Міністерство освіти і науки України
Тернопільський національний технічний університет імені Івана Пулюя

Факультет комп'ютерно-інформаційних систем і програмної інженерії
(повна назва факультету)
Кафедра комп'ютерних наук
(повна назва кафедри)

КВАЛІФІКАЦІЙНА РОБОТА

на здобуття освітнього ступеня

бакалавр
(назва освітнього ступеня)

на тему: Створення застосунку для управління товарами на складі

Виконав: студент IV курсу, групи СН-42
спеціальності 122 Комп’ютерні науки
(шифр і назва спеціальності)

(підпис) Чух Р. В. (прізвище та ініціали)

Керівник (підпис) Готович В. А. (прізвище та ініціали)

Нормоконтроль (підпис) Шимчук Г.В. (прізвище та ініціали)

Завідувач кафедри (підпис) Боднарчук І.О. (прізвище та ініціали)

Рецензент (підпис) Луцик Н.С. (прізвище та ініціали)

Тернопіль
2026

Міністерство освіти і науки України
Тернопільський національний технічний університет імені Івана Пулюя

Факультет комп'ютерно-інформаційних систем і програмної інженерії
(повна назва факультету)
Кафедра комп'ютерних наук
(повна назва кафедри)

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____ Боднарчук І.О.
(підпис) (прізвище та ініціали)
«_25_» _червня_ 2026 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

на здобуття освітнього ступеня Бакалавр
(назва освітнього ступеня)
за спеціальністю 122 Комп'ютерні науки
(шифр і назва спеціальності)
Студенту Чух Роман Васильович
(прізвище, ім'я, по батькові)
1. Тема роботи Створення застосунку для управління товарами на складі

Керівник роботи Готович Володимир Анатолійович, кандидат технічних наук, доцент
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)
Затверджені наказом ректора від «*14*» *травня* 2026 року № *4/9-239*
2. Термін подання студентом завершеної роботи 22 червня 2026 р.
3. Вихідні дані до роботи Офіційна документація системи керування вмістом WordPress та плагіна електронної комерції WooCommerce; технічна документація мов програмування PHP, JavaScript; документація СУБД MySQL; специфікації HTML5 і CSS; стандарти WordPress Coding Standards; нормативні документи з оформлення робіт (ДСТУ 8302:2015, ДСТУ 3008:2015).
4. Зміст роботи (перелік питань, які потрібно розробити)
Вступ. 1. Аналіз предметної області та постановка завдання.
2. Проєктування застосунку.
3. Розробка застосунку та бази даних.
4. Безпека життєдіяльності, основи охорони праці.
Висновки. Перелік використаних джерел. Додатки.

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень, слайдів)
Повний перелік слайдів у презентації, включаючи перший та останній слайди (з номерами, через крапку).

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Безпека життєдіяльності, основи охорони праці			

7. Дата видачі завдання _____

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів роботи	Термін виконання етапів роботи	Примітка
1.	Ознайомлення з завданням до кваліфікаційної роботи	26.01.2026	*Виконано*
2.	Підбір та опрацювання літературних джерел по темі кваліфікаційної роботи	27.01.2026-16.01.2026	*Виконано*
3.	Виконання дослідження щодо методів та засобів управління товарами на складі інтернет-магазину на базі WordPress і WooCommerce	17.01.2026-10.05.2026	*Виконано*
	Розроблення модулів обліку руху товарів, оформлення замовлень з доставкою та звітності	10.05-2026-11.05.2026	*Виконано*
4.	Оформлення розділу «Аналіз предметної області та постановка завдання»	11.05.2026-17.05.2026	*Виконано*
5.	Оформлення розділу «Проєктування застосунку для управління товарами на складі»	18.05.2026-24.05.2026	*Виконано*
6.	Оформлення розділу «Практична реалізація застосунку для управління товарами на складі»	25.05.2026-31.05.2026	*Виконано*
7.	Виконання завдання до підрозділу «Безпека життєдіяльності»	01.06.2026-08.06.2026	*Виконано*
8.	Виконання завдання до підрозділу «Основи охорони	01.06.2026-08.06.2026	*Виконано*
	праці»		
9.	Оформлення кваліфікаційної роботи	09.06.2026-11.06.2026	*Виконано*
10.	Нормоконтроль	12.06.2026-15.06.2026	*Виконано*
11.	Перевірка на плагіат	16.06.2026	*Виконано*
12.	Попередній захист кваліфікаційної роботи	18.06.2026	*Виконано*
13.	Захист кваліфікаційної роботи	22.06.2026	

Студент _____ *Чух Р. В.*
(підпис) (прізвище та ініціали)

Керівник роботи _____ *Готович В. А.*

АНОТАЦІЯ

Створення застосунку для управління товарами на складі // Кваліфікаційна робота освітнього рівня «Бакалавр» // Чух Роман Васильович // Тернопільський національний технічний університет імені Івана Пулюя, факультет комп'ютерно-інформаційних систем і програмної інженерії, кафедра комп'ютерних наук, група СН-42 // Тернопіль, 2026 // С. 64, рис. – 14, табл. – 10, кресл. – , додат. – 4, бібліогр. – 51.

Ключові слова: веб-застосунок, управління товарами, WordPress, WooCommerce, PHP, MySQL, електронна комерція, інтернет-магазин.

Кваліфікаційна робота присвячена розробленню веб-застосунку для управління товарами на складі інтернет-магазину на базі системи керування вмістом WordPress та платформи електронної комерції WooCommerce. Актуальність роботи зумовлена потребою підприємств малого та середнього бізнесу в доступному, інтегрованому інструменті для автоматизації складського обліку безпосередньо в середовищі інтернет-магазину.

У першому розділі проаналізовано предметну область складського обліку інтернет-торгівлі, формалізовано бізнес-процеси руху товарів, здійснено огляд і порівняльний аналіз наявних програмних рішень та платформ електронної комерції, обґрунтовано вибір технологічного стеку, сформульовано функціональні й нефункціональні вимоги до застосунку.

У другому розділі описано проєктування застосунку: розроблено трирівневу архітектуру, структуру бази даних із власними таблицями журналу руху товарів і поставок, побудовано ER-діаграму, діаграми прецедентів та послідовності, спроєктовано модулі звітності, аналітики й підсистему розмежування доступу.

У третьому розділі описано практичну реалізацію та тестування, наведено реалізацію модульного плагіна WordPress з інтеграцією WooCommerce через

систему хуків, модулів обліку залишків, руху товарів, поставок, обробки замовлень і звітності, а також результати функціонального тестування застосунку.

У четвертому розділі розглянуто питання безпеки життєдіяльності та основ охорони праці.

Об’єкт дослідження: процеси управління товарними запасами на складі інтернет-магазину.

Предмет дослідження: методи та програмні засоби автоматизації обліку, контролю й аналізу руху товарів на базі платформи WordPress/WooCommerce.

ABSTRACT

Development of an application for warehouse product management // Qualification work of the educational level «Bachelor» // Chukh Roman Vasylovych // Ternopil Ivan Puluj National Technical University, Faculty of Computer Information Systems and Software Engineering, Department of Computer Sciences, group SN-42 // Ternopil, 2026 // P. 64, fig. – 14, tabl. – 10, draw. – , annexes – 4, references – 51.

Keywords: web application, product management, WordPress, WooCommerce, PHP, MySQL, e-commerce, online store.

The qualification thesis is devoted to the development of a web application for warehouse product management of an online store based on the WordPress content management system and the WooCommerce e-commerce platform. The relevance of the work is determined by the need of small and medium-sized businesses for an affordable, integrated tool for automating warehouse accounting directly within the online store environment.

The first section analyses the subject area of warehouse accounting in online commerce: the business processes of goods movement are formalised, a review and comparative analysis of existing software solutions and e-commerce platforms is carried out, the choice of the technology stack is substantiated, and functional and non-functional requirements for the application are formulated.

The second section describes the application design: a three-tier architecture and a database structure with dedicated tables for the goods movement log and supplies are developed, an ER diagram, use-case and sequence diagrams are built, and the reporting, analytics and access-control subsystems are designed.

The third section covers implementation and testing: the implementation of a modular WordPress plugin with WooCommerce integration via the hooks system,

modules for stock accounting, goods movement, supplies, order processing and reporting are presented, as well as the results of functional testing of the application.

The fourth section addresses issues of life safety and the fundamentals of labour protection.

Object of research: the processes of inventory management in the warehouse of an online store.

Subject of research: methods and software tools for automating the accounting, control and analysis of goods movement based on the WordPress/WooCommerce platform.

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

ABC-аналіз – метод класифікації товарних позицій за рівнем їх значущості для продажів;

AJAX – технологія асинхронного обміну даними між браузером і сервером (Asynchronous JavaScript and XML);

API – програмний інтерфейс застосунку (Application Programming Interface);

CMS – система керування вмістом вебсайту (Content Management System);

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

CSS – каскадні таблиці стилів для оформлення вебсторінок (Cascading Style Sheets);

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

ER-діаграма – діаграма «сутність–зв’язок» для моделювання структури бази даних (Entity–Relationship);

FIFO – метод оцінювання вибуття запасів «перший прийшов – перший пішов» (First In, First Out);

HTML – мова гіпертекстової розмітки вебсторінок (HyperText Markup Language);

HTTP – протокол передавання гіпертексту (HyperText Transfer Protocol);

JavaScript (JS) – мова програмування клієнтської частини застосунку;

MySQL – реляційна система керування базами даних, що використовується для зберігання даних;

PHP – мова програмування серверної частини застосунку (Hypertext Preprocessor);

SQL – мова структурованих запитів до бази даних (Structured Query Language);

UI – інтерфейс користувача (User Interface);

UML – уніфікована мова моделювання (Unified Modeling Language);

URL – уніфікований локатор ресурсу в мережі (Uniform Resource Locator);

WooCommerce – плагін електронної комерції для WordPress, що забезпечує функції інтернет-магазину;

WordPress – система керування вмістом з відкритим вихідним кодом, на базі якої побудовано застосунок;СУБД – система керування базами даних;

ПЗ – програмне забезпечення;

БД – база даних;

Хук (hook) – механізм WordPress для розширення функціональності без зміни ядра системи;

Плагін – програмний модуль, що розширює можливості системи керування вмістом;

ЗМІСТ

ВСТУП

Актуальність теми. Стрімкий розвиток електронної комерції протягом останнього десятиліття суттєво змінив підходи до організації торговельної діяльності. Дедалі більша частка роздрібного товарообігу переходить у цифрове середовище, а інтернет-магазини стають основним або єдиним каналом збуту для значної кількості підприємств малого та середнього бізнесу. За таких умов ефективне управління товарними запасами перетворюється на один з ключових чинників конкурентоспроможності: помилки в обліку, несвоєчасне поповнення складу або відсутність достовірної інформації про залишки безпосередньо впливають на рівень продажів, лояльність клієнтів і прибутковість бізнесу.

Управління товарами на складі інтернет-магазину охоплює широкий спектр процесів – від приймання товару та його розміщення до резервування під замовлення, списання, повернення та доставки кінцевому споживачу. Ручне ведення такого обліку, наприклад за допомогою електронних таблиць, є трудомістким, схильним до помилок і не масштабується зі зростанням асортименту та кількості замовлень. Тому актуальним є створення спеціалізованого програмного забезпечення, яке автоматизує облік руху товарів та інтегрується безпосередньо з вітриною інтернет-магазину.

Серед наявних на ринку рішень переважають або дорогі комплексні системи класу ERP, складні у впровадженні для невеликого бізнесу, або вузькоспеціалізовані сервіси, що потребують додаткової інтеграції з торговельним майданчиком. Водночас значна частина інтернет-магазинів в Україні та світі побудована на системі керування вмістом WordPress з плагіном електронної комерції WooCommerce, які займають провідні позиції за поширеністю. Це робить доцільним розроблення розширення (плагіна) для цієї платформи, що поєднає складський облік, контроль і аналітику в єдиному середовищі без потреби у сторонніх системах.

Актуальність дослідження зумовлена потребою підприємств електронної комерції в доступному, інтегрованому та зручному інструменті для управління

товарними запасами. Розроблення такого застосунку на базі поширеної платформи WordPress/WooCommerce дозволяє знизити вартість володіння рішенням, спростити його впровадження та забезпечити єдину точку керування каталогом, складом, замовленнями та звітністю.

Мета і завдання дослідження. Метою кваліфікаційної роботи є розробка веб-застосунку для управління товарами на складі інтернет-магазину, що поєднує облік, контроль та аналіз руху товарів від моменту їх надходження на склад до продажу й доставки клієнту.

Для досягнення поставленої мети необхідно розв’язати такі **завдання**:

1) проаналізувати предметну область складського обліку інтернет-торгівлі та бізнес-процеси руху товарів;

2) дослідити наявні програмні рішення та визначити їх переваги й недоліки;

3) обґрунтувати вибір технологічного стеку та платформи розроблення;

4) сформулювати функціональні та нефункціональні вимоги до застосунку;

5) спроєктувати архітектуру застосунку, структуру бази даних та інтерфейс користувача;

6) реалізувати застосунок у вигляді плагіна WordPress з інтеграцією WooCommerce;

7) розробити модулі обліку руху товарів, оформлення замовлень з доставкою та звітності;

8) провести тестування розробленого програмного продукту.

Об’єкт і предмет дослідження: процеси управління товарними запасами на складі інтернет-магазину.

Предмет дослідження: методи та програмні засоби автоматизації обліку, контролю й аналізу руху товарів на базі платформи WordPress/WooCommerce.

Методи дослідження. У роботі застосовано методи системного аналізу для дослідження предметної області, методи порівняльного аналізу для оцінювання наявних рішень і технологій, методи об’єктно-орієнтованого та

структурного проєктування для побудови архітектури застосунку, методи моделювання за допомогою мови UML, а також методи функціонального тестування для перевірки коректності роботи програмного продукту.

Практичне значення одержаних результатів. Практичне значення роботи полягає в тому, що розроблений застосунок може бути впроваджений на діючих інтернет-магазинах, побудованих на WordPress/WooCommerce, для автоматизації складського обліку. Це дозволяє скоротити час обробки замовлень, зменшити кількість помилок обліку, забезпечити керівництво достовірною аналітичною інформацією та підвищити загальну ефективність управління товарними запасами.

Структура та обсяг роботи. Кваліфікаційна робота складається зі вступу, чотирьох, висновків, переліку використаних джерел та додатків. У першому розділі проаналізовано предметну область і здійснено постановку завдання. У другому розділі представлено проєктну частину – архітектуру, базу даних та інтерфейс застосунку. У третьому розділі описано практичну реалізацію та тестування. Четвертий розділ містить інформацію основи безпеки життєдіяльності та охорони праці.

РОЗДІЛ 1. АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ ТА ПОСТАНОВКА ЗАВДАННЯ

1.1 Облік та оцінювання запасів продукції на складі

1.1.1 Загальна характеристика предметної області задачі кваліфікаційної роботи

Складський облік є складовою системи управління запасами підприємства й охоплює сукупність процесів реєстрації, контролю та аналізу матеріальних цінностей, що зберігаються на складі. Для підприємства, яке здійснює роздрібну торгівлю через інтернет, склад є проміжною ланкою між постачальниками й кінцевими споживачами, а точність обліку безпосередньо визначає здатність магазину виконувати замовлення вчасно й у повному обсязі [3].

Товар на складі інтернет-магазину перебуває у постійному русі. Життєвий цикл товарної позиції умовно можна поділити на кілька етапів: надходження (приймання від постачальника та оприбуткування), зберігання (розміщення на складі з фіксацією залишків), резервування (виділення під оформлене, але ще не відвантажене замовлення), відвантаження (списання зі складу та передавання у доставку) і, за потреби, повернення (надходження товару назад на склад). На кожному з цих етапів змінюється кількісний та вартісний стан запасів, що має бути коректно відображено в обліковій системі [3].

Ключовими обліковими показниками, з якими оперує система складського обліку, є: фактичний залишок товару, зарезервована кількість, доступна до продажу кількість (різниця між залишком і резервом), рівень мінімального запасу (точка повторного замовлення), собівартість одиниці та сумарна вартість запасів. Своєчасне й достовірне відображення цих показників дозволяє уникнути двох типових проблем електронної торгівлі – дефіциту (коли клієнт оформлює замовлення на відсутній товар) та надлишку (коли заморожуються обігові кошти в неліквідних запасах) [2, 3].

Окремою важливою функцією складського обліку є фіксація історії руху товарів – журналу операцій, у якому реєструється кожна зміна залишку із зазначенням типу операції, кількості, дати, відповідального користувача та пов’язаного документа (замовлення, накладної). Така історія є основою для аудиту, інвентаризації та побудови аналітичної звітності [2].

1.1.2 Методи оцінювання вартості товарних запасів

Важливим аспектом складського обліку є оцінювання вартості запасів, особливо коли той самий товар надходить партіями за різною ціною. У світовій та вітчизняній практиці застосовують кілька методів оцінювання собівартості вибуття запасів. Метод FIFO (First In, First Out – «перший прийшов, перший пішов») передбачає, що товари списуються за собівартістю найдавніших партій; цей метод найкраще відображає реальний фізичний рух товарів зі строком придатності. Метод середньозваженої собівартості розраховує усереднену ціну за всіма наявними партіями, що згладжує коливання закупівельних цін. Метод ідентифікованої собівартості застосовують для унікальних або дорогих товарів, коли можливо точно простежити вартість кожної одиниці [3].

Порівняння методів оцінювання запасів наведено в таблиці 1.1.

Таблиця 1.1 – Порівняння методів оцінювання товарних запасів

Метод	Принцип списання	Сфера застосування
FIFO	За собівартістю найдавніших партій	Товари зі строком придатності
Середньозважений	За усередненою ціною партій	Однорідні масові товари
Ідентифікований	За фактичною ціною конкретної одиниці	Унікальні, дорогі товари

Вибір конкретного методу оцінювання впливає на величину собівартості реалізованих товарів та вартість залишків, а отже – на фінансові показники підприємства. Для інтернет-магазину з широким асортиментом типових товарів

найбільш практичними є методи FIFO та середньозваженої собівартості. У розроблюваному застосунку закладено можливість зберігання закупівельної ціни кожної партії в межах поставки, що створює основу для подальшої реалізації обраного методу оцінювання.

1.2 Аналіз бізнес-процесів управління товарами на складі інтернет-магазину

1.2.1 Узагальнений життєвий цикл товару

Для коректного проєктування застосунку необхідно формалізувати основні бізнес-процеси, які він має автоматизувати. Узагальнений життєвий цикл товарної позиції на складі інтернет-магазину наведено на рисунку 1.1.

Рисунок 1.1 – Життєвий цикл товару на складі інтернет-магазину

Розглянемо ключові процеси руху товарів в інтернет-магазині докладніше.

1.2.2 Процес надходження товару

Процес починається з отримання партії товару від постачальника. Відповідальний працівник перевіряє фактичну кількість і відповідність супровідним документам, після чого оприбутковує товар – збільшує обліковий залишок відповідних позицій. Якщо товарна позиція є новою, її попередньо

створюють у каталозі із зазначенням найменування, артикула (SKU), категорії, ціни та інших атрибутів. Результатом процесу є оновлений залишок і запис в журналі надходжень [3].

1.2.3 Процес оформлення замовлення

Клієнт обирає товари у каталозі, додає їх до кошика та оформлює замовлення, зазначаючи спосіб і адресу доставки. У момент підтвердження замовлення система має перевірити доступність кожної позиції та зарезервувати відповідну кількість, щоб уникнути повторного продажу того самого товару. Замовлення проходить життєвий цикл статусів: «очікує оплати», «обробляється», «комплектується», «відправлено», «виконано» або «скасовано». Зміна статусу впливає на стан запасів: підтвердження зменшує доступну кількість, відвантаження остаточно списує товар зі складу, а скасування повертає зарезервовану кількість у доступну [1].

1.2.4 Процес комплектування та доставки

Після переходу замовлення у стан комплектування працівник складу формує відправлення: відбирає товари, пакує їх і передає службі доставки. У системі при цьому фіксується факт відвантаження, формується інформація про відправлення (служба доставки, номер накладної, орієнтовна дата вручення), а залишки зменшуються на відвантажену кількість. Клієнт отримує сповіщення про відправлення та засіб відстеження.

1.2.5 Процес повернення

У разі відмови клієнта або виявлення дефекту товар може бути повернутий. Процес повернення передбачає приймання товару, перевірку його стану та прийняття рішення про повернення на склад (з відповідним збільшенням

залишку) або списання. Цей процес також має бути відображений в обліку та історії руху [3].

Узагальнено перелічені процеси та їх вплив на стан запасів подано в таблиці 1.2.

Таблиця 1.2 – Бізнес-процеси руху товарів та їх вплив на облік

Процес	**Подія-тригер**	**Вплив на залишок**
Надходження	Оприбуткування партії	Збільшення фактичного залишку
Оформлення замовлення	Підтвердження клієнтом	Резервування кількості
Відвантаження	Передавання у доставку	Списання зі складу
Скасування замовлення	Зміна статусу на «скасовано»	Зняття резерву
Повернення	Приймання повернутого товару	Збільшення залишку або списання
Інвентаризація	Перерахунок фактичних залишків	Коригування облікових даних

1.3 Огляд та аналіз існуючих програмних рішень

На ринку представлено низку програмних продуктів для управління складом та товарними запасами. Розглянемо найбільш поширені рішення, що можуть застосовуватися для інтернет-торгівлі, та визначимо їх сильні й слабкі сторони [1, 2].

1.3.1 Комплексні облікові системи

Системи класу ERP та спеціалізовані облікові програми (наприклад, рішення на базі «1С», BAS, а також SAP Business One для більшого бізнесу) забезпечують потужний функціонал управління запасами, фінансами та логістикою. Їх перевагами є повнота функцій і масштабованість. Водночас для невеликого інтернет-магазину такі системи мають суттєві недоліки: висока вартість ліцензій і впровадження, складність налаштування, потреба у

кваліфікованому персоналі, а також необхідність окремої інтеграції з вітриною інтернет-магазину, що часто реалізується через додаткові конектори.

1.3.2 Хмарні сервіси управління запасами товарів

Хмарні сервіси, такі як Zoho Inventory, inFlow, Sortly чи вітчизняні рішення, надають облік запасів за моделлю підписки. Вони простіші у впровадженні, ніж ERP, мають готові інтеграції з популярними торговельними майданчиками. Проте дані зберігаються на стороні постачальника послуги, функціонал обмежений тарифним планом, а щомісячна оплата зі зростанням обсягів стає відчутною статтею витрат. Крім того, глибина інтеграції з конкретним рушієм магазину часто є обмеженою [2].

1.3.3 Розширення для платформ електронної комерції

Для платформи WooCommerce існує екосистема плагінів управління запасами (зокрема ATUM Inventory Management, WooCommerce Stock Manager та інші). Їх перевагою є безпосередня інтеграція з каталогом і замовленнями магазину. Однак безкоштовні версії зазвичай мають обмежений функціонал, а повноцінна аналітика, керування партіями та розширені звіти доступні лише у платних розширеннях. Це підтверджує доцільність розроблення власного рішення, орієнтованого на конкретні потреби та з можливістю подальшого розвитку.

Порівняння розглянутих категорій рішень за ключовими критеріями наведено в таблиці 1.3.

Проведений аналіз свідчить, що для малого та середнього інтернет-магазину оптимальним за співвідношенням вартості, контролю над даними та можливості доопрацювання є рішення у вигляді власного плагіна для платформи WooCommerce. Саме такий підхід обрано в межах цієї роботи [12, 21].

Таблиця 1.3 – Порівняльний аналіз категорій існуючих рішень

Критерій	**ERP-системи**	**Хмарні сервіси**	**Плагіни WooCommerce**
Вартість впровадження	Висока	Середня (підписка)	Низька
Складність налаштування	Висока	Середня	Низька
Інтеграція з магазином	Потребує конектора	Через API	Вбудована
Глибина аналітики	Висока	Середня	Залежить від версії
Контроль над даними	Повний	Обмежений	Повний
Можливість доопрацювання	Обмежена	Низька	Висока (відкритий код)

Вибір базової платформи є визначальним для архітектури майбутнього застосунку. Розглянемо найпоширеніші системи побудови інтернет-магазинів.

Серед популярних платформ електронної комерції вирізняють: WooCommerce (плагін для WordPress), Shopify (хмарна SaaS-платформа), Magento (Adobe Commerce), PrestaShop та OpenCart. Кожна має свою нішу. Shopify приваблива простотою старту, але є закритою пропрієтарною платформою з обмеженим доступом до серверного коду й обов’язковою підпискою. Magento – потужна, але ресурсомістка й складна в адмініструванні платформа, орієнтована на великий бізнес. PrestaShop та OpenCart – самостійні відкриті рушії з власними екосистемами модулів [1].

Порівняння платформ наведено в таблиці 1.4.

Таблиця 1.4 – Порівняння платформ електронної комерції

Платформа	**Модель**	**Відкритий код**	**Поріг входу**	**Контроль коду**
WooCommerce	Self-hosted	Так	Низький	Повний
Shopify	SaaS	Ні	Низький	Обмежений
Magento	Self-hosted	Так	Високий	Повний
PrestaShop	Self-hosted	Так	Середній	Повний
OpenCart	Self-hosted	Так	Середній	Повний

WooCommerce, своєю чергою, є відкритим розширенням до WordPress – найпоширенішої у світі системи керування вмістом. Це забезпечує низький поріг входу, величезну спільноту розробників, велику кількість готових тем і плагінів, безкоштовність базового функціоналу та повний контроль над кодом і даними завдяки самостійному розгортанню (self-hosted). Поєднання WordPress і WooCommerce займає значну частку ринку інтернет-магазинів, що робить розроблене для нього рішення затребуваним.

На підставі порівняння для реалізації обрано платформу WordPress із плагіном WooCommerce, оскільки вона забезпечує найкраще співвідношення доступності, відкритості та можливостей розширення, що повністю відповідає завданням роботи [21, 31].

1.4 Обґрунтування вибору технологій реалізації

Розроблений застосунок будується за класичною клієнт-серверною моделлю веб-застосунку. Клієнтська частина (фронтенд) відповідає за подання інформації та взаємодію з користувачем і реалізується мовами HTML, CSS та JavaScript. Серверна частина (бекенд) виконує бізнес-логіку, працює з базою даних і реалізується мовою PHP у середовищі WordPress. Дані зберігаються в реляційній базі даних під керуванням MySQL із доступом через мову запитів SQL [7, 23, 24].

HTML забезпечує семантичну розмітку сторінок адміністративної панелі та елементів вітрини. CSS відповідає за візуальне оформлення, адаптивність і узгодженість інтерфейсу зі стилем адміністративної панелі WordPress. JavaScript реалізує інтерактивність – динамічне оновлення таблиць, валідацію форм, асинхронні запити до сервера (AJAX) без перезавантаження сторінки, побудову діаграм у модулі звітності [8, 10, 26].

Мова PHP обрана як основна серверна мова, оскільки саме на ній побудовані WordPress і WooCommerce; це забезпечує безшовну інтеграцію розробленого плагіна з ядром платформи через систему хуків (дій і фільтрів).

MySQL є стандартною СКБД для WordPress, тому використання її та мови SQL для зберігання облікових даних є природним вибором, що не потребує додаткових компонентів інфраструктури [5, 15].

1.5 Постановка завдання та функціональні вимоги до застосунку

На основі проведеного аналізу сформульовано постановку завдання: розробити веб-застосунок у вигляді плагіна WordPress, що інтегрується з WooCommerce та реалізує повний цикл управління товарами на складі інтернет-магазину. Застосунок має забезпечувати виконання таких функціональних вимог:

1. Ведення каталогу товарів з атрибутами (назва, артикул, категорія, ціна, опис, зображення);
2. облік складських залишків з розмежуванням фактичної, зарезервованої та доступної кількості;
3. реєстрація операцій руху товарів (надходження, відвантаження, повернення, коригування) з веденням журналу;
4. автоматичне резервування та списання товарів відповідно до статусів замовлень WooCommerce;
5. керування мінімальними рівнями запасу та сповіщення про необхідність поповнення;
6. оформлення замовлень з вибором способу та параметрів доставки;
7. формування статистичної та аналітичної звітності з продажів і руху товарів;
8. розмежування прав доступу між ролями (адміністратор, менеджер складу).

Перелічені вимоги визначають функціональні модулі застосунку, проєктування яких розглянуто у другому розділі.

Доступ до функцій застосунку розмежовується за ролями користувачів. Розподіл повноважень між ролями наведено в таблиці 1.5.

Таблиця 1.5 – Розмежування повноважень за ролями користувачів

Функція	Менеджер складу	Адміністратор
Перегляд каталогу та залишків	+	+
Оприбуткування поставок	+	+
Обробка замовлень і доставки	+	+
Формування звітів	+	+
Зміна налаштувань системи	–	+
Керування правами доступу	–	+

1.6 Нефункціональні вимоги до застосунку

Окрім функціональних, до застосунку висуваються нефункціональні вимоги, що визначають якісні характеристики системи:

– надійність – коректне опрацювання помилок і збереження цілісності облікових даних навіть за одночасних операцій;

– продуктивність – прийнятний час відгуку інтерфейсу та виконання запитів до бази даних за обсягу в тисячі товарних позицій;

– зручність використання – інтуїтивно зрозумілий інтерфейс, узгоджений зі стилем адміністративної панелі WordPress;

– безпека – перевірка прав доступу, захист від несанкціонованих дій (nonce-токени), екранування даних;

– сумісність – робота з актуальними версіями WordPress, WooCommerce, PHP і MySQL;

– розширюваність – модульна структура, що дозволяє додавати нові функції без переписування ядра;

– адаптивність – коректне відображення інтерфейсу на пристроях з різним розміром екрана.

1.7 Висновки до першого розділу

У першому розділі досліджено предметну область складського обліку інтернет-магазину, формалізовано основні бізнес-процеси руху товарів та визначено їх вплив на стан запасів. Проведено огляд і порівняльний аналіз наявних програмних рішень, який показав, що для малого та середнього бізнесу оптимальним є розроблення власного плагіна для платформи WooCommerce. На підставі порівняння платформ електронної комерції обґрунтовано вибір WordPress і WooCommerce, а також технологічного стеку HTML, CSS, JavaScript, PHP та MySQL. Сформульовано постановку завдання, функціональні та нефункціональні вимоги, що становлять основу для проєктування застосунку у наступному розділі.

РОЗДІЛ 2. ПРОЄКТУВАННЯ ЗАСТОСУНКУ

2.1 Архітектура застосунку

Розроблений застосунок побудовано за трирівневою клієнт-серверною архітектурою, що є типовою для сучасних веб-застосунків і забезпечує розділення відповідальності між рівнями. Така організація спрощує супровід, тестування та подальше розширення системи. Загальну архітектуру застосунку наведено на рисунку 2.1.

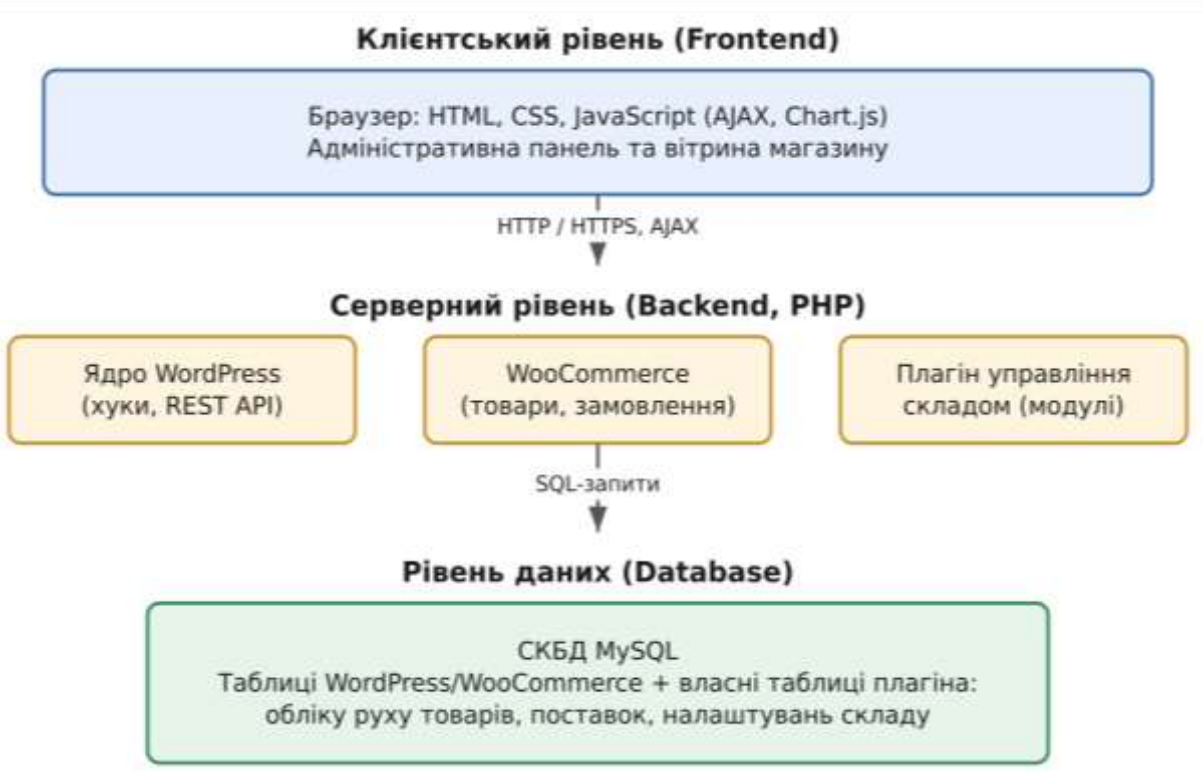

Рисунок 2.1 – Трьохрівнева архітектура застосунку

Клієнтський рівень (рівень подання) виконується у браузері користувача й відповідає за відображення інтерфейсу та взаємодію з користувачем. Він реалізований засобами HTML, CSS і JavaScript. Динамічне оновлення даних без перезавантаження сторінки забезпечується технологією AJAX, а візуалізація аналітичних даних у модулі звітності – бібліотекою побудови діаграм [4, 25].

Серверний рівень (рівень бізнес-логіки) реалізований мовою PHP у середовищі WordPress. Він складається з ядра WordPress, платформи WooCommerce та розробленого плагіна управління складом. Плагін взаємодіє з ядром і WooCommerce через систему хуків (дій і фільтрів), що дозволяє перехоплювати події (наприклад, створення замовлення чи зміну його статусу) і виконувати відповідну облікову логіку без модифікації коду платформи [12, 20].

Рівень даних реалізований на базі реляційної СКБД MySQL. Він включає стандартні таблиці WordPress і WooCommerce (товари, замовлення, метадані), а також власні таблиці плагіна, призначені для зберігання журналу руху товарів, інформації про поставки й налаштувань складу. Доступ до даних здійснюється засобами вбудованого в WordPress класу для роботи з базою даних, що забезпечує захищене формування SQL-запитів [12, 14].

2.2 Обґрунтування технологій проєктування

На етапі проєктування уточнено перелік технологій і підходів, що використовуються в реалізації. Серверна частина будується на основі об’єктно-орієнтованого підходу: функціонал плагіна інкапсулюється у класи, кожен з яких відповідає за окрему предметну область (облік залишків, рух товарів, замовлення, звітність, інтерфейс). Така модульність відповідає принципу єдиної відповідальності та полегшує супровід [16, 17].

Взаємодія з WooCommerce здійснюється через офіційний програмний інтерфейс платформи – функції роботи з товарами та замовленнями, а також через хуки. Для зберігання специфічних для складу даних, які не передбачені стандартною моделлю WooCommerce, проєктуються власні таблиці бази даних. Для клієнт-серверного обміну використовується механізм admin-ajax, вбудований у WordPress, із обов’язковою перевіркою nonce-токенів для захисту від міжсайтової підробки запитів [12, 19].

2.3 Проєктування бази даних

2.3.1 Загальна характеристика бази даних

База даних застосунку поєднує наявну схему WordPress/WooCommerce із власними таблицями плагіна. Товари зберігаються як записи типу product у таблиці wp_posts, а їх атрибути (артикул, ціна, залишок) – у таблиці метаданих. Замовлення в сучасних версіях WooCommerce зберігаються у спеціалізованих таблицях зберігання замовлень. Для реалізації повноцінного складського обліку спроєктовано додаткові таблиці. Концептуальну модель бази даних у вигляді ER-діаграми наведено на рисунку 2.2.

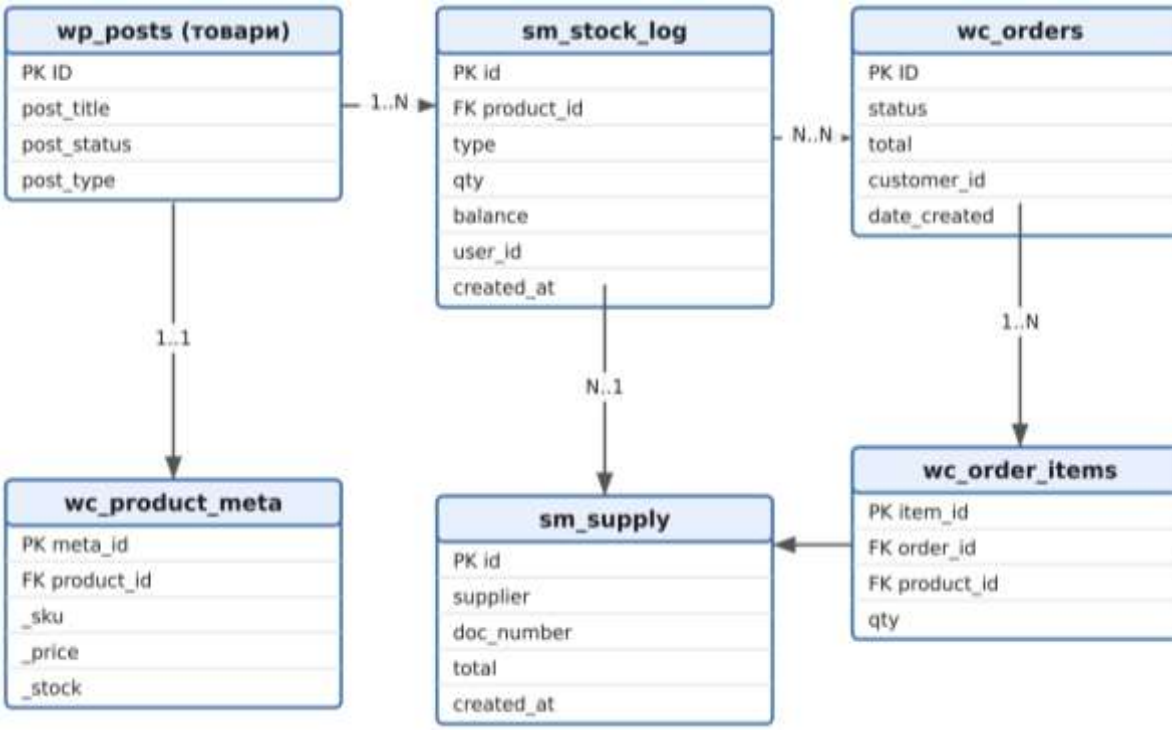

Рисунок 2.2 – ER-діаграма бази даних застосунку

Основною власною таблицею є таблиця журналу руху товарів (sm_stock_log), що фіксує кожну операцію зміни залишку. Її структуру наведено в таблиці 2.1.

Таблиця 2.1 – Структура таблиці sm_stock_log (журнал руху товарів)

Поле	Тип	Призначення
id	BIGINT, PK	Унікальний ідентифікатор операції
product_id	BIGINT, FK	Ідентифікатор товару (wp_posts.ID)
type	VARCHAR(20)	Тип операції: in, out, reserve, return, adjust
qty	INT	Кількість товару в операції
balance	INT	Залишок після операції
reference	VARCHAR(100)	Посилання на документ (замовлення, поставка)
user_id	BIGINT	Користувач, що виконав операцію
note	TEXT	Коментар до операції
created_at	DATETIME	Дата й час операції

Друга власна таблиця (sm_supply) зберігає інформацію про поставки товарів від постачальників: найменування постачальника, номер супровідного документа, загальну суму та дату. Деталізація поставок за позиціями зберігається у пов’язаній таблиці позицій поставки. Третя таблиця (sm_settings) містить налаштування плагіна – порогові значення мінімального запасу, параметри сповіщень тощо [34, 35].

Зв’язки між таблицями організовано так: один товар може мати багато записів у журналі руху (зв’язок «один-до-багатьох»); одне замовлення містить багато позицій, кожна з яких посилається на товар; кожна позиція поставки пов’язана з конкретним товаром. Така схема забезпечує цілісність даних і дає змогу будувати аналітичні запити щодо обороту, динаміки залишків та продажів [14, 24].

2.3.2 Нормалізація структури бази даних

Спроєктована схема бази даних відповідає вимогам третьої нормальної форми (3НФ), що мінімізує надмірність даних та запобігає аномаліям оновлення. Розглянемо дотримання нормальних форм на прикладі власних таблиць плагіна [34, 35].

Перша нормальна форма (1НФ) вимагає атомарності значень полів та відсутності повторюваних груп. У спроєктованих таблицях кожне поле містить єдине неподільне значення: наприклад, у таблиці поставок інформація про окремі товари винесена в окрему таблицю позицій, а не зберігається переліком в одному полі. Друга нормальна форма (2НФ) вимагає, щоб усі неключові атрибути повністю залежали від первинного ключа; у таблицях застосовано прості сурогатні первинні ключі (поле id), тому часткових залежностей не виникає. Третя нормальна форма (3НФ) усуває транзитивні залежності між неключовими атрибутами: у журналі руху товарів зберігається лише ідентифікатор товару, а його найменування та інші атрибути отримуються з відповідних таблиць WooCommerce за потреби, що виключає дублювання.

Дотримання нормальних форм забезпечує узгодженість даних: зміна найменування товару відбувається в одному місці й автоматично відображається в усіх пов’язаних записах. Водночас для пришвидшення формування аналітичних звітів у деяких випадках допускається денормалізація – зокрема, у журналі руху зберігається поле balance (залишок після операції), що технічно є похідним, але дозволяє відновити стан складу на будь-який момент без перерахунку всієї історії [35].

2.4 Проєктування структури плагіна

Плагін організовано як набір PHP-класів із чітко визначеними зонами відповідальності, об’єднаних головним файлом-завантажувачем. Структуру компонентів плагіна наведено на рисунку 2.3.

Головний файл (stock-manager.php) містить метадані плагіна, виконує підключення класів та реєструє хуки активації й деактивації. Клас активатора (class-activator.php) під час встановлення створює власні таблиці бази даних і початкові налаштування. Клас обліку залишків (class-stock.php) реалізує логіку розрахунку фактичної, зарезервованої та доступної кількості. Клас руху товарів (class-movement.php) відповідає за реєстрацію операцій у журналі. Клас

замовлень (class-orders.php) обробляє події WooCommerce, пов’язані зі зміною статусів. Клас звітності (class-reports.php) формує аналітичні дані, а клас інтерфейсу (class-admin.php) будує сторінки адміністративної панелі.

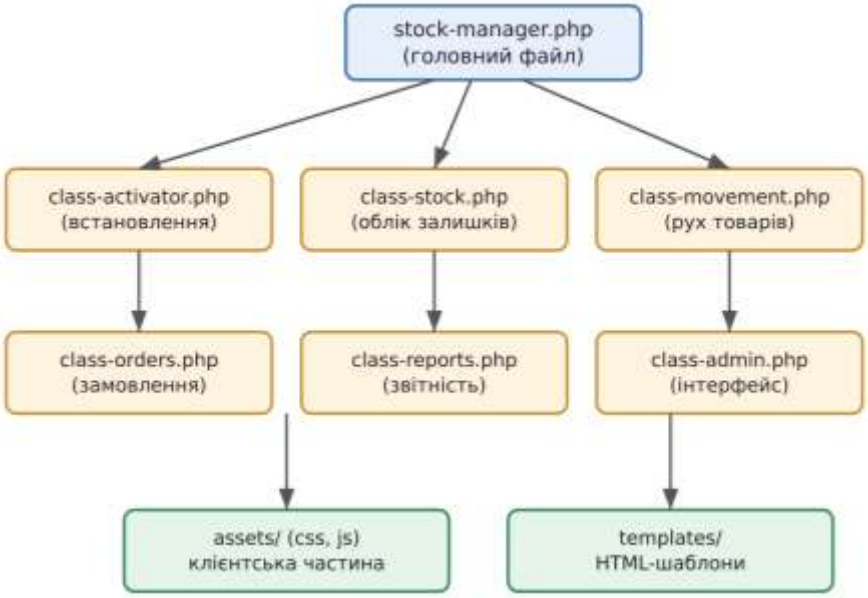

Рисунок 2.3 – Структура програмних компонентів плагіна

Клієнтські ресурси (стилі та сценарії) і HTML-шаблони винесено в окремі каталоги assets та templates відповідно [12, 20]. Перелік основних модулів та їх функцій узагальнено в таблиці 2.2.

Таблиця 2.2 – Функціональні модулі плагіна

Модуль	**Призначення**
Облік залишків	Розрахунок та зберігання залишків, резерву й доступної кількості
Рух товарів	Реєстрація операцій надходження, списання, повернення, коригування
Поставки	Оприбуткування партій товарів від постачальників
Замовлення	Обробка статусів замовлень, резервування та списання товарів
Доставка	Збереження параметрів доставки та формування відправлень
Звітність	Формування статистики продажів та руху товарів
Адміністрування	Побудова інтерфейсу й розмежування прав доступу

Для відображення статичної структури класів та зв’язків між ними побудовано діаграму класів у нотації UML (рисунок 2.4). Кожен клас інкапсулює дані (атрибути) та операції (методи) певної предметної області. Клас обробки замовлень SM_Orders використовує методи класів обліку залишків SM_Stock та руху товарів SM_Movement, що відображає відношення залежності між ними [16].

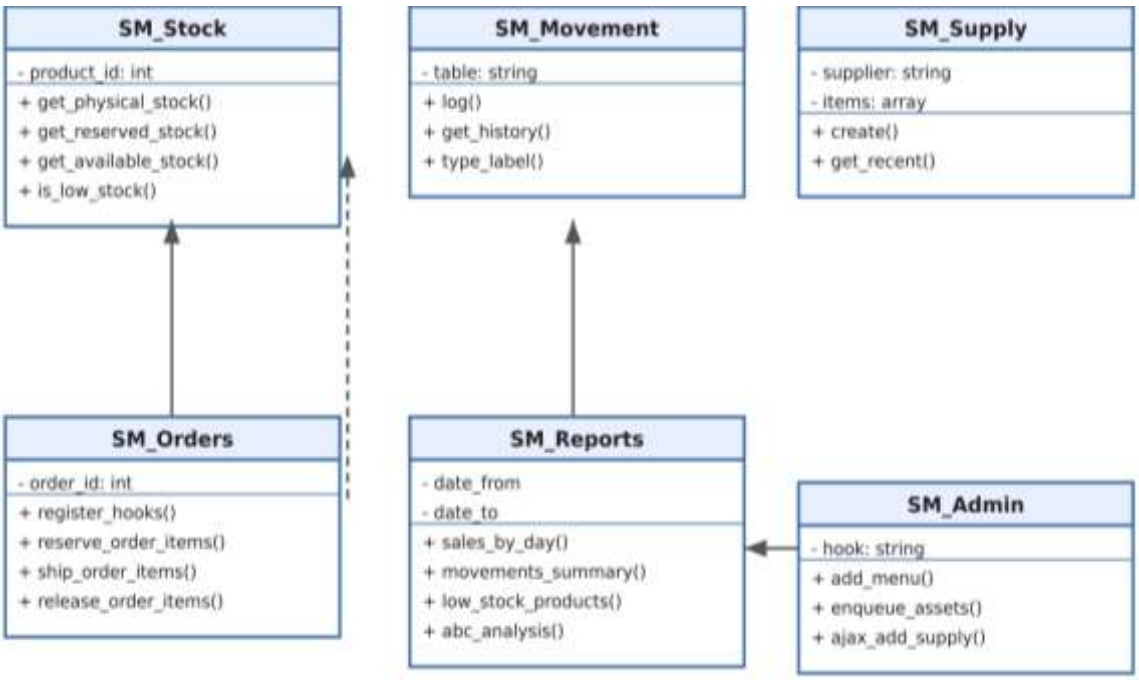

Рисунок 2.4 – Діаграма класів розробленого плагіна

2.5 Проєктування функціональних можливостей програмного рішення: діаграма прецедентів

Для формалізації функціональних вимог із позиції користувачів побудовано діаграму прецедентів (Use Case) у нотації UML, наведену на рисунку 2.5. У системі виокремлено двох основних акторів: менеджера складу та адміністратора.

Менеджер складу виконує повсякденні операції: переглядає каталог, оприбутковує товари, керує залишками, обробляє замовлення, оформлює доставку та формує звіти.

Рисунок 2.5 – Діаграма прецедентів використання системи

Адміністратор додатково має доступ до налаштувань системи та керування правами доступу інших користувачів. Розмежування прав реалізується на базі стандартної системи ролей і можливостей (capabilities) WordPress, що дозволяє гнучко налаштовувати доступ до окремих функцій плагіна [13, 19].

2.6 Проєктування взаємодії при обробці замовлення

Одним з найважливіших сценаріїв є обробка замовлення з автоматичним резервуванням товару. Для опису взаємодії компонентів у часі побудовано діаграму послідовності (рисунок 2.6).

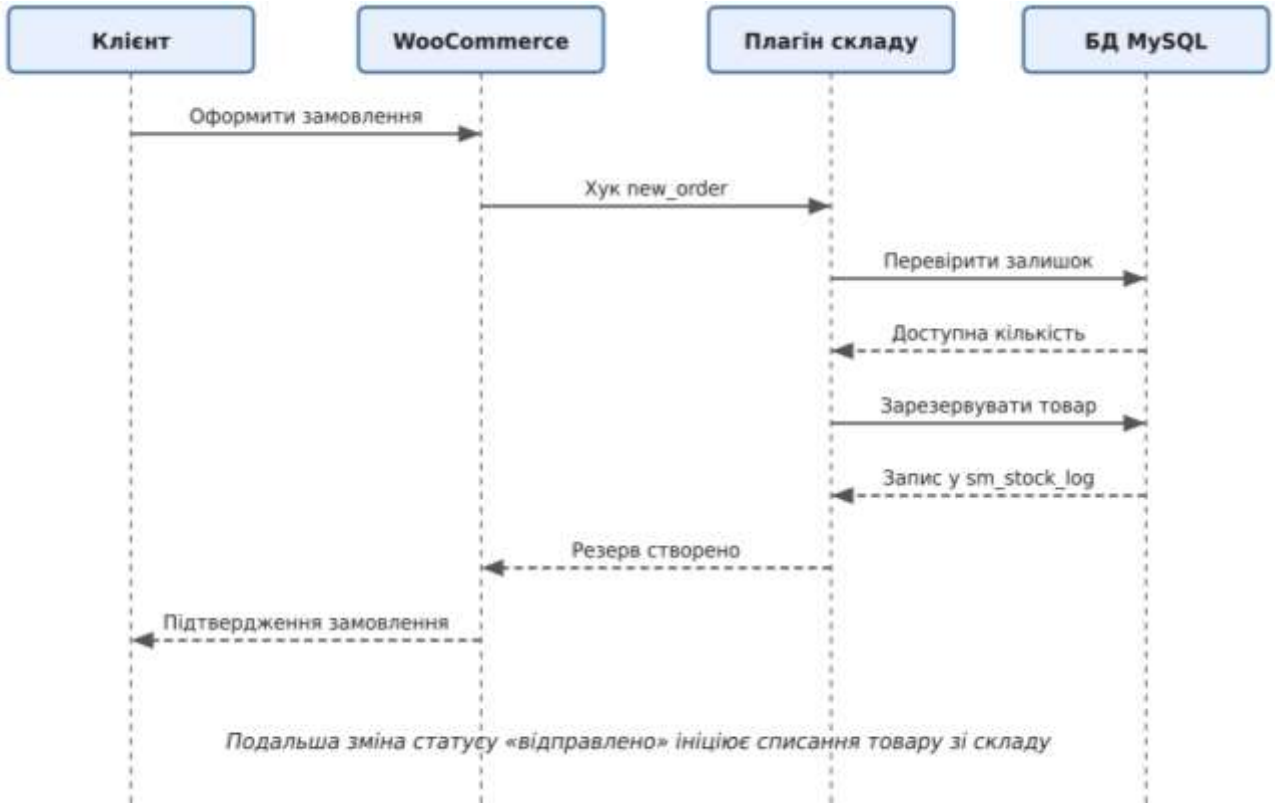

Рисунок 2.6 – Діаграма послідовності обробки замовлення

Сценарій починається з оформлення замовлення клієнтом у вітрині магазину. WooCommerce створює замовлення та ініціює хук його створення, на який підписаний плагін складу. Плагін перевіряє доступну кількість кожної позиції в базі даних і, за наявності товару, резервує відповідну кількість, додаючи запис до журналу руху. Після цього клієнт отримує підтвердження. Подальша зміна статусу замовлення на «відправлено» ініціює остаточне списання зарезервованого товару зі складу, а скасування – повернення резерву в доступну кількість [16].

2.7 Проєктування інтерфейсу користувача

Інтерфейс застосунку проєктується як розширення адміністративної панелі WordPress, що забезпечує візуальну узгодженість зі знайомим користувачам середовищем і знижує час навчання. Плагін додає до головного меню панелі

окремий пункт «Склад» із підпунктами: панель показників, товари й залишки, рух товарів, поставки, замовлення, звіти та налаштування.

Головна сторінка плагіна – інформаційна панель (дашборд) – відображає ключові показники складу у вигляді карток і діаграму динаміки продажів. Прототип головної сторінки наведено на рисунку 2.7.

Рисунок 2.7 – Прототип інформаційної панелі застосунку

Сторінка обліку залишків містить таблицю товарів із пошуком, фільтрами та індикацією стану запасу (достатній, низький, відсутній). Прототип цієї сторінки наведено на рисунку 2.8.

Під час проєктування інтерфейсу враховано принципи зручності використання: мінімальна кількість кроків для виконання типових операцій, наочна індикація стану, підтвердження небезпечних дій, адаптивність до різних розмірів екрана. Кольорове кодування стану запасу (зелений – достатній, жовтий – низький, червоний – відсутній) дозволяє швидко оцінити ситуацію на складі [9, 25].

SKU	Назва товару	Залишок	Резерв	Доступно	Статус
A-1001	Навушники бездротові	120	8	112	OK
A-1002	Смарт-годинник X3	6	4	2	Низький
B-2050	Чохол силіконовий	340	20	320	OK
C-3001	Зарядний пристрій 65W	0	0	0	Немає
D-4012	Кабель USB-C 2м	88	12	76	OK

Рисунок 2.8 – Прототип сторінки обліку залишків товарів

Перейдемо до проєктування модуля звітності та аналітики.

2.8 Проєктування модуля звітності та аналітики

Модуль звітності забезпечує формування аналітичної інформації для підтримки управлінських рішень. Спроєктовано такі види звітів: звіт про продажі за період (з деталізацією за днями, тижнями або місяцями), звіт про рух товарів (надходження й вибуття за обраний період), звіт про залишки та їх вартість, звіт про товари з низьким запасом, а також рейтинг найбільш та найменш продаваних позицій (ABC-аналіз) [3, 26].

Дані для звітів формуються агрегувальними SQL-запитами до журналу руху товарів і таблиць замовлень. Результати відображаються у вигляді таблиць та інтерактивних діаграм (стовпчастих, лінійних, кругових), а також можуть бути експортовані у формат CSV для подальшого опрацювання. Накопичені у журналі руху товарів часові ряди продажів у перспективі можуть слугувати основою для застосування моделебазованих методів аналізу та прогнозування процесів циклічної структури [47, 49, 50]. Перелік спроєктованих звітів наведено в таблиці 2.3.

Таблиця 2.3 – Перелік звітів модуля аналітики

Звіт	Джерело даних	Форма подання
Продажі за період	Замовлення WooCommerce	Лінійна діаграма, таблиця
Рух товарів	Журнал sm_stock_log	Стовпчаста діаграма
Залишки та їх вартість	Метадані товарів	Таблиця
Товари з низьким запасом	Залишки, налаштування	Таблиця зі сповіщеннями
ABC-аналіз товарів	Замовлення, журнал руху	Кругова діаграма, таблиця

2.9 Проєктування підсистеми безпеки та розмежування доступу

Безпека є критично важливою для застосунку, що працює з обліковими даними підприємства. На етапі проєктування визначено комплекс заходів захисту, які реалізуються на кількох рівнях. Контроль доступу базується на стандартній системі ролей і можливостей (capabilities) WordPress: операції зі складом дозволені лише користувачам із можливістю manage_woocommerce, що зазвичай мають адміністратори та менеджери магазину. Перед виконанням кожної дії, яка змінює дані, перевіряється наявність відповідного права [13, 19].

Перелік основних загроз та проєктних заходів протидії узагальнено в таблиці 2.4.

Таблиця 2.4 – Загрози безпеці та заходи протидії

Загроза	Захід протидії
Несанкціонований доступ	Перевірка прав (capabilities) перед кожною дією
SQL-ін’єкції	Підготовлені запити через wpdb->prepare()
Міжсайтова підробка (CSRF)	Перевірка nonce-токенів
Міжсайтовий скриптинг (XSS)	Екранування виводу та санітизація вводу
Прямий доступ до файлів	Перевірка константи ABSPATH

Для захисту від міжсайтової підробки запитів (CSRF) усі форми та AJAX-запити супроводжуються одноразовими токенами (nonce), які генеруються сервером і перевіряються під час опрацювання запиту. Захист від впровадження

SQL-коду забезпечується використанням підготовлених запитів через метод prepare() класу роботи з базою даних, який екранує всі параметри. Для запобігання міжсайтовому скриптингу (XSS) усі дані, що виводяться в інтерфейс, екрануються відповідними функціями (esc_html, esc_attr), а вхідні дані очищуються функціями санітизації [12, 19].

2.10 Висновки до другого розділу

У другому розділі спроєктовано архітектуру застосунку, що базується на трирівневій клієнт-серверній моделі. Обґрунтовано технологічні рішення проєктування, розроблено схему бази даних із власними таблицями для журналу руху товарів, поставок і налаштувань, наведено її ER-діаграму. Спроєктовано модульну структуру плагіна з розподілом відповідальності між класами, побудовано діаграми прецедентів і послідовності, що формалізують функціональні вимоги та сценарії взаємодії. Розроблено прототипи користувацького інтерфейсу та визначено склад модуля звітності. Отримані проєктні рішення є основою для практичної реалізації застосунку, розглянутої в наступному розділі.

РОЗДІЛ 3. РОЗРОБКА ЗАСТОСУНКУ ТА БАЗИ ДАНИХ

3.1 Налаштування середовища розробки

Розроблення застосунку здійснювалося в локальному середовищі, що відтворює умови роботи реального хостингу. Для цього використано програмний стек, який включає веб-сервер Apache, інтерпретатор PHP версії 7.4 або вище та СКБД MySQL. Зручним способом розгортання такого середовища є застосування готових збірок (наприклад, XAMPP, OpenServer або Local), що містять усі необхідні компоненти й не потребують ручного налаштування кожного сервісу [23, 24].

Після розгортання веб-сервера встановлено систему керування вмістом WordPress та активовано плагін електронної комерції WooCommerce, який є обов’язковою залежністю розроблюваного застосунку. Для контролю версій коду використано систему Git, що дозволяє відстежувати зміни, повертатися до попередніх станів і вести розроблення поетапно. Як редактор коду застосовано середовище з підтримкою синтаксису PHP, JavaScript і CSS [23].

Розроблений застосунок оформлено як окремий плагін WordPress, розміщений у каталозі wp-content/plugins/stock-manager. Такий підхід забезпечує ізоляцію коду від ядра платформи та теми оформлення, що відповідає найкращим практикам розроблення під WordPress і гарантує збереження функціональності під час оновлень [12, 20].

3.2 Реалізація структури плагіна та його ініціалізація

Точкою входу плагіна є головний файл stock-manager.php, що містить заголовок з метаданими, оголошення констант, підключення класів модулів та реєстрацію хуків життєвого циклу. Заголовок плагіна у вигляді коментаря зчитується WordPress для відображення інформації про плагін у панелі адміністратора. Фрагмент головного файлу наведено в лістингу 3.1.

Лістинг 3.1 – Ініціалізація плагіна (stock-manager.php)

```
// Константи плагіна.
define( 'SM_VERSION', '1.0.0' );
define( 'SM_PLUGIN_DIR', plugin_dir_path( __FILE__ ) );
define( 'SM_PLUGIN_URL', plugin_dir_url( __FILE__ ) );

// Підключення класів модулів.
require_once SM_PLUGIN_DIR . 'includes/class-activator.php';
require_once SM_PLUGIN_DIR . 'includes/class-stock.php';
require_once SM_PLUGIN_DIR . 'includes/class-movement.php';
// ... інші класи модулів

function sm_init() {
    if ( ! sm_check_woocommerce() ) {
        return;
    }
    $orders = new SM_Orders();
    $orders->register_hooks();
    if ( is_admin() ) {
        $admin = new SM_Admin();
        $admin->register_hooks();
    }
}
add_action( 'plugins_loaded', 'sm_init' );

register_activation_hook( __FILE__, array( 'SM_Activator', 'activate' ) );
```

Функція sm_init() викликається на хуку plugins_loaded після завантаження всіх плагінів, що гарантує доступність WooCommerce. Перед ініціалізацією виконується перевірка наявності WooCommerce; за його відсутності користувачеві виводиться повідомлення, а ініціалізація припиняється. Це запобігає виникненню фатальних помилок через звернення до невизначених функцій [12, 19].

3.3 Створення таблиць бази даних

Під час активації плагіна викликається метод activate() класу SM_Activator, який створює власні таблиці бази даних. Для безпечного створення таблиць використовується вбудована функція WordPress dbDelta(), що порівнює задану структуру з наявною та застосовує лише необхідні зміни. Фрагмент створення таблиці журналу руху товарів наведено в лістингу 3.2.

Лістинг 3.2 – Створення таблиці журналу руху товарів

```
private static function create_tables() {
    global $wpdb;
    $charset_collate = $wpdb->get_charset_collate();
    require_once ABSPATH . 'wp-admin/includes/upgrade.php';

    $log_table = $wpdb->prefix . 'sm_stock_log';
    $sql_log = "CREATE TABLE {$log_table} (
        id BIGINT(20) UNSIGNED NOT NULL AUTO_INCREMENT,
        product_id BIGINT(20) UNSIGNED NOT NULL,
        type VARCHAR(20) NOT NULL,
        qty INT(11) NOT NULL DEFAULT 0,
        balance INT(11) NOT NULL DEFAULT 0,
        reference VARCHAR(100) DEFAULT NULL,
        user_id BIGINT(20) UNSIGNED DEFAULT NULL,
        note TEXT DEFAULT NULL,
        created_at DATETIME NOT NULL DEFAULT CURRENT_TIMESTAMP,
        PRIMARY KEY (id),
        KEY product_id (product_id)
    ) {$charset_collate};";

    dbDelta( $sql_log );
}
```

Використання префікса $wpdb->prefix забезпечує сумісність з будь-якою інсталяцією WordPress незалежно від обраного префікса таблиць. Застосування параметра $charset_collate гарантує коректне кодування даних (utf8mb4), що важливо для зберігання тексту українською мовою [14, 24].

3.4 Реалізація програмного модуля обліку залишків

Логіку обліку залишків інкапсульовано у класі SM_Stock. Фактичний залишок зчитується зі сховища WooCommerce, тоді як зарезервована кількість розраховується за журналом руху товарів агрегувальним SQL-запитом. Доступна до продажу кількість визначається як різниця фактичного залишку та резерву. Реалізацію ключових методів наведено в лістингу 3.3.

Лістинг 3.3 – Розрахунок залишків товару (class-stock.php)

```
public static function get_reserved_stock( $product_id ) {
    global $wpdb;
    $table = $wpdb->prefix . 'sm_stock_log';
    $reserved = $wpdb->get_var( $wpdb->prepare(
        "SELECT COALESCE(SUM(CASE WHEN type = 'reserve' THEN qty
                WHEN type = 'unreserve' THEN -qty ELSE 0 END), 0)
```

```
            FROM {$table} WHERE product_id = %d",
            $product_id
        ) );
        return max( 0, (int) $reserved );
}

public static function get_available_stock( $product_id ) {
        $physical = self::get_physical_stock( $product_id );
        $reserved = self::get_reserved_stock( $product_id );
        return max( 0, $physical - $reserved );
}
```

Застосування методу $wpdb->prepare() для формування SQL-запитів є важливим аспектом безпеки: він екранує параметри й унеможливлює впровадження SQL-коду (SQL-ін’єкції). Підрахунок резерву через одну агрегувальну операцію SUM з умовним виразом CASE дозволяє отримати результат за один запит до бази даних, що позитивно впливає на продуктивність [14, 24].

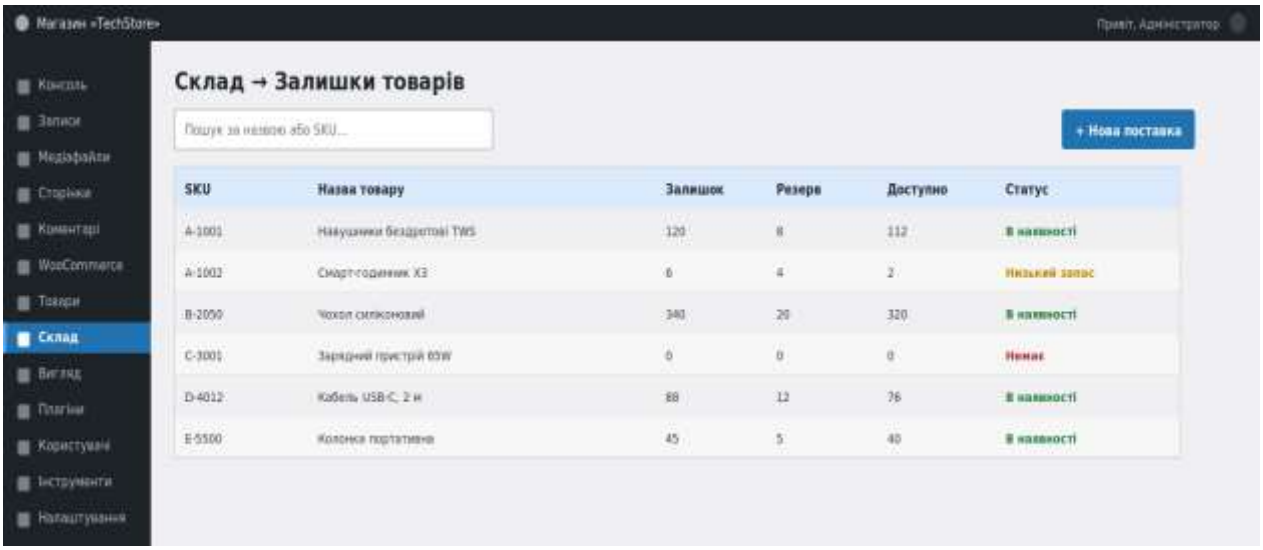

Рисунок 3.1 – Інтерфейс модуля обліку залишків товарів

3.5 Реалізація журналу руху товарів

Кожна зміна стану запасів реєструється у журналі руху товарів методом log() класу SM_Movement. Метод зберігає тип операції, кількість, залишок після операції, посилання на документ, ідентифікатор користувача та позначку часу.

Усі текстові дані попередньо очищуються функціями санітизації WordPress. Реалізацію методу наведено в лістингу 3.4.

Лістинг 3.4 – Реєстрація операції руху (class-movement.php)

```
public static function log( $product_id, $type, $qty, $reference = '', $note =
'' ) {
    global $wpdb;
    $table = $wpdb->prefix . 'sm_stock_log';
    $balance = SM_Stock::get_physical_stock( $product_id );

    $result = $wpdb->insert(
        $table,
        array(
            'product_id' => (int) $product_id,
            'type'       => sanitize_text_field( $type ),
            'qty'        => (int) $qty,
            'balance'    => (int) $balance,
            'reference'  => sanitize_text_field( $reference ),
            'user_id'    => get_current_user_id(),
            'note'       => sanitize_textarea_field( $note ),
            'created_at' => current_time( 'mysql' ),
        ),
        array( '%d', '%s', '%d', '%d', '%s', '%d', '%s', '%s' )
    );
    return $result ? $wpdb->insert_id : false;
```

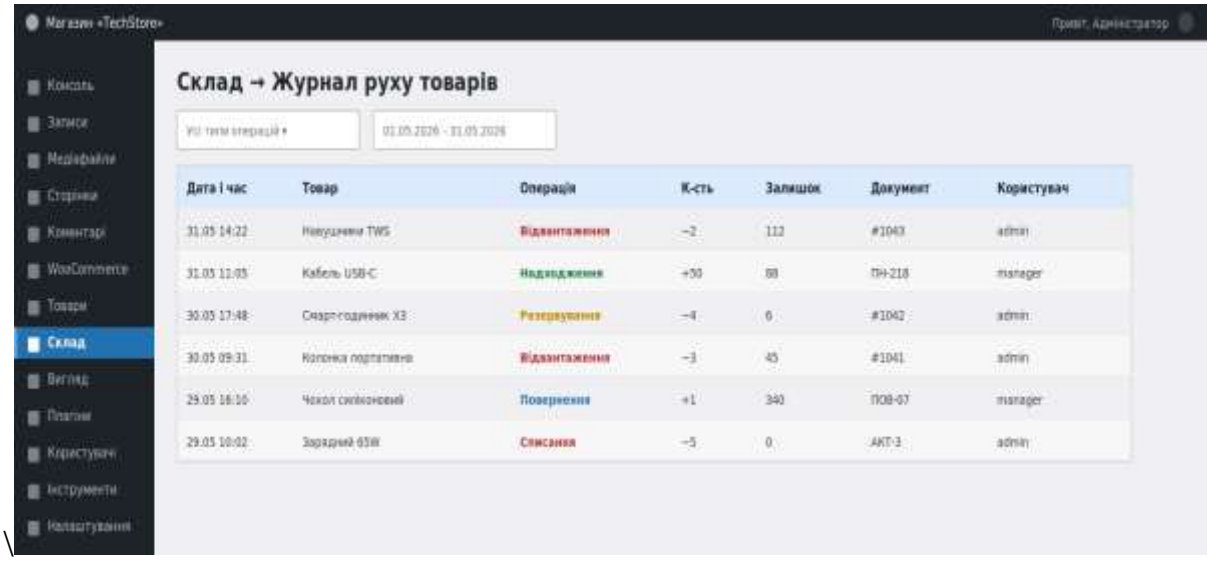

Рисунок 3.2 – Сторінка журналу руху товарів

3.6 Реалізація обробки замовлень та інтеграції з WooCommerce

Інтеграція з WooCommerce реалізована через систему хуків. Клас SM_Orders у методі register_hooks() підписується на події зміни статусів

замовлень. При оформленні замовлення товари резервуються, при переході у статус «виконано» – списуються зі складу, при скасуванні – резерв знімається, а при поверненні коштів – товари повертаються на склад. Реалізацію реєстрації хуків та резервування наведено в лістингу 3.5.

Лістинг 3.5 – Обробка подій замовлень (class-orders.php)

```
public function register_hooks() {
    add_action( 'woocommerce_checkout_order_processed',
        array( $this, 'reserve_order_items' ), 10, 1 );
    add_action( 'woocommerce_order_status_completed',
        array( $this, 'ship_order_items' ), 10, 1 );
    add_action( 'woocommerce_order_status_cancelled',
        array( $this, 'release_order_items' ), 10, 1 );
}

public function ship_order_items( $order_id ) {
    $order = wc_get_order( $order_id );
    if ( ! $order ) { return; }
    foreach ( $order->get_items() as $item ) {
        $product_id = $item->get_product_id();
        $qty        = $item->get_quantity();
        SM_Movement::log( $product_id, 'unreserve', $qty, 'order#' . $order_id
);
        SM_Stock::adjust_physical_stock( $product_id, -$qty );
        SM_Movement::log( $product_id, 'out', $qty, 'order#' . $order_id,
            'Відвантаження за замовленням' );
    }
}
```

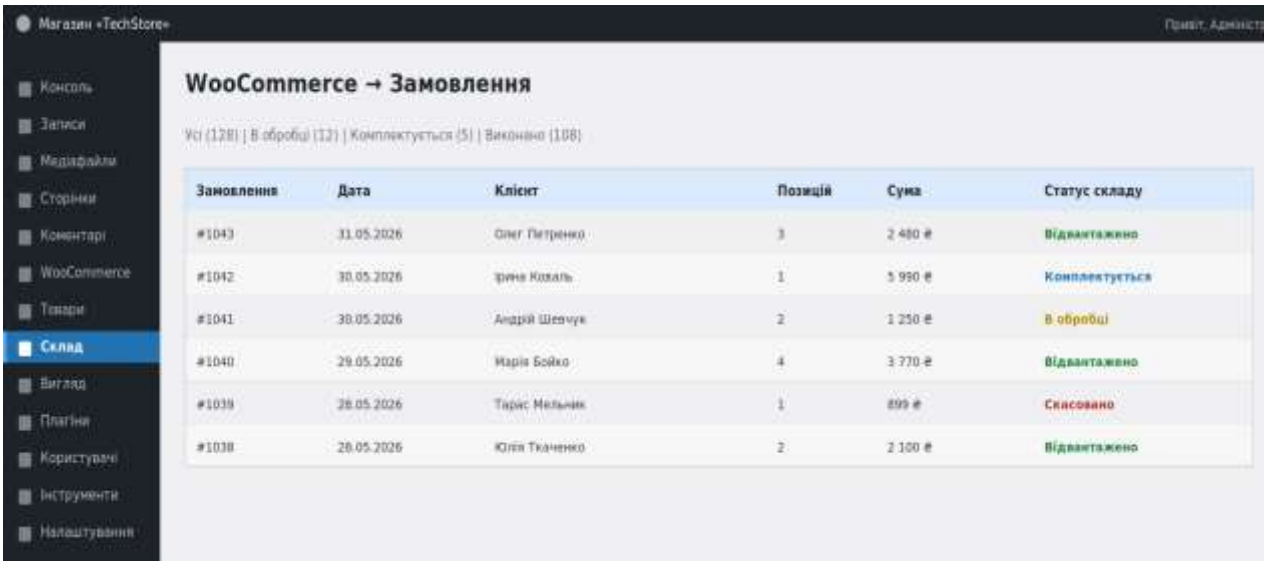

Рисунок 3.3 – Список замовлень в адміністративній панелі

Такий підхід забезпечує повну автоматизацію складського обліку: облікові операції виконуються як реакція на природні події життєвого циклу замовлення без потреби в ручному втручанні оператора. Завдяки слабкому зв’язуванню через хуки плагін не модифікує код WooCommerce і залишається сумісним з його оновленнями [21, 22].

3.7 Реалізація модуля поставок

Оприбуткування партій товарів реалізовано у класі SM_Supply. Метод create() створює запис поставки, додає її позиції, збільшує фактичні залишки товарів і реєструє відповідні операції надходження в журналі руху. Усі дії виконуються в межах єдиної логічної операції, що забезпечує узгодженість даних. Фрагмент реалізації наведено в лістингу 3.6.

Лістинг 3.6 – Оприбуткування поставки (class-supply.php)

```
foreach ( $items as $it ) {
    $product_id = (int) $it['product_id'];
    $qty        = (int) $it['qty'];
    $cost       = (float) $it['cost'];

    $wpdb->insert( $items_table, array(
        'supply_id'  => $supply_id,
        'product_id' => $product_id,
        'qty'        => $qty,
        'cost'       => $cost,
    ), array( '%d', '%d', '%d', '%f' ) );

    // Збільшення залишку та запис у журнал руху.
    SM_Stock::adjust_physical_stock( $product_id, $qty );
    SM_Movement::log( $product_id, 'in', $qty, 'supply#' . $supply_id,
        'Надходження від постачальника: ' . $supplier );
      }
```

Наведений фрагмент ілюструє ключовий принцип роботи модуля поставок: кожна позиція обробляється послідовно, а збільшення фізичного залишку та запис у журнал руху товарів виконуються в межах єдиної логічної операції, що унеможливлює розбіжність між обліковими даними та фактичним станом складу. Результатом виконання методу є оновлений залишок у таблиці

sm_stock_log та відповідний запис типу «in» у журналі руху, який відображається в адміністративній панелі у вигляді форми оформлення нової поставки (рисунок 3.4).

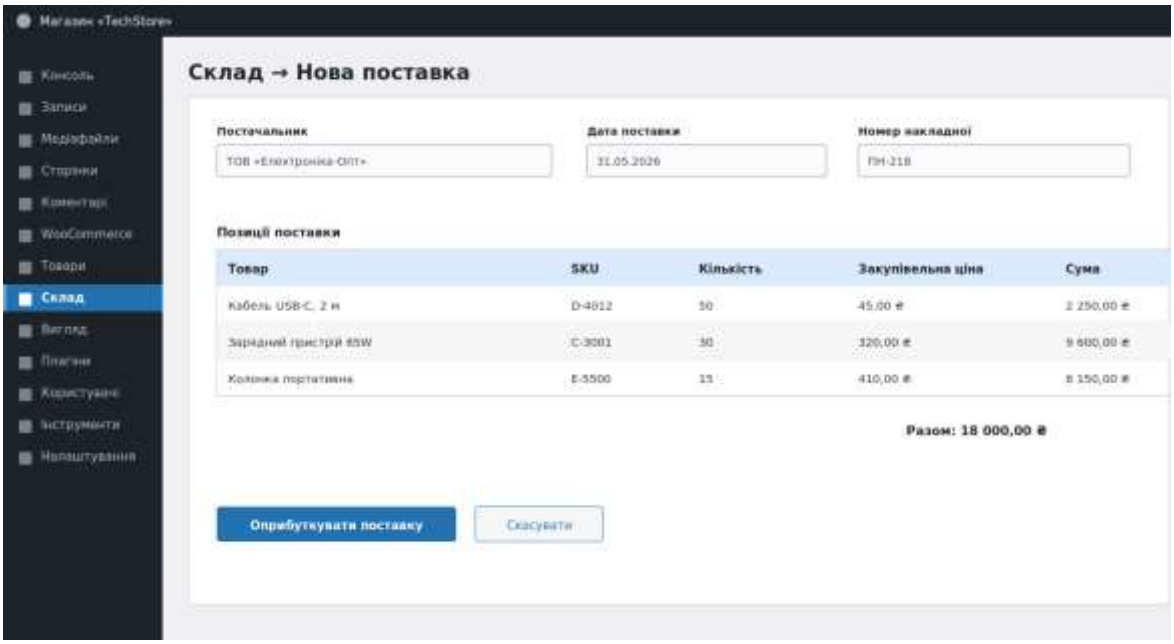

Рисунок 3.4 – Форма оформлення нової поставки

3.8 Реалізація користувацького інтерфейсу та клієнтської частини

Інтерфейс плагіна реалізовано засобами стандартного API WordPress для побудови сторінок адміністративної панелі. Клас SM_Admin реєструє пункт меню «Склад» з підпунктами, підключає стилі та сценарії й обробляє AJAX-запити. Підключення ресурсів відбувається лише на сторінках плагіна, щоб не впливати на роботу інших розділів панелі. Реалізацію підключення ресурсів наведено в лістингу 3.7.

Лістинг 3.7 – Підключення стилів і сценаріїв (class-admin.php)

```
public function enqueue_assets( $hook ) {
    if ( strpos( $hook, 'sm-' ) === false ) {
        return;
    }
    wp_enqueue_style( 'sm-admin',
        SM_PLUGIN_URL . 'assets/css/admin.css', array(), SM_VERSION );
    wp_enqueue_script( 'chartjs',
```

```
        'https://cdn.jsdelivr.net/npm/chart.js', array(), '4.4.0', true );
    wp_enqueue_script( 'sm-admin',
        SM_PLUGIN_URL . 'assets/js/admin.js',
        array( 'jquery', 'chartjs' ), SM_VERSION, true );
    wp_localize_script( 'sm-admin', 'smData', array(
        'ajaxUrl' => admin_url( 'admin-ajax.php' ),
        'nonce'   => wp_create_nonce( 'sm_nonce' ),
    ) );
}
```

Передавання даних із серверної частини до JavaScript здійснюється функцією wp_localize_script(), яка створює об’єкт smData з адресою обробника AJAX та токеном безпеки nonce. Цей токен перевіряється на сервері перед виконанням будь-якої дії, що захищає від міжсайтової підробки запитів (CSRF) [19, 27].

Клієнтська частина реалізована мовою JavaScript із використанням бібліотеки jQuery, що постачається разом із WordPress. Вона забезпечує пошук у таблиці залишків, побудову діаграми продажів та збереження поставок через AJAX. Фрагмент побудови діаграми продажів наведено в лістингу 3.8.

Лістинг 3.8 – Побудова діаграми продажів (admin.js)

```
$.post(smData.ajaxUrl, {
    action: 'sm_sales_data',
    nonce: smData.nonce
}, function (response) {
    if (!response.success) { return; }
    var labels = response.data.map(function (d) { return d.date; });
    var values = response.data.map(function (d) { return d.total; });
    new Chart(canvas.getContext('2d'), {
        type: 'bar',
        data: {
            labels: labels,
            datasets: [{ label: 'Дохід, ₴', data: values,
                backgroundColor: '#3b6ea5' }]
        },
        options: { responsive: true,
            plugins: { legend: { display: false } } }
    });
});
```

Дані для діаграми завантажуються асинхронно: сценарій надсилає AJAX-запит до обробника sm_sales_data, який повертає масив сум продажів за днями у форматі JSON. Отримані дані передаються бібліотеці Chart.js для побудови

стовпчастої діаграми. Такий підхід забезпечує інтерактивність без перезавантаження сторінки [10, 25, 27].

Серверний обробник AJAX-запиту реалізовано як метод класу адміністрування. Він перевіряє nonce-токен, формує дані за допомогою класу звітності та повертає їх у форматі JSON стандартними функціями WordPress. Реалізацію обробника наведено в лістингу 3.9 [12, 21].

Лістинг 3.9 – Серверний обробник AJAX-запиту (class-admin.php)

```
public function ajax_sales_data() {
    check_ajax_referer( 'sm_nonce', 'nonce' );
    $from = sanitize_text_field( $_POST['from']
        ?? date( 'Y-m-d', strtotime( '-7 days' ) ) );
    $to   = sanitize_text_field( $_POST['to'] ?? date( 'Y-m-d' ) );
    $data = SM_Reports::sales_by_day( $from, $to );
    wp_send_json_success( $data );
}
```

3.9 Реалізація модуля звітності

Модуль звітності реалізовано у класі SM_Reports. Він формує дані для діаграм та таблиць аналітики. Одним з найбільш корисних для управління є ABC-аналіз, який розподіляє товари на три групи за внеском у виручку: група A – товари, що дають до 80% виручки, група B – наступні 15%, група C – решта. Реалізацію ABC-аналізу наведено в лістингу 3.10.

Лістинг 3.10 – ABC-аналіз товарів (class-reports.php)

```
uasort( $revenue, function ( $a, $b ) {
    return $b['sum'] <=> $a['sum'];
} );

$total = array_sum( array_column( $revenue, 'sum' ) );
$cumulative = 0;
$result = array();
foreach ( $revenue as $pid => $data ) {
    $cumulative += $data['sum'];
    $share = $total > 0 ? $cumulative / $total : 0;
    $group = $share <= 0.8 ? 'A' : ( $share <= 0.95 ? 'B' : 'C' );
    $result[] = array(
        'product_id' => $pid,
        'name'       => $data['name'],
        'revenue'    => round( $data['sum'], 2 )
```

Алгоритм сортує товарні позиції за спаданням виручки та накопичувально підраховує її частку від загального обсягу: позиції, що формують перші 80 %, потрапляють до групи A, наступні 15 % – до групи B, решта – до групи C. Такий підхід дозволяє керівнику магазину зосередити увагу на товарах групи A, що мають найбільший вплив на виручку, та своєчасно коригувати рівні їх запасів. Результат ABC-аналізу відображається у вигляді зведеної таблиці на панелі звітності (рисунок 3.5).

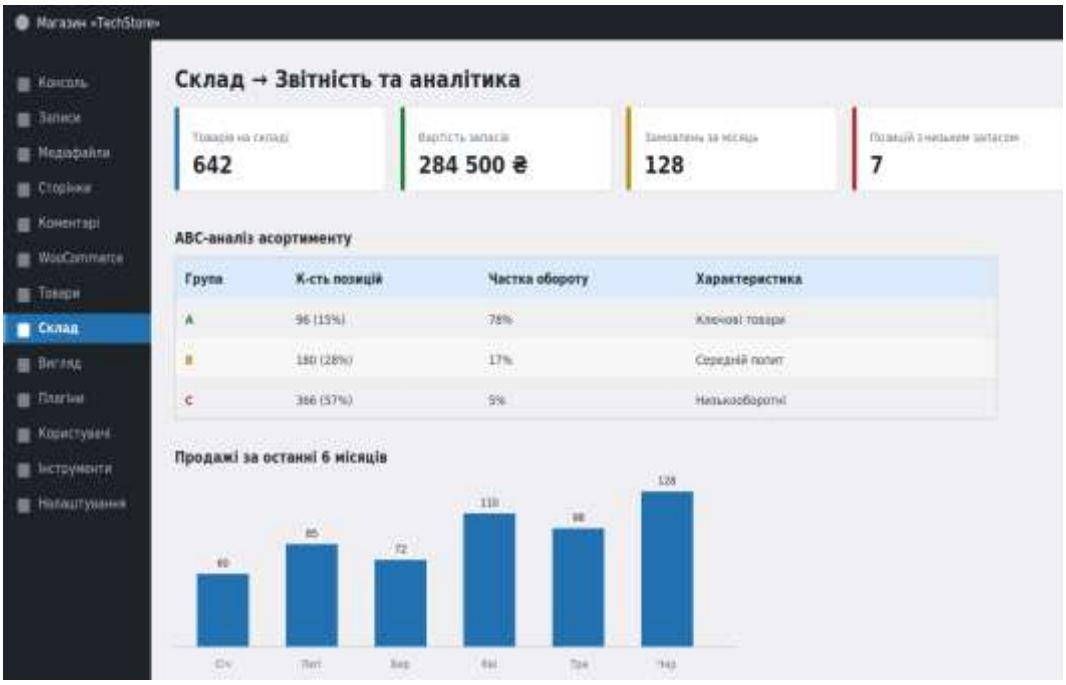

Рисунок 3.5 – Панель звітності з ABC-аналізом

Перейдемо до тестування розробленого застосунку.

3.10 Тестування застосунку

Для перевірки коректності роботи розробленого застосунку проведено функціональне тестування методом чорної скриньки. Тестування полягало у виконанні типових сценаріїв використання та порівнянні фактичного результату з очікуваним. Перевірено основні функціональні можливості: оприбуткування товарів, резервування при оформленні замовлення, списання при відвантаженні,

повернення, формування звітів та індикацію низького запасу. Результати тестування наведено в таблиці 3.1.

Усі тестові сценарії пройдено успішно, що підтверджує відповідність розробленого застосунку поставленим функціональним вимогам. Окремо перевірено аспекти безпеки: спроби виконати дії без належних прав доступу або без коректного nonce-токена відхиляються системою. Також перевірено коректність відображення інтерфейсу на екранах різної ширини, що підтвердило адаптивність розробленого рішення [17, 18].

Таблиця 3.1 – Результати функціонального тестування

Тестовий сценарій	**Очікуваний результат**	**Статус**
Оприбуткування поставки 50 од.	Залишок збільшено на 50, запис у журналі	Пройдено
Оформлення замовлення на 3 од.	Резерв 3 од., доступно зменшено	Пройдено
Переведення замовлення у «виконано»	Залишок зменшено, резерв знято	Пройдено
Скасування замовлення	Резерв повернено в доступну кількість	Пройдено
Повернення коштів за замовленням	Товар повернено на склад	Пройдено
Залишок нижче порогу	Статус «Низький», поява у звіті	Пройдено
Формування ABC-аналізу	Коректний розподіл на групи A, B, C	Пройдено
Пошук товару за SKU	Фільтрація таблиці залишків	Пройдено

Поряд із функціональним тестуванням методом чорної скриньки застосовано модульний підхід до перевірки окремих методів. Зокрема, методи розрахунку залишків перевірено на граничних значеннях: нульовому залишку, від’ємному результаті резервування (який коректно обмежується нулем), а також на великій кількості операцій у журналі руху. Інтеграційне тестування полягало у перевірці взаємодії плагіна з WooCommerce: створювалися реальні тестові замовлення, після чого контролювалася відповідність складських залишків очікуваним значенням на кожному етапі життєвого циклу замовлення [18, 21].

3.11 Аналіз продуктивності

Для оцінювання продуктивності застосунку проведено вимірювання часу виконання основних операцій на тестовій базі даних, що містила близько двох тисяч товарних позицій та десять тисяч записів у журналі руху. Розрахунок доступного залишку окремого товару, що реалізований через єдиний агрегувальний запит з індексом за полем product_id, виконувався за час, непомітний для користувача. Формування сторінки залишків зі ста товарами та сторінки звітів з ABC-аналізом також відбувалося у прийнятних межах.

Для забезпечення продуктивності під час проєктування бази даних передбачено індекси за полями, що найчастіше використовуються в умовах вибірки та групування: ідентифікатором товару, типом операції та датою створення запису. Це суттєво пришвидшує виконання аналітичних запитів. Подальшими напрямами оптимізації за значного зростання обсягу даних можуть бути кешування результатів звітів та періодичне архівування застарілих записів журналу руху [18, 24].

3.12 Розгортання та рекомендації щодо використання

Розгортання застосунку на робочому сервері здійснюється копіюванням каталогу плагіна до теки wp-content/plugins та його активацією через панель адміністратора. Під час активації автоматично створюються необхідні таблиці бази даних. Після активації в головному меню панелі з’являється розділ «Склад», через який здійснюється керування товарними запасами.

Для повноцінного використання застосунку рекомендується: налаштувати пороговий рівень мінімального запасу відповідно до специфіки бізнесу; регулярно оприбутковувати поставки для підтримання актуальності залишків; використовувати звіти для планування закупівель та аналізу асортименту. Завдяки модульній архітектурі плагін може бути розширений додатковими

функціями, такими як інтеграція зі службами доставки, друк етикеток або підтримка кількох складів [20, 28].

3.13 Висновки до третього розділу

У третьому розділі описано практичну реалізацію застосунку для управління товарами на складі. Налаштовано середовище розробки, реалізовано структуру плагіна WordPress та механізм його ініціалізації. Розроблено модулі обліку залишків, журналу руху товарів, поставок, обробки замовлень з інтеграцією WooCommerce, а також модуль звітності з ABC-аналізом. Реалізовано користувацький інтерфейс на базі адміністративної панелі WordPress та клієнтську частину з асинхронною взаємодією й побудовою діаграм. Проведене функціональне тестування підтвердило коректність роботи всіх ключових функцій та відповідність застосунку поставленим вимогам. Наведено рекомендації щодо розгортання й подальшого розвитку рішення.

РОЗДІЛ 4. БЕЗПЕКА ЖИТТЄДІЯЛЬНОСТІ, ОСНОВИ ОХОРОНИ ПРАЦІ

4.1 Безпека життєдіяльності

Цей розділ виконано відповідно до методичних вказівок та навчально-методичних матеріалів кафедри з дисципліни „Безпека життєдіяльності, основи охорони праці“ [45, 46], які визначають структуру й зміст відповідного розділу кваліфікаційної роботи здобувача освітнього ступеня „бакалавр“.

Трудова діяльність людини, зокрема розумова праця інженера-програміста, тісно пов’язана з функціонуванням центральної нервової системи (ЦНС), яка є головним регулятором усіх процесів життєдіяльності організму та забезпечує його взаємодію із зовнішнім середовищем. Центральна нервова система складається з головного та спинного мозку, які утворюються мільярдами нервових клітин – нейронів, об’єднаних у складні функціональні зв’язки. Саме ЦНС сприймає інформацію від органів чуття, опрацьовує її, формує відповідні реакції та координує роботу всіх систем організму під час виконання будь-якого виду праці [36, 38].

Робота центральної нервової системи ґрунтується на рефлекторному принципі. Розрізняють безумовні рефлекси, які є вродженими та забезпечують основні життєві функції, і умовні рефлекси, що формуються протягом життя людини на основі набутого досвіду. Учення про вищу нервову діяльність, розроблене І. П. Павловим, пояснює, що саме умовні рефлекси лежать в основі навчання, формування трудових навичок та професійних умінь. У процесі багаторазового повторення певних робочих операцій у корі головного мозку утворюється так званий динамічний стереотип – стійка система умовно-рефлекторних зв’язків, завдяки якій звичні дії виконуються швидко, точно та з меншими витратами нервової енергії [37, 39].

Під час трудової діяльності центральна нервова система виконує низку найважливіших функцій. Вона забезпечує сприйняття та опрацювання інформації, концентрацію та переключення уваги, оперативну і довготривалу

пам’ять, мислення й ухвалення рішень, а також координацію рухів і регуляцію емоційного стану працівника. Для розумової праці, до якої належить розроблення програмного забезпечення, характерне переважання інформаційних та аналітичних навантажень: працівник тривалий час сприймає й аналізує великі обсяги даних, утримує в пам’яті складні логічні конструкції, ухвалює рішення в умовах обмеженого часу. Усе це створює значне навантаження саме на вищі відділи центральної нервової системи – кору великих півкуль головного мозку [39].

Працездатність людини протягом робочого дня не є сталою величиною й безпосередньо залежить від функціонального стану ЦНС. На початку роботи відбувається період впрацьовування, коли поступово підвищується збудливість нервових центрів і налагоджуються потрібні робочі зв’язки. Далі настає фаза стійкої високої працездатності, після чого внаслідок розвитку втоми починається її зниження. Втома – це закономірний фізіологічний стан, що виникає через виснаження ресурсів нервових клітин та розвиток у корі головного мозку охоронного гальмування, яке захищає нервову систему від надмірного перенапруження. За розумової праці втома часто має прихований характер і супроводжується погіршенням уваги, сповільненням реакцій, збільшенням кількості помилок [39].

Особливістю праці програміста є тривале перебування у вимушеній робочій позі, монотонність окремих операцій, постійне нервово-емоційне напруження та робота в умовах дефіциту часу. Монотонність і одноманітність дій спричиняють розвиток гальмівних процесів у корі головного мозку, що знижує тонус нервової системи й призводить до передчасної втоми. Натомість надмірне інформаційне навантаження та емоційна напруга можуть викликати перезбудження нервових центрів, порушення сну, дратівливість і навіть розвиток неврозів. Тому раціональна організація праці з урахуванням закономірностей роботи ЦНС є необхідною умовою збереження високої працездатності та здоров’я працівника [36, 38].

Для підтримання оптимального функціонального стану центральної нервової системи застосовують науково обґрунтований режим праці та відпочинку, що передбачає чергування періодів роботи з регламентованими перервами. Короткі перерви дозволяють відновити ресурси нервових центрів і запобігти накопиченню втоми. Важливе значення мають також зміна видів діяльності, виконання фізичних вправ для зняття статичного напруження, провітрювання приміщення, дотримання правильного освітлення робочого місця та оптимального мікроклімату. Достатній нічний сон, раціональне харчування та регулярна рухова активність сприяють відновленню працездатності нервової системи. Урахування ролі центральної нервової системи в трудовому процесі дає змогу організувати працю так, щоб забезпечити її високу ефективність за одночасного збереження здоров’я та працездатності людини [37, 39].

4.2 Основи охорони праці

Пожежна безпека є однією з найважливіших складових охорони праці, оскільки пожежа становить серйозну загрозу для життя й здоров’я працівників, а також призводить до значних матеріальних збитків. Особливо актуальним питання запобігання поширенню пожежі є для виробничих цехів та приміщень, де зосереджено технологічне обладнання, електроустановки, горючі матеріали, а також перебуває значна кількість людей. Горіння можливе за одночасної наявності трьох умов: горючої речовини, окисника (як правило, кисню повітря) та джерела запалювання. Тому система протипожежного захисту приміщення спрямована насамперед на унеможливлення виникнення цих умов, а в разі виникнення пожежі – на обмеження її поширення та якнайшвидшу ліквідацію [40, 41].

Відповідно до вимог нормативних документів усі приміщення за вибухопожежною та пожежною небезпекою поділяють на категорії А, Б, В, Г і Д залежно від властивостей речовин і матеріалів, що в них обертаються. Визначення категорії приміщення є основою для встановлення вимог до його

вогнестійкості, планування, систем протипожежного захисту та евакуації. Приміщення, у яких розташовані персональні комп'ютери та офісне обладнання, належать переважно до категорії В (пожежонебезпечні), оскільки містять тверді горючі матеріали – корпуси техніки, кабельну ізоляцію, меблі, паперові документи. Основними причинами виникнення пожеж у таких приміщеннях є несправність електропроводки та електроустановок, коротке замикання, перевантаження електромережі, порушення правил експлуатації обладнання та необережне поводження з вогнем [41, 44].

Заходи щодо запобігання поширенню пожежі поділяють на об'ємно-планувальні, конструктивні, технічні та організаційні. Об'ємно-планувальні та конструктивні рішення передбачають поділ будівлі на протипожежні відсіки за допомогою протипожежних перешкод – стін, перегородок, перекриттів і протипожежних дверей, виконаних із негорючих матеріалів з нормованою межею вогнестійкості. Такі перешкоди перешкоджають переходу вогню та продуктів горіння з одного приміщення до іншого, локалізуючи осередок пожежі в межах одного відсіку. Важливе значення має також застосування будівельних конструкцій і оздоблювальних матеріалів з відповідним ступенем вогнестійкості та класом пожежної небезпеки, а на території підприємства – дотримання протипожежних розривів між будівлями [44].

Технічні заходи передбачають обладнання приміщень системами автоматичної пожежної сигналізації, які за допомогою сповіщувачів реагують на появу диму, підвищення температури або відкритого полум'я й передають сигнал на приймально-контрольний прилад. Спрацювання сигналізації забезпечує своєчасне виявлення осередку загоряння та автоматичне ввімкнення систем оповіщення й управління евакуацією людей. Для гасіння пожежі на ранній стадії застосовують автоматичні установки пожежогасіння (спринклерні та дренчерні водяні системи, газові, порошкові й аерозольні установки), вибір яких залежить від призначення приміщення та властивостей матеріалів. У приміщеннях з електронним обладнанням перевагу надають газовому пожежогасінню, яке не пошкоджує техніку. Обов'язковим є також обладнання

приміщень системами видалення диму та підпору повітря, що перешкоджають задимленню шляхів евакуації [43, 44].

Невід'ємною складовою протипожежного захисту є первинні засоби пожежогасіння – вогнегасники, пожежні крани, ящики з піском, пожежний інвентар, які розміщують у доступних і помітних місцях. Для приміщень з електроустановками та обчислювальною технікою застосовують вуглекислотні та порошкові вогнегасники, придатні для гасіння електрообладнання під напругою. Кожне приміщення має бути забезпечене розрахунковою кількістю вогнегасників відповідно до його площі та категорії за пожежною небезпекою. Працівники повинні знати розташування засобів пожежогасіння та вміти ними користуватися [43].

Особливе значення для безпеки людей під час пожежі мають шляхи евакуації та евакуаційні виходи. Вони повинні забезпечувати безперешкодний і своєчасний вихід усіх працівників із приміщення в безпечну зону. Кількість, розміри й розташування евакуаційних виходів визначаються нормативними вимогами залежно від площі приміщення та кількості людей. Шляхи евакуації не можна захаращувати, а двері на них повинні відчинятися в напрямку виходу. Будівлі обладнують евакуаційним освітленням та світловими покажчиками напрямку руху, а на видних місцях розміщують плани евакуації [43, 44].

Організаційні заходи з пожежної безпеки включають розроблення та затвердження інструкцій про заходи пожежної безпеки, призначення осіб, відповідальних за пожежну безпеку, проведення з працівниками вступного та періодичних протипожежних інструктажів і навчань, а також періодичних тренувань з евакуації. Важливим є дотримання протипожежного режиму: заборона куріння у невстановлених місцях, контроль за справністю електромережі та своєчасне технічне обслуговування обладнання й систем протипожежного захисту. Комплексне поєднання конструктивних, технічних та організаційних заходів дає змогу не лише запобігти виникненню пожежі в цеху чи приміщенні, а й суттєво обмежити її поширення, забезпечивши безпеку працівників та збереження матеріальних цінностей [42, 43].

ВИСНОВКИ

У межах кваліфікаційної роботи розроблено веб-застосунок для управління товарами на складі інтернет-магазину у вигляді плагіна для системи керування вмістом WordPress з інтеграцією платформи електронної комерції WooCommerce. Поставлену мету досягнуто, а всі визначені завдання виконано в повному обсязі.

Основні результати роботи полягають в наступно му:

1. Проаналізовано предметну область складського обліку інтернет-торгівлі, формалізовано основні бізнес-процеси руху товарів – надходження, зберігання, резервування, відвантаження та повернення – і визначено їх вплив на стан запасів. Це дозволило сформувати чітке уявлення про функції майбутнього застосунку.

2. Здійснено огляд та порівняльний аналіз наявних програмних рішень (ERP-систем, хмарних сервісів і плагінів електронної комерції), за результатами якого обґрунтовано доцільність розроблення власного плагіна для платформи WooCommerce як оптимального за співвідношенням вартості, контролю над даними та можливості доопрацювання рішення для малого й середнього бізнесу.

3. Обґрунтовано вибір технологічного стеку – HTML, CSS, JavaScript для клієнтської частини, PHP для серверної логіки та MySQL для зберігання даних, а також платформи WordPress і WooCommerce як основи застосунку.

4. Сформульовано функціональні та нефункціональні вимоги до застосунку, що стали основою для проєктування.

5. Спроєктовано трирівневу архітектуру застосунку, розроблено структуру бази даних із власними таблицями журналу руху товарів, поставок і налаштувань, побудовано ER-діаграму, діаграми прецедентів та послідовності, а також прототипи користувацького інтерфейсу.

6. Реалізовано застосунок у вигляді модульного плагіна WordPress, що складається з класів обліку залишків, руху товарів, поставок, обробки замовлень,

звітності та адміністрування. Завдяки використанню системи хуків досягнуто безшовної інтеграції з WooCommerce без модифікації його коду.

7. Розроблено модулі обліку руху товарів з автоматичним резервуванням і списанням за статусами замовлень, оформлення поставок, а також звітності з динамікою продажів, ABC-аналізом і контролем низького запасу. Реалізовано інтерактивний інтерфейс із асинхронною взаємодією та візуалізацією даних.

8. Проведено функціональне тестування застосунку, яке підтвердило коректність роботи всіх ключових функцій та відповідність поставленим вимогам.

Практичне значення одержаних результатів полягає в тому, що розроблений застосунок є готовим до впровадження інструментом, який дозволяє підприємствам малого та середнього бізнесу автоматизувати складський облік безпосередньо в середовищі свого інтернет-магазину. Це сприяє скороченню часу обробки замовлень, зменшенню кількості облікових помилок та підвищенню обґрунтованості управлінських рішень завдяки аналітичній звітності.

Перспективами подальшого розвитку застосунку є додавання підтримки кількох складів, інтеграція з вітчизняними службами доставки та платіжними системами, реалізація прогнозування попиту на основі історичних даних, друк складських документів і штрихкодів, а також розроблення мобільного інтерфейсу для працівників складу. Зокрема, для прогнозування попиту та аналізу динаміки продажів доцільно застосувати математичні моделі та методи опрацювання даних у формі циклічних випадкових процесів, що добре зарекомендували себе в задачах моделювання й прогнозування процесів споживання ресурсів [47, 48, 49, 50, 51]. Реалізована модульна архітектура створює належну основу для впровадження зазначених удосконалень.

ПЕРЕЛІК ДЖЕРЕЛ

1. Бойчук О. С. Електронна комерція : навч. посібник / О. С. Бойчук. – Київ : Кондор, 2019. – 320 с.

2. Гужва В. М. Інформаційні системи і технології на підприємствах : навч. посібник / В. М. Гужва. – Київ : КНЕУ, 2018. – 400 с.

3. Левицький С. І. Управління запасами підприємства : монографія / С. І. Левицький. – Львів : Магнолія, 2020. – 256 с.

4. Морзе Н. В. Основи інформаційних технологій : підручник / Н. В. Морзе. – Київ : Видавнича група BHV, 2017. – 352 с.

5. Пасічник В. В. Веб-технології та веб-дизайн : підручник / В. В. Пасічник, О. В. Пасічник, Д. І. Угрин. – Львів : Магнолія, 2018. – 336 с.

6. Глинський Я. М. Інтернет. Сервіси, HTML і веб-дизайн / Я. М. Глинський, В. А. Ряжська. – Львів : СПД Глинський, 2016. – 240 с.

7. Велінг Л. Розробка веб-застосунків за допомогою PHP і MySQL / Люк Велінг, Лаура Томсон ; пер. з англ. – 5-те вид. – Москва : Вільямс, 2017. – 768 с.

8. Ніксон Р. Створюємо динамічні веб-сайти за допомогою PHP, MySQL, JavaScript, CSS та HTML5 / Робін Ніксон ; пер. з англ. – Санкт-Петербург : Пітер, 2019. – 816 с.

9. Дакетт Дж. HTML і CSS. Розробка та дизайн веб-сайтів / Джон Дакетт ; пер. з англ. – Москва : Ексмо, 2017. – 480 с.

10. Фленаган Д. JavaScript. Детальний посібник / Девід Фленаган ; пер. з англ. – 7-ме вид. – Москва : Вільямс, 2021. – 720 с.

11. Хавербеке М. Виразний JavaScript / Марейн Хавербеке ; пер. з англ. – 3-тє вид. – Санкт-Петербург : Пітер, 2019. – 480 с.

12. Вільямс Б. Професійна розробка плагінів для WordPress / Бред Вільямс, Джастін Тедлок, Джон Джеймс Джекобі ; пер. з англ. – Москва : Діалектика, 2020. – 560 с.

13. Стерн Х. WordPress для професіоналів / Хел Стерн, Девід Дамстра, Бренден Вільямс ; пер. з англ. – Москва : Вільямс, 2018. – 464 с.

14. Дюбуа П. MySQL. Збірник рецептів / Поль Дюбуа ; пер. з англ. – Санкт-Петербург : Символ-Плюс, 2017. – 1056 с.

15. Бейлі Л. Вивчаємо PHP і MySQL / Лінн Бейлі, Майкл Моррісон ; пер. з англ. – Москва : Ексмо, 2018. – 768 с.

16. Фаулер М. UML. Основи / Мартін Фаулер ; пер. з англ. – 3-тє вид. – Санкт-Петербург : Символ-Плюс, 2016. – 192 с.

17. Мартін Р. Чистий код. Створення, аналіз і рефакторинг / Роберт Мартін ; пер. з англ. – Санкт-Петербург : Пітер, 2019. – 464 с.

18. Макконнелл С. Досконалий код. Майстер-клас / Стів Макконнелл ; пер. з англ. – Москва : Російська редакція, 2017. – 896 с.

19. WordPress Developer Resources : офіційна документація для розробників. – URL: https://developer.wordpress.org/ (дата звернення: 15.04.2025).

20. WordPress Plugin Handbook : посібник з розробки плагінів. – URL: https://developer.wordpress.org/plugins/ (дата звернення: 15.04.2025).

21. WooCommerce Documentation : офіційна документація платформи. – URL: https://woocommerce.com/documentation/ (дата звернення: 18.04.2025).

22. WooCommerce Code Reference : довідник з програмного інтерфейсу. – URL: https://woocommerce.github.io/code-reference/ (дата звернення: 18.04.2025).

23. PHP Manual : офіційна документація мови PHP. – URL: https://www.php.net/manual/uk/ (дата звернення: 20.04.2025).

24. MySQL 8.0 Reference Manual : офіційна документація MySQL. – URL: https://dev.mysql.com/doc/refman/8.0/en/ (дата звернення: 20.04.2025).

25. MDN Web Docs : документація з веб-технологій HTML, CSS, JavaScript. – URL: https://developer.mozilla.org/ (дата звернення: 22.04.2025).

26. Chart.js Documentation : документація бібліотеки побудови діаграм. – URL: https://www.chartjs.org/docs/latest/ (дата звернення: 25.04.2025).

27. jQuery API Documentation : довідник з бібліотеки jQuery. – URL: https://api.jquery.com/ (дата звернення: 25.04.2025).

28. WordPress Coding Standards : стандарти оформлення коду. – URL: https://developer.wordpress.org/coding-standards/ (дата звернення: 27.04.2025).

29. W3C HTML5 Specification : специфікація мови розмітки. – URL: https://www.w3.org/TR/html52/ (дата звернення: 28.04.2025).

30. W3Schools Online Web Tutorials : навчальні матеріали з веб-розробки. – URL: https://www.w3schools.com/ (дата звернення: 28.04.2025).

31. BuiltWith Technology Trends : статистика використання платформ електронної комерції. – URL: https://trends.builtwith.com/ (дата звернення: 30.04.2025).

32. ДСТУ 8302:2015. Інформація та документація. Бібліографічне посилання. Загальні положення та правила складання. – Київ : ДП «УкрНДНЦ», 2016. – 17 с.

33. ДСТУ 3008:2015. Інформація та документація. Звіти у сфері науки і техніки. Структура та правила оформлювання. – Київ : ДП «УкрНДНЦ», 2016. – 26 с.

34. Гайдаржи В. І. Бази даних та інформаційні системи : навч. посібник / В. І. Гайдаржи, О. А. Дацюк. – Київ : Політехніка, 2019. – 288 с.

35. Карпенко С. Г. Системи управління базами даних : навч. посібник / С. Г. Карпенко. – Київ : МАУП, 2018. – 192 с.

36. Желібо Є. П. Безпека життєдіяльності : навчальний посібник / Є. П. Желібо, Н. М. Заверуха, В. В. Зацарний. – Київ : Каравела, 2019. – 344 с.

37. Запорожець О. І. Безпека життєдіяльності : підручник / О. І. Запорожець, Б. Д. Халмурадов, В. І. Применко [та ін.]. – Київ : Центр учбової літератури, 2018. – 448 с.

38. Пістун І. П. Безпека життєдіяльності : навчальний посібник / І. П. Пістун, Ю. В. Кіт, А. П. Березовецький. – Суми : Університетська книга, 2017. – 301 с.

39. Крушельницька Я. В. Фізіологія і психологія праці : підручник / Я. В. Крушельницька. – Київ : КНЕУ, 2017. – 367 с.

40. Гандзюк М. П. Основи охорони праці : підручник / М. П. Гандзюк, Є. П. Желібо, М. О. Халімовський. – Київ : Каравела, 2018. – 384 с.

41. Жидецький В. Ц. Основи охорони праці : підручник / В. Ц. Жидецький. – Львів : Афіша, 2019. – 352 с.

42. Кодекс цивільного захисту України від 02.10.2012 № 5403-VI (зі змінами). – URL: https://zakon.rada.gov.ua/laws/show/5403-17 (дата звернення: 05.06.2025).

43. Правила пожежної безпеки в Україні : затв. наказом МВС України від 30.12.2014 № 1417. – URL: https://zakon.rada.gov.ua/laws/show/z0252-15 (дата звернення: 05.06.2025).

44. ДБН В.1.1-7:2016. Пожежна безпека об'єктів будівництва. Загальні вимоги. – Київ : Мінрегіон України, 2017. – 41 с.

45. Методичні вказівки для написання розділу „Безпека життєдіяльності, основи охорони праці" в кваліфікаційних роботах здобувачів освітнього рівня „бакалавр" / Укладачі: О. Я. Гурик, І. Б. Окіпний. – Тернопіль : ТНТУ імені Івана Пулюя, 2021. – 20 с.

46. Навчально-методичний посібник до практичних занять з дисципліни «Безпека життєдіяльності, основи охорони праці» для студентів освітнього ступеня „бакалавр" усіх спеціальностей та форм навчання / Укладачі: О. Я. Гурик, І. Б. Окіпний, В. С. Сенчишин, С. Ю. Мариненко, О. І. Король. – Тернопіль : ТНТУ імені Івана Пулюя, 2025. – 123 с.

47. Lytvynenko I. Mathematical model of gas consumption process in the form of cyclic random process / I. Lytvynenko, S. Lupenko, O. Nazarevych, H. Shymchuk, V. Hotovych // Proceedings of the 16th IEEE International Conference on Computer Sciences and Information Technologies (CSIT). – Lviv, 2021. – Vol. 1. – P. 232–235. DOI: 10.1109/CSIT52700.2021.9648621.

48. Scherbak L. Mathematical model of the energy resource consumption process in the form of a random process with piecewise homogeneous components / L. Scherbak, I. Lytvynenko, S. Kharchenko, O. Nazarevych, V. Hotovych // CEUR Workshop Proceedings. – 2022. – Vol. 3309. – P. 150–159. URL: http://ceur-ws.org/Vol-3309/paper11.pdf

49. Lupenko S. Concept of design, requirements and generalized architectures of components of the integrated onto-oriented information environment of simulation and processing of cyclic signals / S. Lupenko, I. Lytvynenko, V. Hotovych, A. Zozuli, N. Chizoba, O. Volyanyk // Вісник Тернопільського національного технічного університету. – 2021. – Т. 102, № 2. – С. 96–108. DOI: 10.33108/visnyk_tntu2021.02.147

50. Lytvynenko I. Additive mathematical model of gas consumption process / I. Lytvynenko, S. Lupenko, O. Nazarevych, H. Shymchuk, V. Hotovych // Вісник Тернопільського національного технічного університету. – 2022. – Т. 105, № 1. – С. 87–97. DOI: 10.33108/visnyk_tntu2021.04.087

51. Lytvynenko I. Simulation of gas consumption process based on the mathematical model in the form of cyclic random process / I. Lytvynenko, S. Lupenko, N. Kunanets, O. Nazarevych, H. Shymchuk, V. Hotovych // Proceedings of the IEEE International Conference on Advanced Computer Information Technologies (ACIT). – 2021. – P. 23–26. URL: http://ceur-ws.org/Vol-3039/paper23.pdf

ДОДАТКИ

Додаток А

Лістинг А.1 – Головний файл плагіна stock-manager.php

```
<?php
/**
 * Plugin Name:       Stock Manager – Управління товарами на складі
 * Plugin URI:        https://example.com/stock-manager
 * Description:       Облік, контроль та аналіз руху товарів на складі інтернет-
магазину на базі WooCommerce.
 * Version:           1.0.0
 * Author:            Студент спеціальності «Комп'ютерні науки»
 * License:           GPL-2.0+
 * Text Domain:       stock-manager
 * Requires PHP:      7.4
 * WC requires at least: 6.0
 *
 * Головний файл-завантажувач плагіна. Підключає класи, реєструє
 * хуки активації/деактивації та ініціалізує модулі застосунку.
 *
 * @package StockManager
 */


// Заборона прямого доступу до файлу.
if ( ! defined( 'ABSPATH' ) ) {
    exit;
}


// Константи плагіна.
define( 'SM_VERSION', '1.0.0' );
define( 'SM_PLUGIN_DIR', plugin_dir_path( __FILE__ ) );
define( 'SM_PLUGIN_URL', plugin_dir_url( __FILE__ ) );


// Підключення класів модулів.
require_once SM_PLUGIN_DIR . 'includes/class-activator.php';
require_once SM_PLUGIN_DIR . 'includes/class-stock.php';
require_once SM_PLUGIN_DIR . 'includes/class-movement.php';
require_once SM_PLUGIN_DIR . 'includes/class-supply.php';
require_once SM_PLUGIN_DIR . 'includes/class-orders.php';
require_once SM_PLUGIN_DIR . 'includes/class-reports.php';
require_once SM_PLUGIN_DIR . 'includes/class-admin.php';


/**
 * Перевірка наявності активованого WooCommerce.
 * Без нього плагін не має сенсу, тому показуємо повідомлення й зупиняємось.
 */
function sm_check_woocommerce() {
    if ( ! class_exists( 'WooCommerce' ) ) {
        add_action( 'admin_notices', function () {
            echo '<div class="notice notice-error"><p>'
                . esc_html__( 'Для роботи плагіна «Stock Manager» потрібен
активований WooCommerce.', 'stock-manager' )
                . '</p></div>';
        } );
        return false;
    }
    return true;
}
```

```
/**
 * Ініціалізація плагіна після завантаження всіх плагінів.
 */
function sm_init() {
    if ( ! sm_check_woocommerce() ) {
        return;
    }

    // Створення екземплярів модулів та реєстрація їх хуків.
    $orders  = new SM_Orders();
    $orders->register_hooks();

    if ( is_admin() ) {
        $admin = new SM_Admin();
        $admin->register_hooks();
    }
}
add_action( 'plugins_loaded', 'sm_init' );

// Реєстрація хуків активації та деактивації плагіна.
register_activation_hook( __FILE__, array( 'SM_Activator', 'activate' ) );
register_deactivation_hook( __FILE__, array( 'SM_Activator', 'deactivate' ) );
```

Лістинг А.2 – Клас активації class-activator.php

```
<?php
/**
 * Клас активації плагіна.
 *
 * Створює власні таблиці бази даних під час активації плагіна
 * та виконує початкові налаштування.
 *
 * @package StockManager
 */

if ( ! defined( 'ABSPATH' ) ) {
    exit;
}

class SM_Activator {

    /**
     * Дії під час активації плагіна: створення таблиць та опцій.
     */
    public static function activate() {
        self::create_tables();
        self::set_default_options();
        // Скидання правил перезапису URL (на випадок реєстрації кінцевих
точок).
        flush_rewrite_rules();
    }

    /**
     * Дії під час деактивації плагіна.
     * Таблиці навмисно не видаляються, щоб зберегти облікові дані.
     */
    public static function deactivate() {
        flush_rewrite_rules();
    }

    /**
     * Створення власних таблиць плагіна за допомогою dbDelta().
     */
    private static function create_tables() {
        global $wpdb;
        $charset_collate = $wpdb->get_charset_collate();
        require_once ABSPATH . 'wp-admin/includes/upgrade.php';

        $log_table    = $wpdb->prefix . 'sm_stock_log';
        $supply_table = $wpdb->prefix . 'sm_supply';
        $items_table  = $wpdb->prefix . 'sm_supply_items';

        // Таблиця журналу руху товарів.
        $sql_log = "CREATE TABLE {$log_table} (
            id BIGINT(20) UNSIGNED NOT NULL AUTO_INCREMENT,
            product_id BIGINT(20) UNSIGNED NOT NULL,
            type VARCHAR(20) NOT NULL,
            qty INT(11) NOT NULL DEFAULT 0,
            balance INT(11) NOT NULL DEFAULT 0,
            reference VARCHAR(100) DEFAULT NULL,
            user_id BIGINT(20) UNSIGNED DEFAULT NULL,
            note TEXT DEFAULT NULL,
            created_at DATETIME NOT NULL DEFAULT CURRENT_TIMESTAMP,
```

```
            PRIMARY KEY (id),
            KEY product_id (product_id),
            KEY type (type),
            KEY created_at (created_at)
        ) {$charset_collate};";

        // Таблиця поставок.
        $sql_supply = "CREATE TABLE {$supply_table} (
            id BIGINT(20) UNSIGNED NOT NULL AUTO_INCREMENT,
            supplier VARCHAR(255) NOT NULL,
            doc_number VARCHAR(100) DEFAULT NULL,
            total DECIMAL(12,2) NOT NULL DEFAULT 0,
            user_id BIGINT(20) UNSIGNED DEFAULT NULL,
            created_at DATETIME NOT NULL DEFAULT CURRENT_TIMESTAMP,
            PRIMARY KEY (id)
        ) {$charset_collate};";

        // Таблиця позицій поставки.
        $sql_items = "CREATE TABLE {$items_table} (
            id BIGINT(20) UNSIGNED NOT NULL AUTO_INCREMENT,
            supply_id BIGINT(20) UNSIGNED NOT NULL,
            product_id BIGINT(20) UNSIGNED NOT NULL,
            qty INT(11) NOT NULL DEFAULT 0,
            cost DECIMAL(12,2) NOT NULL DEFAULT 0,
            PRIMARY KEY (id),
            KEY supply_id (supply_id),
            KEY product_id (product_id)
        ) {$charset_collate};";

        dbDelta( $sql_log );
        dbDelta( $sql_supply );
        dbDelta( $sql_items );
    }

    /**
     * Встановлення налаштувань за замовчуванням.
     */
    private static function set_default_options() {
        if ( false === get_option( 'sm_settings' ) ) {
            add_option( 'sm_settings', array(
                'low_stock_threshold' => 5,
                'enable_notifications' => 1,
            ) );
        }
    }
}
```

Додаток Б

Лістинг Б.1 – Клас обліку залишків class-stock.php

```
<?php
/**
 * Клас обліку складських залишків.
 *
 * Інкапсулює логіку отримання фактичного, зарезервованого та
 * доступного залишку товарів. Фактичний залишок зберігається у
 * метаданих товару WooCommerce (_stock), резерв розраховується за
 * журналом руху товарів.
 *
 * @package StockManager
 */

if ( ! defined( 'ABSPATH' ) ) {
    exit;
}

class SM_Stock {

    /**
     * Повертає фактичний залишок товару зі сховища WooCommerce.
     *
     * @param int $product_id Ідентифікатор товару.
     * @return int Фактична кількість.
     */
    public static function get_physical_stock( $product_id ) {
        $product = wc_get_product( $product_id );
        if ( ! $product ) {
            return 0;
        }
        return (int) $product->get_stock_quantity();
    }

    /**
     * Повертає зарезервовану кількість товару.
     * Резерв – це сума активних резервів за журналом руху.
     *
     * @param int $product_id Ідентифікатор товару.
     * @return int Зарезервована кількість.
     */
    public static function get_reserved_stock( $product_id ) {
        global $wpdb;
        $table = $wpdb->prefix . 'sm_stock_log';
        // Сума резервів мінус зняті резерви для товару.
        $reserved = $wpdb->get_var( $wpdb->prepare(
            "SELECT COALESCE(SUM(CASE WHEN type = 'reserve' THEN qty
                    WHEN type = 'unreserve' THEN -qty ELSE 0 END), 0)
             FROM {$table} WHERE product_id = %d",
            $product_id
        ) );
        return max( 0, (int) $reserved );
    }

    /**
     * Повертає доступну до продажу кількість.
     *
     * @param int $product_id Ідентифікатор товару.
     * @return int Доступна кількість.
```

```
     */
    public static function get_available_stock( $product_id ) {
        $physical = self::get_physical_stock( $product_id );
        $reserved = self::get_reserved_stock( $product_id );
        return max( 0, $physical - $reserved );
    }

    /**
     * Перевіряє, чи є товар у статусі низького запасу.
     *
     * @param int $product_id Ідентифікатор товару.
     * @return bool
     */
    public static function is_low_stock( $product_id ) {
        $settings  = get_option( 'sm_settings', array() );
        $threshold = isset( $settings['low_stock_threshold'] ) ? (int)
$settings['low_stock_threshold'] : 5;
        return self::get_available_stock( $product_id ) <= $threshold;
    }

    /**
     * Змінює фактичний залишок товару на задану величину.
     *
     * @param int $product_id Ідентифікатор товару.
     * @param int $delta      Зміна кількості (може бути від'ємною).
     * @return int Новий залишок.
     */
    public static function adjust_physical_stock( $product_id, $delta ) {
        $product = wc_get_product( $product_id );
        if ( ! $product ) {
            return 0;
        }
        $current = (int) $product->get_stock_quantity();
        $new     = max( 0, $current + (int) $delta );
        $product->set_stock_quantity( $new );
        $product->save();
        return $new;
    }
}
```

Лістинг Б.2 – Клас руху товарів class-movement.php

```
<?php
/**
 * Клас руху товарів.
 *
 * Реєструє операції руху товарів у журналі sm_stock_log та
 * надає методи для отримання історії руху.
 *
 * @package StockManager
 */

if ( ! defined( 'ABSPATH' ) ) {
    exit;
}

class SM_Movement {

    /**
     * Реєструє операцію руху товару в журналі.
     *
     * @param int    $product_id Ідентифікатор товару.
     * @param string $type       Тип операції (in, out, reserve, unreserve,
return, adjust).
     * @param int    $qty        Кількість.
     * @param string $reference  Посилання на документ.
     * @param string $note       Коментар.
     * @return int|false Ідентифікатор запису або false у разі помилки.
     */
    public static function log( $product_id, $type, $qty, $reference = '', $note
= '' ) {
        global $wpdb;
        $table = $wpdb->prefix . 'sm_stock_log';

        // Поточний фактичний залишок для збереження стану після операції.
        $balance = SM_Stock::get_physical_stock( $product_id );

        $result = $wpdb->insert(
            $table,
            array(
                'product_id' => (int) $product_id,
                'type'       => sanitize_text_field( $type ),
                'qty'        => (int) $qty,
                'balance'    => (int) $balance,
                'reference'  => sanitize_text_field( $reference ),
                'user_id'    => get_current_user_id(),
                'note'       => sanitize_textarea_field( $note ),
                'created_at' => current_time( 'mysql' ),
            ),
            array( '%d', '%s', '%d', '%d', '%s', '%d', '%s', '%s' )
        );

        return $result ? $wpdb->insert_id : false;
    }

    /**
     * Повертає історію руху товару.
     *
     * @param int $product_id Ідентифікатор товару (0 – усі товари).
     * @param int $limit      Максимальна кількість записів.
```

```
     * @return array Масив записів.
     */
    public static function get_history( $product_id = 0, $limit = 50 ) {
        global $wpdb;
        $table = $wpdb->prefix . 'sm_stock_log';

        if ( $product_id ) {
            return $wpdb->get_results( $wpdb->prepare(
                "SELECT * FROM {$table} WHERE product_id = %d ORDER BY
created_at DESC LIMIT %d",
                $product_id,
                $limit
            ) );
        }

        return $wpdb->get_results( $wpdb->prepare(
            "SELECT * FROM {$table} ORDER BY created_at DESC LIMIT %d",
            $limit
        ) );
    }

    /**
     * Людиночитна назва типу операції.
     *
     * @param string $type Код типу.
     * @return string Назва українською.
     */
    public static function type_label( $type ) {
        $labels = array(
            'in'        => 'Надходження',
            'out'       => 'Списання',
            'reserve'   => 'Резервування',
            'unreserve' => 'Зняття резерву',
            'return'    => 'Повернення',
            'adjust'    => 'Коригування',
        );
        return isset( $labels[ $type ] ) ? $labels[ $type ] : $type;
    }
}
```

Лістинг Б.3 – Клас поставок class-supply.php

```
<?php
/**
 * Клас поставок товарів.
 *
 * Реалізує оприбуткування партій товарів від постачальників:
 * створює запис поставки, її позиції та збільшує залишки.
 *
 * @package StockManager
 */

if ( ! defined( 'ABSPATH' ) ) {
    exit;
}

class SM_Supply {

    /**
     * Створює поставку та оприбутковує товари.
     *
     * @param string $supplier   Постачальник.
     * @param string $doc_number Номер документа.
     * @param array  $items      Масив позицій: [ ['product_id'=>, 'qty'=>,
'cost'=>], ... ].
     * @return int|false Ідентифікатор поставки або false.
     */
    public static function create( $supplier, $doc_number, $items ) {
        global $wpdb;
        $supply_table = $wpdb->prefix . 'sm_supply';
        $items_table  = $wpdb->prefix . 'sm_supply_items';

        $total = 0.0;
        foreach ( $items as $it ) {
            $total += (float) $it['cost'] * (int) $it['qty'];
        }

        // Створення запису поставки.
        $ok = $wpdb->insert( $supply_table, array(
            'supplier'   => sanitize_text_field( $supplier ),
            'doc_number' => sanitize_text_field( $doc_number ),
            'total'      => $total,
            'user_id'    => get_current_user_id(),
            'created_at' => current_time( 'mysql' ),
        ), array( '%s', '%s', '%f', '%d', '%s' ) );

        if ( ! $ok ) {
            return false;
        }
        $supply_id = $wpdb->insert_id;

        // Додавання позицій та оприбуткування.
        foreach ( $items as $it ) {
            $product_id = (int) $it['product_id'];
            $qty        = (int) $it['qty'];
            $cost       = (float) $it['cost'];

            $wpdb->insert( $items_table, array(
                'supply_id'  => $supply_id,
                'product_id' => $product_id,
```

```
                'qty'        => $qty,
                'cost'       => $cost,
            ), array( '%d', '%d', '%d', '%f' ) );

            // Збільшення фактичного залишку та запис у журнал руху.
            SM_Stock::adjust_physical_stock( $product_id, $qty );
            SM_Movement::log( $product_id, 'in', $qty, 'supply#' . $supply_id,
                'Надходження від постачальника: ' . $supplier );
        }

        return $supply_id;
    }

    /**
     * Повертає список останніх поставок.
     *
     * @param int $limit Кількість записів.
     * @return array
     */
    public static function get_recent( $limit = 20 ) {
        global $wpdb;
        $table = $wpdb->prefix . 'sm_supply';
        return $wpdb->get_results( $wpdb->prepare(
            "SELECT * FROM {$table} ORDER BY created_at DESC LIMIT %d",
            $limit
        ) );
    }
}
```

Додаток В

Лістинг В.1 – Клас обробки замовлень class-orders.php

```
<?php
/**
 * Клас обробки замовлень.
 *
 * Підписується на хуки WooCommerce, пов'язані зі зміною статусів
 * замовлень, та виконує відповідні складські операції: резервування,
 * списання й повернення товарів.
 *
 * @package StockManager
 */

if ( ! defined( 'ABSPATH' ) ) {
    exit;
}

class SM_Orders {

    /**
     * Реєстрація хуків WooCommerce.
     */
    public function register_hooks() {
        // Нове замовлення – резервування товарів.
        add_action( 'woocommerce_checkout_order_processed', array( $this,
'reserve_order_items' ), 10, 1 );
        // Замовлення відправлено (статус «completed») – списання зі складу.
        add_action( 'woocommerce_order_status_completed', array( $this,
'ship_order_items' ), 10, 1 );
        // Замовлення скасовано – зняття резерву.
        add_action( 'woocommerce_order_status_cancelled', array( $this,
'release_order_items' ), 10, 1 );
        add_action( 'woocommerce_order_status_refunded', array( $this,
'return_order_items' ), 10, 1 );
    }

    /**
     * Резервує товари при оформленні замовлення.
     *
     * @param int $order_id Ідентифікатор замовлення.
     */
    public function reserve_order_items( $order_id ) {
        $order = wc_get_order( $order_id );
        if ( ! $order ) {
            return;
        }
        foreach ( $order->get_items() as $item ) {
            $product_id = $item->get_product_id();
            $qty        = $item->get_quantity();
            SM_Movement::log( $product_id, 'reserve', $qty, 'order#' .
$order_id,
                'Резервування під замовлення' );
        }
    }

    /**
     * Списує товари зі складу при відвантаженні.
     *
```

```
     * @param int $order_id Ідентифікатор замовлення.
     */
    public function ship_order_items( $order_id ) {
        $order = wc_get_order( $order_id );
        if ( ! $order ) {
            return;
        }
        foreach ( $order->get_items() as $item ) {
            $product_id = $item->get_product_id();
            $qty        = $item->get_quantity();
            // Знімаємо резерв і фактично списуємо товар.
            SM_Movement::log( $product_id, 'unreserve', $qty, 'order#' .
$order_id );
            SM_Stock::adjust_physical_stock( $product_id, -$qty );
            SM_Movement::log( $product_id, 'out', $qty, 'order#' . $order_id,
                'Відвантаження за замовленням' );
        }
    }

    /**
     * Знімає резерв при скасуванні замовлення.
     *
     * @param int $order_id Ідентифікатор замовлення.
     */
    public function release_order_items( $order_id ) {
        $order = wc_get_order( $order_id );
        if ( ! $order ) {
            return;
        }
        foreach ( $order->get_items() as $item ) {
            $product_id = $item->get_product_id();
            $qty        = $item->get_quantity();
            SM_Movement::log( $product_id, 'unreserve', $qty, 'order#' .
$order_id,
                'Скасування замовлення' );
        }
    }

    /**
     * Повертає товари на склад при поверненні коштів.
     *
     * @param int $order_id Ідентифікатор замовлення.
     */
    public function return_order_items( $order_id ) {
        $order = wc_get_order( $order_id );
        if ( ! $order ) {
            return;
        }
        foreach ( $order->get_items() as $item ) {
            $product_id = $item->get_product_id();
            $qty        = $item->get_quantity();
            SM_Stock::adjust_physical_stock( $product_id, $qty );
            SM_Movement::log( $product_id, 'return', $qty, 'order#' . $order_id,
                'Повернення товару' );
        }
    }
}
```

Лістинг В.2 – Клас звітності class-reports.php

```
<?php
/**
```

```
 * Клас звітності.
 *
 * Формує аналітичні дані: продажі за період, рух товарів,
 * товари з низьким запасом, ABC-аналіз.
 *
 * @package StockManager
 */

if ( ! defined( 'ABSPATH' ) ) {
    exit;
}

class SM_Reports {

    /**
     * Динаміка продажів за період (сума замовлень за днями).
     *
     * @param string $date_from Дата початку (Y-m-d).
     * @param string $date_to   Дата кінця (Y-m-d).
     * @return array Масив [ ['date'=>, 'total'=>], ... ].
     */
    public static function sales_by_day( $date_from, $date_to ) {
        $orders = wc_get_orders( array(
            'status'       => array( 'wc-completed', 'wc-processing' ),
            'date_created' => $date_from . '...' . $date_to,
            'limit'        => -1,
        ) );

        $result = array();
        foreach ( $orders as $order ) {
            $day = $order->get_date_created()->date( 'Y-m-d' );
            if ( ! isset( $result[ $day ] ) ) {
                $result[ $day ] = 0;
            }
            $result[ $day ] += (float) $order->get_total();
        }
        ksort( $result );

        $out = array();
        foreach ( $result as $day => $total ) {
            $out[] = array( 'date' => $day, 'total' => round( $total, 2 ) );
        }
        return $out;
    }

    /**
     * Рух товарів за період, згрупований за типом операції.
     *
     * @param string $date_from Дата початку.
     * @param string $date_to   Дата кінця.
     * @return array
     */
    public static function movements_summary( $date_from, $date_to ) {
        global $wpdb;
        $table = $wpdb->prefix . 'sm_stock_log';
        return $wpdb->get_results( $wpdb->prepare(
            "SELECT type, SUM(qty) AS total_qty, COUNT(*) AS operations
             FROM {$table}
             WHERE created_at BETWEEN %s AND %s
             GROUP BY type",
            $date_from . ' 00:00:00',
```

```
            $date_to . ' 23:59:59'
        ) );
    }

    /**
     * Перелік товарів з низьким запасом.
     *
     * @return array Масив товарів.
     */
    public static function low_stock_products() {
        $products = wc_get_products( array(
            'limit'        => -1,
            'stock_status' => 'instock',
        ) );

        $result = array();
        foreach ( $products as $product ) {
            $id = $product->get_id();
            if ( SM_Stock::is_low_stock( $id ) ) {
                $result[] = array(
                    'id'        => $id,
                    'name'      => $product->get_name(),
                    'sku'       => $product->get_sku(),
                    'available' => SM_Stock::get_available_stock( $id ),
                );
            }
        }
        return $result;
    }

    /**
     * ABC-аналіз товарів за виручкою.
     * Групи: A – до 80% виручки, B – наступні 15%, C – решта 5%.
     *
     * @param string $date_from Дата початку.
     * @param string $date_to   Дата кінця.
     * @return array
     */
    public static function abc_analysis( $date_from, $date_to ) {
        $orders = wc_get_orders( array(
            'status'       => array( 'wc-completed' ),
            'date_created' => $date_from . '...' . $date_to,
            'limit'        => -1,
        ) );

        $revenue = array();
        foreach ( $orders as $order ) {
            foreach ( $order->get_items() as $item ) {
                $pid = $item->get_product_id();
                if ( ! isset( $revenue[ $pid ] ) ) {
                    $revenue[ $pid ] = array( 'name' => $item->get_name(), 'sum'
=> 0 );
                }
                $revenue[ $pid ]['sum'] += (float) $item->get_total();
            }
        }

        // Сортування за спаданням виручки.
        uasort( $revenue, function ( $a, $b ) {
            return $b['sum'] <=> $a['sum'];
        } );
```

```
        $total = array_sum( array_column( $revenue, 'sum' ) );
        $cumulative = 0;
        $result = array();
        foreach ( $revenue as $pid => $data ) {
            $cumulative += $data['sum'];
            $share = $total > 0 ? $cumulative / $total : 0;
            $group = $share <= 0.8 ? 'A' : ( $share <= 0.95 ? 'B' : 'C' );
            $result[] = array(
                'product_id' => $pid,
                'name'       => $data['name'],
                'revenue'    => round( $data['sum'], 2 ),
                'group'      => $group,
            );
        }
        return $result;
    }
}
```

Лістинг В.3 – Клас адміністрування class-admin.php

```
<?php
/**
 * Клас адміністративного інтерфейсу.
 *
 * Реєструє пункти меню, підключає стилі та сценарії, обробляє
 * AJAX-запити та виводить сторінки плагіна.
 *
 * @package StockManager
 */

if ( ! defined( 'ABSPATH' ) ) {
    exit;
}

class SM_Admin {

    /**
     * Реєстрація хуків адміністративної частини.
     */
    public function register_hooks() {
        add_action( 'admin_menu', array( $this, 'add_menu' ) );
        add_action( 'admin_enqueue_scripts', array( $this, 'enqueue_assets' ) );
        // AJAX-обробники.
        add_action( 'wp_ajax_sm_add_supply', array( $this, 'ajax_add_supply' )
);
        add_action( 'wp_ajax_sm_sales_data', array( $this, 'ajax_sales_data' )
);
    }

    /**
     * Додавання пунктів меню «Склад» до адміністративної панелі.
     */
    public function add_menu() {
        add_menu_page(
            'Управління складом',
            'Склад',
            'manage_woocommerce',
            'sm-dashboard',
            array( $this, 'render_dashboard' ),
            'dashicons-archive',
            56
        );
        add_submenu_page( 'sm-dashboard', 'Панель', 'Панель',
'manage_woocommerce', 'sm-dashboard', array( $this, 'render_dashboard' ) );
        add_submenu_page( 'sm-dashboard', 'Залишки товарів', 'Залишки',
'manage_woocommerce', 'sm-stock', array( $this, 'render_stock' ) );
        add_submenu_page( 'sm-dashboard', 'Рух товарів', 'Рух товарів',
'manage_woocommerce', 'sm-movements', array( $this, 'render_movements' ) );
        add_submenu_page( 'sm-dashboard', 'Поставки', 'Поставки',
'manage_woocommerce', 'sm-supply', array( $this, 'render_supply' ) );
        add_submenu_page( 'sm-dashboard', 'Звіти', 'Звіти',
'manage_woocommerce', 'sm-reports', array( $this, 'render_reports' ) );
    }

    /**
     * Підключення CSS та JS лише на сторінках плагіна.
     *
     * @param string $hook Поточна сторінка.
     */
```

```
    public function enqueue_assets( $hook ) {
        if ( strpos( $hook, 'sm-' ) === false ) {
            return;
        }
        wp_enqueue_style( 'sm-admin', SM_PLUGIN_URL . 'assets/css/admin.css',
array(), SM_VERSION );
        // Бібліотека діаграм Chart.js (CDN).
        wp_enqueue_script( 'chartjs', 'https://cdn.jsdelivr.net/npm/chart.js',
array(), '4.4.0', true );
        wp_enqueue_script( 'sm-admin', SM_PLUGIN_URL . 'assets/js/admin.js',
array( 'jquery', 'chartjs' ), SM_VERSION, true );
        // Передавання параметрів та nonce у JavaScript.
        wp_localize_script( 'sm-admin', 'smData', array(
            'ajaxUrl' => admin_url( 'admin-ajax.php' ),
            'nonce'   => wp_create_nonce( 'sm_nonce' ),
        ) );
    }

    /* ---------- Сторінки ---------- */

    public function render_dashboard() {
        include SM_PLUGIN_DIR . 'templates/dashboard.php';
    }

    public function render_stock() {
        $products = wc_get_products( array( 'limit' => 100 ) );
        include SM_PLUGIN_DIR . 'templates/stock-list.php';
    }

    public function render_movements() {
        $history = SM_Movement::get_history( 0, 100 );
        include SM_PLUGIN_DIR . 'templates/movements.php';
    }

    public function render_supply() {
        $products = wc_get_products( array( 'limit' => 200 ) );
        $supplies = SM_Supply::get_recent( 20 );
        include SM_PLUGIN_DIR . 'templates/supply.php';
    }

    public function render_reports() {
        include SM_PLUGIN_DIR . 'templates/reports.php';
    }

    /* ---------- AJAX-обробники ---------- */

    /**
     * Додавання поставки через AJAX.
     */
    public function ajax_add_supply() {
        check_ajax_referer( 'sm_nonce', 'nonce' );
        if ( ! current_user_can( 'manage_woocommerce' ) ) {
            wp_send_json_error( 'Недостатньо прав' );
        }

        $supplier   = sanitize_text_field( $_POST['supplier'] ?? '' );
        $doc_number = sanitize_text_field( $_POST['doc_number'] ?? '' );
        $items      = isset( $_POST['items'] ) ? (array) $_POST['items'] :
array();
```

```
        $clean = array();
        foreach ( $items as $it ) {
            $clean[] = array(
                'product_id' => (int) ( $it['product_id'] ?? 0 ),
                'qty'        => (int) ( $it['qty'] ?? 0 ),
                'cost'       => (float) ( $it['cost'] ?? 0 ),
            );
        }

        $supply_id = SM_Supply::create( $supplier, $doc_number, $clean );
        if ( $supply_id ) {
            wp_send_json_success( array( 'supply_id' => $supply_id ) );
        }
        wp_send_json_error( 'Помилка створення поставки' );
    }

    /**
     * Дані продажів для діаграми через AJAX.
     */
    public function ajax_sales_data() {
        check_ajax_referer( 'sm_nonce', 'nonce' );
        $from = sanitize_text_field( $_POST['from'] ?? date( 'Y-m-d', strtotime(
'-7 days' ) ) );
        $to   = sanitize_text_field( $_POST['to'] ?? date( 'Y-m-d' ) );
        $data = SM_Reports::sales_by_day( $from, $to );
        wp_send_json_success( $data );
    }
}
```

Додаток Д

Лістинг Д.1 – Клієнтський сценарій admin.js

```
/**
 * Клієнтський сценарій плагіна Stock Manager.
 * Реалізує пошук у таблиці, побудову діаграми продажів та
 * додавання поставки через AJAX.
 */
(function ($) {
    'use strict';

    $(function () {

        /* --- Пошук у таблиці залишків --- */
        $('#sm-search').on('keyup', function () {
            var query = $(this).val().toLowerCase();
            $('.sm-table tbody tr').each(function () {
                var text = $(this).text().toLowerCase();
                $(this).toggle(text.indexOf(query) !== -1);
            });
        });

        /* --- Діаграма продажів на панелі --- */
        var canvas = document.getElementById('sm-sales-chart');
        if (canvas && typeof Chart !== 'undefined') {
            $.post(smData.ajaxUrl, {
                action: 'sm_sales_data',
                nonce: smData.nonce
            }, function (response) {
                if (!response.success) { return; }
                var labels = response.data.map(function (d) { return d.date; });
                var values = response.data.map(function (d) { return d.total;
});
                new Chart(canvas.getContext('2d'), {
                    type: 'bar',
                    data: {
                        labels: labels,
                        datasets: [{
                            label: 'Дохід, ₴',
                            data: values,
                            backgroundColor: '#3b6ea5'
                        }]
                    },
                    options: {
                        responsive: true,
                        plugins: { legend: { display: false } },
                        scales: { y: { beginAtZero: true } }
                    }
                });
            });
        }

        /* --- Додавання рядка позиції поставки --- */
        $('#sm-add-row').on('click', function (e) {
            e.preventDefault();
            var $firstRow = $('#sm-supply-items tbody tr:first');
            var $clone = $firstRow.clone();
            $clone.find('input').val(function (i, val) {
                return $(this).hasClass('sm-item-qty') ? 1 : 0;
            });
```

```
            $('#sm-supply-items tbody').append($clone);
        });

        /* --- Збереження поставки через AJAX --- */
        $('#sm-save-supply').on('click', function (e) {
            e.preventDefault();
            var items = [];
            $('#sm-supply-items tbody tr').each(function () {
                items.push({
                    product_id: $(this).find('.sm-item-product').val(),
                    qty: $(this).find('.sm-item-qty').val(),
                    cost: $(this).find('.sm-item-cost').val()
                });
            });

            $.post(smData.ajaxUrl, {
                action: 'sm_add_supply',
                nonce: smData.nonce,
                supplier: $('#sm-supplier').val(),
                doc_number: $('#sm-doc').val(),
                items: items
            }, function (response) {
                if (response.success) {
                    alert('Поставку оприбутковано (№ ' + response.data.supply_id
+ ')');
                    location.reload();
                } else {
                    alert('Помилка: ' + response.data);
                }
            });
        });

    });

})(jQuery);
```

Лістинг Д.2 – Стилі admin.css

```
/**
 * Стилі адміністративного інтерфейсу плагіна Stock Manager.
 */

.sm-wrap {
    max-width: 1100px;
}

/* Картки показників */
.sm-cards {
    display: flex;
    gap: 20px;
    margin: 20px 0;
    flex-wrap: wrap;
}

.sm-card {
    flex: 1 1 220px;
    background: #fff;
    border: 1px solid #ccd0d4;
    border-left-width: 4px;
    border-radius: 6px;
    padding: 18px 20px;
    display: flex;
    flex-direction: column;
    box-shadow: 0 1px 2px rgba(0, 0, 0, 0.05);
}

.sm-card-label {
    font-size: 13px;
    color: #646970;
}

.sm-card-value {
    font-size: 28px;
    font-weight: 700;
    margin-top: 6px;
}

.sm-card-blue   { border-left-color: #3b6ea5; }
.sm-card-green  { border-left-color: #2e8b57; }
.sm-card-orange { border-left-color: #c8902a; }

/* Панелі */
.sm-panel {
    background: #fff;
    border: 1px solid #ccd0d4;
    border-radius: 6px;
    padding: 16px 20px;
    margin: 20px 0;
}

.sm-panel h2 {
    margin-top: 0;
}

/* Таблиці */
```

```
.sm-table {
    width: 100%;
    margin-top: 12px;
}

.sm-search {
    width: 320px;
    max-width: 100%;
    padding: 6px 10px;
    margin: 12px 0;
}

/* Бейджі статусу */
.sm-badge {
    display: inline-block;
    padding: 2px 10px;
    border-radius: 12px;
    font-size: 12px;
    color: #fff;
}

.sm-badge-green  { background: #2e8b57; }
.sm-badge-orange { background: #c8902a; }
.sm-badge-red    { background: #c0392b; }

/* Адаптивність */
@media (max-width: 782px) {
    .sm-cards {
        flex-direction: column;
    }
}
```

Лістинг Д.3 – SQL-схема таблиць schema.sql

```
-- ============================================================
-- Stock Manager – схема власних таблиць плагіна
-- СКБД: MySQL 5.7+ / MariaDB 10.3+
-- Префікс wp_ слід замінити на фактичний префікс таблиць WordPress.
-- ============================================================

-- Журнал руху товарів: фіксує кожну операцію зміни залишку.
CREATE TABLE IF NOT EXISTS `wp_sm_stock_log` (
    `id`         BIGINT(20) UNSIGNED NOT NULL AUTO_INCREMENT,
    `product_id` BIGINT(20) UNSIGNED NOT NULL,
    `type`       VARCHAR(20) NOT NULL,
    `qty`        INT(11) NOT NULL DEFAULT 0,
    `balance`    INT(11) NOT NULL DEFAULT 0,
    `reference`  VARCHAR(100) DEFAULT NULL,
    `user_id`    BIGINT(20) UNSIGNED DEFAULT NULL,
    `note`       TEXT DEFAULT NULL,
    `created_at` DATETIME NOT NULL DEFAULT CURRENT_TIMESTAMP,
    PRIMARY KEY (`id`),
    KEY `product_id` (`product_id`),
    KEY `type` (`type`),
    KEY `created_at` (`created_at`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_ci;

-- Поставки товарів від постачальників.
CREATE TABLE IF NOT EXISTS `wp_sm_supply` (
    `id`         BIGINT(20) UNSIGNED NOT NULL AUTO_INCREMENT,
    `supplier`   VARCHAR(255) NOT NULL,
    `doc_number` VARCHAR(100) DEFAULT NULL,
    `total`      DECIMAL(12,2) NOT NULL DEFAULT 0.00,
    `user_id`    BIGINT(20) UNSIGNED DEFAULT NULL,
    `created_at` DATETIME NOT NULL DEFAULT CURRENT_TIMESTAMP,
    PRIMARY KEY (`id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_ci;

-- Позиції поставки (деталізація за товарами).
CREATE TABLE IF NOT EXISTS `wp_sm_supply_items` (
    `id`         BIGINT(20) UNSIGNED NOT NULL AUTO_INCREMENT,
    `supply_id`  BIGINT(20) UNSIGNED NOT NULL,
    `product_id` BIGINT(20) UNSIGNED NOT NULL,
    `qty`        INT(11) NOT NULL DEFAULT 0,
    `cost`       DECIMAL(12,2) NOT NULL DEFAULT 0.00,
    PRIMARY KEY (`id`),
    KEY `supply_id` (`supply_id`),
    KEY `product_id` (`product_id`),
    CONSTRAINT `fk_supply_items_supply`
        FOREIGN KEY (`supply_id`) REFERENCES `wp_sm_supply` (`id`)
        ON DELETE CASCADE
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_ci;
```