First Expired First Out);

–облік умов зберігання (температура, вологість);

–контроль ваги або об’єму товару;

–управління статусами товару (придатний/забракований/на перевірці).

8. Аналітика і звітність

Після збору даних система повинна формувати звіти:

–залишки товару за період;

–рух товарів;

–звіти по інвентаризації;

–відхилення і втрати;

–KPI працівників складу;

–швидкість обробки замовлень;

–рентабельність зберігання по зонах.

Додаткові задачі:

–Розмежування доступу: різні ролі користувачів (складський оператор, адміністратор, керівник логістики).

–Контроль версій змін: логування дій (хто і коли змінив інформацію).

–Інтеграція з зовнішніми системами: бухгалтерія, CRM, логістичні платформи.

1.3 Програмні рішення для автоматизації складських процесів

Автоматизація складських процесів реалізується за допомогою спеціалізованих програмних рішень, які об’єднуються загальним терміном — WMS (Warehouse Management System), або системи управління складом. Ці програмні комплекси забезпечують цифровий контроль над усіма етапами логістичних операцій: від приймання товарів до відвантаження, зберігання, інвентаризації та аналітики.

Основні функції програм складського обліку:

- Реєстрація всіх логістичних подій;
- Облік залишків у реальному часі;
- Відстеження партій, серійних номерів і термінів придатності;
- Планування зон зберігання;
- Оптимізація маршрутів збору замовлень;

– Формування звітів, актів та накладних;

– Управління користувачами, контроль доступу;

– Інтеграція з ERP, CRM, бухгалтерськими та TMS-системами.

Класифікація програмних рішень для складу:

1. Системи класу A (Enterprise WMS)

–Призначені для великих складських і логістичних центрів з високою складністю внутрішніх процесів.

–Підтримують автоматизацію зони приймання, динамічне зберігання, інтелектуальний підбір, контроль якості, багаторівневу інтеграцію.

–Приклади: *G.O.L.D. Stock WMS*, *Manhattan WMS*, *SAP EWM*, *Oracle SCM*.

–Переваги: масштабованість, багатофункціональність, висока надійність.

–Недоліки: висока вартість ліцензування та впровадження, складність адаптації під малий бізнес.

2. Системи середнього рівня

– Призначені для складів середнього розміру або компаній із простими логістичними схемами.

– Автоматизують базові процеси: приймання, зберігання, відвантаження, інвентаризація.

– Приклади: *1С:Склад*, *Instock WMS*, *Axelot WMS*, *ABM Cloud WMS*.

– Широко використовуються в Україні завдяки підтримці локального законодавства та можливості гнучкого налаштування.

3. Спеціалізовані або власноруч створені рішення

– Розробляються з урахуванням специфіки конкретного підприємства.

– Мають індивідуальну структуру бази даних, адаптований функціонал, простий інтерфейс.

– Реалізуються за допомогою сучасних фреймворків (наприклад, Python + Flask, JavaScript + React, PHP + Laravel), з використанням SQLite, MySQL або PostgreSQL для бази даних.

– Переваги: мінімальна вартість, повна адаптація до бізнес-процесів, можливість швидких змін.

– Недоліки: потреба у фахівцеві-розробнику, початковий час на проєктування.

4. Хмарні сервіси

– SaaS-платформи, які не потребують локального встановлення. Доступ до системи здійснюється через веб-браузер.

– Приклади: *Odoo Inventory*, *Zoho Inventory*, *NetSuite WMS*.

– Переваги: швидкий старт, мобільність, автоматичні оновлення.

– Недоліки: періодична абонплата, обмеження кастомізації.

Таблиця 1.1 – Порівняння популярних WMS-рішень в Україні

Назва системи	Масштаб	Ціна (тис. дол.)	Підтримка локалізації	Гнучкість	Інтеграція з ERP
G.O.L.D Stock	Високий	150–300	Часткова	Висока	Так
Instock WMS	Середній	50–120	Повна	Висока	Так
1С:Склад 8.3	Середній	5–30	Повна	Середня	Так
Axelot WMS	Середній	10–25	Повна	Висока	Так
ABM Cloud WMS	Малий	2–10	Часткова	Середня	Частково
Odoo Inventory	Малий	0–50 (SaaS)	Часткова	Висока	Так

Таблиця 1.2 – Інноваційні технології, що використовуються в WMS

Технологія	Опис
RFID	Безконтактна ідентифікація товарів і палет
Штрихкоди	Стандартний метод сканування під час приймання та комплектації
IoT	Датчики контролю температури, вологості, переміщення
Bluetooth/Wi-Fi	Безпровідна передача даних з терміналів збору
WCS (Warehouse Control System)	Контроль ваги, габаритів, маршрутизація
VDT (Voice Direct Technology)	Голосове управління комплектуванням

Вибір програмного рішення для ПП «Анжіо»

Оскільки підприємство належить до малого або середнього бізнесу, доцільним є впровадження власної програмно-апаратної моделі. Такий підхід забезпечить:

- мінімальні витрати на розробку;
- можливість інтеграції з наявним обладнанням (сканери, ПК);
- адаптацію до конкретної логіки бізнес-процесів;
- простоту використання без зайвої надбудови.

Розробка власної системи дозволяє сконцентруватися лише на потрібному функціоналі (наприклад, облік приймання, зберігання, видачі товару) без зайвих ліцензійних і підтримуючих витрат.

1.4 Апаратні засоби для автоматизації складу

У сучасній логістиці автоматизація складських процесів ґрунтується не лише на програмному забезпеченні, але й на апаратному забезпеченні — пристроях, що фіксують фізичні параметри товару, забезпечують ідентифікацію, контроль доступу, збір інформації в реальному часі. Апаратна частина є «очима» та «руками» складської системи, без яких цифрова система не зможе повноцінно функціонувати.

1. Класифікація апаратних засобів за функціональним призначенням

Апаратні компоненти можна умовно поділити на кілька функціональних груп:

Таблиця 1.3 – Класифікація апаратних засобів за функціями

Функція	Типи пристроїв
Ідентифікація продукції	Сканери штрихкодів, RFID-зчитувачі
Візуалізація та друк	Принтери етикеток, термопринтери, екрани, дисплеї
Збір даних	Термінали збору даних, мобільні сканери, планшети
Контроль фізичних параметрів	Вагові датчики, температурні сенсори, датчики вологості
Локальне керування	Контролери Arduino, Raspberry Pi, мікроконтролери ESP

Функція	Типи пристроїв
Автоматизоване переміщення	Конвеєри, ліфти, стелажні каруселі, маніпулятори
Безпека	Камери, електронні замки, сканери відбитків пальців

Основні типи апаратних засобів: опис, принцип роботи, доцільність

Сканери штрихкодів

Призначення: Швидке ідентифікування товарів при прийманні, переміщенні, видачі.

Типи: Провідні, безпровідні, стаціонарні, вбудовані у POS-термінали. Переваги: Низька ціна, простота впровадження, сумісність з будь-якою WMS.

Недоліки: Залежність від стану етикеток, потреба в прямій видимості.

RFID-системи

Принцип роботи: RFID-мітка передає ідентифікаційний код через радіохвилі до зчитувача.

Переваги: Безконтактна ідентифікація, можна зчитувати десятки міток одночасно.

Застосування: Управління палетами, тарою, контроль доступу в зони складу.

Доцільність для ПП «Анжіо»: Можна застосувати для обліку тари, або відслідковування одиниць продукції з багаторазовим використанням.

Термінали збору даних (ТЗД)

Суть: Поєднують функції сканера та мобільного комп’ютера. Працюють автономно, можуть синхронізуватись з базою по Wi-Fi або Bluetooth.

Використовуються для: інвентаризацій, відбору замовлень, приймання, контролю залишків.

Типовий пристрій: Android-смартфон із вбудованим сканером або додатком.

Принтери етикеток

Роль: Друк унікальних штрихкодів, QR-кодів, RFID-етикеток, що закріплюються на продукції, комірках, палетах.

Переваги: Формування динамічної маркувальної системи.

Доцільність: Незамінний елемент при створенні системи адресного зберігання.

Контролери та мікрокомп'ютери

Arduino / Raspberry Pi / ESP32 — відкриті апаратні платформи, які дозволяють створити індивідуальні автоматизовані рішення.

Приклади застосування:

– зчитування ваги з тензодатчиків при прийманні продукції;

– підключення до датчика температури на складі з небезпечним товаром;

– керування індикатором, який сигналізує оператору про завершення пакування;

– облік відкривання дверей з записом часу і користувача.

Переваги:

– дуже низька вартість (від 10 до 30 дол. за модуль);

– можливість підключення до Wi-Fi та віддаленого керування;

– відкритий код, що дозволяє швидко адаптувати до потреб бізнесу.

Таблиця 1.4 – Сенсорні модулі та датчики

Тип датчика	Призначення
Температурний	Контроль температури у холодильних зонах
Ваговий	Автоматичне визначення маси при прийманні
ІЧ-датчик руху	Реєстрація переміщення персоналу/товару
Датчик вологості	Моніторинг складу з паперовою/тканинною продукцією
Датчик освітлення	Автоматичне вмикання освітлення в зоні збору

Приклад інтегрованого апаратного рішення

У межах дипломного проєкту для ПП «Анжіо» можна реалізувати спрощену програмно-апаратну модель з таких компонентів:

Таблиця 1.5 – Взаємозв’язок пристроїв та їх функцій з складськими процесами

Компонент	Пристрій	Функція
Облік приймання	Сканер + ТЗД	Зчитування штрихкоду з товару
Контроль ваги	Arduino + тензодатчик	Автоматична реєстрація маси
Контроль умов	Датчик DHT22 (темп./вологість)	Моніторинг мікроклімату складу
Індикація	LED-індикатори	Підказка про завершення задачі
Передача даних	ESP8266	Wi-Fi підключення до бази даних
Вивід інформації	LCD-екран або браузер	Інтерфейс для персоналу

Переваги впровадження апаратної частини

– Зниження навантаження на персонал: автоматичне зчитування даних зменшує потребу у ручному вводі.

– Підвищення точності: відсутність помилок під час інвентаризації або комплектації.

– Прозорість операцій: кожна дія фіксується, можна відстежити маршрут товару.

– Економічна ефективність: рішення на базі відкритих платформ коштують у 5–10 разів дешевше за промислові аналоги.

– Масштабованість: у майбутньому можна розширити функціонал або кількість пристроїв без повної перебудови системи.

Висновки до розділу 1

У першому розділі було розглянуто теоретичні основи автоматизації складських процесів, актуальні підходи до управління логістикою, а також сучасні програмні й апаратні засоби, які забезпечують ефективне функціонування складу.

Проведений аналіз дозволяє зробити такі узагальнення:

– Складська логістика є ключовим елементом у системі матеріального постачання та збуту, від ефективності якої напряму залежить ритмічність і економічність роботи підприємства.

– Основні задачі складського обліку включають облік надходження, зберігання, переміщення, інвентаризацію та відвантаження товарів. Вони повинні вирішуватись оперативно, точно й автоматизовано.

– Програмні рішення, зокрема WMS-системи, забезпечують цифрову фіксацію логістичних подій, управління залишками, формування звітності та інтеграцію з іншими модулями підприємства. Для малого підприємства, такого як ПП «Анжіо», доцільним є створення власного програмного продукту, орієнтованого на конкретні бізнес-процеси.

– Апаратна частина (сканери, сенсори, контролери, RFID, ТЗД) є критично важливою для автоматичного збору фізичних даних і з’єднання «реального складу» з цифровою обліковою системою. Оптимальною для підприємства є реалізація програмно-апаратної моделі на основі недорогих відкритих платформ.

Таким чином, створення індивідуальної програмно-апаратної моделі складського обліку дозволить ПП «Анжіо» значно підвищити точність, швидкість і прозорість управління товарними залишками, зменшити вплив людського фактора, забезпечити контроль логістичних операцій і отримати конкурентну перевагу.

РОЗДІЛ 2. АНАЛІЗ ДІЯЛЬНОСТІ ПП «АНЖІО» ТА ПОСТАНОВКА ЗАДАЧІ АВТОМАТИЗАЦІЇ

2.1 Характеристика приватної фірми «Анжіо»

Приватна фірма «Анжіо» (код ЄДРПОУ 21475635) — українське підприємство, засноване 28 квітня 1993 року в місті Києві.

Основні напрями діяльності

Згідно з даними Єдиного державного реєстру, основний вид діяльності ПФ «Анжіо» — оптова торгівля деревиною, будівельними матеріалами та санітарно-технічним обладнанням (КВЕД 46.73) .(

Компанія також здійснює такі види діяльності:

- виробництво меблів для офісів і підприємств торгівлі;
- виробництво кухонних меблів;
- виробництво інших меблів;
- лісопильне та стругальне виробництво;
- виробництво фарб, лаків, клеїв та інших хімічних продуктів;
- оптова та роздрібна торгівля широким асортиментом товарів, включаючи текстиль, одяг, побутову техніку, меблі, косметику та інші господарські товари .

Інфраструктура та мережа

ПФ «Анжіо» має мережу з 12 складів-магазинів, розташованих у Києві та Київській області . Формат "склад-магазин" дозволяє компанії ефективно поєднувати функції зберігання та роздрібної торгівлі, забезпечуючи оперативне обслуговування клієнтів.

Партнерство та репутація

За роки діяльності компанія здобула репутацію надійного партнера, що підтверджується співпрацею з відомими брендами, такими як URSA, Makita, STIHL, KRAUSE, Tikkurila та Kolorit)

В даному дипломні роботі, я буду розглядати особливості складського обліку окремо виділеної філії Анжіо-Українка.

Організаційна структура філії

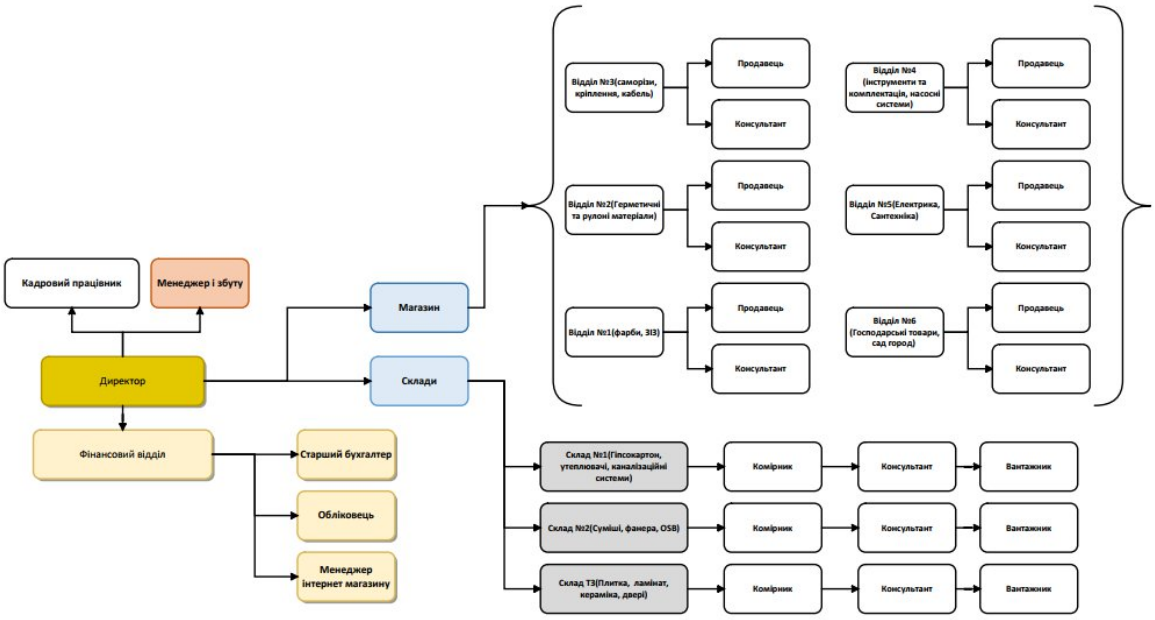


Рис. 2.1 — Організаційна структура управління торговельного підприємства

На рис. 2.1 представлена організаційна структура управління торговельно-складським комплексом, що спеціалізується на реалізації будівельних та господарських товарів. За своєю суттю, структура є лінійно-функціональною, ієрархічного типу. В її основі лежить принцип поділу за функціональними напрямками (фінанси, торгівля, логістика) з подальшою глибокою спеціалізацією підрозділів за товарними групами.

Найвищий рівень управління представлений посадою Директора, який здійснює загальне стратегічне та операційне керівництво. Йому безпосередньо підпорядковуються кадровий працівник та менеджер зі збуту, що забезпечують адміністративні та комерційні функції.

Діяльність підприємства диференційована на три основні функціональні блоки, підзвітні директору:

Фінансовий відділ: Відповідає за фінансово-економічну діяльність підприємства. У його структуру входять старший бухгалтер, обліковець та

менеджер інтернет-магазину. Підпорядкування менеджера e-commerce фінансовому відділу є особливістю даної структури і може свідчити про пріоритетність фінансового контролю та обліку в онлайн-каналі продажів.

Магазин (торговельний зал): Цей блок відповідає за роздрібні продажі. Він сегментований на відділи за принципом товарної спеціалізації (наприклад, "Відділ №2: електрика, сантехніка", "Відділ №1: фарби, піни" та ін.). Штатна структура кожного відділу є стандартизованою і включає посади продавця та консультанта, що дозволяє забезпечити як процес продажу, так і кваліфіковану допомогу клієнтам.

Склади (складський комплекс): Цей блок забезпечує логістичні та складські операції. Структура складів також побудована за товарно-габаритним принципом (наприклад, "Склад №1: гіпсокартон, утеплювач", "Склад №2: суміші, фанера"). Операційний процес на складах забезпечується трьома ключовими посадами: комірник (облік та зберігання), консультант (комплектація замовлень, допомога клієнтам) та вантажник (фізичне переміщення товару).

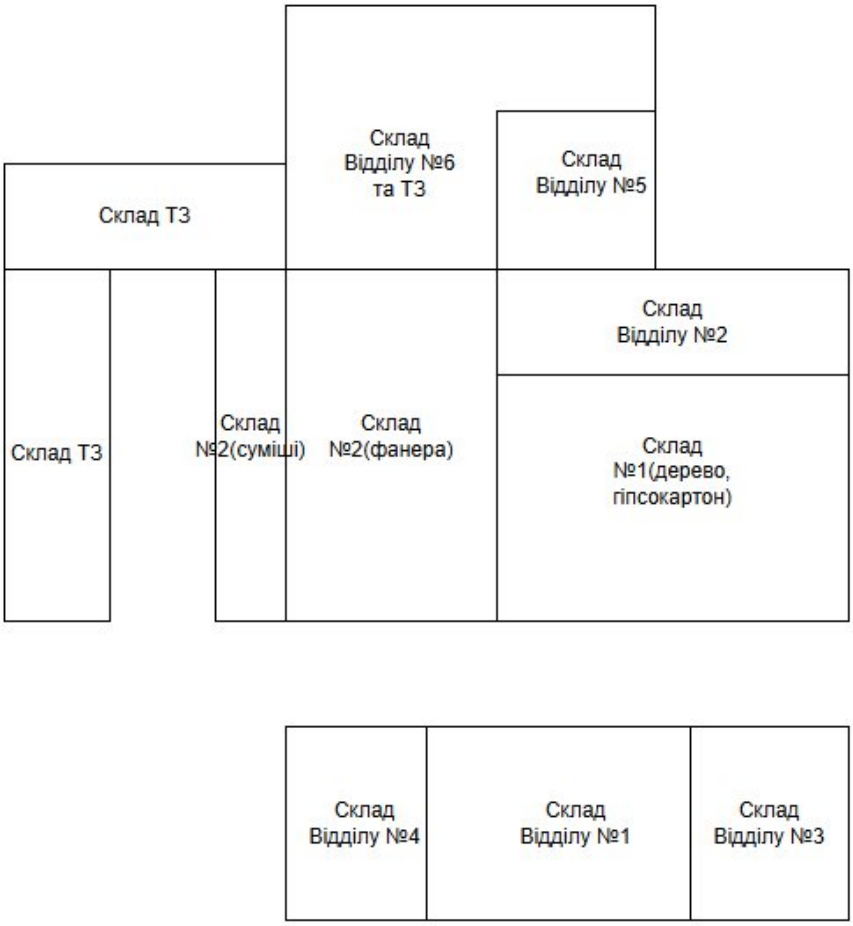


Рис. 2.2– Схема зонування складського комплексу підприємства

На рис. 2.2 представлена концептуальна схема фізичного розміщення та функціонального зонування складських приміщень торговельного підприємства. Дана схема ілюструє як статичну структуру розміщення зон зберігання, так і динаміку основних логістичних процесів, таких як приймання та видача товару.

1. Фізична структура та зони зберігання

Структура складського комплексу базується на принципі децентралізованого зберігання, де товарні запаси розподілені по декількох окремих приміщеннях, спеціалізованих за товарними групами або за приналежністю до певного торговельного відділу. Відповідно до схеми, можна виділити наступні ключові зони зберігання:

Спеціалізовані склади відділів: Основна частина запасів зберігається на складах, що закріплені за конкретними торговими відділами (наприклад, Склад №1, Склад №2, Склад ТЗ, Склад Відділу №1, №2, №3, №4, №5, №6).

Деталізовані зони зберігання: Деякі склади мають додатковий внутрішній поділ за типом продукції. Наприклад, Склад №2 поділяється на зони для зберігання "сумішей" та "фанери", а Склад №1 має велику зону для "дерева, гіпсокартону".

2. Функціональне зонування логістичних процесів

На основі наданої інформації, весь логістичний процес на території комплексу можна розділити на три ключові функціональні зони, організація яких описана нижче.

Зона зберігання: Це фізичні приміщення складів, що безпосередньо зображені на схемі. Кожен склад призначений для зберігання визначеної номенклатури товарів, що дозволяє оптимізувати умови зберігання та внутрішні складські операції.

Зона приймання та розвантаження: Згідно з описом, для кожного складу відділу передбачена власна прилегла зона для розвантаження та приймання товару. Такий децентралізований підхід дозволяє уникнути черг та скупчення транспорту в єдиній точці. Винятком є склади Відділів №5 та №6, для яких ці операції централізовано виконуються біля "Складу ТЗ". Це обумовлено специфікою товару та оптимізацією використання персоналу та техніки.

Зона видачі та завантаження: Процес видачі товару клієнтам або завантаження для переміщення організований за аналогічним принципом. Більшість складів мають власну прилеглу зону видачі, що дозволяє оперативно обслуговувати клієнтів відповідного відділу. Як і у випадку з прийманням, для товарів зі складів Відділів №5 та №6 видача та завантаження відбуваються централізовано біля "Складу ТЗ". Водночас, для дрібних товарів у невеликих кількостях передбачена можливість видачі безпосередньо у відповідних

торгових відділах, що значно прискорює обслуговування клієнтів з невеликими покупками.

Висновки до розділу 2

У даному розділі було проведено комплексний аналіз діяльності приватного підприємства «Анжіо», його організаційної структури та специфіки складських процесів на прикладі окремої філії.

На основі аналізу було встановлено, що ПФ «Анжіо» є підприємством у сфері оптової та роздрібної торгівлі будівельними матеріалами з розвиненою мережею складів-магазинів. Організаційна структура філії є лінійно-функціональною та ієрархічною, з чітким поділом на фінансовий, торговельний та складський блоки, підпорядковані директору. Складський комплекс підприємства функціонує за децентралізованим принципом, де товарні запаси розподілені по спеціалізованих складах, закріплених за відповідними торговими відділами. Логістичні процеси, такі як приймання та видача товару, також переважно децентралізовані, однак мають елементи централізації для обробки певних товарних груп.

Проведений аналіз дозволив виявити ключові передумови для автоматизації. Поточна модель управління, попри свою структурованість, при ручному або частково автоматизованому обліку створює ризики виникнення помилок, пов'язаних з людським фактором, призводить до затримок в оновленні даних про залишки та ускладнює проведення оперативної інвентаризації. Відсутність єдиної цифрової системи в реальному часі не дозволяє досягти необхідного рівня прозорості та контролю над рухом товарно-матеріальних цінностей.

Постановка задачі автоматизації. На основі виявлених потреб формулюється задача розробки програмного продукту — автоматизованої системи управління складом. Система повинна вирішувати наступні ключові завдання:

Створити єдиний інформаційний простір для ведення обліку товарних залишків у розрізі складів та слотів.

Автоматизувати документообіг для всіх основних складських операцій: приходу, продажу, списання, внутрішнього переміщення та інвентаризаційної корекції.

Забезпечити точну ідентифікацію товару, його ціни та місця зберігання на кожному етапі логістичного ланцюга.

Надати зручний інтерфейс для персоналу з різними ролями (комірник, менеджер, бухгалтер) для виконання їхніх посадових обов'язків.

Підвищити швидкість та точність облікових операцій, мінімізувати ймовірність помилок та забезпечити керівництво актуальною інформацією для прийняття управлінських рішень.

Таким чином, розробка власної WMS-системи є логічним та обґрунтованим кроком для підвищення ефективності діяльності ПП «Анжіо».

РОЗДІЛ 3. ТЕХНІЧНА РЕАЛІЗАЦІЯ СИСТЕМИ УПРАВЛІННЯ СКЛАДОМ

3.1. Вибір інструментарію та середовища розробки

Для створення програмного продукту було обрано набір технологій, що є поширеним у веб-розробці та дозволяє створити гнучку та масштабовану систему.

Локальне середовище розробки XAMPP

Розробка та тестування системи проводились на локальному сервері, розгорнутому за допомогою програмного пакета XAMPP. XAMPP — це безкоштовна, кросплатформна збірка веб-сервера, що містить усі необхідні компоненти для повноцінної розробки. Абревіатура XAMPP розшифровується як:

- X (Cross-platform) – працює на різних операційних системах.
- A (Apache) – найпопулярніший у світі веб-сервер, що відповідає за обробку HTTP-запитів.
- M (MariaDB) – система управління базами даних (СУБД), що є відгалуженням MySQL і повністю сумісна з нею. Використовується для зберігання всієї інформації системи.
- P (PHP) – мова програмування, на якій написана серверна логіка додатку.
- P (Perl) – ще одна мова програмування, що входить до складу пакета.

Вибір XAMPP обумовлений його простотою у встановленні та налаштуванні, а також наявністю всіх необхідних інструментів "з коробки", що значно прискорює процес розробки.

Мова програмування PHP

Серверна частина (бек-енд) системи розроблена з використанням мови програмування PHP (Hypertext Preprocessor). PHP є скриптовою мовою, що виконується на стороні сервера і ідеально підходить для веб-розробки. Її ключові переваги для даного проєкту:

- Взаємодія з базами даних: PHP має потужні та зручні інструменти для роботи з СУБД MariaDB/MySQL.
- Обробка веб-форм: Мова дозволяє легко приймати, валідувати та обробляти дані, що надходять від користувача через HTML-форми (наприклад, при авторизації, створенні документів).
- Генерація динамічного контенту: PHP дозволяє створювати веб-сторінки, вміст яких змінюється залежно від дій користувача, даних з бази даних та логіки програми.

3.2. Проєктування та структура бази даних

Основою будь-якої інформаційної системи є база даних (БД). Для даного проєкту була спроєктована реляційна база даних під назвою warehouses. Вона слугує централізованим сховищем для всієї інформації про товари, операції, користувачів та складську структуру.

	Таблица	Действие						Строки	Тип	Сравнение	Размер
☐	categories	☆	Обзор	Структура	Поиск	Вставить	Очистить	Удалить	0	InnoDB	utf8mb4_unicode_ci	16,0 КиБ
☐	documents	☆	Обзор	Структура	Поиск	Вставить	Очистить	Удалить	11	InnoDB	utf8mb4_general_ci	80,0 КиБ
☐	document_items	☆	Обзор	Структура	Поиск	Вставить	Очистить	Удалить	11	InnoDB	utf8mb4_general_ci	80,0 КиБ
☐	inventory	☆	Обзор	Структура	Поиск	Вставить	Очистить	Удалить	5	InnoDB	utf8mb4_unicode_ci	48,0 КиБ
☐	products	☆	Обзор	Структура	Поиск	Вставить	Очистить	Удалить	2	InnoDB	utf8mb4_unicode_ci	48,0 КиБ
☐	product_prices	☆	Обзор	Структура	Поиск	Вставить	Очистить	Удалить	2	InnoDB	utf8mb4_unicode_ci	32,0 КиБ
☐	product_requests	☆	Обзор	Структура	Поиск	Вставить	Очистить	Удалить	0	InnoDB	utf8mb4_unicode_ci	48,0 КиБ
☐	slots	☆	Обзор	Структура	Поиск	Вставить	Очистить	Удалить	25	InnoDB	utf8mb4_unicode_ci	32,0 КиБ
☐	suppliers	☆	Обзор	Структура	Поиск	Вставить	Очистить	Удалить	1	InnoDB	utf8mb4_unicode_ci	16,0 КиБ
☐	users	☆	Обзор	Структура	Поиск	Вставить	Очистить	Удалить	1	InnoDB	utf8mb4_general_ci	32,0 КиБ
☐	warehouses	☆	Обзор	Структура	Поиск	Вставить	Очистить	Удалить	1	InnoDB	utf8mb4_unicode_ci	16,0 КиБ

Рис.3.1 Таблиці в Базі даних warehouses

Детальний опис сутностей та атрибутів бази даних

Структура бази даних warehouses, представлена на рисунку, спроєктована для комплексного зберігання інформації, що необхідна для автоматизації процесів складського обліку. Кожна таблиця представляє окрему сутність бізнес-процесу, а її поля (стовпці) виступають атрибутами цієї сутності.

Таблиця users призначена для зберігання даних про користувачів системи. Кожен рядок цієї таблиці є унікальним обліковим записом. Таблиця містить унікальний числовий ідентифікатор id (первинний ключ), логін username для входу, поле password для зберігання хешу пароля, що забезпечує безпеку автентифікаційних даних, а також full_name для ідентифікації користувача в інтерфейсі системи. Поле created_at фіксує дату створення облікового запису.

Таблиці warehouses та slots описують фізичну структуру зберігання. Сутність warehouses представляє окремий склад або складське приміщення. Її атрибутами є унікальний ідентифікатор id, назва складу name, його адреса address та логічне поле active, що дозволяє тимчасово виключати склад з операційної діяльності. Таблиця slots деталізує структуру, описуючи конкретні місця зберігання всередині складу — стелажі, ячейки, палети. Кожен слот характеризується ідентифікатором id, посиланням на свій склад warehouse_id, унікальною назвою name (наприклад, «А-01-5»), описом description, показником максимальної місткості capacity та статусом активності active.

Таблиці-довідники suppliers та categories слугують для класифікації та зберігання допоміжної інформації. Таблиця suppliers містить перелік постачальників товарів. Кожен постачальник описується такими атрибутами, як назва компанії name, контактна особа contact_person, телефон phone, електронна пошта email та адреса. Таблиця categories є простим довідником товарних категорій, що складається з ідентифікатора та назви name, і використовується для групування товарів.

Таблиці products та product_prices формують ядро товарного каталогу. Основна таблиця products містить повний перелік товарної номенклатури. Кожен товар (рядок таблиці) має унікальний артикул article, штрих-код barcode, назву name, опис description, одиницю виміру unit та посилання на свою категорію category_id. Таблиця product_prices розширює інформацію про товар, дозволяючи зберігати множинні цінові показники. Вона містить закупівельну ціну purchase_price, роздрібну ціну selling_price, відсоток націнки

markup_percent та прапорець active, що дозволяє мати історію цін та активувати актуальну.

Таблиця inventory є центральною обліковою таблицею системи. Вона не зберігає довідкову інформацію, а фіксує поточний, кількісний стан складу. Кожен рядок цієї таблиці означає, що певна кількість (quantity) конкретного товару (product_id) знаходиться в конкретному слоті (slot_id). Поля product_id та slot_id разом утворюють унікальний ключ, що унеможливлює дублювання записів. Атрибут last_updated автоматично фіксує час останньої зміни залишку.

Таблиці documents та document_items реалізують документообіг, що є основою для фіксації всіх господарських операцій. Таблиця documents виконує роль "шапки" документа. Вона містить загальні атрибути операції: унікальний номер id, тип операції type (прихід, продаж, списання, переміщення, коригування), номер первинного документа number та номер пов'язаного рахунку-фактури invoice_ref_number, інформацію про контрагента counterparty, а також посилання на постачальника supplier_id та користувача, що створив документ, created_by. Таблиця document_items, у свою чергу, є табличною частиною документа. Кожен її рядок — це конкретна товарна позиція в рамках операції, що описується посиланнями на документ document_id та товар product_id, а також кількістю quantity та ціною price. Важливими є поля source_slot_id та destination_slot_id, які фіксують, з якої ячейки товар було взято та/або в яку переміщено.

Таблиця product_requests призначена для автоматизації внутрішніх потреб. Вона дозволяє користувачам створювати запити на товар. Кожен запит містить інформацію про товар product_id, бажану кількість requested_quantity, ініціатора запиту requester_id та статус обробки status (напр., «в очікуванні», «схвалено», «відхилено»).

Аналіз зв'язків між сутностями

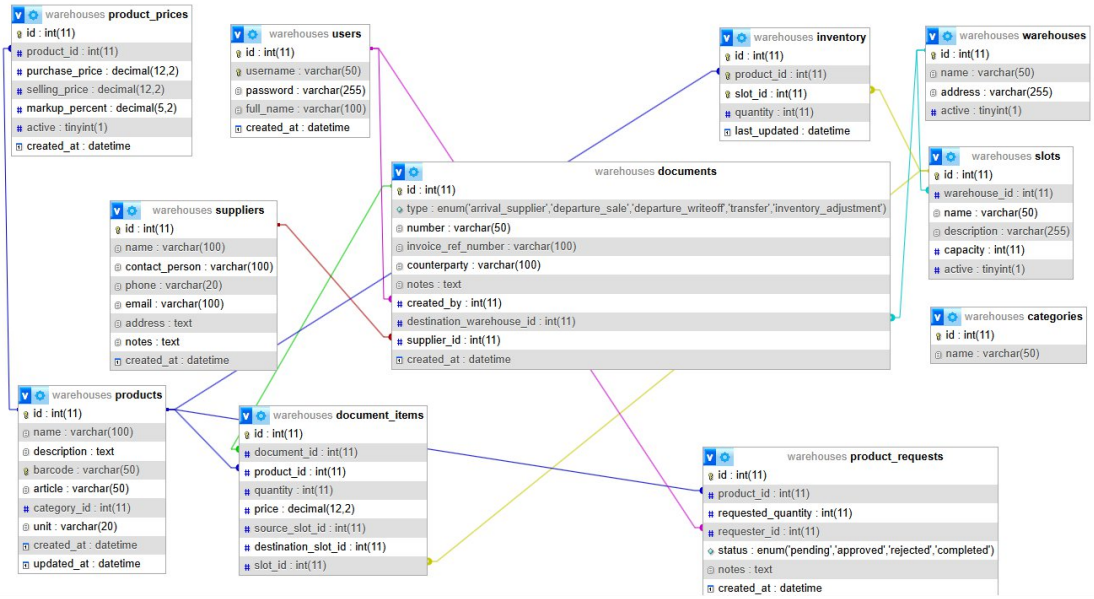


Рис. 3.2. Схема бази даних

Представлена на рисунку схема бази даних утворює логічно пов'язану структуру, де таблиці взаємодіють між собою за допомогою первинних та зовнішніх ключів, формуючи переважно зв'язки типу "один-до-багатьох".

Центральною сутністю всього каталогу є таблиця products. Вона пов'язана зі своїми атрибутивними довідниками: зв'язок "багато-до-одного" з таблицею categories дозволяє групувати товари, а зв'язок "один-до-багатьох" з product_prices — гнучко керувати ціноутворенням.

Фізична структура складу реалізована через ієрархічний зв'язок "один-до-багатьох" між таблицями warehouses та slots. Для фактичного обліку залишків використовується таблиця inventory, яка виступає сполучною ланкою між товарами та місцями їх зберігання, реалізуючи зв'язок "багато-до-багатьох" між products та slots.

Операційна діяльність фіксується через пару таблиць documents та document_items. documents є головною таблицею, що агрегує дані про операцію, маючи зв'язки "багато-до-одного" з таблицями users (для ідентифікації автора документа) та suppliers (для документів приходу). Таблиця document_items є підпорядкованою і реалізує зв'язок "один-до-багатьох" з documents. Водночас document_items є ключовим інтеграційним елементом, оскільки вона

посилається на products для ідентифікації товару та на slots (через поля source_slot_id та destination_slot_id) для фіксації його фізичного руху. Саме через цю таблицю будь-яка операція знаходить своє відображення у зміні залишків в таблиці inventory.

Таким чином, спроєктована архітектура БД дозволяє комплексно та взаємопов'язано зберігати всю інформацію, необхідну для автоматизації складських процесів від ведення довідників до фіксації та аналізу товарних рухів.

Інтерфейс користувача (UI) є ключовим компонентом розробленої системи, що забезпечує взаємодію персоналу з її функціоналом. Розробка інтерфейсу була зосереджена на досягненні простоти, інтуїтивності та функціональності, щоб мінімізувати час на навчання персоналу та знизити ймовірність помилок при роботі. Для стилізації та створення адаптивної верстки був використаний CSS-фреймворк Bootstrap.

3.3. Сторінка автентифікації

Призначення та загальний вигляд

Сторінка автентифікації (реалізована у файлі index.php) є першою точкою контакту користувача з системою та виконує критично важливу функцію контролю доступу. Її головне завдання — перевірити автентичність користувача, перш ніж надати йому доступ до внутрішніх модулів програми.

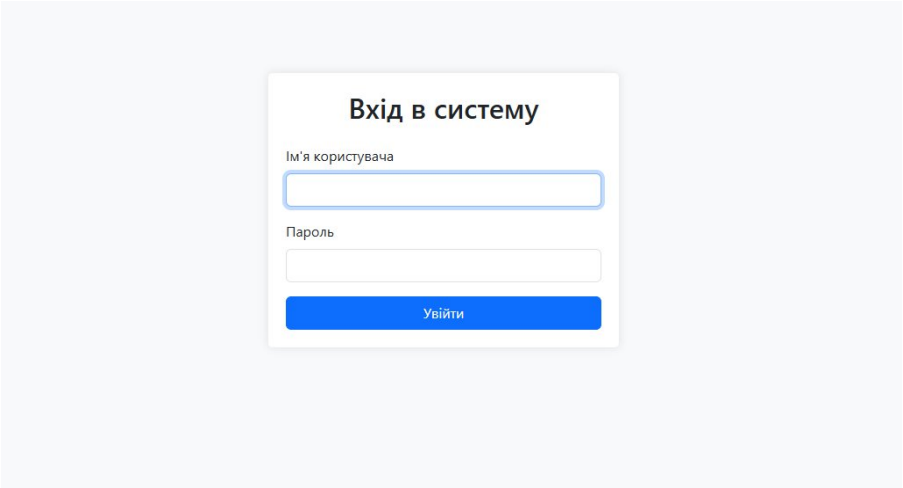


Рис.3.3 Сторінка автентифікації

Дизайн сторінки реалізовано за принципом мінімалізму. Такий підхід дозволяє не відволікати користувача від головного завдання — введення автентифікаційних даних. За допомогою фреймворку Bootstrap, основні елементи форми вирівняні по центру екрана у спеціальному контейнері, що забезпечує візуальну чистоту та коректне відображення на пристроях з різними розмірами екранів.

Основні елементи інтерфейсу

Форма входу складається з наступних стандартних елементів:

- Поле вводу "Ім'я користувача" (username): Стандартне текстове поле, призначене для введення унікального логіну користувача. Є обов'язковим для заповнення.
- Поле вводу "Пароль" (password): Поле типу password, яке автоматично приховує введені символи (замінюючи їх на крапки), що забезпечує базовий рівень конфіденційності при введенні. Також є обов'язковим для заповнення.
- Кнопка "Увійти" (submit): Основний елемент управління, що ініціює процес автентифікації. При натисканні дані з полів форми відправляються на сервер для перевірки.
- Блок повідомлення про помилку: У випадку введення невірних даних, над формою динамічно з'являється блок з повідомленням "Неверное имя

пользователя или пароль", що чітко інформує користувача про невдалу спробу входу та необхідність перевірити введені дані.

Логіка взаємодії з користувачем

Процес взаємодії зі сторінкою побудований наступним чином:

1. При першому завантаженні сторінки серверний скрипт перевіряє, чи не є користувач вже авторизованим (через наявність даних у сесії). Якщо сесія активна, користувач автоматично перенаправляється на головну робочу сторінку системи (dashboard.php), минаючи повторний вхід.
2. Користувач вводить свої логін та пароль у відповідні поля.
3. Після натискання кнопки "Увійти" дані з форми відправляються методом POST на цей же скрипт index.php для обробки.
4. Серверна логіка отримує дані, виконує перевірку в таблиці users бази даних. У разі успіху, дані про користувача записуються в сесію, і він перенаправляється на головну сторінку. У разі невдачі — сторінка перезавантажується, і користувач бачить повідомлення про помилку.

3.4. Головна панель та навігація (dashboard.php)

Загальне призначення та структура

Після успішної автентифікації користувач перенаправляється на головну панель системи (файл dashboard.php), яка є центральним вузлом для доступу до всіх функціональних модулів. Ця сторінка виступає у ролі "домашнього екрана" додатку.

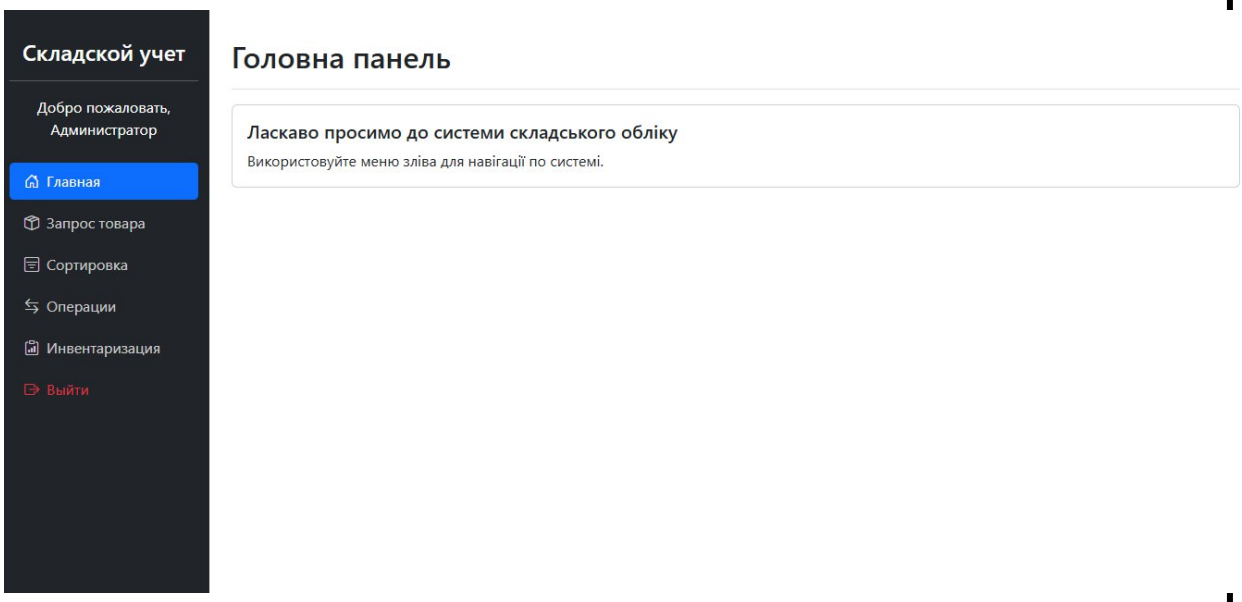


Рис.3.4 Головна панель та навігація

Інтерфейс сторінки, як і всіх наступних внутрішніх сторінок системи, реалізовано за поширеною двоколонковою схемою, що є стандартом для сучасних інформаційних систем. Вона включає:

- фіксовану бічну панель (сайдбар) для навігації;
- основну робочу область для відображення контенту обраного модуля.

Такий підхід забезпечує зручну та інтуїтивно зрозумілу роботу з програмою, оскільки навігаційні елементи завжди залишаються на видноті.

Основні елементи інтерфейсу

1. Бічна навігаційна панель (сайдбар) Сайдбар є статичним елементом, присутнім на всіх основних сторінках системи, що забезпечує швидкий та зручний перехід між розділами.
    - Блок привітання: У верхній частині панелі відображається персоналізоване вітальне повідомлення з іменем поточного користувача (Вітаємо, [Ім'я користувача]), яке отримується з даних сесії.
    - Навігаційне меню: Основну частину панелі займає вертикальне меню, яке містить посилання на ключові модулі системи. Поточна

активна сторінка підсвічується в меню для кращої орієнтації користувача. Перелік модулів, доступних для навігації, включає:

- Головна: посилання на стартову сторінку dashboard.php.
- Запит товару: перехід до модуля створення внутрішніх запитів на товари.
- Сортування: доступ до функціоналу сортування та розміщення товарів.
- Прибуття/Відбуття (Операції): перехід до центрального модуля управління складськими операціями.
- Інвентаризація: доступ до модуля перегляду залишків та проведення інвентаризації.

- Функція виходу: В нижній частині сайдбару розташоване посилання "Вийти", яке веде на скрипт logout.php. Цей скрипт завершує поточну сесію користувача та перенаправляє його на сторінку автентифікації.

2. Основна робоча область Робоча область сторінки dashboard.php виконує роль "домашнього екрана" та інформаційного центру. На поточному етапі розробки вона містить вітальну картку, що інформує користувача про успішний вхід та пропонує використовувати меню для навігації по системі. У перспективі розвитку системи цей простір може бути використаний для виведення ключових показників ефективності (KPI), графіків надходжень та продажів, списку останніх операцій або важливих сповіщень для персоналу.

3.5. Модуль "Запит товару" (product_request.php)

Призначення та загальний вигляд

Даний модуль призначений для пошуку товарів у загальній номенклатурі та створення внутрішніх запитів на їх отримання. Це ключовий інструмент для

комунікації між різними підрозділами (наприклад, між торговим залом та складом) щодо потреби в певних товарних позиціях.

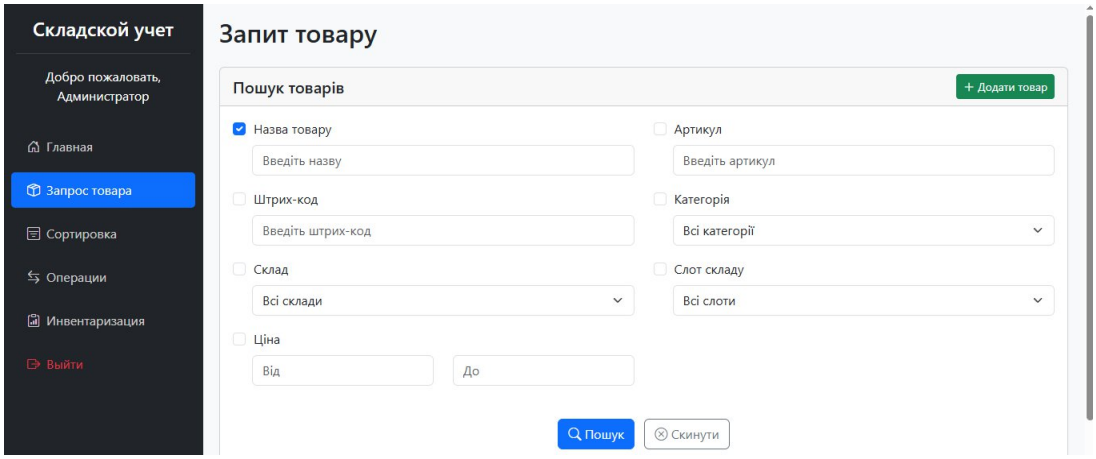


Рис.3.5 Модуль Запит товару

Інтерфейс сторінки логічно розділений на дві основні частини:

1. Форма пошуку товарів: Розширений блок з великою кількістю фільтрів для точного пошуку.
2. Блок результатів та створення запиту: Динамічна область, де відображаються знайдені товари та з'являється форма для створення запиту.

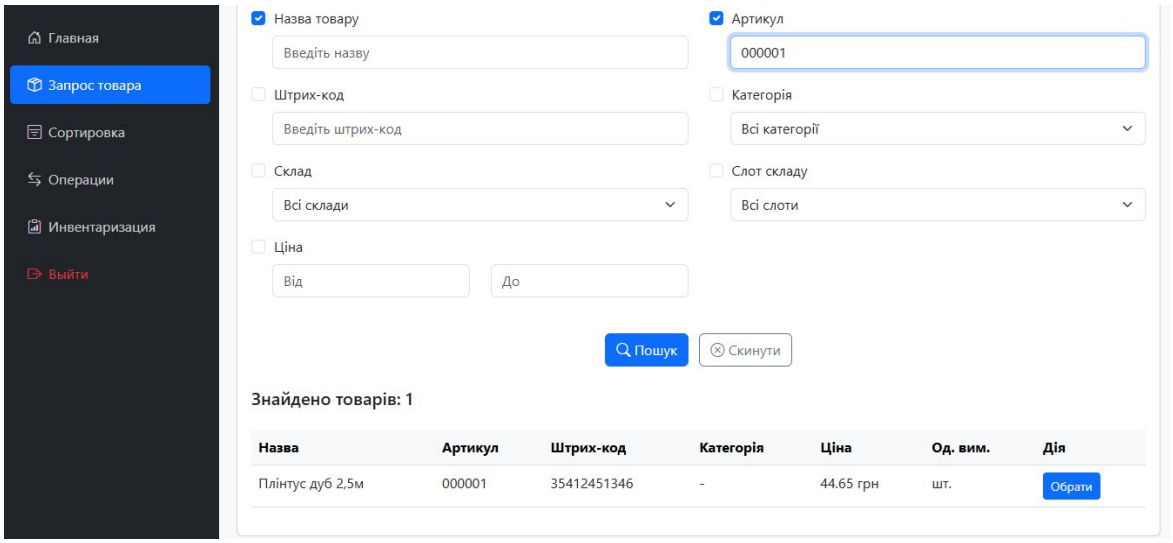


Рис.3.6 Обробка запиту

Основні елементи інтерфейсу

1. Форма пошуку:

- Параметри пошуку: Користувачу надається великий набір критеріїв для пошуку, кожен з яких можна активувати за допомогою прапорця (checkbox): назва товару, артикул, штрих-код, категорія, склад та конкретний слот. Це дозволяє здійснювати як широкий, так і дуже точковий пошук.
- Поля вводу та випадаючі списки: Для кожного критерію передбачено відповідний елемент: текстові поля для назви/артикулу/штрих-коду та випадаючі списки для вибору категорії, складу або слота. Дані для списків завантажуються з відповідних таблиць бази даних (categories, warehouses, slots).
- Кнопки управління: Форма містить кнопки "Пошук" для відправки запиту та "Скинути" для очищення всіх полів.

2. Таблиця результатів:
   - Після виконання пошуку знайдені товари відображаються у вигляді таблиці. У таблиці виводиться ключова інформація про товар: назва, артикул, штрих-код, категорія, ціна та одиниця виміру.
   - Кожен рядок таблиці є інтерактивним. При натисканні на нього система фіксує вибір товару для подальших дій.
3. Форма створення запиту:
   - Поява форми: Ця форма за замовчуванням прихована і з'являється лише після того, як користувач обрав товар з таблиці результатів.
   - Склад форми: Вона містить поле з назвою обраного товару (лише для читання), поле для введення необхідної кількості, випадаючий список для вибору складу призначення (куди потрібен товар) та поле для приміток.
   - Відправка запиту: Кнопка "Надіслати запит" відправляє дані на сервер, де створюється новий запис у таблиці product_requests.

Логіка взаємодії з користувачем

Робота з модулем відбувається за наступним сценарієм:

1. Користувач заповнює одне або декілька полів у формі пошуку та натискає "Пошук".
2. Сторінка перезавантажується, і PHP-скрипт виконує SQL-запит до бази даних з урахуванням обраних критеріїв.
3. Результати пошуку відображаються у таблиці.
4. Користувач натискає на потрібний товар у таблиці.
5. JavaScript-сценарій робить видимою форму створення запиту, автоматично підставляючи в неї назву та ID обраного товару. Сторінка прокручується до цієї форми для зручності.
6. Користувач вказує необхідну кількість, склад призначення, за потреби додає примітку і натискає "Надіслати запит".
7. Дані відправляються на сервер, запит реєструється в системі, і користувач бачить повідомлення про успішне створення запиту.

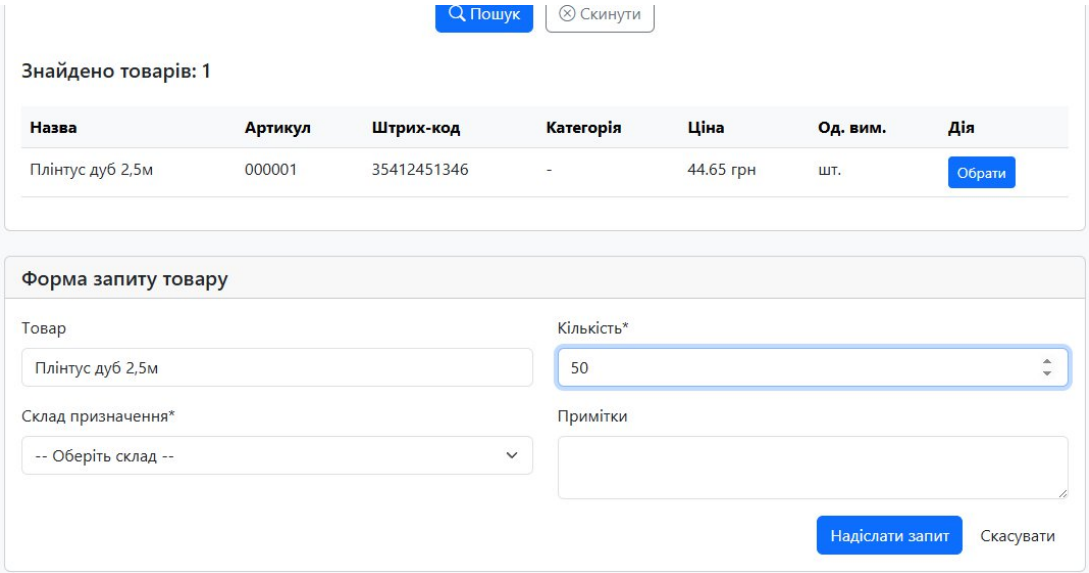


Рис.3.7 Обробка запитів на товар

3.6. Модуль "Додавання товару" (products_add.php)

Призначення та спосіб доступу

Модуль "Додавання товару" призначений для розширення номенклатурного довідника системи шляхом створення нових товарних позицій та визначення їхніх базових цін. Доступ до цього модуля здійснюється не через

головне навігаційне меню, а контекстуально. На сторінці "Запит товару" (product_request.php) розташована спеціальна кнопка "Додати товар", що дозволяє користувачеві оперативно перейти до створення нового товару, якщо його не було знайдено під час пошуку.

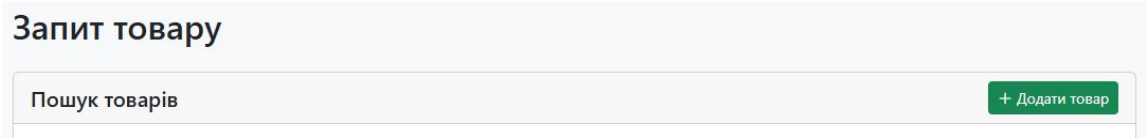


Рис. 3.8. Кнопка для додавання нової позиції товару

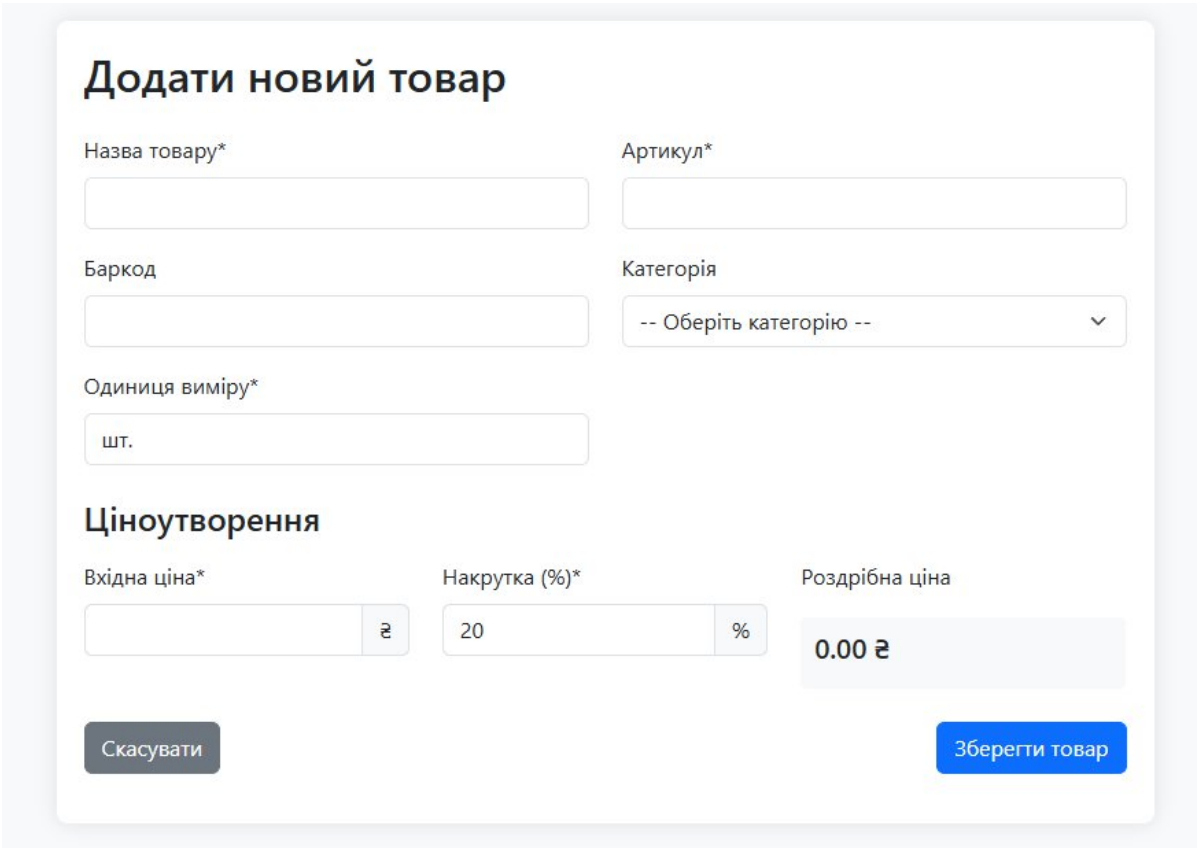


Рис. 3.9 Форма для додавання нової позиції товару

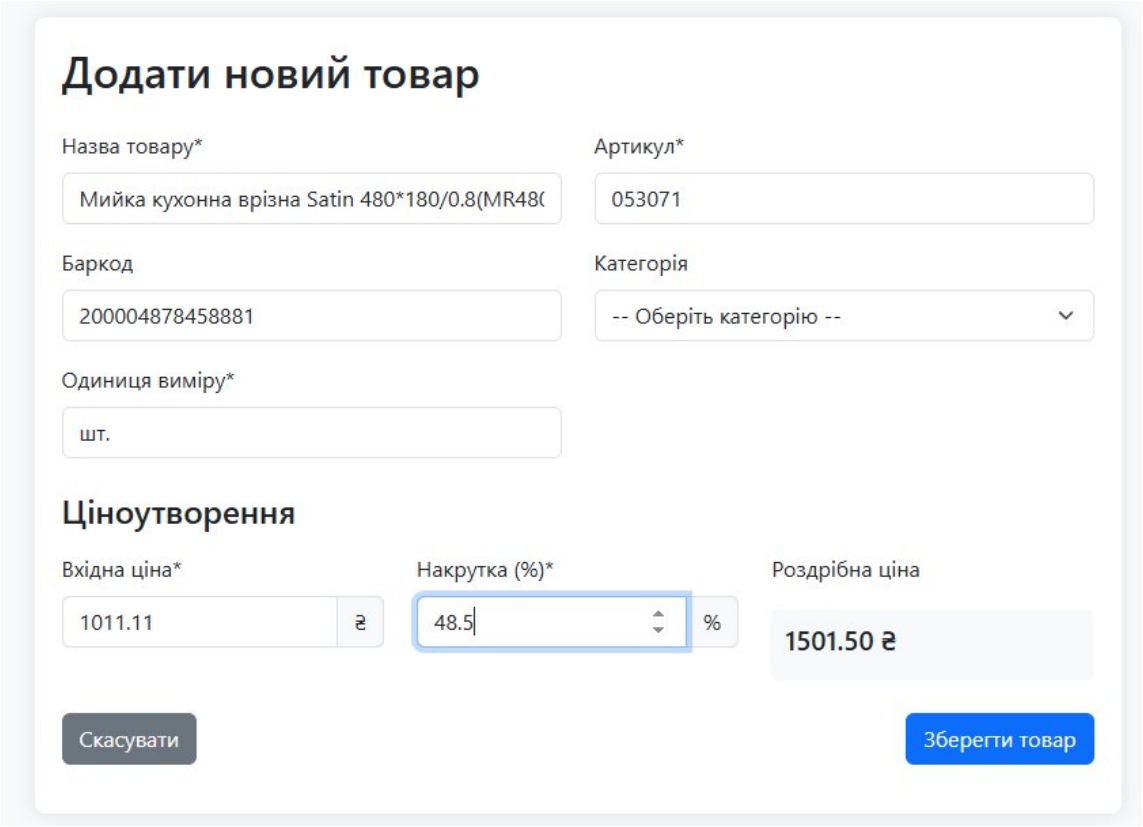


Рис.3.10 Введення інформації по нову позицію товару

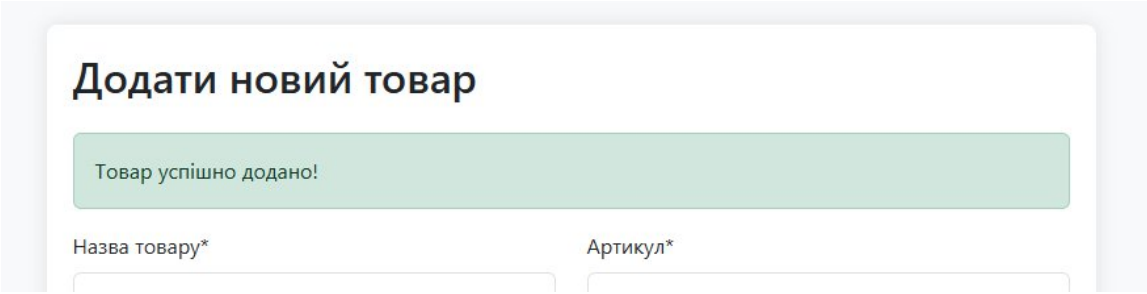


Рис.3.11 Результат обробки запиту

Загальний вигляд та основні елементи інтерфейсу

Сторінка products_add.php являє собою єдину HTML-форму, що згрупована у два логічні блоки: "Основна інформація" та "Ціноутворення".

1. Блок основної інформації:
   - Назва товару та Артикул: Два основні текстові поля (name, article), які є обов'язковими для заповнення та слугують для ідентифікації товару в системі.

- Баркод та Категорія: Необов'язкові поля. Поле barcode призначене для введення штрих-коду, а випадаючий список category_id дозволяє віднести товар до однієї з існуючих категорій. Список категорій завантажується з таблиці categories.
- Одиниця виміру: Текстове поле unit зі значенням за замовчуванням "шт.", що дозволяє вказати, в яких одиницях обліковується товар.

2. Блок ціноутворення:
   - Вхідна ціна: Числове поле purchase_price для введення закупівельної ціни товару.
   - Накрутка (%): Числове поле markup_percent для введення торговельної націнки у відсотках.
   - Роздрібна ціна: Це поле не редагується користувачем безпосередньо. Його значення (selling_price_display) розраховується автоматично за допомогою JavaScript-сценарію на основі вхідної ціни та націнки, що дозволяє користувачеві миттєво бачити результат.

Логіка взаємодії та збереження

При зміні значень у полях "Вхідна ціна" або "Накрутка (%)" клієнтський скрипт перераховує роздрібну ціну та оновлює її на екрані. Після заповнення всіх необхідних полів користувач натискає кнопку "Зберегти товар". Дані відправляються на сервер, де PHP-скрипт виконує наступні дії:

1. Створює новий запис у таблиці products з основними атрибутами товару.
2. Створює пов'язаний запис у таблиці product_prices з розрахованими цінами (вхідною, роздрібною) та націнкою. Весь процес збереження відбувається в межах однієї транзакції для забезпечення цілісності даних. У разі успіху користувач бачить відповідне повідомлення. На сторінці також є кнопка "Скасувати", що дозволяє повернутися на попередню сторінку, не зберігаючи дані.

3.7. Модуль "Сортування" (sorting.php)

Призначення та загальний вигляд

Модуль, реалізований у файлі sorting.php, виконує фундаментальну адміністративну функцію — створення та управління фізичною топологією складського комплексу. Він дозволяє створювати нові склади, а також деталізувати їхню структуру шляхом додавання місць зберігання (слотів). Цей модуль є первинним, оскільки без визначеної структури складів неможливе виконання будь-яких подальших операцій з товарами.

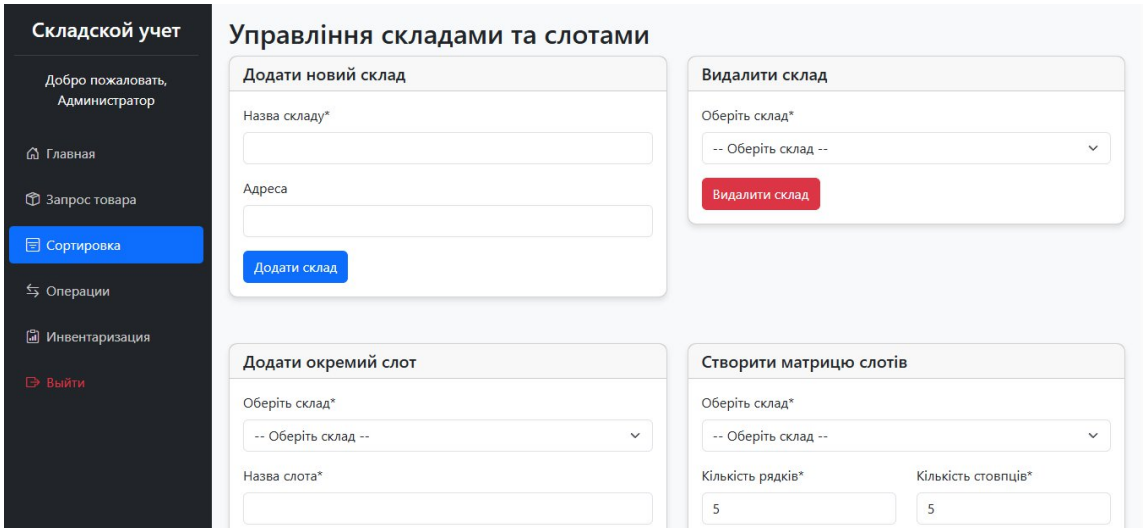


Рис.3.12 Інтерфейс модуля "Сортування"

Інтерфейс сторінки можна розділити на три основні логічні зони:

1. Блок форм для створення складів.

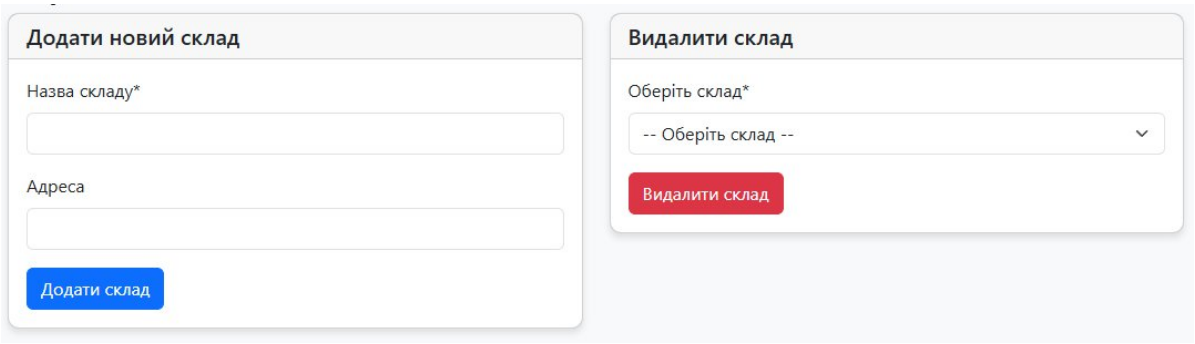


Рис.3.13 Блок форми додавання та видалення товару

2. Блок форм для створення слотів (поодиноко або матрицею).

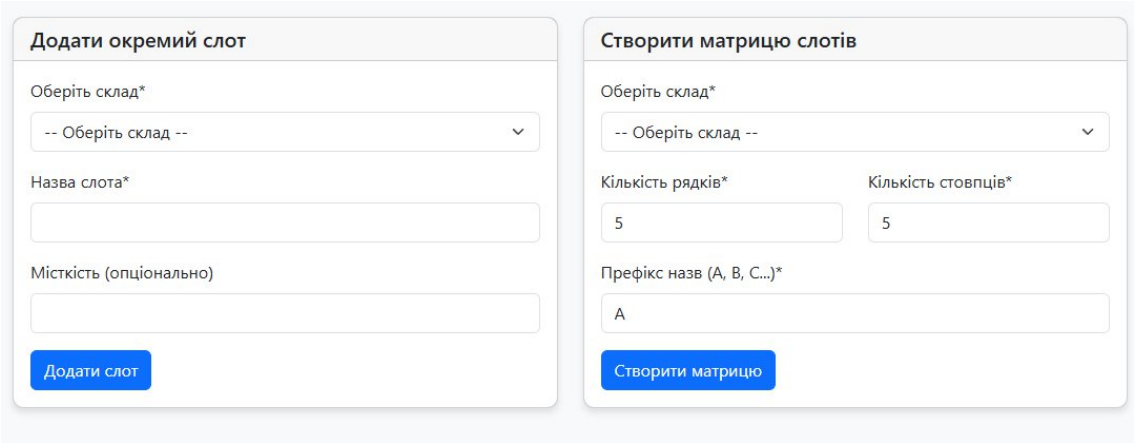


Рис.3.14 Форма для створення слоті

3. Блок візуального представлення всіх створених складів та їх слотів.

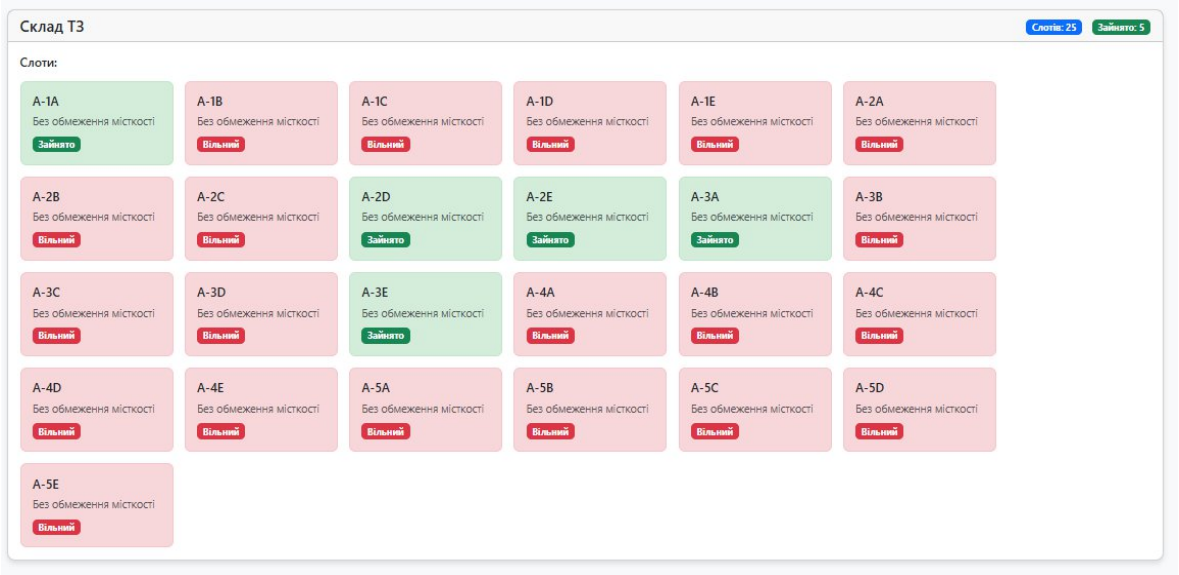


Рис.3.15 Візуальна інтерпретація слотів

Основні функціональні блоки

1. Управління складами:
   - Додавання складу: Користувачеві надається форма для створення нового складу. Вона містить поля для введення назви складу та його адреси. Після відправки форми створюється новий запис у таблиці warehouses.
   - Видалення складу: Окрема форма дозволяє обрати існуючий склад зі списку та "видалити" його. Фактично, для збереження цілісності

даних, запис у базі даних не видаляється, а лише позначається як неактивний (UPDATE warehouses SET active = 0...).

2. Управління слотами:
   - Додавання окремого слота: Форма дозволяє точково створити один слот, вказавши його назву, місткість (опціонально) та обравши склад, до якого він належить.
   - Створення матриці слотів: Для швидкого створення великої кількості однотипних слотів (наприклад, для нового стелажа) реалізовано функцію генерації матриці. Користувач вказує склад, кількість рядків та стовпців, а також префікс для назв (наприклад, "A"). Система автоматично генерує та додає в базу даних слоти з іменами на кшталт "A-1A", "A-1B" і т.д..
3. Візуалізація структури складу:
   - Картки складів: Кожен активний склад відображається у вигляді окремої інформаційної картки (warehouse-card). У заголовку картки виводиться назва складу, адреса та зведена статистика: загальна кількість слотів, кількість зайнятих слотів та відсоток загальної заповненості, якщо для слотів вказана місткість.
   - Картки слотів: Всередині кожної картки складу відображаються всі належні йому слоти у вигляді менших карток (slot-card). Колір картки слота залежить від його стану: зелений (slot-occupied) — якщо у слоті є товар, червоний (slot-empty) — якщо він порожній. Якщо для слота задана місткість, на картці також відображається візуальний індикатор заповненості (slot-capacity-bar).

Інтерактивна взаємодія

Ключовим елементом взаємодії є модальне вікно (slotModal), яке з'являється при натисканні на картку будь-якого слота.

- Завантаження даних: При відкритті вікна за допомогою AJAX-запиту до файлу get_slot_data.php підвантажується детальна інформація про вміст обраного слота.
- Управління вмістом: У модальному вікні користувач може:
  - Змінити кількість існуючого товару у слоті.
  - Додати до слота новий товар зі списку та вказати його кількість.
- Оновлення даних: Усі зміни, зроблені у модальному вікні, відправляються на сервер до скрипта update_inventory.php, який вносить відповідні корективи до таблиці inventory. Після успішного збереження сторінка автоматично перезавантажується для відображення актуального стану.

Звісно, переходимо до опису центрального модуля системи — управління операціями, з урахуванням наданого файлу operations.php.

3.8. Модуль "Управління Операціями" (operations.php)

Призначення та загальний вигляд

Модуль "Управління Операціями", реалізований у файлі operations.php, є ядром всієї системи складського обліку. Він призначений для реєстрації всіх типів руху товарно-матеріальних цінностей. На відміну від інших модулів, які мають вузькоспеціалізоване призначення, цей модуль є універсальним інструментом для виконання більшості щоденних складських завдань.

Ключовою особливістю інтерфейсу є його динамічність. Замість створення окремих сторінок для приходу, продажу чи списання, реалізовано єдину форму, яка адаптує свій вигляд та набір полів залежно від типу операції, обраної користувачем. Інтерфейс сторінки реалізовано за двоколонковою схемою, що включає фіксовану бічну панель для навігації та основну робочу область.

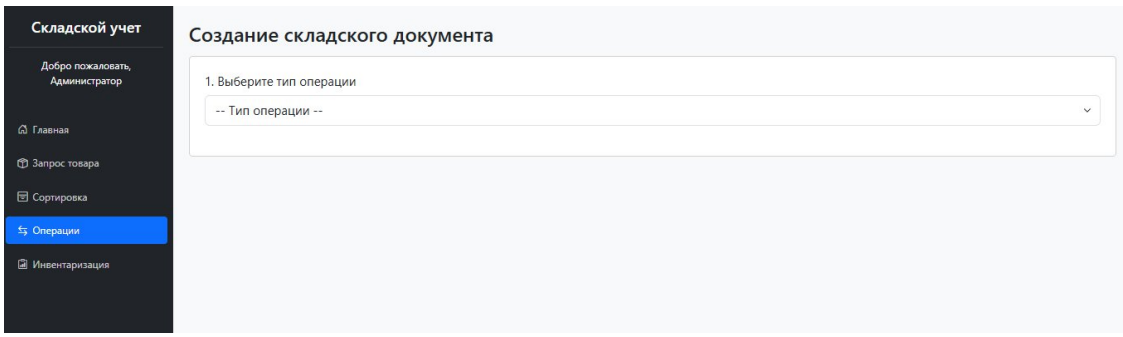


Рис.3.16 Інтерфейс модуля "Операцій"

Основні елементи інтерфейсу

1. Селектор типу операції:
   - Це головний керуючий елемент сторінки — випадаючий список (operationType).
   - Він пропонує користувачеві обрати один з доступних типів операцій: "Приход от поставщика", "Продажа", "Списание" або "Перемещение". Вибір у цьому списку ініціює зміну решти елементів форми за допомогою JavaScript.

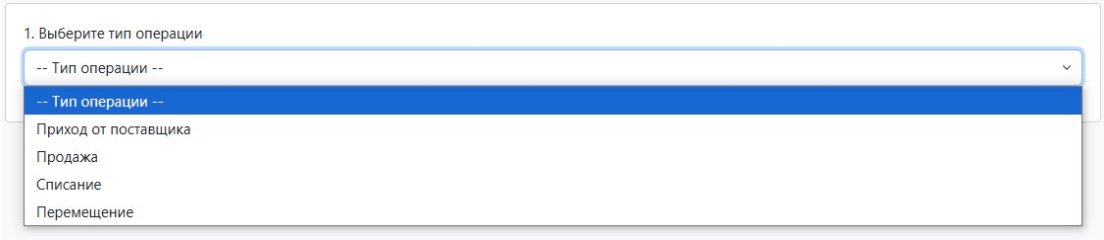


Рис.3.17 Вибір операцій у селекторі

2. Блок даних документа:
   - Цей блок (formFields) містить поля для заповнення "шапки" документа. Його вміст є динамічним:
     - Для приходу з'являються поля для вибору постачальника (supplier_id) та введення номера накладної (invoice_number).

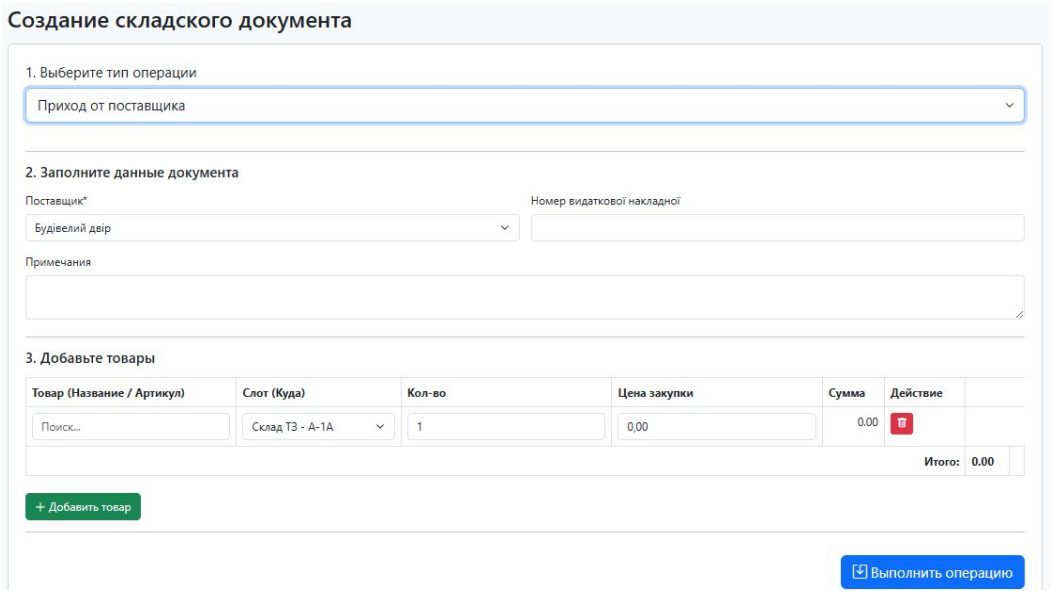


Рис.3.18 Форма прихід від постачальника

- Для продажу — поля для введення назви клієнта (customer), номера видаткової накладної (invoice_number) та номера рахунку-фактури (invoice_ref_number).

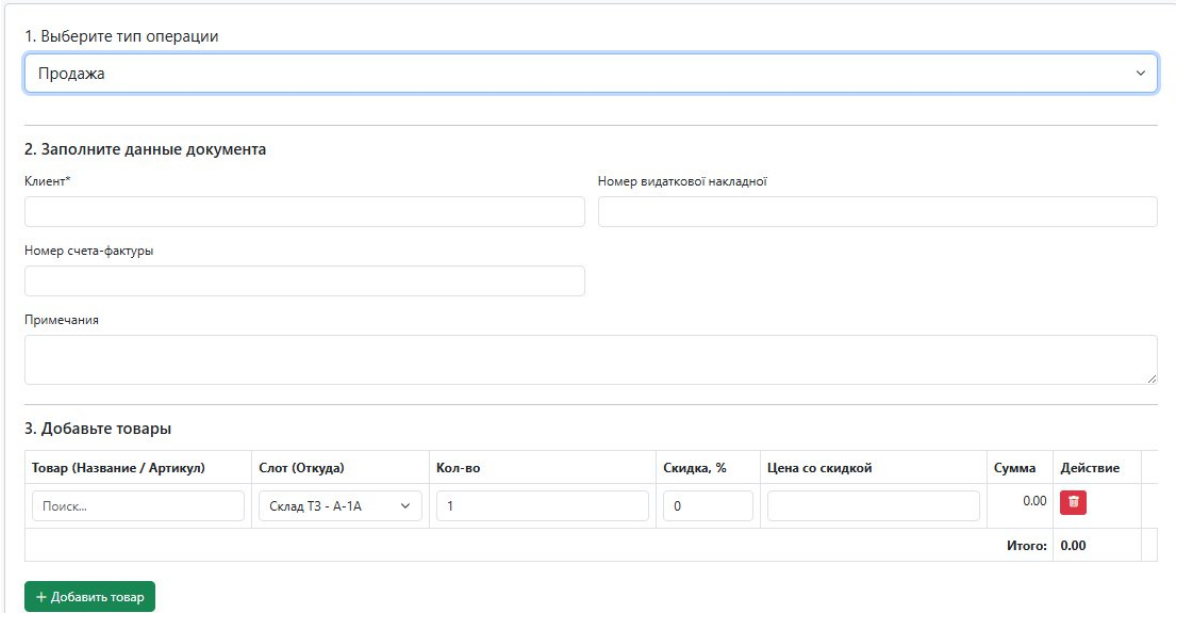


Рис.3.19 Форма продаж для клієнтів та клієнтів за безготівковою формою, тобто організацій

- Для списання — поля для причини списання (reason) та номера рахунку-фактури (invoice_ref_number).

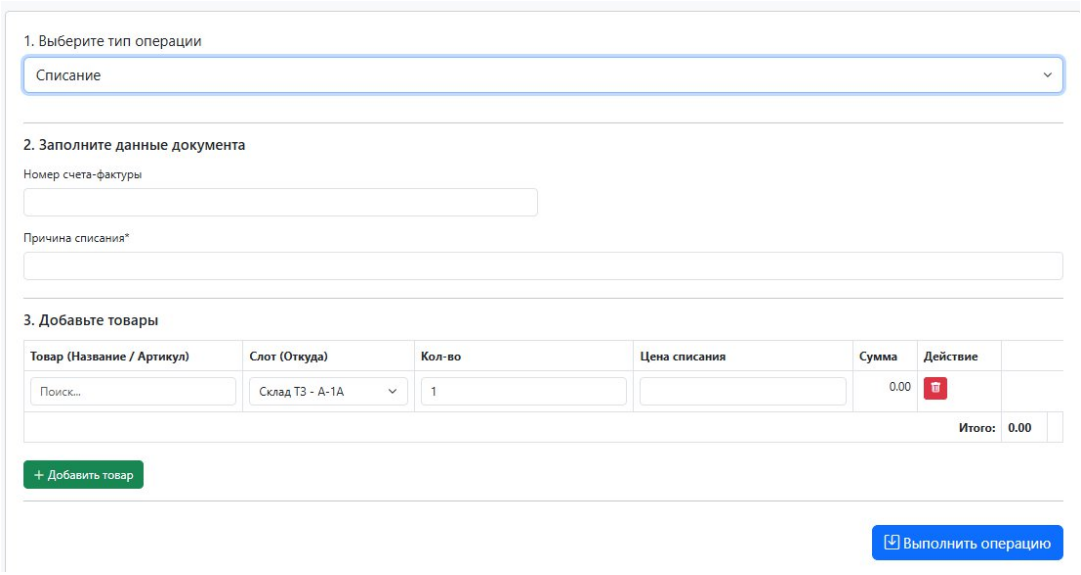


Рис.3.20 Форма списання

- Для переміщення та інших операцій доступне поле для приміток (notes).

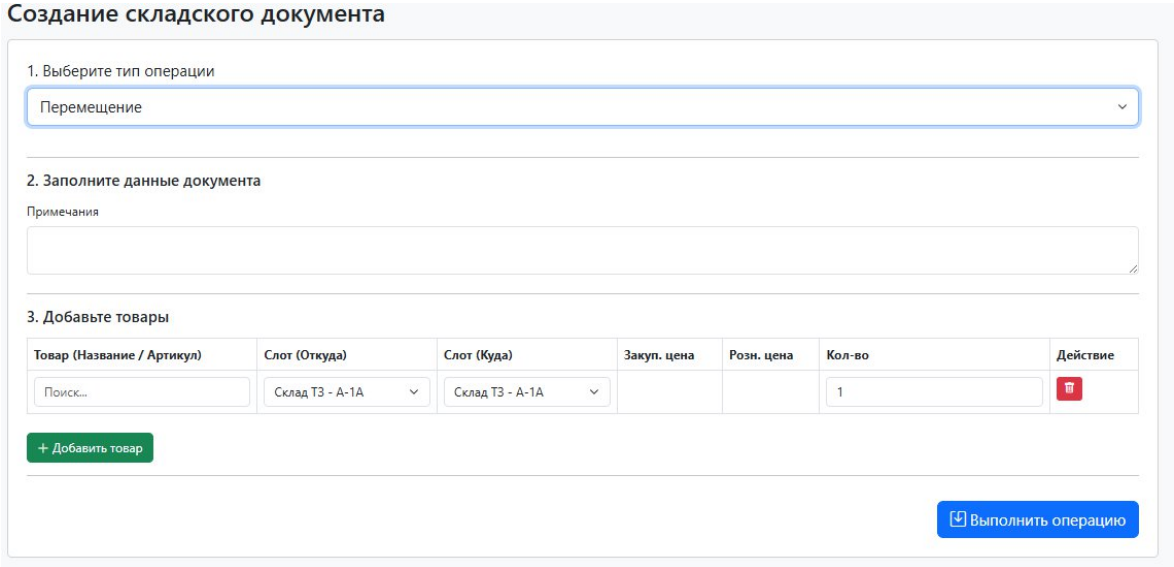


Рис.3.21 Форма Переміщення

3. Таблиця товарів:
   - Це основна інтерактивна частина модуля, де користувач додає товарні позиції до документа.
   - Додавання товару: Здійснюється через кнопку "Добавить товар" (addProductBtn), яка додає новий порожній рядок у таблицю.

- Пошук товару: У кожному рядку є текстове поле (product-search), яке при введенні символів надсилає AJAX-запит до скрипта search_products.php та відображає список знайдених товарів за назвою або артикулом.
- Колонки таблиці: Набір колонок у таблиці також динамічно змінюється. Наприклад, для операції "Перемещение" відображаються колонки "Слот (Откуда)" та "Слот (Куда)", а для "Продажи" — колонка "Скидка, %".
- Розрахунок цін: Система автоматично розраховує ціни залежно від типу операції:
  - При списанні ціна фіксована ("вхідна ціна + 10%") і не редагується.
  - При продажу користувач може вказати відсоток знижки, і фінальна ціна перераховується автоматично.
  - При приході користувач може редагувати закупівельну ціну вручну.

Логіка взаємодії з користувачем

1. Користувач обирає потрібний тип операції у верхньому випадаючому списку.
2. Інтерфейс миттєво адаптується: з'являються відповідні поля для заповнення даних документа та таблиця з потрібними колонками.
3. Користувач натискає кнопку "Добавить товар" для створення нового рядка в таблиці.
4. У рядку він починає вводити назву або артикул товару, обирає потрібний товар зі списку, що з'явився.
5. Система автоматично заповнює дані про товар та розраховує ціну згідно з правилами обраної операції.

6. Користувач вводить кількість, обирає слоти (якщо потрібно) та, за потреби, вказує знижку.
7. Після додавання всіх товарних позицій користувач натискає кнопку "Выполнить операцию". Дані відправляються на сервер, де PHP-скрипт проводить валідацію, виконує відповідні зміни в базі даних (в таблицях documents, document_items, inventory) та виводить повідомлення про успішне виконання операції.

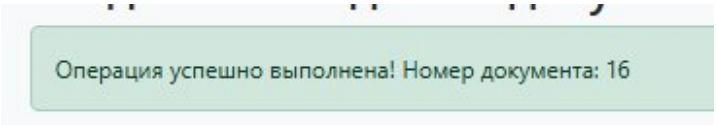


Рис.3.22 Результат обробки документів

Звісно, продовжуємо опис модулів.

3.9. Модуль "Інвентаризація та залишки" (inventory.php)

Призначення та загальний вигляд

Модуль "Інвентаризація та залишки", реалізований у файлі inventory.php, є ключовим інструментом для контролю та коригування товарних запасів. Він виконує дві основні функції: надає актуальну інформацію про кількість товарів у розрізі місць їх зберігання та забезпечує механізм для проведення інвентаризації.

Інтерфейс сторінки складається з двох функціональних блоків:

- Блок проведення інвентаризації: інтерактивна форма для вибору складу/слота та введення фактичних залишків.
- Блок перегляду залишків: зведена таблиця, що відображає поточний стан складу.

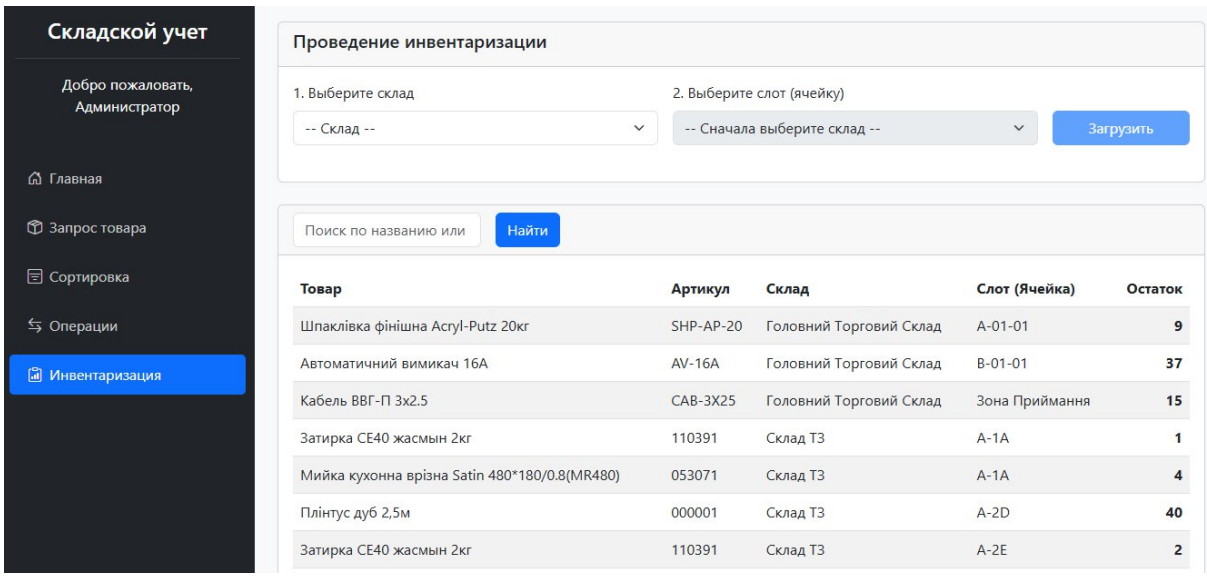


Рис.3.23 Інтерфейс модуля " Інвентаризація"

Основні функціональні блоки

1. Проведення інвентаризації:
   - Вибір місця: Користувач послідовно обирає склад (warehouse_select) та слот (slot_select) для перерахунку. Список слотів динамічно завантажується за допомогою AJAX-запиту до api_get_slots.php після вибору складу.
   - Завантаження даних: Після натискання кнопки "Загрузить" (load_slot_btn) система за допомогою AJAX-запиту до api_get_inventory_for_slot.php підвантажує та відображає таблицю з товарами, що числяться в обраному слоті.
   - Таблиця звірки: У таблиці для кожного товару відображається "Учетное кол-во (БД)" (кількість з бази даних) та надається поле "Фактическое кол-во" (actual-qty-input) для введення результатів фізичного перерахунку. JavaScript "на льоту" розраховує та показує різницю.
   - Збереження результатів: Кнопка "Сохранить результаты инвентаризации" відправляє дані на сервер для обробки.
2. Перегляд залишків:

- Фільтрація: Блок містить форму пошуку, що дозволяє фільтрувати дані за назвою або артикулом товару.
- Таблиця залишків: Основний елемент блоку — це зведена таблиця, що показує повний перелік товарних залишків у системі. Кожен рядок містить назву товару, його артикул, склад, назву слота та точну кількість.

Логіка взаємодії з користувачем

1. Користувач відкриває сторінку inventory.php та бачить загальний список залишків.
2. Для проведення інвентаризації він обирає склад та слот у верхньому блоці.

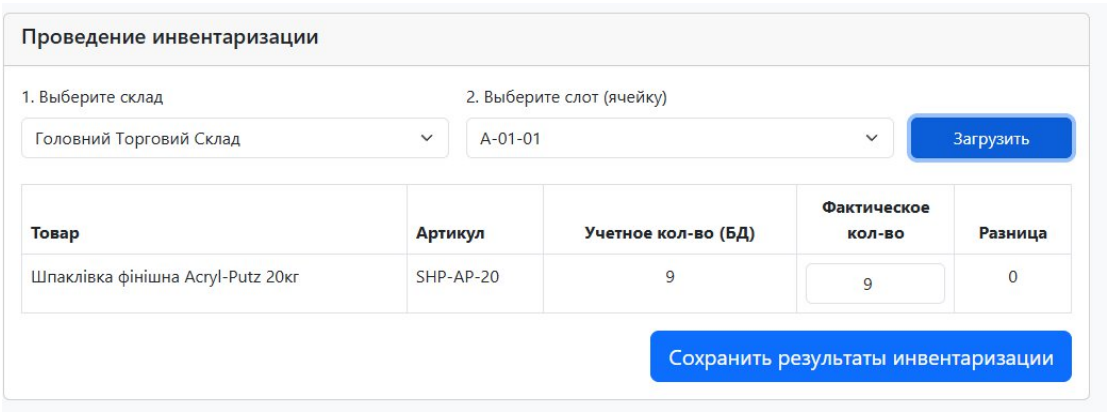


Рис.3.24 Форма інвентаризацій

3. Після завантаження даних вводить фактичну кількість товарів у відповідні поля.
4. Після натискання кнопки "Сохранить результаты" дані відправляються на сервер. PHP-скрипт аналізує розбіжності та, якщо вони є, створює документ типу inventory_adjustment та оновлює дані в таблиці inventory до фактичних значень.
5. Сторінка перезавантажується, і користувач бачить повідомлення про успішне завершення операції, а таблиця залишків відображає вже скориговані дані.

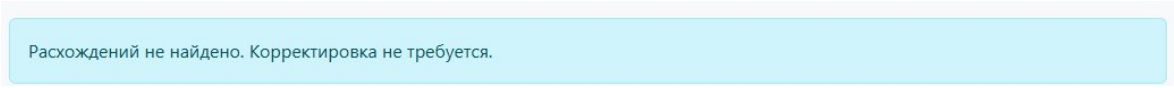


Рис.3.25 Повідомлення після корегування залишків

Звісно, переходимо до фінального етапу — опису тестування для вашої дипломної роботи.

3.10. Тестування

Метою етапу тестування є перевірка працездатності розробленої системи управління складом, виявлення та усунення потенційних помилок, а також підтвердження відповідності реалізованого функціоналу поставленим вимогам.

Для досягнення цієї мети були поставлені наступні завдання:

1. Функціональне тестування: Перевірити коректність роботи всіх ключових бізнес-процесів системи (прихід, розхід, інвентаризація тощо).
2. Тестування інтерфейсу користувача (UI Testing): Перевірити коректність відображення всіх елементів інтерфейсу, їхню інтерактивність та зручність для користувача.
3. Перевірка цілісності даних: Впевнитися, що всі операції коректно відображаються в базі даних, не порушуючи логічних зв'язків та цілісності інформації.

План тестування та тестові сценарії

Тестування проводилося за методом "чорної скриньки", де система перевірялася з точки зору кінцевого користувача без заглиблення у внутрішню структуру коду. Були розроблені тестові сценарії, що охоплюють основні шляхи взаємодії користувача з системою.

Основні сценарії для тестування:

1. Створення базової структури складу (додавання складів, слотів).
2. Створення нових товарних позицій.
3. Проведення операції "Прихід товару" та перевірка оновлення залишків.
4. Проведення операції "Списання товару" та перевірка оновлення залишків.

5. Проведення повної процедури інвентаризації та перевірка коректності коригування.

Опис та результати тестових випадків (Test Cases)

Нижче наведено опис ключових тестових випадків, що були виконані для підтвердження працездатності системи.

Тестовий випадок №1: Авторизація користувача

- Опис: Перевірка можливості входу в систему з коректними даними та відмови в доступі з некоректними.
- Кроки для відтворення:
    1. Відкрити сторінку index.php.
    2. Ввести невірний логін та пароль у відповідні поля (username, password), натиснути "Увійти".
    3. Ввести вірний логін та пароль, натиснути "Увійти".
- Очікуваний результат:
    1. На кроці 2 система повинна показати повідомлення про помилку "Неверное имя пользователя или пароль".

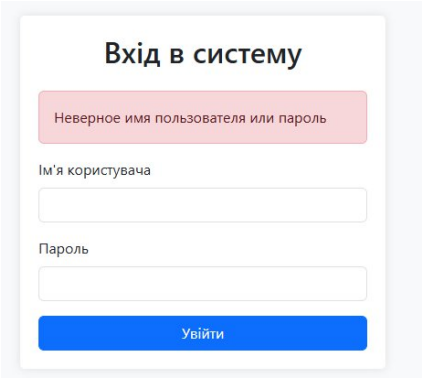


Рис. 3.26 Результат

2. На кроці 3 система повинна успішно перенаправити користувача на головну панель (dashboard.php).

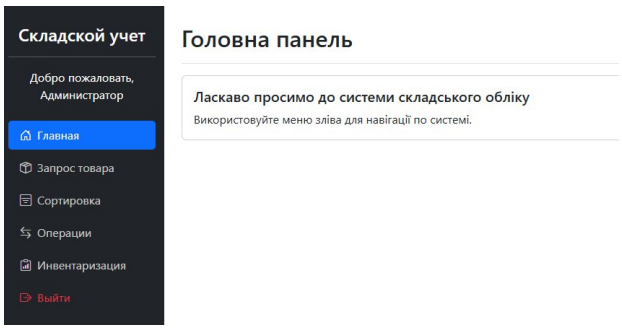


Рис. 3.27 Результат

- Фактичний результат: Відповідає очікуваному.

Тестовий випадок №2: Створення нового товару

- Опис: Перевірка коректності створення нової товарної позиції та розрахунку роздрібної ціни.
- Кроки для відтворення:
    1. Перейти до модуля "Запит товару", натиснути кнопку "Додати товар", щоб відкрити products_add.php.
    2. Заповнити поля: Назва="Тестовий товар", Артикул="TEST-001", Вхідна ціна=100, Накрутка=25%.
    3. Натиснути "Зберегти товар".
- Очікуваний результат:
    1. Роздрібна ціна повинна автоматично розрахуватися як 125.00.

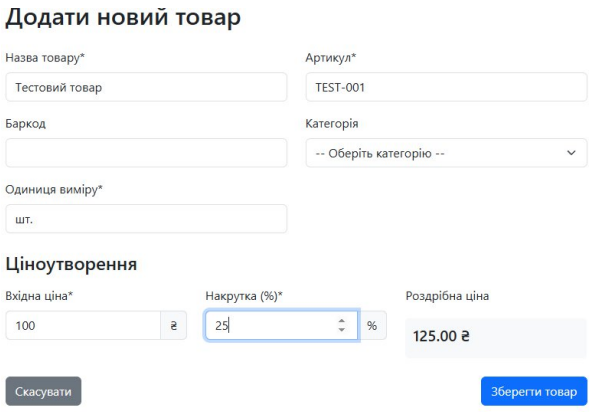


Рис. 3.28 Результат

2. Після збереження з'являється повідомлення про успішне додавання.
3. У таблицях products та product_prices з'являються відповідні нові записи.

- Фактичний результат: Відповідає очікуваному.

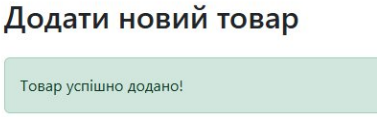


Рис. 3.29 Результат

Тестовий випадок №3: Операція "Прихід товару"

- Опис: Перевірка оприбуткування товару та оновлення залишків.
- Кроки для відтворення:
    1. На сторінці operations.php обрати тип операції "Приход от поставщика".
    2. Заповнити дані документа.
    3. Додати в табличну частину "Тестовий товар" у кількості 10 шт. у слот "A-01-01".
    4. Натиснути "Выполнить операцию".
    5. Перейти на сторінку inventory.php.
- Очікуваний результат:
    1. Операція завершується успішно.

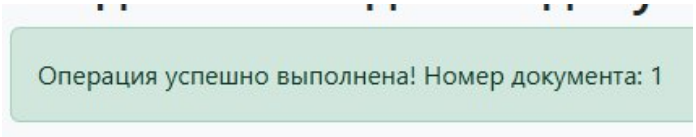
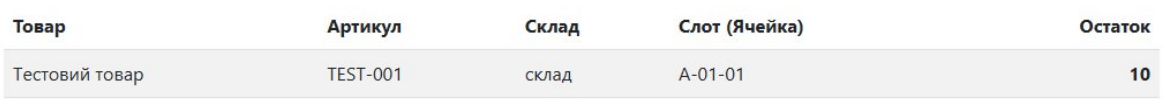


Рис. 3.30 Результат

2. На сторінці інвентаризації для товару "Тестовий товар" у слоті "A-01-01" з'являється залишок 10 шт.



Товар	Артикул	Склад	Слот (Ячейка)	Остаток
Тестовий товар	TEST-001	склад	A-01-01	**10**

Рис. 3.31 Результат

- Фактичний результат: Відповідає очікуваному.

  Тестовий випадок №4: Операція "Списання товару" з контролем залишків
- Опис: Перевірка списання товару та коректного зменшення залишку.
- Кроки для відтворення:
  1. На сторінці operations.php обрати тип операції "Списание".
  2. Додати в табличну частину "Тестовий товар" у кількості 3 шт. зі слота "A-01-01".
  3. Натиснути "Выполнить операцию".
  4. Перейти на сторінку inventory.php.
- Очікуваний результат:
  1. Операція завершується успішно.

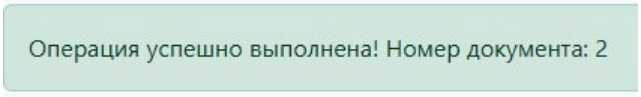
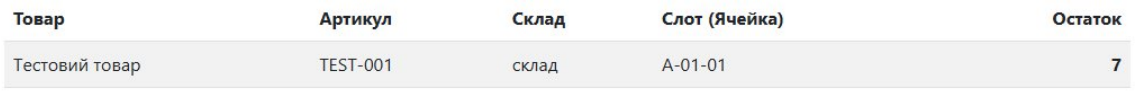


Рис 3.32 Результат

  2. На сторінці інвентаризації залишок товару "Тестовий товар" у слоті "A-01-01" дорівнює 7 шт. (10 - 3).



Товар	Артикул	Склад	Слот (Ячейка)	Остаток
Тестовий товар	TEST-001	склад	A-01-01	7

Рис. 3.33 Результат

- Фактичний результат: Відповідає очікуваному.

  Тестовий випадок №5: Проведення інвентаризації
- Опис: Перевірка процедури коригування залишків.
- Кроки для відтворення:
  1. На сторінці inventory.php обрати склад та слот "A-01-01", натиснути "Загрузить".
  2. Система показує "Тестовий товар" з обліковою кількістю 7 шт.
  3. У поле "Фактическое кол-во" ввести значення 5.

4. Натиснути "Сохранить результаты инвентаризации".

- Очікуваний результат:
  1. Операція завершується успішно, створено документ коригування.

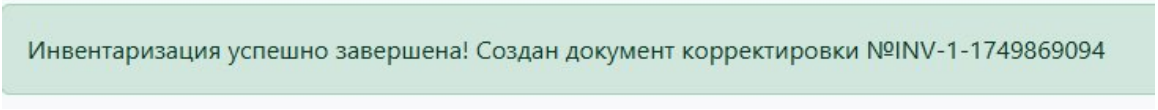


Рис. 3.34 Результат

2. При повторному перегляді сторінки inventory.php залишок товару "Тестовий товар" у слоті "A-01-01" дорівнює 5 шт.

Товар	Артикул	Склад	Слот (Ячейка)	Остаток
Тестовий товар	TEST-001	склад	A-01-01	5

Рис 3.35 Результат

- Фактичний результат: Відповідає очікуваному.

3.11. Інтеграція та розгортання системи

Для розгортання системи у робочому середовищі (production) замість пакета XAMPP, що є інструментом для розробки, пропонується використання класичного та надійного стеку технологій LEMP. Цей підхід забезпечує вищу продуктивність, безпеку та гнучкість у налаштуванні. Абревіатура LEMP розшифровується як:

- L (Linux): В якості операційної системи для сервера пропонується використати дистрибутив Ubuntu Server 22.04 LTS. Це стабільна, довгостроково підтримувана та одна з найпопулярніших ОС для веб-серверів.
- E (Nginx): Замість Apache буде використано веб-сервер Nginx (вимовляється як "Engine-X"). Він відомий своєю високою

продуктивністю, ефективністю при обробці великої кількості одночасних з'єднань та низьким споживанням ресурсів.

- M (MariaDB): Система управління базами даних MariaDB залишається без змін, оскільки вона є надійною, продуктивною та повністю сумісною з розробленою структурою даних.
- P (PHP-FPM): Для обробки PHP-скриптів буде використано PHP-FPM (FastCGI Process Manager). Це стандартний та найбільш ефективний спосіб інтеграції PHP з веб-сервером Nginx.

Такий стек є галузевим стандартом для розгортання надійних та швидких веб-додатків.

Архітектура мережевої інфраструктури

Для візуалізації та проєктування мережі було використано симулятор Cisco Packet Tracer. Запропонована топологія (рисунок 5.1) побудована за ієрархічною моделлю та використовує віртуальні локальні мережі (VLAN) для сегментації трафіку, підвищення безпеки та керованості.

Компоненти інфраструктури:

1. Маршрутизатор (Router): Використовується для забезпечення зв'язку між різними VLAN (inter-VLAN routing). У моделі це може бути пристрій класу Cisco 2901.
2. Центральний комутатор (Core Switch): Є ядром мережі, до якого підключаються сервер та комутатори рівня доступу. Це може бути керований комутатор 3-го рівня, наприклад, Cisco Catalyst 3560, що також може виконувати функції маршрутизації.
3. Комутатори доступу (Access Switches): Звичайні керовані комутатори 2-го рівня (напр., Cisco Catalyst 2960), встановлені у кожній фізичній зоні (офіс, торговий зал, склади) для підключення кінцевих пристроїв.
4. Сервер: Фізичний сервер з встановленим стеком LEMP, підключений до центрального комутатора.

5. Віртуальні локальні мережі (VLAN): Для логічного розділення мережі пропонується створити наступні VLAN:
    - VLAN 10 (Management): Для серверів та мережевого обладнання.
    - VLAN 20 (Office): Для комп'ютерів фінансового відділу та директорату (~5 ПК).
    - VLAN 30 (Sales): Для комп'ютерів у торговельних відділах (12 ПК).
    - VLAN 40 (Logistics): Для комп'ютерів на складах (3 ПК).

Схема мережі, змодельована в Cisco Packet Tracer:

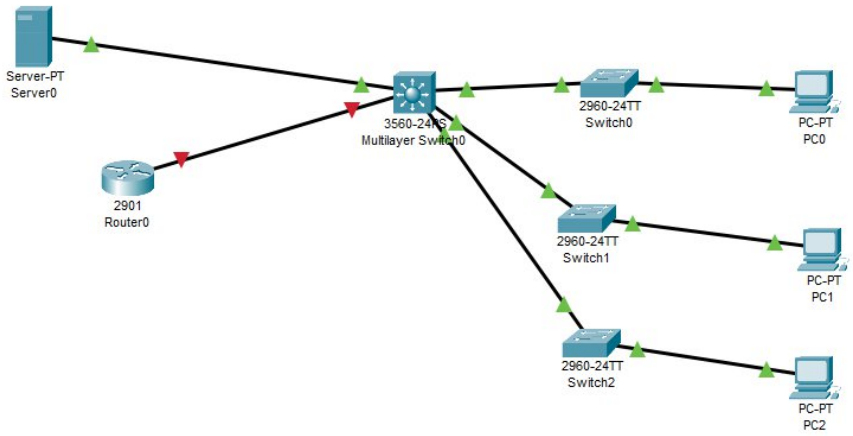


Рис. 3.36 Концептуальна схема мережі, розроблена в Cisco Packet Tracer

План розгортання та інтеграції

1. Налаштування сервера: Встановлення ОС Ubuntu Server, оновлення системи, встановлення та базова конфігурація Nginx, MariaDB та PHP-FPM. Налаштування статичної IP-адреси.
2. Налаштування мережевого обладнання: Конфігурація комутаторів, створення VLAN та налаштування транкових портів між комутаторами. Налаштування маршрутизатора для забезпечення зв'язку між VLAN.
3. Розгортання додатку: Копіювання файлів проєкту на сервер. Створення бази даних та користувача в MariaDB. Імпорт структури БД. Налаштування файлу db_connect.php. Створення та активація

конфігураційного файлу "серверного блоку" (virtual host) в Nginx для сайту.

4. Налаштування клієнтських місць та запуск: Підключення комп'ютерів до відповідних портів комутаторів, перевірка доступу до сервера. Створення облікових записів та проведення навчання персоналу.

Висновки до розділу 3

У даному розділі було детально описано процес технічної реалізації програмної системи для автоматизації складського обліку. На основі проведеної роботи можна зробити наступні висновки:

1. Обґрунтовано вибір технологічного стеку. Для розробки було використано середовище XAMPP, що включає веб-сервер Apache, СУБД MariaDB та мову програмування PHP. Даний набір інструментів дозволив швидко розгорнути локальне середовище та ефективно реалізувати всю серверну логіку, зокрема обробку форм, взаємодію з базою даних та генерацію динамічних веб-сторінок.

2. Спроєктовано та реалізовано комплексну реляційну базу даних `warehouses`. Розроблена структура БД є фундаментом усієї системи. Вона логічно розділяє дані на довідкові (користувачі, товари, категорії, постачальники), структурні (склади, слоти) та операційні (документи, товарні позиції, залишки). Така архітектура забезпечує цілісність даних, мінімізує їх дублювання та створює гнучку основу для фіксації всіх господарських операцій.

3. Розроблено ключові програмні модулі та інтерфейс користувача. Було реалізовано основний функціонал системи у вигляді окремих, але взаємопов'язаних модулів:

Модуль автентифікації забезпечує безпечний доступ до системи.

Модулі управління товарами та структурою складу (`products_add.php`, `sorting.php`) надають адміністративні інструменти для наповнення та налаштування довідників.

Центральний модуль "Управління Операціями" (`operations.php`) є ядром системи, що дозволяє реєструвати всі типи руху товару через єдиний динамічний інтерфейс.

Модуль "Інвентаризація" (`inventory.php`) надає інструменти для контролю та коригування залишків.

Інтерфейс користувача, створений з використанням фреймворку Bootstrap, є адаптивним та інтуїтивно зрозумілим, що спрощує роботу персоналу з системою.

Таким чином, у третьому розділі було повністю описано процес перетворення теоретичних вимог та завдань, поставлених у попередніх розділах, на готовий до тестування програмний продукт з продуманою архітектурою бази даних та реалізованим користувацьким функціоналом.

ВИСНОВКИ

У ході виконання дипломної роботи було успішно досягнуто поставленої мети — розроблено та протестовано програмну систему для автоматизації складського обліку на прикладі філії підприємства ПП «Анжіо. На основі проведеного дослідження, проєктування, розробки та тестування можна зробити наступні узагальнюючі висновки:

Проведений у першому розділі аналіз теоретичних основ складської логістики підтвердив, що в сучасних ринкових умовах ефективне управління складом є критично важливим фактором конкурентоспроможності підприємства. Було встановлено, що ручні методи обліку призводять до помилок та фінансових втрат, тоді як впровадження спеціалізованих WMS-систем дозволяє мінімізувати вплив людського фактора, підвищити точність та швидкість операцій. Для підприємства малого та середнього бізнесу, як-от ПП «Анжіо», було обґрунтовано доцільність розробки власного, адаптованого програмного рішення.

На основі аналізу діяльності ПП «Анжіо», проведеного в другому розділі, було виявлено ключові потреби в автоматизації.Існуюча лінійно-функціональна структура та децентралізована модель складських процесів, хоч і є логічною, створює значні ризики при відсутності єдиної інформаційної системи. Було сформульовано чітку постановку задачі: створити централізований веб-додаток для ведення документообігу, обліку залишків у розрізі слотів та надання персоналу зручного інтерфейсу для виконання операцій.

У третьому розділі було успішно виконано технічну реалізацію системи Було обрано технологічний стек PHP та MariaDB, розгорнутий на локальному сервері XAMPP для розробки. Спроєктовано та створено комплексну реляційну базу даних `warehouses`, що забезпечує цілісність та логічну зв'язність всієї інформації. Було розроблено ключові програмні модулі системи: автентифікації, управління структурою складу, ведення номенклатури, а також

універсальний модуль для виконання всіх типів складських операцій та модуль для проведення інвентаризації.

Проведено функціональне тестування розробленого продукту. За допомогою набору тестових випадків, що імітували реальні сценарії роботи користувачів (авторизація, створення товару, прихід, списання, інвентаризація), було підтверджено, що система працює коректно, а її функціонал відповідає поставленим вимогам. Усі тестові випадки показали, що дані в системі обробляються правильно, а залишки товарів оновлюються згідно з логікою виконаних операцій.

Запропоновано план розгортання системи на реальному підприємстві. Для забезпечення надійності та продуктивності було запропоновано використання серверного стеку LEMP (Linux, Nginx, MariaDB, PHP-FPM. Також, за допомогою симулятора Cisco Packet Tracer, було розроблено архітектуру локальної мережі з використанням VLAN для сегментації трафіку та підвищення безпеки.

Таким чином, у ході виконання дипломної роботи було пройдено всі етапи розробки програмного продукту: від аналізу проблеми та постановки задачі до технічної реалізації, тестування та планування впровадження. Результатом роботи є повноцінний програмний прототип системи управління складом, готовий до розгортання та експлуатації на підприємстві ПП «Анжіо».

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Аналіз ринку систем автоматизації складу в Україні // "Логіст.Today". – Режим доступу: https://logist.today/
2. Довідник з WMS-систем. [Електронний ресурс] // "Склад и Техника". – Режим доступу: https://sitmag.ru/category/wms
3. Дудар Т.Г., Волошин Р.В. Основи логістики. Київ: Центр учбової літератури, 2012. 202 ст.
4. Крикавський Є. В. Логістика. Основи теорії: Підручник. – Львів: «Інтелект-Захід», 2018. – 416 с.
5. Офіційна документація MariaDB. [Електронний ресурс]. Режим доступу: https://mariadb.org/documentation/
6. Офіційна документація PHP. [Електронний ресурс]. Режим доступу: https://www.php.net/manual/uk/
7. Офіційний сайт фреймворку Bootstrap. [Електронний ресурс]. Режим доступу: https://getbootstrap.com/
8. Business to Government (B2G), Business-to-Consumer (B2C), Businessto-Business (B2B). URL: https://www.investopedia.com/terms
9. Classification of Modern Warehousing Complexes. URL: http://starpm.by/en/inform/type/detal.php?ELEMENT_ID=1653
10. Deng Mingxing, Mao Jian, Gan Xingwen. Development of Automated Warehouse Management System. 2018. URL: https://www.matec92 conferences.org/articles/matecconf/pdf/2018/91/matecconf_eitce2018_03051.pdf
11. ER diagrams vs. EER diagrams: What's the difference? URL:https://cacoo.com/blog/er-diagrams-vs-eer-diagrams-whats-thedifference/
12. Frazelle, E. (2016). World-Class Warehousing and Material Handling. McGraw-Hill Education.

13. I.Bychkov,G.Oparin,A.Tchernykh,A.Feoktistov,V.Bogdanov,Yu.Dyadk i n,V.Andrukhov,O.Basharin. Simulation modeling in heterogeneous distributed computing environments to support decisions making in warehouse logistics. 2017. URL: https://www.sciencedirect.com/science/article/pii/S1877705817341802
14. Kardex. URL: https://www.kardex.com/en/
15. Natesan Andiyappillai. Factors Influencing the Successful Implementation of the Warehouse Management System (WMS). 2020.URL: https://www.ijais.org/archives/volume12/number35/andiyappillai-2020-ijais-451896.pdf
16. Nikola Dujmešić, Ivona Bajor, Tomislav Rožić. Intelligent WarehouseManagement System. 2018. URL: https://hrcak.srce.hr/204475
17. Richards, G. (2017). Warehouse Management: A Complete Guide to Improving Efficiency and Minimizing Costs in the Modern Warehouse. Kogan Page.
18. Samir Yerpude,Dr. Tarun Kumar Singhal. SMART Warehouse with Internet of Things supported Inventory Management System. 2018. URL:https://acadpubl.eu/hub/2018-118-24/3/533.pdf
19. Tim King, George Reese, Randy Yarger, Hugh E. Williams, Managing &Using MySQL: Open Source SQL Databases for Managing Information Web Sites. USA: O'Reilly Media. 2002. 444 p.

ДОДАТКИ

Додаток А

Лістинг програми

Модуль Index.php

```
<?php
session_start();

// Подключение к базе данных
$db = new mysqli('localhost', 'root', '', 'warehouses');

if ($db->connect_error) {
    die("Connection failed: " . $db->connect_error);
}

// Обработка формы входа
if ($_SERVER['REQUEST_METHOD'] == 'POST') {
    $username = trim($_POST['username']);
    $password = trim($_POST['password']);

    $stmt = $db->prepare("SELECT id, username, password, full_name FROM users
WHERE username = ?");
    $stmt->bind_param("s", $username);
    $stmt->execute();
    $result = $stmt->get_result();

    if ($result->num_rows == 1) {
        $user = $result->fetch_assoc();

        if (password_verify($password, $user['password'])) {
            $_SESSION['user_id'] = $user['id'];
```

```
            $_SESSION['username'] = $user['username'];
            $_SESSION['full_name'] = $user['full_name'];

            header("Location: dashboard.php");
            exit();
        } else {
            $error = "Неверное имя пользователя или пароль";
        }
    } else {
        $error = "Неверное имя пользователя или пароль";
    }
}
?>
<!DOCTYPE html>
<html lang="uk">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Вхід в систему</title>
    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/css/bootstrap.min.css"
rel="stylesheet">
    <style>
        body {
            background-color: #f8f9fa;
        }
        .login-container {
            max-width: 400px;
            margin: 100px auto;
            padding: 20px;
            background-color: #fff;
```

```
            border-radius: 5px;
            box-shadow: 0 0 10px rgba(0,0,0,0.1);
        }
    </style>
</head>
<body>
    <div class="container">
        <div class="login-container">
            <h2 class="text-center mb-4">Вхід в систему</h2>
            <?php if (isset($error)): ?>
                <div class="alert alert-danger"><?php echo $error; ?></div>
            <?php endif; ?>
            <form method="POST" action="">
                <div class="mb-3">
                    <label for="username" class="form-label">Ім'я користувача</label>
                    <input type="text" class="form-control" id="username" name="username"
required>
                </div>
                <div class="mb-3">
                    <label for="password" class="form-label">Пароль</label>
                    <input type="password" class="form-control" id="password"
name="password" required>
                </div>
                <button type="submit" class="btn btn-primary w-100">Увійти</button>
            </form>
        </div>
    </div>
</body>
</html>
```

Модуль dashboard.php

```
<?php
session_start();

if (!isset($_SESSION['user_id'])) {
    header("Location: index.php");
    exit();
}

// Подключение к базе данных
$db = new mysqli('localhost', 'root', '', 'warehouses');

if ($db->connect_error) {
    die("Connection failed: " . $db->connect_error);
}
?>
<!DOCTYPE html>
<html lang="uk">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Складський облік - Головна</title>
    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/css/bootstrap.min.css"
rel="stylesheet">
    <link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/bootstrap-
icons@1.10.0/font/bootstrap-icons.css">
    <style>
        .sidebar { min-height: 100vh; background-color: #212529; color: white; padding-top:
20px; flex-shrink: 0; display: flex; flex-direction: column; }
        .sidebar .nav-link { color: rgba(255, 255, 255, 0.75); margin-bottom: 5px; border-
radius: 5px; transition: all 0.3s; }
```

```
        .sidebar .nav-link:hover, .sidebar .nav-link.active { color: white; background-color: #0d6efd; }
        .main-content {
            padding: 20px;
        }
    </style>

    </style>
</head>
<body>
    <div class="container-fluid">
        <div class="row">
            <!-- Сайдбар -->
            <div class="col-md-3 col-lg-2 d-md-block sidebar collapse">
                <div class="position-sticky pt-3">
                    <div class="text-center mb-4">
                        <h4>Складской учет</h4>
                        <hr>
                        <p>Добро пожаловать, <?php echo htmlspecialchars($_SESSION['full_name']); ?></p>
                    </div>
                    <ul class="nav flex-column">
            <li class="nav-item"><a class="nav-link active" href="dashboard.php"><i class="bi bi-house-door me-2"></i>Главная</a></li>
            <li class="nav-item"><a class="nav-link" href="product_request.php"><i class="bi bi-box-seam me-2"></i>Запрос товара</a></li>
            <li class="nav-item"><a class="nav-link" href="sorting.php"><i class="bi bi-filter-square me-2"></i>Сортировка</a></li>
            <li class="nav-item"><a class="nav-link" href="operations.php"><i class="bi bi-arrow-left-right me-2"></i>Операции</a></li>
```

```
        <li class="nav-item"><a class="nav-link" href="inventory.php"><i class="bi bi-clipboard-data me-2"></i>Инвентаризация</a></li>
      </ul>
      <ul class="nav flex-column mt-auto mb-3">
        <li class="nav-item"><a class="nav-link text-danger" href="logout.php"><i class="bi bi-box-arrow-right me-2"></i>Выйти</a></li>
      </ul>
        </div>
      </div>

      <!-- Основной контент -->
      <main class="col-md-9 ms-sm-auto col-lg-10 px-md-4 main-content">
        <div class="d-flex justify-content-between flex-wrap flex-md-nowrap align-items-center pt-3 pb-2 mb-3 border-bottom">
          <h1 class="h2">Головна панель</h1>
        </div>

        <div class="row">
          <div class="col-md-12">
            <div class="card">
              <div class="card-body">
                <h5 class="card-title">Ласкаво просимо до системи складського обліку</h5>
                <p class="card-text">Використовуйте меню зліва для навігації по системі.</p>
              </div>
            </div>
          </div>
        </div>
      </main>
```

```
        </div>
    </div>

    <script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/js/bootstrap.bundle.min.js"></scrip
t>
</body>
</html>
```

Модуль db_connect.php

```
<?php
$db = new mysqli('localhost', 'root', '', 'warehouses');

if ($db->connect_error) {
    die("Connection failed: " . $db->connect_error);
}

$db->set_charset("utf8mb4");
?>
```

Модуль logout.php

```
<?php
session_start();
session_unset();
session_destroy();
header("Location: index.php");
exit();
?>
```

Модуль sorting.php

```
<?php
session_start();
require_once 'db_connect.php';
```

```
if (!isset($_SESSION['user_id'])) {
    header("Location: index.php");
    exit();
}

// Функція для створення матриці слотів
function createSlotsMatrix($db, $warehouse_id, $rows, $cols, $prefix) {
    $db->begin_transaction();
    try {
        for ($row = 1; $row <= $rows; $row++) {
            for ($col = 1; $col <= $cols; $col++) {
                $name = $prefix . '-' . $row . chr(64 + $col); // Наприклад: A-1A, A-1B, ...
                $db->query("INSERT INTO slots (warehouse_id, name) VALUES
($warehouse_id, '$name')");
            }
        }
        $db->commit();
        return true;
    } catch (Exception $e) {
        $db->rollback();
        return false;
    }
}

// Обробка форм
$message = '';
if ($_SERVER['REQUEST_METHOD'] == 'POST') {
    if (isset($_POST['add_warehouse'])) {
        // Додавання складу
```

```
        $name = $db->real_escape_string(trim($_POST['name']));
        $address = $db->real_escape_string(trim($_POST['address']));
        $db->query("INSERT INTO warehouses (name, address) VALUES ('$name',
'$address')");
        $message = '<div class="alert alert-success">Склад успішно додано!</div>';
    }
    elseif (isset($_POST['delete_warehouse'])) {
        // Видалення складу
        $id = (int)$_POST['warehouse_id'];
        $db->query("UPDATE warehouses SET active = 0 WHERE id = $id");
        $message = '<div class="alert alert-success">Склад успішно видалено!</div>';
    }
    elseif (isset($_POST['add_single_slot'])) {
        // Додавання окремого слота
        $warehouse_id = (int)$_POST['warehouse_id'];
        $name = $db->real_escape_string(trim($_POST['name']));
        $capacity = !empty($_POST['capacity']) ? (int)$_POST['capacity'] : NULL;
        $db->query("INSERT INTO slots (warehouse_id, name, capacity) VALUES
($warehouse_id, '$name', " . ($capacity ?: 'NULL') . ")");
        $message = '<div class="alert alert-success">Слот успішно додано!</div>';
    }
    elseif (isset($_POST['add_slots_matrix'])) {
        // Додавання матриці слотів
        $warehouse_id = (int)$_POST['warehouse_id'];
        $rows = (int)$_POST['rows'];
        $cols = (int)$_POST['cols'];
        $prefix = $db->real_escape_string(trim($_POST['prefix']));

        if (createSlotsMatrix($db, $warehouse_id, $rows, $cols, $prefix)) {
```

```
            $message = '<div class="alert alert-success">Матриця слотів ('.$rows.'x'.$cols.') успішно створена!</div>';
        } else {
            $message = '<div class="alert alert-danger">Помилка при створенні матриці слотів</div>';
        }
    }
    elseif (isset($_POST['update_inventory'])) {
        // Оновлення інвентарю
        $slot_id = (int)$_POST['slot_id'];
        $product_id = (int)$_POST['product_id'];
        $quantity = (int)$_POST['quantity'];

        if ($quantity > 0) {
            $db->query("INSERT INTO inventory (product_id, slot_id, quantity)
                        VALUES ($product_id, $slot_id, $quantity)
                        ON DUPLICATE KEY UPDATE quantity = $quantity");
        } else {
            $db->query("DELETE FROM inventory WHERE product_id = $product_id AND slot_id = $slot_id");
        }
        $message = '<div class="alert alert-success">Інвентар оновлено!</div>';
    }
}

// Отримання даних
$warehouses = $db->query("SELECT * FROM warehouses WHERE active = 1")->fetch_all(MYSQLI_ASSOC);
$products = $db->query("SELECT p.id, p.name FROM products p")->fetch_all(MYSQLI_ASSOC);
```

```
?>

<!DOCTYPE html>
<html lang="uk">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Управління складами | Склад</title>
    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/css/bootstrap.min.css" rel="stylesheet">
    <link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/bootstrap-icons@1.10.0/font/bootstrap-icons.css">
    <style>
        body { display: flex; min-height: 100vh; margin: 0; }
        .sidebar { width: 250px; background-color: #212529; color: white; padding: 20px 0; flex-shrink: 0; }
        .sidebar .nav-link { color: rgba(255, 255, 255, 0.75); padding: 10px 20px; margin: 5px 10px; border-radius: 5px; transition: all 0.3s; }
        .sidebar .nav-link:hover, .sidebar .nav-link.active { color: white; background-color: rgba(255, 255, 255, 0.1); }
        .sidebar .nav-link.active { background-color: #0d6efd; }
        .main-content { flex-grow: 1; padding: 20px; background-color: #f8f9fa; }
        .warehouse-card { margin-bottom: 30px; border-radius: 10px; box-shadow: 0 4px 6px rgba(0, 0, 0, 0.1); }
        .slots-container { display: flex; flex-wrap: wrap; gap: 15px; margin-top: 15px; }
        .slot-card { width: 200px; border-radius: 8px; padding: 15px; position: relative; transition: transform 0.2s; cursor: pointer; }
        .slot-card:hover { transform: translateY(-3px); }
        .slot-occupied { background-color: #d4edda; border: 1px solid #c3e6cb; }
        .slot-empty { background-color: #f8d7da; border: 1px solid #f5c6cb; }
```

```
        .slot-capacity { height: 5px; background-color: #e9ecef; border-radius: 3px; margin-top: 10px; overflow: hidden; }
        .slot-capacity-bar { height: 100%; background-color: #28a745; }
        .progress-text { font-size: 12px; margin-top: 5px; text-align: center; }
        .matrix-controls { background-color: #f8f9fa; padding: 15px; border-radius: 8px; margin-bottom: 20px; }
    </style>
</head>
<body>
    <div class="sidebar">
        <div class="text-center mb-4 px-3">
            <h4>Складской учет</h4>
            <hr class="text-white-50">
            <p>Добро пожаловать, <br><?= htmlspecialchars($_SESSION['full_name'] ?? 'Пользователь') ?></p>
        </div>
        <ul class="nav flex-column">
            <li class="nav-item"><a class="nav-link" href="dashboard.php"><i class="bi bi-house-door me-2"></i>Главная</a></li>
            <li class="nav-item"><a class="nav-link" href="product_request.php"><i class="bi bi-box-seam me-2"></i>Запрос товара</a></li>
            <li class="nav-item"><a class="nav-link active" href="sorting.php"><i class="bi bi-filter-square me-2"></i>Сортировка</a></li>
            <li class="nav-item"><a class="nav-link" href="operations.php"><i class="bi bi-arrow-left-right me-2"></i>Операции</a></li>
            <li class="nav-item"><a class="nav-link" href="inventory.php"><i class="bi bi-clipboard-data me-2"></i>Инвентаризация</a></li>
        </ul>
        <ul class="nav flex-column mt-auto mb-3">
```

```
            <li class="nav-item"><a class="nav-link text-danger" href="logout.php"><i class="bi bi-box-arrow-right me-2"></i>Выйти</a></li>
        </ul>
    </div>
    <!-- Основний контент -->
    <div class="main-content">
        <h2>Управління складами та слотами</h2>

        <?= $message ?>

        <!-- Форми управління складами -->
        <div class="row">
            <div class="col-md-6">
                <div class="card warehouse-card">
                    <div class="card-header">
                        <h5 class="mb-0">Додати новий склад</h5>
                    </div>
                    <div class="card-body">
                        <form method="POST">
                            <input type="hidden" name="add_warehouse" value="1">
                            <div class="mb-3">
                                <label class="form-label">Назва складу*</label>
                                <input type="text" class="form-control" name="name" required>
                            </div>
                            <div class="mb-3">
                                <label class="form-label">Адреса</label>
                                <input type="text" class="form-control" name="address">
                            </div>
                            <button type="submit" class="btn btn-primary">Додати склад</button>
                        </form>
```

```
            </div>
        </div>
    </div>

    <div class="col-md-6">
        <div class="card warehouse-card">
            <div class="card-header">
                <h5 class="mb-0">Видалити склад</h5>
            </div>
            <div class="card-body">
                <form method="POST">
                    <input type="hidden" name="delete_warehouse" value="1">
                    <div class="mb-3">
                        <label class="form-label">Оберіть склад*</label>
                        <select class="form-select" name="warehouse_id" required>
                            <option value="">-- Оберіть склад --</option>
                            <?php foreach ($warehouses as $warehouse): ?>
                                <option value="<?= $warehouse['id'] ?>"><?=
htmlspecialchars($warehouse['name']) ?></option>
                            <?php endforeach; ?>
                        </select>
                    </div>
                    <button type="submit" class="btn btn-danger">Видалити
склад</button>
                </form>
            </div>
        </div>
    </div>
```

```
<!-- Форми додавання слотів -->
<div class="row mt-4">
  <div class="col-md-6">
    <div class="card warehouse-card">
      <div class="card-header">
        <h5 class="mb-0">Додати окремий слот</h5>
      </div>
      <div class="card-body">
        <form method="POST">
          <input type="hidden" name="add_single_slot" value="1">
          <div class="mb-3">
            <label class="form-label">Оберіть склад*</label>
            <select class="form-select" name="warehouse_id" required>
              <option value="">-- Оберіть склад --</option>
              <?php foreach ($warehouses as $warehouse): ?>
                <option value="<?= $warehouse['id'] ?>"><?=
htmlspecialchars($warehouse['name']) ?></option>
              <?php endforeach; ?>
            </select>
          </div>
          <div class="mb-3">
            <label class="form-label">Назва слота*</label>
            <input type="text" class="form-control" name="name" required>
          </div>
          <div class="mb-3">
            <label class="form-label">Місткість (опціонально)</label>
            <input type="number" class="form-control" name="capacity"
min="1">
          </div>
          <button type="submit" class="btn btn-primary">Додати слот</button>
```

```
                </form>
            </div>
        </div>
    </div>

    <div class="col-md-6">
        <div class="card warehouse-card">
            <div class="card-header">
                <h5 class="mb-0">Створити матрицю слотів</h5>
            </div>
            <div class="card-body">
                <form method="POST">
                    <input type="hidden" name="add_slots_matrix" value="1">
                    <div class="mb-3">
                        <label class="form-label">Оберіть склад*</label>
                        <select class="form-select" name="warehouse_id" required>
                            <option value="">-- Оберіть склад --</option>
                            <?php foreach ($warehouses as $warehouse): ?>
                                <option value="<?= $warehouse['id'] ?>"><?=
htmlspecialchars($warehouse['name']) ?></option>
                            <?php endforeach; ?>
                        </select>
                    </div>
                    <div class="row">
                        <div class="col-md-6 mb-3">
                            <label class="form-label">Кількість рядків*</label>
                            <input type="number" class="form-control" name="rows" min="1"
max="20" value="5" required>
                        </div>
                        <div class="col-md-6 mb-3">
```

```
                        <label class="form-label">Кількість стовпців*</label>
                        <input type="number" class="form-control" name="cols" min="1"
max="20" value="5" required>
                    </div>
                </div>
                <div class="mb-3">
                    <label class="form-label">Префікс назв (A, B, C...)*</label>
                    <input type="text" class="form-control" name="prefix"
maxlength="3" value="A" required>
                </div>
                <button type="submit" class="btn btn-primary">Створити
матрицю</button>
            </form>
        </div>
    </div>
</div>

<!-- Список складів з слотами -->
<?php foreach ($warehouses as $warehouse): ?>
    <?php
    // Отримання слотів для поточного складу
    $slots = $db->query("
        SELECT s.*,
            (SELECT SUM(quantity) FROM inventory WHERE slot_id = s.id) as
occupied,
            COUNT(i.product_id) as product_count
        FROM slots s
        LEFT JOIN inventory i ON s.id = i.slot_id
        WHERE s.warehouse_id = {$warehouse['id']} AND s.active = 1
```

```
            GROUP BY s.id
            ORDER BY s.name
        ")->fetch_all(MYSQLI_ASSOC);

        // Розрахунок зайнятості складу
        $total_slots = count($slots);
        $occupied_slots = 0;
        $total_capacity = 0;
        $used_capacity = 0;

        foreach ($slots as $slot) {
            if ($slot['product_count'] > 0) $occupied_slots++;
            if ($slot['capacity']) {
                $total_capacity += $slot['capacity'];
                $used_capacity += min($slot['occupied'] ?? 0, $slot['capacity']);
            }
        }

        $warehouse_occupancy = $total_capacity > 0 ? ($used_capacity / $total_capacity)
* 100 : 0;
        ?>

        <div class="card mb-4 warehouse-card mt-4">
            <div class="card-header d-flex justify-content-between align-items-center">
                <div>
                    <h5 class="mb-0"><?= htmlspecialchars($warehouse['name']) ?></h5>
                    <small class="text-muted"><?=
htmlspecialchars($warehouse['address']) ?></small>
                </div>
                <div>
```

```
                <span class="badge bg-primary">Слотів: <?= $total_slots ?></span>
                <span class="badge bg-success ms-2">Зайнято: <?=
$occupied_slots ?></span>
                <?php if ($total_capacity > 0): ?>
                    <span class="badge bg-info ms-2">Заповненість: <?=
round($warehouse_occupancy, 1) ?>%</span>
                <?php endif; ?>
            </div>
        </div>
        <div class="card-body">
            <?php if ($total_capacity > 0): ?>
                <div class="mb-3">
                    <div class="d-flex justify-content-between mb-1">
                        <span>Загальна зайнятість:</span>
                        <span><?= round($warehouse_occupancy, 1) ?>%</span>
                    </div>
                    <div class="progress" style="height: 20px;">
                        <div class="progress-bar" role="progressbar"
                            style="width: <?= $warehouse_occupancy ?>%"
                            aria-valuenow="<?= $warehouse_occupancy ?>"
                            aria-valuemin="0"
                            aria-valuemax="100">
                        </div>
                    </div>
                </div>
            <?php endif; ?>

            <h6>Слоти:</h6>
            <div class="slots-container">
                <?php foreach ($slots as $slot): ?>
```

```
                    <?php
                    $is_occupied = $slot['product_count'] > 0;
                    $occupancy_percent = $slot['capacity'] > 0 ?
                       min(100, ($slot['occupied'] / $slot['capacity']) * 100) : 0;
                    ?>
                    <div class="slot-card <?= $is_occupied ? 'slot-occupied' : 'slot-
empty' ?>"
                         data-bs-toggle="modal" data-bs-target="#slotModal"
                         data-slot-id="<?= $slot['id'] ?>">
                        <h6><?= htmlspecialchars($slot['name']) ?></h6>

                        <?php if ($slot['capacity']): ?>
                            <div class="d-flex justify-content-between small">
                                <span>Місткість:</span>
                                <span><?= $slot['occupied'] ?? 0 ?>/<?=
$slot['capacity'] ?></span>
                            </div>
                            <div class="slot-capacity">
                                <div class="slot-capacity-bar" style="width: <?=
$occupancy_percent ?>%"></div>
                            </div>
                            <div class="progress-text">
                                <?= round($occupancy_percent, 1) ?>% заповнено
                            </div>
                        <?php else: ?>
                            <div class="small text-muted">Без обмеження місткості</div>
                        <?php endif; ?>

                        <?php if ($is_occupied): ?>
                            <span class="badge bg-success mt-2">Зайнято</span>
```

```
                        <?php else: ?>
                            <span class="badge bg-danger mt-2">Вільний</span>
                        <?php endif; ?>
                    </div>
                <?php endforeach; ?>
            </div>
        </div>
    <?php endforeach; ?>
</div>

<!-- Модальне вікно для управління слотом -->
<div class="modal fade" id="slotModal" tabindex="-1" aria-hidden="true">
    <div class="modal-dialog modal-lg">
        <div class="modal-content">
            <div class="modal-header">
                <h5 class="modal-title" id="slotModalTitle">Управління слотом</h5>
                <button type="button" class="btn-close" data-bs-dismiss="modal" aria-label="Close"></button>
            </div>
            <div class="modal-body modal-slot-content" id="slotModalContent">
                <!-- Вміст буде завантажено через AJAX -->
            </div>
            <div class="modal-footer">
                <button type="button" class="btn btn-secondary" data-bs-dismiss="modal">Закрити</button>
            </div>
        </div>
    </div>
</div>
```

```
    <script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/js/bootstrap.bundle.min.js"></scrip
t>
    <script>
        // Завантаження даних слота при відкритті модального вікна
        document.getElementById('slotModal').addEventListener('show.bs.modal',
function(event) {
            const button = event.relatedTarget;
            const slotId = button.getAttribute('data-slot-id');

            fetch(`get_slot_data.php?slot_id=${slotId}`)
                .then(response => response.text())
                .then(data => {
                    document.getElementById('slotModalContent').innerHTML = data;
                });
        });

        // Оновлення кількості товару в слоті
        function updateInventory(slotId, productId) {
            const quantity = document.getElementById(`quantity_${productId}`).value;

            fetch('update_inventory.php', {
                method: 'POST',
                headers: {
                    'Content-Type': 'application/x-www-form-urlencoded',
                },
                body:
`slot_id=${slotId}&product_id=${productId}&quantity=${quantity}&update_inventory=
1`
```

```
        })
        .then(response => response.json())
        .then(data => {
            if (data.success) {
                // Оновити модальне вікно
                const modal =
bootstrap.Modal.getInstance(document.getElementById('slotModal'));
                modal._element.addEventListener('hidden.bs.modal', function() {
                    location.reload();
                });
                modal.hide();
            } else {
                alert('Помилка: ' + (data.message || 'Невідома помилка'));
            }
        });
    }

    // Додавання товару до слота
    function addProductToSlot(slotId) {
        const productId = document.getElementById('add_product_id').value;
        const quantity = document.getElementById('add_quantity').value;

        if (!productId) {
            alert('Будь ласка, оберіть товар');
            return;
        }

        fetch('update_inventory.php', {
            method: 'POST',
            headers: {
```

```
                'Content-Type': 'application/x-www-form-urlencoded',
            },
            body:
`slot_id=${slotId}&product_id=${productId}&quantity=${quantity}&update_inventory=
1`
        })
        .then(response => response.json())
        .then(data => {
            if (data.success) {
                // Оновити модальне вікно
                const modal =
bootstrap.Modal.getInstance(document.getElementById('slotModal'));
                modal._element.addEventListener('hidden.bs.modal', function() {
                    location.reload();
                });
                modal.hide();
            } else {
                alert('Помилка: ' + (data.message || 'Невідома помилка'));
            }
        });
    }
  </script>
</body>
</html>
Модуль api_get_inventory_for_slots.php
<?php
require_once 'db_connect.php';

if (!isset($_GET['slot_id'])) {
    die('<div class="alert alert-danger">Не выбран слот.</div>');
```

```
}

$slot_id = (int)$_GET['slot_id'];

$query = "
    SELECT i.quantity, p.id as product_id, p.name as product_name, p.article
    FROM inventory i
    JOIN products p ON i.product_id = p.id
    WHERE i.slot_id = ?
    ORDER BY p.name
";
$stmt = $db->prepare($query);
$stmt->bind_param("i", $slot_id);
$stmt->execute();
$items = $stmt->get_result()->fetch_all(MYSQLI_ASSOC);
?>

<form method="POST">
    <input type="hidden" name="inventory_slot_id" value="<?= $slot_id ?>">
    <table class="table table-bordered">
        <thead>
            <tr>
                <th>Товар</th>
                <th>Артикул</th>
                <th class="text-center">Учетное кол-во (БД)</th>
                <th class="text-center" style="width: 15%;">Фактическое кол-во</th>
                <th class="text-center">Разница</th>
            </tr>
        </thead>
        <tbody>
```

```
        <?php if (empty($items)): ?>
            <tr><td colspan="5" class="text-center">В этом слоте по базе нет товаров.</td></tr>
        <?php else: ?>
            <?php foreach($items as $item): ?>
                <tr>
                    <td><?= htmlspecialchars($item['product_name']) ?></td>
                    <td><?= htmlspecialchars($item['article']) ?></td>
                    <td class="text-center recorded-qty"><?= $item['quantity'] ?></td>
                    <td>
                        <input type="hidden" name="products[<?= $item['product_id'] ?>][recorded]" value="<?= $item['quantity'] ?>">
                        <input type="number" class="form-control text-center actual-qty-input"
                               name="products[<?= $item['product_id'] ?>][actual]"
                               value="<?= $item['quantity'] ?>" min="0" required>
                    </td>
                    <td class="text-center difference">0</td>
                </tr>
            <?php endforeach; ?>
        <?php endif; ?>
    </tbody>
</table>
<div class="text-end mt-3">
    <button type="submit" class="btn btn-primary btn-lg">Сохранить результаты инвентаризации</button>
</div>
</form>
```

Модуль api_get_slots.php

```
<?php
require_once 'db_connect.php';
```

```
header('Content-Type: application/json');

if (!isset($_GET['warehouse_id'])) {
    echo json_encode([]);
    exit();
}

$warehouse_id = (int)$_GET['warehouse_id'];

$stmt = $db->prepare("SELECT id, name FROM slots WHERE warehouse_id = ? AND
active = 1 ORDER BY name");
$stmt->bind_param("i", $warehouse_id);
$stmt->execute();
$slots = $stmt->get_result()->fetch_all(MYSQLI_ASSOC);

echo json_encode($slots);
?>
```

Модуль get_slot_data.php

```
<?php
require_once 'db_connect.php';

$slot_id = (int)$_GET['slot_id'];

// Отримання інформації про слот
$slot = $db->query("SELECT s.*, w.name as warehouse_name
                    FROM slots s
                    JOIN warehouses w ON s.warehouse_id = w.id
                    WHERE s.id = $slot_id")->fetch_assoc();

// Отримання товарів у слоті
```

```
$products_in_slot = $db->query("
    SELECT p.id, p.name, p.unit, i.quantity
    FROM inventory i
    JOIN products p ON i.product_id = p.id
    WHERE i.slot_id = $slot_id
")->fetch_all(MYSQLI_ASSOC);

// Отримання всіх доступних товарів
$all_products = $db->query("
    SELECT p.id, p.name
    FROM products p
    WHERE p.id NOT IN (SELECT product_id FROM inventory WHERE slot_id =
$slot_id)
")->fetch_all(MYSQLI_ASSOC);

// Розрахунок зайнятості
$used_quantity = array_sum(array_column($products_in_slot, 'quantity'));
$capacity_percent = $slot['capacity'] > 0 ? min(100, ($used_quantity / $slot['capacity']) *
100) : 0;
?>

<div class="container">
    <h4><?= htmlspecialchars($slot['name']) ?></h4>
    <p><strong>Склад:</strong> <?= htmlspecialchars($slot['warehouse_name']) ?></p>
    <?php if ($slot['description']): ?>
        <p><?= htmlspecialchars($slot['description']) ?></p>
    <?php endif; ?>

    <?php if ($slot['capacity']): ?>
        <div class="mb-3">
```

```
        <div class="d-flex justify-content-between">
            <span>Місткість:</span>
            <span>
                <?= $used_quantity ?> / <?= $slot['capacity'] ?>
            </span>
        </div>
        <div class="progress" style="height: 20px;">
            <div class="progress-bar" role="progressbar"
                 style="width: <?= $capacity_percent ?>%"
                 aria-valuenow="<?= $used_quantity ?>"
                 aria-valuemin="0"
                 aria-valuemax="<?= $slot['capacity'] ?>">
            </div>
        </div>
        <div class="text-end small">
            <?= round($capacity_percent, 1) ?>% заповнено
        </div>
    </div>
<?php endif; ?>

<h5 class="mt-4">Товари в слоті:</h5>
<?php if (!empty($products_in_slot)): ?>
    <div class="table-responsive">
        <table class="table table-striped">
            <thead>
                <tr>
                    <th>Товар</th>
                    <th>Кількість</th>
                    <th>Од. вим.</th>
                    <th>Дія</th>
```

```
                </tr>
            </thead>
            <tbody>
                <?php foreach ($products_in_slot as $product): ?>
                    <tr>
                        <td><?= htmlspecialchars($product['name']) ?></td>
                        <td>
                            <input type="number" id="quantity_<?= $product['id'] ?>"
                                   class="form-control form-control-sm"
                                   value="<?= $product['quantity'] ?>" min="0">
                        </td>
                        <td><?= $product['unit'] ?></td>
                        <td>
                            <button class="btn btn-sm btn-primary"
                                    onclick="updateInventory(<?= $slot_id ?>, <?=
$product['id'] ?>)">
                                Оновити
                            </button>
                        </td>
                    </tr>
                <?php endforeach; ?>
            </tbody>
        </table>
    </div>
<?php else: ?>
    <div class="alert alert-info">Слот порожній</div>
<?php endif; ?>

<h5 class="mt-4">Додати товар:</h5>
<?php if (!empty($all_products)): ?>
```

```
        <div class="row g-3">
            <div class="col-md-6">
                <select class="form-select" id="add_product_id">
                    <option value="">-- Оберіть товар --</option>
                    <?php foreach ($all_products as $product): ?>
                        <option value="<?= $product['id'] ?>"><?=
htmlspecialchars($product['name']) ?></option>
                    <?php endforeach; ?>
                </select>
            </div>
            <div class="col-md-3">
                <input type="number" class="form-control" id="add_quantity" min="1"
value="1">
            </div>
            <div class="col-md-3">
                <button class="btn btn-success w-100"
                        onclick="addProductToSlot(<?= $slot_id ?>)">
                    <i class="bi bi-plus-lg"></i> Додати
                </button>
            </div>
        </div>
    <?php else: ?>
        <div class="alert alert-info">Всі товари вже додані до цього слота</div>
    <?php endif; ?>
</div>
```

Модуль inventory.php

```
<?php
session_start();
require_once 'db_connect.php';
```

```
if (!isset($_SESSION['user_id'])) {
    header("Location: index.php");
    exit();
}

$message = '';
$user_id = (int)$_SESSION['user_id'];

// --- ОБРАБОТКА ФОРМЫ ИНВЕНТАРИЗАЦИИ ---
if ($_SERVER['REQUEST_METHOD'] == 'POST' &&
isset($_POST['inventory_slot_id'])) {
    $slot_id = (int)$_POST['inventory_slot_id'];
    $products_data = $_POST['products'] ?? [];

    if (empty($products_data)) {
        $message = '<div class="alert alert-warning">Не было отправлено ни одного
товара для корректировки.</div>';
    } else {
        $db->begin_transaction();
        try {
            // Создаем один документ для всей операции инвентаризации
            $doc_number = 'INV-' . $slot_id . '-' . time();
            $notes = "Инвентаризация слота ID: {$slot_id}";

            $stmt_doc = $db->prepare("INSERT INTO documents (type, number, notes,
created_by) VALUES ('inventory_adjustment', ?, ?, ?)");
            $stmt_doc->bind_param("ssi", $doc_number, $notes, $user_id);
            $stmt_doc->execute();
            $document_id = $db->insert_id;
```

```
        $has_changes = false;
        foreach ($products_data as $product_id => $data) {
            $product_id = (int)$product_id;
            $recorded_qty = (int)$data['recorded'];
            $actual_qty = (int)$data['actual'];

            $delta = $actual_qty - $recorded_qty;

            // Если есть разница, создаем запись и корректируем остаток
            if ($delta != 0) {
                $has_changes = true;
                // Добавляем запись в документ, количество - это разница (может быть отрицательным)
                $stmt_item = $db->prepare("INSERT INTO document_items (document_id, product_id, quantity, source_slot_id) VALUES (?, ?, ?, ?)");
                $stmt_item->bind_param("iiii", $document_id, $product_id, $delta, $slot_id);
                $stmt_item->execute();

                // Устанавливаем точное фактическое количество в таблице остатков
                $stmt_inv = $db->prepare("UPDATE inventory SET quantity = ? WHERE product_id = ? AND slot_id = ?");
                $stmt_inv->bind_param("iii", $actual_qty, $product_id, $slot_id);
                $stmt_inv->execute();
            }
        }

        if (!$has_changes) {
            // Если изменений не было, откатываем создание документа, чтобы не плодить пустые
            $db->rollback();
```

```
            $message = '<div class="alert alert-info">Расхождений не найдено. Корректировка не требуется.</div>';
        } else {
            $db->commit();
            $message = '<div class="alert alert-success">Инвентаризация успешно завершена! Создан документ корректировки №' . $doc_number . '</div>';
        }

    } catch (Exception $e) {
        $db->rollback();
        $message = '<div class="alert alert-danger">Ошибка при проведении инвентаризации: ' . $e->getMessage() . '</div>';
    }
  }
}


// --- ПОЛУЧЕНИЕ ДАННЫХ ДЛЯ ОТОБРАЖЕНИЯ ---
$warehouses = $db->query("SELECT id, name FROM warehouses WHERE active = 1 ORDER BY name")->fetch_all(MYSQLI_ASSOC);

$search_term = $_GET['search'] ?? '';
$where_clause = '';
$params = [];
$types = '';

if (!empty($search_term)) {
    $where_clause = "WHERE p.name LIKE ? OR p.article LIKE ?";
    $params[] = '%' . $search_term . '%';
    $params[] = '%' . $search_term . '%';
```

```
    $types = 'ss';
}

$query = "
    SELECT i.quantity, p.name as product_name, p.article as product_article, s.name as
slot_name, w.name as warehouse_name
    FROM inventory i
    JOIN products p ON i.product_id = p.id
    JOIN slots s ON i.slot_id = s.id
    JOIN warehouses w ON s.warehouse_id = w.id
    {$where_clause}
    ORDER BY w.name, s.name, p.name
";

$stmt = $db->prepare($query);
if (!empty($params)) {
    $stmt->bind_param($types, ...$params);
}
$stmt->execute();
$inventory_items = $stmt->get_result()->fetch_all(MYSQLI_ASSOC);
?>
<!DOCTYPE html>
<html lang="ru">
<head>
    <meta charset="UTF-8">
    <title>Инвентаризация и остатки | Склад</title>
    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/css/bootstrap.min.css"
rel="stylesheet">
    <link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/bootstrap-
icons@1.10.0/font/bootstrap-icons.css">
```

```
    <style>
        body { display: flex; min-height: 100vh; margin: 0; background-color: #f8f9fa;}
        .sidebar { width: 280px; background-color: #212529; color: white; padding-top: 20px; flex-shrink: 0; display: flex; flex-direction: column; }
        .sidebar .nav-link { color: rgba(255, 255, 255, 0.75); padding: 10px 20px; margin: 5px 10px; border-radius: 5px; transition: all 0.3s; }
        .sidebar .nav-link:hover, .sidebar .nav-link.active { color: white; background-color: #0d6efd; }
        .main-content { flex-grow: 1; padding: 25px; }
        .diff-positive { color: green; font-weight: bold; }
        .diff-negative { color: red; font-weight: bold; }
    </style>
</head>
<body>
    <div class="sidebar">
      <div class="text-center mb-4 px-3">
          <h4>Складской учет</h4>
          <hr class="text-white-50">
          <p>Добро пожаловать, <br><?= htmlspecialchars($_SESSION['full_name'] ?? 'Пользователь') ?></p>
      </div>
      <ul class="nav flex-column">
          <li class="nav-item"><a class="nav-link" href="dashboard.php"><i class="bi bi-house-door me-2"></i>Главная</a></li>
          <li class="nav-item"><a class="nav-link" href="product_request.php"><i class="bi bi-box-seam me-2"></i>Запрос товара</a></li>
          <li class="nav-item"><a class="nav-link" href="sorting.php"><i class="bi bi-filter-square me-2"></i>Сортировка</a></li>
          <li class="nav-item"><a class="nav-link" href="operations.php"><i class="bi bi-arrow-left-right me-2"></i>Операции</a></li>
```

```
            <li class="nav-item"><a class="nav-link active" href="inventory.php"><i class="bi bi-clipboard-data me-2"></i>Инвентаризация</a></li>
        </ul>
        <ul class="nav flex-column mt-auto mb-3">
            <li class="nav-item"><a class="nav-link text-danger" href="logout.php"><i class="bi bi-box-arrow-right me-2"></i>Выйти</a></li>
        </ul>
    </div>

    <div class="main-content">
        <?php if (!empty($message)) echo $message; ?>

        <div class="card mb-4">
            <div class="card-header">
                <h5>Проведение инвентаризации</h5>
            </div>
            <div class="card-body">
                <div class="row g-3">
                    <div class="col-md-5">
                        <label for="warehouse_select" class="form-label">1. Выберите склад</label>
                        <select id="warehouse_select" class="form-select">
                            <option value="">-- Склад --</option>
                            <?php foreach ($warehouses as $warehouse): ?>
                                <option value="<?= $warehouse['id'] ?>"><?= htmlspecialchars($warehouse['name']) ?></option>
                            <?php endforeach; ?>
                        </select>
                    </div>
                    <div class="col-md-5">
```

```
                    <label for="slot_select" class="form-label">2. Выберите слот (ячейку)</label>
                    <select id="slot_select" class="form-select" disabled>
                        <option value="">-- Сначала выберите склад --</option>
                    </select>
                </div>
                 <div class="col-md-2 d-flex align-items-end">
                    <button id="load_slot_btn" class="btn btn-primary w-100" disabled>Загрузить</button>
                </div>
            </div>
            <div id="inventory_form_container" class="mt-4"></div>
        </div>
    </div>


    <div class="card">
        <div class="card-header">
            <form method="GET" class="row g-3 align-items-center">
                <div class="col-auto"><input type="text" class="form-control" name="search" placeholder="Поиск по названию или артикулу..." value="<?= htmlspecialchars($search_term) ?>"></div>
                <div class="col-auto"><button type="submit" class="btn btn-primary">Найти</button></div>
            </form>
        </div>
        <div class="card-body">
            <div class="table-responsive"><table class="table table-striped table-hover">
                <thead><tr><th>Товар</th><th>Артикул</th><th>Склад</th><th>Слот (Ячейка)</th><th class="text-end">Остаток</th></tr></thead>
```

```
                <tbody>
                    <?php if (empty($inventory_items)): ?>
                        <tr><td colspan="5" class="text-center">Нет данных об остатках.</td></tr>
                    <?php else: ?>
                        <?php foreach ($inventory_items as $item): ?>
                            <tr><td><?= htmlspecialchars($item['product_name']) ?></td><td><?= htmlspecialchars($item['product_article']) ?></td><td><?= htmlspecialchars($item['warehouse_name']) ?></td><td><?= htmlspecialchars($item['slot_name']) ?></td><td class="text-end fw-bold"><?= $item['quantity'] ?></td></tr>
                        <?php endforeach; ?>
                    <?php endif; ?>
                </tbody>
            </table></div>
        </div>
    </div>
</div>

<script>
    // Загрузка слотов при выборе склада
    const warehouseSelect = document.getElementById('warehouse_select');
    const slotSelect = document.getElementById('slot_select');
    const loadBtn = document.getElementById('load_slot_btn');

    warehouseSelect.addEventListener('change', function() {
        const warehouseId = this.value;
        slotSelect.innerHTML = '<option value="">-- Загрузка... --</option>';
        slotSelect.disabled = true;
```

```
        loadBtn.disabled = true;

        if (!warehouseId) {
            slotSelect.innerHTML = '<option value="">-- Сначала выберите склад --
</option>';
            return;
        }

        fetch(`api_get_slots.php?warehouse_id=${warehouseId}`)
            .then(response => response.json())
            .then(slots => {
                slotSelect.innerHTML = '<option value="">-- Выберите слот --</option>';
                slots.forEach(slot => {
                    slotSelect.innerHTML += `<option
value="${slot.id}">${slot.name}</option>`;
                });
                slotSelect.disabled = false;
            });
    });

    slotSelect.addEventListener('change', function() {
        loadBtn.disabled = !this.value;
    });

    // Загрузка формы инвентаризации
    loadBtn.addEventListener('click', function() {
        const slotId = slotSelect.value;
        if (!slotId) return;

        const container = document.getElementById('inventory_form_container');
```

```
    container.innerHTML = '<p>Загрузка данных по слоту...</p>';

    fetch(`api_get_inventory_for_slot.php?slot_id=${slotId}`)
        .then(response => response.text())
        .then(html => {
            container.innerHTML = html;
            // Навешиваем обработчики для расчета разницы
            container.querySelectorAll('.actual-qty-input').forEach(input => {
                input.addEventListener('input', calculateDifference);
            });
        });
});

function calculateDifference(event) {
    const input = event.target;
    const row = input.closest('tr');
    const recordedQty = parseInt(row.querySelector('.recorded-qty').textContent);
    const actualQty = parseInt(input.value) || 0;
    const diff = actualQty - recordedQty;

    const diffCell = row.querySelector('.difference');
    diffCell.textContent = diff;
    diffCell.className = 'difference '; // сброс классов
    if (diff > 0) {
        diffCell.classList.add('diff-positive');
        diffCell.textContent = '+' + diff;
    } else if (diff < 0) {
        diffCell.classList.add('diff-negative');
    }
}
```

```
    </script>
</body>
</html>
```

Модуль operations.php

```
<?php
ini_set('display_errors', 1);
error_reporting(E_ALL);

session_start();
require_once 'db_connect.php';

$message = '';
$suppliers = [];
$all_slots = [];

// --- ПОЛУЧЕНИЕ ДАННЫХ ДЛЯ ФОРМ (ЗАЩИЩЕННАЯ ВЕРСИЯ) ---
if (isset($db) && $db) {
    // Получаем поставщиков
    $suppliers_result = $db->query("SELECT id, name FROM suppliers ORDER BY name");
    if ($suppliers_result) {
        $suppliers = $suppliers_result->fetch_all(MYSQLI_ASSOC);
    } else {
        $message .= '<div class="alert alert-danger">Ошибка загрузки списка поставщиков: ' . $db->error . '</div>';
    }

    // Получаем слоты
```

```
    $slots_result = $db->query("SELECT s.id, s.name, w.name as warehouse_name FROM slots s JOIN warehouses w ON s.warehouse_id = w.id WHERE s.active = 1 ORDER BY w.name, s.name");
    if ($slots_result) {
        $all_slots = $slots_result->fetch_all(MYSQLI_ASSOC);
    } else {
        $message .= '<div class="alert alert-danger">Ошибка загрузки списка слотов: ' . $db->error . '</div>';
    }
} else {
     $message .= '<div class="alert alert-danger">Критическая ошибка: нет соединения с базой данных.</div>';
}

// --- ОБРАБОТЧИК POST-ЗАПРОСА (ФИНАЛЬНАЯ ВЕРСИЯ) ---
if ($_SERVER['REQUEST_METHOD'] == 'POST'
&& !empty($_POST['operation_type'])) {
    $operation_type = $_POST['operation_type'];
    $user_id = (int)$_SESSION['user_id'];

    $db->begin_transaction();
    try {
        if (empty($_POST['products'])) {
            throw new Exception("Не добавлено ни одного товара в документ.");
        }

        $document_id = null;
        $notes = $_POST['notes'] ?? '';

        // --- Создаем "шапку" документа ---
```

```
        $stmt_doc = null;
        switch ($operation_type) {
            case 'arrival_supplier':
                $stmt_doc = $db->prepare("INSERT INTO documents (type, number,
supplier_id, notes, created_by) VALUES (?, ?, ?, ?, ?)");
                $stmt_doc->bind_param("ssisi", $operation_type, $_POST['invoice_number'],
$_POST['supplier_id'], $notes, $user_id);
                break;
            case 'departure_sale':
                $stmt_doc = $db->prepare("INSERT INTO documents (type, number,
invoice_ref_number, counterparty, notes, created_by) VALUES (?, ?, ?, ?, ?, ?)");
                $stmt_doc->bind_param("sssssi", $operation_type, $_POST['invoice_number'],
$_POST['invoice_ref_number'], $_POST['customer'], $notes, $user_id);
                break;
            case 'departure_writeoff':
                $doc_number = 'WR-' . time();
                $stmt_doc = $db->prepare("INSERT INTO documents (type, number,
invoice_ref_number, notes, created_by) VALUES (?, ?, ?, ?, ?)");
                $stmt_doc->bind_param("ssssi", $operation_type, $doc_number,
$_POST['invoice_ref_number'], $_POST['reason'], $user_id);
                break;
            case 'transfer':
                $doc_number = 'TR-' . time();
                $stmt_doc = $db->prepare("INSERT INTO documents (type, number, notes,
created_by) VALUES (?, ?, ?, ?)");
                $stmt_doc->bind_param("sssi", $operation_type, $doc_number, $notes,
$user_id);
                break;
            default:
                throw new Exception("Неверный тип операции.");
```

```
    }

    $stmt_doc->execute();
    $document_id = $db->insert_id;

    // --- Обрабатываем товары в документе ---
    foreach ($_POST['products'] as $product_data) {
        $product_id = (int)($product_data['id'] ?? 0);
        $quantity = (int)($product_data['quantity'] ?? 0);
        if ($product_id === 0 || $quantity === 0) continue;

        if ($operation_type === 'arrival_supplier') {
            $price = (float)$product_data['price'];
            $stmt_item = $db->prepare("INSERT INTO document_items (document_id,
product_id, quantity, price, destination_slot_id) VALUES (?, ?, ?, ?, ?)");
            $stmt_item->bind_param("iiidi", $document_id, $product_id, $quantity, $price,
$product_data['dest_slot_id']);
            $stmt_item->execute();
            $stmt_inv = $db->prepare("INSERT INTO inventory (product_id, slot_id,
quantity) VALUES (?, ?, ?) ON DUPLICATE KEY UPDATE quantity = quantity +
VALUES(quantity)");
            $stmt_inv->bind_param("iii", $product_id, $product_data['dest_slot_id'],
$quantity);
            $stmt_inv->execute();
        } elseif (in_array($operation_type, ['departure_sale', 'departure_writeoff'])) {
            $source_slot_id = (int)$product_data['source_slot_id'];
            $price = (float)$product_data['price'];
            $inv_check = $db->prepare("SELECT quantity FROM inventory WHERE
product_id = ? AND slot_id = ?");
            $inv_check->bind_param("ii", $product_id, $source_slot_id);
```

```
            $inv_check->execute();
            $stock = $inv_check->get_result()->fetch_assoc();
            if (!$stock || $stock['quantity'] < $quantity) { throw new
Exception("Недостаточно товара (ID: {$product_id}) в слоте."); }
            $stmt_item = $db->prepare("INSERT INTO document_items (document_id,
product_id, quantity, price, source_slot_id) VALUES (?, ?, ?, ?, ?)");
            $stmt_item->bind_param("iiidi", $document_id, $product_id, $quantity, $price,
$source_slot_id);
            $stmt_item->execute();
            $stmt_inv = $db->prepare("UPDATE inventory SET quantity = quantity - ?
WHERE product_id = ? AND slot_id = ?");
            $stmt_inv->bind_param("iii", $quantity, $product_id, $source_slot_id);
            $stmt_inv->execute();
        } elseif ($operation_type === 'transfer') {
            $source_slot_id = (int)$product_data['source_slot_id'];
            $dest_slot_id = (int)$product_data['dest_slot_id'];
            if ($source_slot_id == $dest_slot_id) { throw new Exception("Слот-источник и
слот-получатель не могут совпадать."); }
            $inv_check = $db->prepare("SELECT quantity FROM inventory WHERE
product_id = ? AND slot_id = ?");
            $inv_check->bind_param("ii", $product_id, $source_slot_id);
            $inv_check->execute();
            $stock = $inv_check->get_result()->fetch_assoc();
            if (!$stock || $stock['quantity'] < $quantity) { throw new
Exception("Недостаточно товара (ID: {$product_id}) в слоте-источнике."); }
            $stmt_item = $db->prepare("INSERT INTO document_items (document_id,
product_id, quantity, source_slot_id, destination_slot_id) VALUES (?, ?, ?, ?, ?)");
            $stmt_item->bind_param("iiiii", $document_id, $product_id, $quantity,
$source_slot_id, $dest_slot_id);
            $stmt_item->execute();
```

```
            $stmt_inv_out = $db->prepare("UPDATE inventory SET quantity = quantity - ?
WHERE product_id = ? AND slot_id = ?");
            $stmt_inv_out->bind_param("iii", $quantity, $product_id, $source_slot_id);
            $stmt_inv_out->execute();
            $stmt_inv_in = $db->prepare("INSERT INTO inventory (product_id, slot_id,
quantity) VALUES (?, ?, ?) ON DUPLICATE KEY UPDATE quantity = quantity +
VALUES(quantity)");
            $stmt_inv_in->bind_param("iii", $product_id, $dest_slot_id, $quantity);
            $stmt_inv_in->execute();
        }
    }
    $db->commit();
    $message = '<div class="alert alert-success">Операция успешно выполнена!
Номер документа: ' . $document_id . '</div>';
  } catch (Exception $e) {
    $db->rollback();
    $message = '<div class="alert alert-danger">Ошибка: ' . $e->getMessage() . '</div>';
  }
}
?>
<!DOCTYPE html>
<html lang="ru">
<head>
  <meta charset="UTF-8">
  <title>Управление Операциями | Склад</title>
  <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/css/bootstrap.min.css"
rel="stylesheet">
  <link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/bootstrap-
icons@1.10.0/font/bootstrap-icons.css">
  <style>
```

```
        body { display: flex; min-height: 100vh; margin: 0; background-color: #f8f9fa;}
        .sidebar { width: 280px; background-color: #212529; color: white; padding-top: 20px; flex-shrink: 0; display: flex; flex-direction: column; }
        .sidebar .nav-link { color: rgba(255, 255, 255, 0.75); padding: 10px 20px; margin: 5px 10px; border-radius: 5px; transition: all 0.3s; }
        .sidebar .nav-link:hover, .sidebar .nav-link.active { color: white; background-color: #0d6efd; }
        .main-content { flex-grow: 1; padding: 25px; }
        .dynamic-field { display: none; }
        .product-search-results { position: absolute; z-index: 1000; width: 100%; background: white; border: 1px solid #ddd; max-height: 200px; overflow-y: auto; }
        .product-search-results .list-group-item { cursor: pointer; }
    </style>
</head>
<body>
    <div class="sidebar">
        <div class="text-center mb-4 px-3">
            <h4>Складской учет</h4>
            <hr class="text-white-50">
            <p>Добро пожаловать, <br><?= htmlspecialchars($_SESSION['full_name'] ?? 'Пользователь') ?></p>
        </div>
        <ul class="nav flex-column">
            <li class="nav-item"><a class="nav-link" href="dashboard.php"><i class="bi bi-house-door me-2"></i>Главная</a></li>
            <li class="nav-item"><a class="nav-link" href="product_request.php"><i class="bi bi-box-seam me-2"></i>Запрос товара</a></li>
            <li class="nav-item"><a class="nav-link" href="sorting.php"><i class="bi bi-filter-square me-2"></i>Сортировка</a></li>
```

```
        <li class="nav-item"><a class="nav-link active" href="operations.php"><i class="bi bi-arrow-left-right me-2"></i>Операции</a></li>
        <li class="nav-item"><a class="nav-link" href="inventory.php"><i class="bi bi-clipboard-data me-2"></i>Инвентаризация</a></li>
      </ul>
      <ul class="nav flex-column mt-auto mb-3">
        <li class="nav-item"><a class="nav-link text-danger" href="logout.php"><i class="bi bi-box-arrow-right me-2"></i>Выйти</a></li>
      </ul>
    </div>

    <div class="main-content">
      <h2>Создание складского документа</h2>

      <?php if (!empty($message)) echo $message; ?>

      <form id="operationForm" method="POST" class="card p-4 mt-3">
        <div class="row">
          <div class="col-md-12 mb-4">
            <label for="operationType" class="form-label fs-5">1. Выберите тип операции</label>
            <select id="operationType" name="operation_type" class="form-select form-select-lg" required>
              <option value="" selected>-- Тип операции --</option>
              <option value="arrival_supplier">Приход от поставщика</option>
              <option value="departure_sale">Продажа</option>
              <option value="departure_writeoff">Списание</option>
              <option value="transfer">Перемещение</option>
            </select>
          </div>
```

```
</div>

<div id="formFields" class="dynamic-field">
  <hr><h5 class="mb-3">2. Заполните данные документа</h5>
  <div class="row g-3">
    <div class="dynamic-field field-arrival_supplier col-md-6"><label class="form-label">Поставщик*</label><select name="supplier_id" class="form-select"><?php foreach ($suppliers as $s) echo "<option value='{$s['id']}'>".htmlspecialchars($s['name'])."</option>"; ?></select></div>
    <div class="dynamic-field field-departure_sale col-md-6"><label class="form-label">Клиент*</label><input type="text" name="customer" class="form-control"></div>
    <div class="dynamic-field field-arrival_supplier field-departure_sale col-md-6"><label class="form-label">Номер видаткової накладної</label><input type="text" name="invoice_number" class="form-control"></div>
    <div class="dynamic-field field-departure_sale field-departure_writeoff col-md-6"><label class="form-label">Номер счета-фактуры</label><input type="text" name="invoice_ref_number" class="form-control"></div>
    <div class="dynamic-field field-departure_writeoff col-12"><label class="form-label">Причина списания*</label><input type="text" name="reason" class="form-control"></div>
    <div class="dynamic-field field-arrival_supplier field-departure_sale field-transfer col-12"><label class="form-label">Примечания</label><textarea name="notes" class="form-control" rows="2"></textarea></div>
  </div>
  <hr class="mt-4"><h5 class="mb-3">3. Добавьте товары</h5>
  <div class="table-responsive"><table class="table table-bordered" id="productsTable"><thead id="productsTableHeader"></thead><tbody></tbody><tfoot id="productsTableFooter" style="display:none;"><tr><td colspan="6" class="text-
```

```
end"><strong>Итого:</strong></td><td id="documentTotal" class="fw-bold">0.00</td><td></td></tr></tfoot></table></div>
            <button type="button" id="addProductBtn" class="btn btn-success mt-2"><i class="bi bi-plus-lg"></i> Добавить товар</button>
        </div>
        <hr id="submitDivider" style="display:none;"><div id="submitButtonContainer" style="display:none;" class="text-end mt-3"><button type="submit" class="btn btn-primary btn-lg"><i class="bi bi-save"></i> Выполнить операцию</button></div>
    </form>
</div>

<script>
const allSlots = <?= json_encode($all_slots) ?>;
let productRowCounter = 0;

document.getElementById('operationType').addEventListener('change', function() {
    const type = this.value;
    const formFields = document.getElementById('formFields');
    document.querySelectorAll('.dynamic-field').forEach(el => el.style.display = 'none');
    document.querySelector('#productsTable tbody').innerHTML = '';
    productRowCounter = 0;
    updateTableHeader(null);
    ['productsTableFooter', 'submitDivider', 'submitButtonContainer'].forEach(id => document.getElementById(id).style.display = 'none');

    if (!type) { formFields.style.display = 'none'; return; }

    formFields.style.display = 'block';
    ['submitDivider', 'submitButtonContainer'].forEach(id => document.getElementById(id).style.display = 'block');
```

```
        document.querySelectorAll(`.field-${type}`).forEach(el => el.style.display = 'block');
        updateTableHeader(type);
        addProductRow();
    });

    document.getElementById('addProductBtn').addEventListener('click', addProductRow);

    function updateTableHeader(type) {
        const header = document.getElementById('productsTableHeader');
        if (!type) { header.innerHTML = ''; return; }
        let html = '<tr><th>Товар (Название / Артикул)</th>';
        if (type === 'transfer') html += '<th>Слот (Откуда)</th><th>Слот (Куда)</th><th>Закуп. цена</th><th>Розн. цена</th>';
        else if (type === 'arrival_supplier') html += '<th>Слот (Куда)</th>';
        else if (['departure_sale', 'departure_writeoff'].includes(type)) html += '<th>Слот (Откуда)</th>';
        html += '<th>Кол-во</th>';
        if (type === 'arrival_supplier') html += '<th>Цена закупки</th><th>Сумма</th>';
        else if (type === 'departure_sale') html += '<th>Скидка, %</th><th>Цена со скидкой</th><th>Сумма</th>';
        else if (type === 'departure_writeoff') html += '<th>Цена списания</th><th>Сумма</th>';
        html += '<th>Действие</th></tr>';
        header.innerHTML = html;
        document.getElementById('productsTableFooter').style.display = ['arrival_supplier', 'departure_sale', 'departure_writeoff'].includes(type) ? '' : 'none';
    }

    function addProductRow() {
        const type = document.getElementById('operationType').value;
```

```
        if (!type) return;
        const tableBody = document.querySelector('#productsTable tbody');
        const rowId = `productRow_${productRowCounter++}`;
        const row = document.createElement('tr');
        row.id = rowId;
        const slotsOptions = allSlots.map(s => `<option
value="${s.id}">${s.warehouse_name} - ${s.name}</option>`).join('');

        let html = `<td><input type="text" class="form-control product-search"
placeholder="Поиск..."><input type="hidden" name="products[${rowId}][id]"><div
class="product-search-results list-group"></div></td>`;
        if (type === 'transfer') html += `<td><select
name="products[${rowId}][source_slot_id]" class="form-select"
required>${slotsOptions}</select></td><td><select
name="products[${rowId}][dest_slot_id]" class="form-select"
required>${slotsOptions}</select></td><td class="purchase-price-display text-
end"></td><td class="selling-price-display text-end"></td>`;
        else if (type === 'arrival_supplier') html += `<td><select
name="products[${rowId}][dest_slot_id]" class="form-select"
required>${slotsOptions}</select></td>`;
        else if (['departure_sale', 'departure_writeoff'].includes(type)) html += `<td><select
name="products[${rowId}][source_slot_id]" class="form-select"
required>${slotsOptions}</select></td>`;

        html += `<td><input type="number" name="products[${rowId}][quantity]"
class="form-control quantity" min="1" value="1" required></td>`;

        if (type === 'arrival_supplier') html += `<td><input type="number"
name="products[${rowId}][price]" class="form-control price" min="0" step="0.01"
value="0.00"></td><td class="total text-end">0.00</td>`;
```

```
        else if (type === 'departure_sale') html += `<td><input type="number" class="form-control discount" name="products[${rowId}][discount]" value="0" min="0" max="10" step="1"></td><td><input type="text" name="products[${rowId}][price]" class="form-control price" readonly></td><td class="total text-end">0.00</td>`;
        else if (type === 'departure_writeoff') html += `<td><input type="text" name="products[${rowId}][price]" class="form-control price" readonly></td><td class="total text-end">0.00</td>`;

        html += `<td><button type="button" class="btn btn-danger btn-sm" onclick="removeRow('${rowId}')"><i class="bi bi-trash"></i></button></td>`;

        row.innerHTML = html;
        tableBody.appendChild(row);
        initRow(row);
    }

    function initRow(row) {
        row.querySelector('.product-search').addEventListener('input', function() { /* ... код поиска ... */ });
        row.querySelector('.quantity').addEventListener('input', () => calculateRowTotal(row));
        const priceInput = row.querySelector('.price');
        if (priceInput && !priceInput.readOnly) priceInput.addEventListener('input', () => calculateRowTotal(row));
        const discountInput = row.querySelector('.discount');
        if (discountInput) discountInput.addEventListener('input', () => calculateRowTotal(row));
    }

    function selectProduct(row, product) {
```

```
        row.querySelector('.product-search').value = `${product.name} (Арт: ${product.article})`;
        row.querySelector('input[name*="[id]"]').value = product.id;
        row.dataset.purchasePrice = product.purchase_price || 0;
        row.dataset.sellingPrice = product.selling_price || 0;
        const purchasePriceDisplay = row.querySelector('.purchase-price-display');
        if (purchasePriceDisplay) purchasePriceDisplay.textContent = (parseFloat(row.dataset.purchasePrice)).toFixed(2);
        const sellingPriceDisplay = row.querySelector('.selling-price-display');
        if (sellingPriceDisplay) sellingPriceDisplay.textContent = (parseFloat(row.dataset.sellingPrice)).toFixed(2);
        calculateRowTotal(row);
    }

    function removeRow(rowId) { document.getElementById(rowId).remove(); calculateDocumentTotal(); }

    function calculateRowTotal(row) {
        const totalCell = row.querySelector('.total');
        const quantity = parseFloat(row.querySelector('.quantity')?.value) || 0;
        const priceInput = row.querySelector('.price');
        const type = document.getElementById('operationType').value;
        let finalPrice = 0;
        const basePurchasePrice = parseFloat(row.dataset.purchasePrice) || 0;
        const baseSellingPrice = parseFloat(row.dataset.sellingPrice) || 0;
        if (type === 'arrival_supplier') {
            finalPrice = parseFloat(priceInput.value) || 0;
        } else if (type === 'departure_writeoff') {
            finalPrice = basePurchasePrice * 1.10;
        } else if (type === 'departure_sale') {
```

```
            const discount = parseFloat(row.querySelector('.discount')?.value) || 0;
            finalPrice = baseSellingPrice * (1 - discount / 100);
        }
        if (priceInput) priceInput.value = finalPrice.toFixed(2);
        if (totalCell) totalCell.textContent = (quantity * finalPrice).toFixed(2);
        calculateDocumentTotal();
    }

    function calculateDocumentTotal() {
        let total = 0;
        document.querySelectorAll('#productsTable tbody .total').forEach(cell => total +=
parseFloat(cell.textContent) || 0);
        document.getElementById('documentTotal').textContent = total.toFixed(2);
    }
    // Код для поиска внутри initRow
    function initRow(row) {
        const searchInput = row.querySelector('.product-search');
        const resultsContainer = row.querySelector('.product-search-results');
        searchInput.addEventListener('input', function() {
            if (this.value.length < 2) { resultsContainer.style.display = 'none'; return; }
            fetch(`search_products.php?term=${encodeURIComponent(this.value)}`)
                .then(response => response.json()).then(products => {
                    resultsContainer.innerHTML = '';
                    products.forEach(product => {
                        const item = document.createElement('a');
                        item.href = "#"; item.className = 'list-group-item list-group-item-action';
                        item.innerHTML = `<strong>${product.name}</strong> <small
class="text-muted">(Арт: ${product.article})</small>`;
                        item.onclick = e => { e.preventDefault(); selectProduct(row, product);
resultsContainer.style.display = 'none'; };
```

```
                resultsContainer.appendChild(item);
            });
            resultsContainer.style.display = 'block';
        });
    });
    document.addEventListener('click', e => { if (!e.target.closest('.product-search-
results')) { resultsContainer.style.display = 'none'; } });
    row.querySelector('.quantity').addEventListener('input', () =>
calculateRowTotal(row));
    const priceInput = row.querySelector('.price');
    if (priceInput && !priceInput.readOnly) priceInput.addEventListener('input', () =>
calculateRowTotal(row));
    const discountInput = row.querySelector('.discount');
    if (discountInput) discountInput.addEventListener('input', () =>
calculateRowTotal(row));
  }
  </script>
</body>
</html>
```

Модуль product_requets.php

```
<?php
session_start();
require_once 'db_connect.php';

// Перевірка авторизації
if (!isset($_SESSION['user_id'])) {
  header("Location: index.php");
  exit();
}
```

```
// Створення таблиць, якщо вони не існують
$db->query("CREATE TABLE IF NOT EXISTS categories (
    id INT(11) AUTO_INCREMENT PRIMARY KEY,
    name VARCHAR(50) NOT NULL
) COLLATE utf8mb4_unicode_ci");

$db->query("CREATE TABLE IF NOT EXISTS products (
    id INT(11) AUTO_INCREMENT PRIMARY KEY,
    name VARCHAR(100) NOT NULL,
    article VARCHAR(50),
    barcode VARCHAR(50),
    category_id INT(11),
    unit VARCHAR(20) DEFAULT 'шт.',
    created_at DATETIME DEFAULT CURRENT_TIMESTAMP,
    FOREIGN KEY (category_id) REFERENCES categories(id)
) COLLATE utf8mb4_unicode_ci");

$db->query("CREATE TABLE IF NOT EXISTS product_prices (
    id INT(11) AUTO_INCREMENT PRIMARY KEY,
    product_id INT(11) NOT NULL,
    purchase_price DECIMAL(12,2) NOT NULL,
    selling_price DECIMAL(12,2) NOT NULL,
    markup_percent DECIMAL(5,2) NOT NULL,
    active TINYINT(1) DEFAULT 1,
    created_at DATETIME DEFAULT CURRENT_TIMESTAMP,
    FOREIGN KEY (product_id) REFERENCES products(id)
) COLLATE utf8mb4_unicode_ci");

$db->query("CREATE TABLE IF NOT EXISTS warehouses (
```

```
    id INT(11) AUTO_INCREMENT PRIMARY KEY,
    name VARCHAR(50) NOT NULL,
    address VARCHAR(255),
    active TINYINT(1) DEFAULT 1
) COLLATE utf8mb4_unicode_ci");

$db->query("CREATE TABLE IF NOT EXISTS slots (
    id INT(11) AUTO_INCREMENT PRIMARY KEY,
    warehouse_id INT(11) NOT NULL,
    name VARCHAR(50) NOT NULL,
    description VARCHAR(255),
    capacity INT(11),
    active TINYINT(1) DEFAULT 1,
    FOREIGN KEY (warehouse_id) REFERENCES warehouses(id)
) COLLATE utf8mb4_unicode_ci");

$db->query("CREATE TABLE IF NOT EXISTS inventory (
    id INT(11) AUTO_INCREMENT PRIMARY KEY,
    product_id INT(11) NOT NULL,
    slot_id INT(11) NOT NULL,
    quantity INT(11) DEFAULT 0,
    last_updated DATETIME DEFAULT CURRENT_TIMESTAMP ON UPDATE
CURRENT_TIMESTAMP,
    UNIQUE KEY product_slot (product_id, slot_id),
    FOREIGN KEY (product_id) REFERENCES products(id),
    FOREIGN KEY (slot_id) REFERENCES slots(id)
) COLLATE utf8mb4_unicode_ci");

$db->query("CREATE TABLE IF NOT EXISTS product_requests (
    id INT(11) AUTO_INCREMENT PRIMARY KEY,
```

```
    product_id INT(11) NOT NULL,
    requested_quantity INT(11) NOT NULL,
    requester_id INT(11) NOT NULL,
    warehouse_id INT(11) NOT NULL,
    status ENUM('pending','approved','rejected','completed') DEFAULT 'pending',
    notes TEXT,
    created_at DATETIME DEFAULT CURRENT_TIMESTAMP,
    updated_at DATETIME ON UPDATE CURRENT_TIMESTAMP,
    FOREIGN KEY (product_id) REFERENCES products(id),
    FOREIGN KEY (warehouse_id) REFERENCES warehouses(id),
    FOREIGN KEY (requester_id) REFERENCES users(id)
) COLLATE utf8mb4_unicode_ci");

// Отримання даних для форми
$categories = $db->query("SELECT id, name FROM
categories")->fetch_all(MYSQLI_ASSOC);
$warehouses = $db->query("SELECT id, name FROM warehouses WHERE active =
1")->fetch_all(MYSQLI_ASSOC);
$slots = $db->query("SELECT s.id, s.name, w.name as warehouse_name
                FROM slots s JOIN warehouses w ON s.warehouse_id = w.id
                WHERE s.active = 1")->fetch_all(MYSQLI_ASSOC);

// Обробка форми
$message = '';
$products = [];

if ($_SERVER['REQUEST_METHOD'] == 'POST') {
    if (isset($_POST['request_product'])) {
        // Обробка запиту товару
        $product_id = (int)$_POST['product_id'];
```

```
    $quantity = (int)$_POST['quantity'];
    $warehouse_id = (int)$_POST['warehouse_id'];
    $notes = $db->real_escape_string(trim($_POST['notes']));
    $requester_id = (int)$_SESSION['user_id'];

    $db->query("INSERT INTO product_requests
            (product_id, requested_quantity, requester_id, warehouse_id, notes)
            VALUES ($product_id, $quantity, $requester_id, $warehouse_id, '$notes')");

    $message = '<div class="alert alert-success">Запит успішно створено!</div>';
} elseif (isset($_POST['search_products'])) {
    // Обробка пошуку
    $where = [];
    $params = [];
    $types = '';

    if (!empty($_POST['name']) && isset($_POST['search_by_name'])) {
        $where[] = "p.name LIKE ?";
        $params[] = '%'.$_POST['name'].'%';
        $types .= 's';
    }

    if (!empty($_POST['article']) && isset($_POST['search_by_article'])) {
        $where[] = "p.article LIKE ?";
        $params[] = '%'.$_POST['article'].'%';
        $types .= 's';
    }

    if (!empty($_POST['barcode']) && isset($_POST['search_by_barcode'])) {
        $where[] = "p.barcode LIKE ?";
```

```
    $params[] = '%'.$_POST['barcode'].'%';
    $types .= 's';
}

if (!empty($_POST['category_id']) && isset($_POST['search_by_category'])) {
    $where[] = "p.category_id = ?";
    $params[] = (int)$_POST['category_id'];
    $types .= 'i';
}

if (!empty($_POST['warehouse_id']) && isset($_POST['search_by_warehouse'])) {
    $where[] = "i.warehouse_id = ?";
    $params[] = (int)$_POST['warehouse_id'];
    $types .= 'i';
}

if (!empty($_POST['slot_id']) && isset($_POST['search_by_slot'])) {
    $where[] = "i.slot_id = ?";
    $params[] = (int)$_POST['slot_id'];
    $types .= 'i';
}

$query = "SELECT p.id, p.name, p.article, p.barcode, p.unit,
                 pp.selling_price, c.name as category_name
          FROM products p
          LEFT JOIN product_prices pp ON p.id = pp.product_id AND pp.active = 1
          LEFT JOIN categories c ON p.category_id = c.id
          LEFT JOIN inventory i ON p.id = i.product_id";

if (!empty($where)) {
```

```
            $query .= " WHERE " . implode(" AND ", $where);
        }

        $query .= " GROUP BY p.id LIMIT 100";

        $stmt = $db->prepare($query);
        if (!empty($params)) {
            $stmt->bind_param($types, ...$params);
        }
        $stmt->execute();
        $products = $stmt->get_result()->fetch_all(MYSQLI_ASSOC);
    }
}
?>

<!DOCTYPE html>
<html lang="uk">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Запит товару | Склад</title>
    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/css/bootstrap.min.css"
rel="stylesheet">
    <link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/bootstrap-
icons@1.10.0/font/bootstrap-icons.css">
    <style>
        body {
            display: flex;
            min-height: 100vh;
            margin: 0;
```

```
}
.sidebar {
    width: 250px;
    background-color: #212529;
    color: white;
    padding: 20px 0;
    flex-shrink: 0;
}
.sidebar .nav-link {
    color: rgba(255, 255, 255, 0.75);
    padding: 10px 20px;
    margin: 5px 10px;
    border-radius: 5px;
    transition: all 0.3s;
}
.sidebar .nav-link:hover, .sidebar .nav-link.active {
    color: white;
    background-color: rgba(255, 255, 255, 0.1);
}
.sidebar .nav-link.active {
    background-color: #0d6efd;
}
.main-content {
    flex-grow: 1;
    padding: 20px;
    background-color: #f8f9fa;
}
.search-param {
    margin-bottom: 15px;
}
```

```
        .search-param .form-check-input {
            margin-right: 10px;
        }
        .product-card {
            cursor: pointer;
            transition: transform 0.2s;
        }
        .product-card:hover {
            transform: translateY(-3px);
        }
        #requestFormContainer {
            display: none;
        }
    </style>
</head>
<body>
    <div class="sidebar">
        <div class="text-center mb-4 px-3">
            <h4>Складской учет</h4>
            <hr class="text-white-50">
            <p>Добро пожаловать, <br><?= htmlspecialchars($_SESSION['full_name'] ?? 'Пользователь') ?></p>
        </div>
        <ul class="nav flex-column">
            <li class="nav-item"><a class="nav-link" href="dashboard.php"><i class="bi bi-house-door me-2"></i>Главная</a></li>
            <li class="nav-item"><a class="nav-link active" href="product_request.php"><i class="bi bi-box-seam me-2"></i>Запрос товара</a></li>
            <li class="nav-item"><a class="nav-link" href="sorting.php"><i class="bi bi-filter-square me-2"></i>Сортировка</a></li>
```

```
        <li class="nav-item"><a class="nav-link" href="operations.php"><i class="bi bi-arrow-left-right me-2"></i>Операции</a></li>
        <li class="nav-item"><a class="nav-link" href="inventory.php"><i class="bi bi-clipboard-data me-2"></i>Инвентаризация</a></li>
      </ul>
      <ul class="nav flex-column mt-auto mb-3">
        <li class="nav-item"><a class="nav-link text-danger" href="logout.php"><i class="bi bi-box-arrow-right me-2"></i>Выйти</a></li>
      </ul>
    </div>
    <!-- Основний контент -->
    <div class="main-content">
      <h2>Запит товару</h2>

      <?= $message ?>

      <div class="card mt-4">
        <div class="card-header d-flex justify-content-between align-items-center">
          <h5 class="mb-0">Пошук товарів</h5>
          <a href="products_add.php" class="btn btn-success btn-sm">
            <i class="bi bi-plus-lg"></i> Додати товар
          </a>
        </div>
        <div class="card-body">
          <form method="POST">
            <input type="hidden" name="search_products" value="1">

            <div class="row">
              <!-- Назва -->
              <div class="col-md-6 search-param">
```

```
                    <div class="form-check">
                        <input class="form-check-input" type="checkbox"
name="search_by_name" id="searchByName" checked>
                        <label class="form-check-label" for="searchByName">Назва
товару</label>
                        <input type="text" class="form-control mt-2" name="name"
placeholder="Введіть назву">
                    </div>
                </div>

                <!-- Артикул -->
                <div class="col-md-6 search-param">
                    <div class="form-check">
                        <input class="form-check-input" type="checkbox"
name="search_by_article" id="searchByArticle">
                        <label class="form-check-label"
for="searchByArticle">Артикул</label>
                        <input type="text" class="form-control mt-2" name="article"
placeholder="Введіть артикул">
                    </div>
                </div>

                <!-- Баркод -->
                <div class="col-md-6 search-param">
                    <div class="form-check">
                        <input class="form-check-input" type="checkbox"
name="search_by_barcode" id="searchByBarcode">
                        <label class="form-check-label" for="searchByBarcode">Штрих-
код</label>
```

```
                    <input type="text" class="form-control mt-2" name="barcode"
placeholder="Введіть штрих-код">
                </div>
            </div>

            <!-- Категорія -->
            <div class="col-md-6 search-param">
                <div class="form-check">
                    <input class="form-check-input" type="checkbox"
name="search_by_category" id="searchByCategory">
                    <label class="form-check-label"
for="searchByCategory">Категорія</label>
                    <select class="form-select mt-2" name="category_id">
                        <option value="">Всі категорії</option>
                        <?php foreach ($categories as $cat): ?>
                            <option value="<?= $cat['id'] ?>"><?=
htmlspecialchars($cat['name']) ?></option>
                        <?php endforeach; ?>
                    </select>
                </div>
            </div>

            <!-- Склад -->
            <div class="col-md-6 search-param">
                <div class="form-check">
                    <input class="form-check-input" type="checkbox"
name="search_by_warehouse" id="searchByWarehouse">
                    <label class="form-check-label"
for="searchByWarehouse">Склад</label>
                    <select class="form-select mt-2" name="warehouse_id">
```

```
                        <option value="">Всі склади</option>
                        <?php foreach ($warehouses as $wh): ?>
                            <option value="<?= $wh['id'] ?>"><?=
htmlspecialchars($wh['name']) ?></option>
                        <?php endforeach; ?>
                    </select>
                </div>
            </div>

            <!-- Слот -->
            <div class="col-md-6 search-param">
                <div class="form-check">
                    <input class="form-check-input" type="checkbox"
name="search_by_slot" id="searchBySlot">
                    <label class="form-check-label" for="searchBySlot">Слот
складу</label>
                    <select class="form-select mt-2" name="slot_id">
                        <option value="">Всі слоти</option>
                        <?php foreach ($slots as $slot): ?>
                            <option value="<?= $slot['id'] ?>">
                                <?= htmlspecialchars($slot['warehouse_name']) ?> - <?=
htmlspecialchars($slot['name']) ?>
                            </option>
                        <?php endforeach; ?>
                    </select>
                </div>
            </div>

            <!-- Ціна -->
            <div class="col-md-6 search-param">
```

```
                    <div class="form-check">
                        <input class="form-check-input" type="checkbox"
name="search_by_price" id="searchByPrice">
                        <label class="form-check-label" for="searchByPrice">Ціна</label>
                        <div class="row mt-2">
                            <div class="col">
                                <input type="number" class="form-control" name="price_min"
placeholder="Від" step="0.01">
                            </div>
                            <div class="col">
                                <input type="number" class="form-control" name="price_max"
placeholder="До" step="0.01">
                            </div>
                        </div>
                    </div>
                </div>

                <div class="text-center mt-4">
                    <button type="submit" class="btn btn-primary me-2">
                        <i class="bi bi-search"></i> Пошук
                    </button>
                    <button type="reset" class="btn btn-outline-secondary">
                        <i class="bi bi-x-circle"></i> Скинути
                    </button>
                </div>
            </form>

            <?php if (!empty($products)): ?>
                <div class="mt-4">
```

```
<h5>Знайдено товарів: <?= count($products) ?></h5>

<div class="table-responsive">
  <table class="table table-hover mt-3">
    <thead class="table-light">
      <tr>
        <th>Назва</th>
        <th>Артикул</th>
        <th>Штрих-код</th>
        <th>Категорія</th>
        <th>Ціна</th>
        <th>Од. вим.</th>
        <th>Дія</th>
      </tr>
    </thead>
    <tbody>
      <?php foreach ($products as $product): ?>
        <tr class="product-card" onclick="selectProduct(<?=
$product['id'] ?>, '<?= htmlspecialchars($product['name'], ENT_QUOTES) ?>')">
          <td><?= htmlspecialchars($product['name']) ?></td>
          <td><?= htmlspecialchars($product['article'] ?? '-') ?></td>
          <td><?= htmlspecialchars($product['barcode'] ?? '-') ?></td>
          <td><?= htmlspecialchars($product['category_name'] ?? '-
') ?></td>
          <td><?= $product['selling_price'] ?> грн</td>
          <td><?= htmlspecialchars($product['unit']) ?></td>
          <td>
            <button class="btn btn-sm btn-primary"
                onclick="event.stopPropagation(); selectProduct(<?=
$product['id'] ?>, '<?= htmlspecialchars($product['name'], ENT_QUOTES) ?>')">
```

```
                                    Обрати
                                </button>
                            </td>
                        </tr>
                    <?php endforeach; ?>
                </tbody>
            </table>
        </div>
    </div>
    <?php elseif ($_SERVER['REQUEST_METHOD'] == 'POST'): ?>
        <div class="alert alert-info mt-4">Товари за вказаними параметрами не знайдено.</div>
    <?php endif; ?>
</div>
</div>

<!-- Форма запиту товару -->
<div class="card mt-4" id="requestFormContainer">
    <div class="card-header">
        <h5 class="mb-0">Форма запиту товару</h5>
    </div>
    <div class="card-body">
        <form method="POST">
            <input type="hidden" name="request_product" value="1">
            <input type="hidden" name="product_id" id="selectedProductId">

            <div class="row mb-3">
                <div class="col-md-6">
                    <label class="form-label">Товар</label>
```

```
                    <input type="text" class="form-control" id="selectedProductName"
readonly>
                </div>
                <div class="col-md-6">
                    <label class="form-label">Кількість*</label>
                    <input type="number" class="form-control" name="quantity" min="1"
required>
                </div>
            </div>

            <div class="row mb-3">
                <div class="col-md-6">
                    <label class="form-label">Склад призначення*</label>
                    <select class="form-select" name="warehouse_id" required>
                        <option value="">-- Оберіть склад --</option>
                        <?php foreach ($warehouses as $wh): ?>
                            <option value="<?= $wh['id'] ?>"><?=
htmlspecialchars($wh['name']) ?></option>
                        <?php endforeach; ?>
                    </select>
                </div>
                <div class="col-md-6">
                    <label class="form-label">Примітки</label>
                    <textarea class="form-control" name="notes" rows="2"></textarea>
                </div>
            </div>

            <div class="text-end">
                <button type="submit" class="btn btn-primary">Надіслати
запит</button>
```

```
                    <a href="product_request.php" class="btn btn-secondary>
                <i class="bi bi-arrow-left"></i>Скасувати
            </a>



                </div>
            </form>
        </div>
    </div>
</div>

<script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/js/bootstrap.bundle.min.js"></scrip
t>
<script>
    function selectProduct(id, name) {
        document.getElementById('selectedProductId').value = id;
        document.getElementById('selectedProductName').value = name;
        document.getElementById('requestFormContainer').style.display = 'block';

        // Прокрутка до форми
        document.getElementById('requestFormContainer').scrollIntoView({
            behavior: 'smooth'
        });
    }

    function cancelRequest() {
        document.getElementById('requestFormContainer').style.display = 'none';
    }
```

```
        // Автоматичний пошук при зміні параметрів
        document.querySelectorAll('input[type="text"], input[type="number"],
select').forEach(input => {
            input.addEventListener('change', function() {
                if (this.value.trim() !== '') {
                    this.closest('.search-param').querySelector('.form-check-input').checked =
true;
                }
            });
        });
    </script>
</body>
</html>
```

Модуль products_add.php

```
<<?php
session_start();
require_once 'db_connect.php';

if (!isset($_SESSION['user_id'])) {
    header("Location: index.php");
    exit();
}

$message = '';
$categories = $db->query("SELECT id, name FROM
categories")->fetch_all(MYSQLI_ASSOC);

if ($_SERVER['REQUEST_METHOD'] == 'POST') {
    $name = trim($_POST['name']);
    $article = trim($_POST['article']); // Додано поле артикулу
```

```
    $barcode = trim($_POST['barcode']);
    $category_id = $_POST['category_id'] ?: NULL;
    $unit = trim($_POST['unit']);
    $purchase_price = (float)$_POST['purchase_price'];
    $markup_percent = (float)$_POST['markup_percent'];
    $selling_price = $purchase_price * (1 + $markup_percent / 100);

    try {
        $db->begin_transaction();

        // Додавання товару з артикулом
        $stmt = $db->prepare("INSERT INTO products (name, article, barcode, category_id, unit) VALUES (?, ?, ?, ?, ?)");
        $stmt->bind_param("sssis", $name, $article, $barcode, $category_id, $unit);
        $stmt->execute();
        $product_id = $stmt->insert_id;

        // Додавання цін
        $stmt = $db->prepare("INSERT INTO product_prices (product_id, purchase_price, selling_price, markup_percent) VALUES (?, ?, ?, ?)");
        $stmt->bind_param("iddd", $product_id, $purchase_price, $selling_price, $markup_percent);
        $stmt->execute();

        $db->commit();
        $message = '<div class="alert alert-success">Товар успішно додано!</div>';
    } catch (Exception $e) {
        $db->rollback();
        $message = '<div class="alert alert-danger">Помилка: ' . $e->getMessage() . '</div>';
```

```
    }
}
?>

<!DOCTYPE html>
<html lang="uk">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Додати товар | Склад</title>
    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/css/bootstrap.min.css"
rel="stylesheet">
    <style>
        body {
            background-color: #f8f9fa;
        }
        .form-container {
            max-width: 800px;
            margin: 30px auto;
            padding: 20px;
            background-color: #fff;
            border-radius: 8px;
            box-shadow: 0 0 15px rgba(0,0,0,0.1);
        }
        .price-preview {
            background-color: #f8f9fa;
            padding: 10px;
            border-radius: 5px;
            margin-top: 10px;
        }
```

```
    </style>
</head>
<body>

    <div class="main-content">
        <div class="container">
            <div class="form-container">
                <h2 class="mb-4">Додати новий товар</h2>

                <?php echo $message; ?>

                <form method="POST">
                    <div class="row mb-3">
                        <div class="col-md-6">
                            <label class="form-label">Назва товару*</label>
                            <input type="text" name="name" class="form-control" required>
                        </div>
                        <div class="col-md-6">
                            <label class="form-label">Артикул*</label>
                            <input type="text" name="article" class="form-control" required>
                        </div>
                    </div>

                    <div class="row mb-3">
                        <div class="col-md-6">
                            <label class="form-label">Баркод</label>
                            <input type="text" name="barcode" class="form-control">
                        </div>
                        <div class="col-md-6">
                            <label class="form-label">Категорія</label>
```

```
                    <select name="category_id" class="form-select">
                        <option value="">-- Оберіть категорію --</option>
                        <?php foreach ($categories as $category): ?>
                            <option value="<?= $category['id'] ?>"><?=
htmlspecialchars($category['name']) ?></option>
                        <?php endforeach; ?>
                    </select>
                </div>
            </div>

            <div class="row mb-3">
                <div class="col-md-6">
                    <label class="form-label">Одиниця виміру*</label>
                    <input type="text" name="unit" class="form-control" value="шт."
required>
                </div>
            </div>

            <h4 class="mt-4 mb-3">Ціноутворення</h4>
            <div class="row mb-3">
                <div class="col-md-4">
                    <label class="form-label">Вхідна ціна*</label>
                    <div class="input-group">
                        <input type="number" name="purchase_price" id="purchase_price"
                               class="form-control" step="0.01" min="0" required
                               oninput="calculateSellingPrice()">
                        <span class="input-group-text">₴</span>
                    </div>
                </div>
                <div class="col-md-4">
```

```
                    <label class="form-label">Накрутка (%)*</label>
                    <div class="input-group">
                        <input type="number" name="markup_percent" id="markup_percent"
                               class="form-control" step="0.1" min="0" value="20" required
                               oninput="calculateSellingPrice()">
                        <span class="input-group-text">%</span>
                    </div>
                </div>
                <div class="col-md-4">
                    <label class="form-label">Роздрібна ціна</label>
                    <div class="price-preview">
                        <h5 id="selling_price_display">0.00 ₴</h5>
                        <input type="hidden" name="selling_price" id="selling_price">
                    </div>
                </div>
            </div>

            <div class="d-flex justify-content-between mt-4">
                <a href="products.php" class="btn btn-secondary">
                    <i class="bi bi-arrow-left"></i> Скасувати
                </a>
                <button type="submit" class="btn btn-primary">
                    <i class="bi bi-save"></i> Зберегти товар
                </button>
            </div>
        </form>
    </div>
</div>
</div>
```

```
    <script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/js/bootstrap.bundle.min.js"></scrip
t>
    <script>
        // Функція для розрахунку роздрібної ціни
        function calculateSellingPrice() {
            const purchasePrice =
parseFloat(document.getElementById('purchase_price').value) || 0;
            const markupPercent =
parseFloat(document.getElementById('markup_percent').value) || 0;
            const sellingPrice = purchasePrice * (1 + markupPercent / 100);

            document.getElementById('selling_price').value = sellingPrice.toFixed(2);
            document.getElementById('selling_price_display').textContent =
sellingPrice.toFixed(2) + ' ₴';
        }

        // Розрахувати ціну при завантаженні сторінки
        document.addEventListener('DOMContentLoaded', calculateSellingPrice);
    </script>
</body>
</html>
```

Модуль product_search.php

```
<?php
require_once 'db_connect.php';

header('Content-Type: application/json');

if (!isset($_GET['term']) || strlen($_GET['term']) < 2) {
    echo json_encode([]);
```

```
    exit();
}

$term = '%' . $_GET['term'] . '%';
$searchBy = $_GET['search_by'] ?? 'both'; // 'article', 'name' або 'both'

$query = "SELECT p.id, p.name, p.article, p.barcode, p.unit,
                 pp.purchase_price, pp.selling_price
          FROM products p
          JOIN product_prices pp ON p.id = pp.product_id AND pp.active = 1
          WHERE ";

if ($searchBy === 'article') {
    $query .= "p.article LIKE ?";
    $params = [$term];
} elseif ($searchBy === 'name') {
    $query .= "p.name LIKE ?";
    $params = [$term];
} else {
    $query .= "p.name LIKE ? OR p.article LIKE ? OR p.barcode LIKE ?";
    $params = [$term, $term, $term];
}

$query .= " LIMIT 10";

$stmt = $db->prepare($query);

if ($searchBy === 'both') {
    $stmt->bind_param(str_repeat('s', count($params)), ...$params);
} else {
```

```
    $stmt->bind_param('s', $params[0]);
}

$stmt->execute();
$result = $stmt->get_result();

$products = [];
while ($row = $result->fetch_assoc()) {
    $products[] = $row;
}

echo json_encode($products);
?>
```

Модуль product_search1.php

```
<?php
require_once 'db_connect.php';

header('Content-Type: application/json');

if (empty($_GET['term'])) {
    echo json_encode([]);
    exit();
}

$searchTerm = '%' . $_GET['term'] . '%';

$query = "SELECT p.id, p.name, p.article, p.barcode, p.unit,
                 pp.purchase_price,
                 pp.selling_price
          FROM products p
```

```
        LEFT JOIN product_prices pp ON p.id = pp.product_id AND pp.active = 1
        WHERE p.name LIKE ? OR p.article LIKE ?
        LIMIT 15";

$stmt = $db->prepare($query);
$stmt->bind_param("ss", $searchTerm, $searchTerm);
$stmt->execute();
$result = $stmt->get_result();
$products = $result->fetch_all(MYSQLI_ASSOC);

echo json_encode($products);
?>
```

Модуль transfer_to_slot.php

```
<?php
session_start();
require 'db_connect.php';

if ($_SERVER['REQUEST_METHOD'] == 'POST') {
    try {
        $db->begin_transaction();

        // 1. Создаем документ перемещения
        $stmt = $db->prepare("
            INSERT INTO documents
            (type, number, counterparty, created_by)
            VALUES ('transfer', ?, 'Internal Transfer', ?)
        ");
        $stmt->bind_param("si", $_POST['transfer_number'], $_SESSION['user_id']);
        $stmt->execute();
        $document_id = $db->insert_id;
```

```
        // 2. Обрабатываем каждое перемещение
        foreach ($_POST['transfers'] as $transfer) {
            $product_id = (int)$transfer['product_id'];
            $slot_id = (int)$transfer['slot_id'];
            $quantity = (int)$transfer['quantity'];

            // Проверяем доступное количество
            $available = $db->query("
                SELECT quantity FROM inventory WHERE product_id = $product_id
            ")->fetch_row()[0];

            if ($available < $quantity) {
                throw new Exception("Недостаточно товара ID $product_id (доступно: $available)");
            }

            // Добавляем в документ
            $stmt = $db->prepare("
                INSERT INTO document_items
                (document_id, product_id, quantity, price, slot_id)
                VALUES (?, ?, ?, 0, ?)
            ");
            $stmt->bind_param("iiii", $document_id, $product_id, $quantity, $slot_id);
            $stmt->execute();

            // Обновляем общие остатки
            $db->query("
                UPDATE inventory SET quantity = quantity - $quantity
                WHERE product_id = $product_id
```

```
            ");

            // Обновляем остатки в слоте
            $db->query("
                INSERT INTO slot_inventory
                (product_id, slot_id, quantity)
                VALUES ($product_id, $slot_id, $quantity)
                ON DUPLICATE KEY UPDATE quantity = quantity + VALUES(quantity)
            ");
        }

        $db->commit();
        echo json_encode(['success' => true, 'document_id' => $document_id]);
    } catch (Exception $e) {
        $db->rollback();
        echo json_encode(['success' => false, 'message' => $e->getMessage()]);
    }
    exit;
}

// Получаем товары с общими остатками
$products = $db->query("
    SELECT p.id, p.name, p.article, i.quantity
    FROM products p
    JOIN inventory i ON p.id = i.product_id
    WHERE i.quantity > 0
")->fetch_all(MYSQLI_ASSOC);

// Получаем доступные слоты
$slots = $db->query("
```

```
    SELECT s.id, s.name, w.name as warehouse_name
    FROM slots s
    JOIN warehouses w ON s.warehouse_id = w.id
    WHERE s.active = 1
")->fetch_all(MYSQLI_ASSOC);
?>

<!-- HTML форма для перемещения -->
<div class="container">
    <h2>Распределение товаров по слотам</h2>
    <form id="transferForm">
        <div class="mb-3">
            <label>Номер перемещения</label>
            <input type="text" name="transfer_number" class="form-control" required>
        </div>

        <div id="transferItems">
            <!-- Динамически добавляемые строки -->
        </div>

        <button type="button" class="btn btn-success" onclick="addTransferRow()">
            Добавить товар
        </button>

        <button type="submit" class="btn btn-primary">
            Сохранить перемещение
        </button>
    </form>
</div>
```

```
<script>
// JavaScript для управления формой
function addTransferRow(productId = '', quantity = '') {
    const html = `
    <div class="transfer-row mb-3 border p-3">
        <div class="row">
            <div class="col-md-5">
                <select name="transfers[][product_id]" class="form-select" required>
                    <option value="">Выберите товар</option>
                    <?php foreach ($products as $p): ?>
                        <option value="<?= $p['id'] ?>"
                            ${productId == <?= $p['id'] ?> ? 'selected' : ''}>
                            <?= htmlspecialchars($p['name']) ?> (<?= $p['quantity'] ?> шт.)
                        </option>
                    <?php endforeach; ?>
                </select>
            </div>
            <div class="col-md-3">
                <select name="transfers[][slot_id]" class="form-select" required>
                    <option value="">Выберите слот</option>
                    <?php foreach ($slots as $s): ?>
                        <option value="<?= $s['id'] ?>">
                            <?= htmlspecialchars($s['warehouse_name']) ?> - <?=
htmlspecialchars($s['name']) ?>
                        </option>
                    <?php endforeach; ?>
                </select>
            </div>
            <div class="col-md-3">
                <input type="number" name="transfers[][quantity]"
```

```
                    class="form-control" min="1" value="${quantity}" required>
            </div>
            <div class="col-md-1">
                <button type="button" class="btn btn-danger" onclick="this.closest('.transfer-row').remove()">
                    ×
                </button>
            </div>
        </div>
    </div>`;
    document.getElementById('transferItems').insertAdjacentHTML('beforeend', html);
}

document.getElementById('transferForm').addEventListener('submit', async function(e) {
    e.preventDefault();
    const formData = new FormData(this);

    const response = await fetch('transfer_to_slot.php', {
        method: 'POST',
        body: formData
    });

    const result = await response.json();
    if (result.success) {
        alert(`Перемещение #${result.document_id} успешно создано!`);
        location.reload();
    } else {
        alert('Ошибка: ' + result.message);
    }
});
```

```
// Добавляем первую строку при загрузке
addTransferRow();
</script>
Модуль update_inventory.php
<?php
require_once 'db_connect.php';

header('Content-Type: application/json');

if ($_SERVER['REQUEST_METHOD'] == 'POST' &&
isset($_POST['update_inventory'])) {
    $slot_id = (int)$_POST['slot_id'];
    $product_id = (int)$_POST['product_id'];
    $quantity = (int)$_POST['quantity'];

    try {
        if ($quantity > 0) {
            $db->query("INSERT INTO inventory (product_id, slot_id, quantity)
                    VALUES ($product_id, $slot_id, $quantity)
                    ON DUPLICATE KEY UPDATE quantity = $quantity");
        } else {
            $db->query("DELETE FROM inventory WHERE product_id = $product_id AND
slot_id = $slot_id");
        }

        echo json_encode(['success' => true]);
    } catch (Exception $e) {
        echo json_encode(['success' => false, 'message' => $e->getMessage()]);
    }
```

```
} else {
    echo json_encode(['success' => false, 'message' => 'Невірний запит']);
}
```

Додаток Б

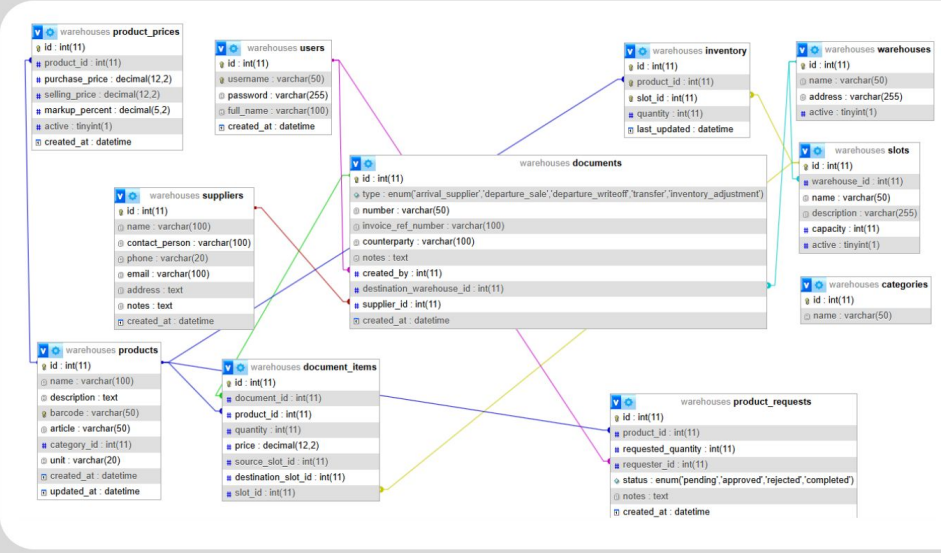
Архітектура БД

Архітектура бази даних

Назва БД: warehouses.
Тип: Реляційна, розроблена в СУБД MariaDB.
Ключові сутності:

- products (довідник товарів), product_prices (ціни).
- warehouses (склади), slots (місця зберігання).
- inventory (таблиця фактичних залишків).
- documents та document_items (документообіг).

Принцип: Структура забезпечує цілісність даних та фіксацію всіх операцій.



Додаток В

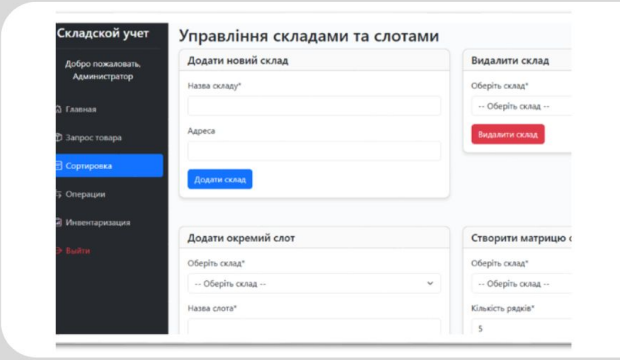
Модуль «Управління складами та слотами»

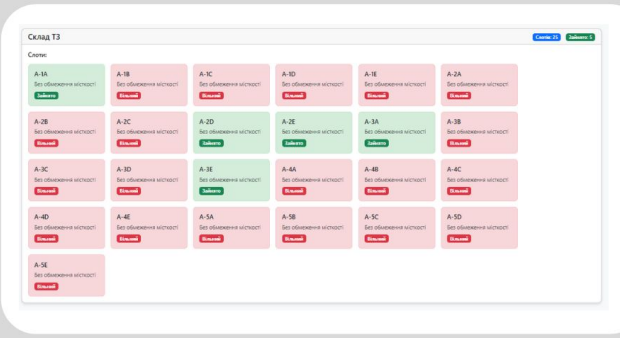
Опис ключових модулів системи

Модуль "Управління складами та слотами"

Ключові функції:

* Створення та "видалення" (деактивація) складських приміщень.
* Додавання окремих слотів із зазначенням місткості.
* Автоматична генерація матриці слотів для швидкого наповнення.
* Візуальне представлення заповненості складів та стану кожного слота (вільний/зайнятий).





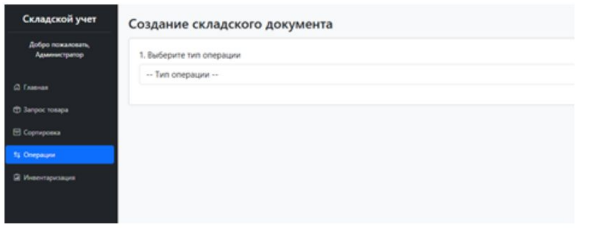
Додаток Г

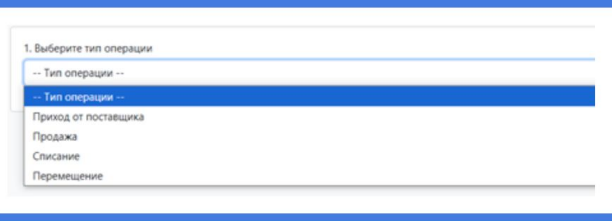
Модуль «Управління операціями» Частина 1

Модуль "Управління Операціями" Частина 1

Ключові особливості:

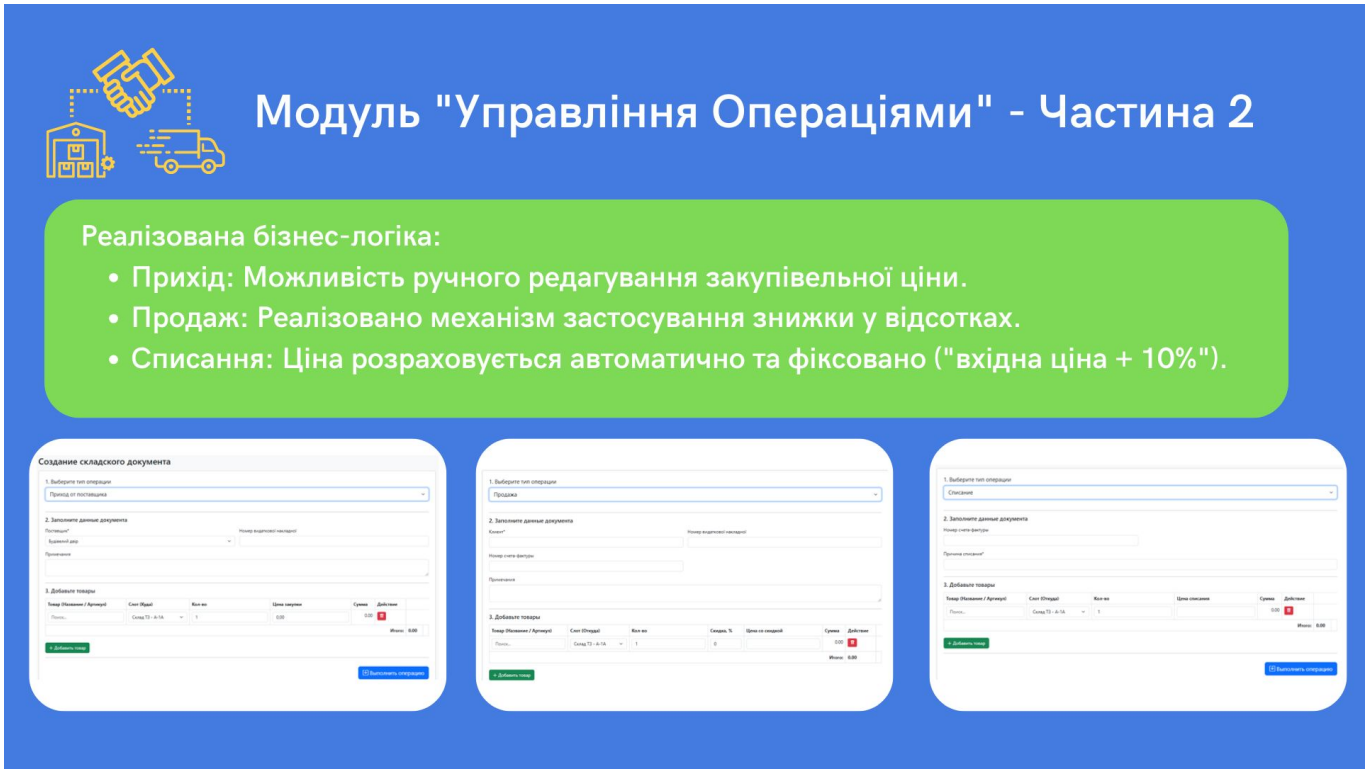
- Центральний модуль для реєстрації всіх типів руху ТМЦ.
- Реалізовано універсальний інтерфейс замість окремих сторінок.
- Форма динамічно адаптується під обраний тип операції за допомогою JavaScript.
- Доступні операції: "Приход от поставщика", "Продажа", "Списание", "Перемещение".





Додаток Д

Модуль «Управління операціями» Частина 2



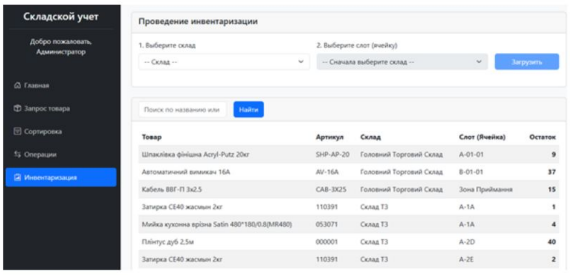
Додаток Е

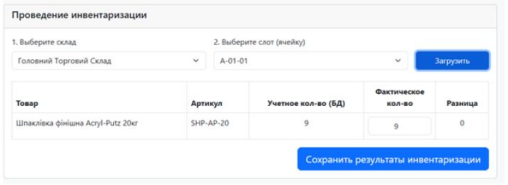
Модуль «Інвентаризація та залишки»

Модуль "Інвентаризація та залишки"

Ключові функції:

- Перегляд актуальних залишків усіх товарів у розрізі складів та слотів.
- Інтерактивна форма для проведення інвентаризації по конкретному слоту.
- Автоматичний розрахунок розбіжностей між обліковою та фактичною кількістю.
- Створення коригуючого документа (inventory_adjustment) для фіксації результатів інвентаризації.





Додаток Ж

Модуль «Тестування системи»

Тестування системи

Метод: "Чорна скринька".

Основні тестові сценарії:

- Авторизація користувача.
- Створення нового товару.
- Проведення операції "Прихід товару" та перевірка оновлення залишків.
- Проведення операції "Списання товару".
- Проведення інвентаризації та перевірка корекції залишків.

Результат: "Фактичний результат: Відповідає очікуваному".