НАЦІОНАЛЬНИЙ УНІВЕРСТИТЕТ БІОРЕСУРСІВ І ПРИРОДОКОРИСТУВАННЯ УКРАЇНИ

ФАКУЛЬТЕТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

ПОГОДЖЕНО **Декан факультету**	**ДОПУСКАЄТЬСЯ ДО ЗАХИСТУ** **Завідувач кафедри інформаційних систем і технологій**
_____ **Ігор БОЛБОТ** (підпис)	_____ **Михайло ШВИДЕНКО** (підпис)
«___» _____ 2026 р.	«___» _____ 2026 р.

БАКАЛАВРСЬКА КВАЛІФІКАЦІЙНА РОБОТА

на тему: «Розробка web орієнтованої інформаційної системи управління малим торговельним підприємством»

Спеціальність «**Інформаційні системи і технології»**

Освітньо-професійна програма **«Інформаційні системи і технології»**

Гарант освітньої програми

к.е.н., доцент _____ **Максим Мокрієв**

науковий ступінь та вчене звання (підпис)

Керівник бакалаврської кваліфікаційної роботи (проєкту) _____ **Вікторія Смолій**

професор, доктор технічних наук

науковий ступінь та вчене звання (підпис)

Виконала _____ **Вікторія Єсіп**

(підпис)

Київ – 2026

НАЦІОНАЛЬНИЙ УНІВЕРСТИТЕТ БІОРЕСУРСІВ І
ПРИРОДОКОРИСТУВАННЯ УКРАЇНИ
ФАКУЛЬТЕТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

ЗАТВЕРДЖУЮ
Завідувач кафедри
Інформаційних систем і технології
доцент,к.е.н, _____**Михайло ШВИДЕНКО**
“_____” ____2026р.

З А В Д А Н Н Я
ДО ВИКОНАННЯ БАКАЛАВРСЬКОЇ КВАЛІФІКАЦІЙНОЇ РОБОТИ
ЗДОБУВАЧКИ
Єсіп Вікторії Іванівни

Спеціальність «126 Інформаційні системи і технології»
Освітньо-професійна програма «Інформаційні системи і технології»
Тема бакалаврської кваліфікаційної роботи
«Розробка web орієнтованої інформаційної системи управління малим торговельним підприємством»
затверджена наказом від «19» грудня 2025 року № 3205«С»
Термін подання завершеної роботи на кафедру « » червня 2026 року
Вихідні дані до бакалаврської кваліфікаційної роботи:
Процеси управління малим торговельним підприємством, включаючи облік товарів, складські операції, продаж товарів, роботу з клієнтами та постачальниками; дані про номенклатуру товарів, залишки на складі, ціни, замовлення та фінансові показники; бізнес-процеси закупівлі, реалізації та аналітики продажів; вимоги до web-орієнтованих інформаційних систем; сучасні веб-технології (HTML, CSS, JavaScript, Node.js, Express.js, PostgreSQL, MySQL); існуючі програмні рішення для автоматизації торгівлі та управління малим бізнесом; інструменти візуалізації даних та формування звітності.
Перелік питань, що підлягають дослідженню:
аналіз предметної області управління малим торговельним підприємством та існуючих веб-систем автоматизації торгівлі; визначення вимог до системи; розробка структури бази даних та механізмів обробки транзакцій; проєктування архітектури та реалізація основних модулів вебзастосунку; розробка людино-машинного інтерфейсу; тестування системи; оцінювання ефективності рішення та підготовка рекомендацій щодо його впровадження.

Дата видачі завдання «19» грудня 2025 р.

Керівник бакалаврської кваліфікаційної роботи _____ (підпис) **Вікторія СМОЛІЙ**

Завдання взяв до виконання _____ (підпис) **Вікторія ЄСІП**

Календарний план

№ з/п	Назва етапів виконання бакалаврської кваліфікаційної роботи	Строк виконання етапів бакалаврської кваліфікаційної роботи	Примітка
1	Дослідження предметної області та постановка задачі	Листопад 2025 - Січень 2026	Включає аналіз предметної області, існуючих ІТ, визначення вимог та розробку технічного завдання (розділ 1).
2	Проектування інструментальних засобів створення та використання інформаційних систем і технологій	Січень 2026 – Березень 2026	Включає визначення проектних рішень, розробку логічної/фізичної структури, декомпозицію, розробку ЛМІ (розділ 2).
3	Реалізація та тестування розробленої інформаційної системи	Березень 2026 - Травень 2026	Включає опис модулів, розробку ЛМІ, визначення технічних характеристик, тестування системи (розділ 3).
4	Перезахист	1 червня	Демонстрація сайту
5	Захист дипломної роботи	10 червня	

Керівник бакалаврської кваліфікаційної роботи _____ (підпис) **Вікторія СМОЛІЙ**

Завдання взяв до виконання _____ (підпис) **Вікторія ЄСІП**

Зміст

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

ACID - Atomicity, Consistency, Isolation, Durability - атомарність, узгодженість, ізольованість, довговічність транзакцій

API - Application Programming Interface - інтерфейс програмування застосунків

БД - база даних

CRUD - Create, Read, Update, Delete - створення, читання, оновлення, видалення даних

CSS / CSS3 - Cascading Style Sheets - каскадні таблиці стилів

DOM - Document Object Model - об'єктна модель документа

ER - Entity-Relationship - «сутність-зв'язок» (тип діаграм для моделювання БД)

ES6 - ECMAScript 6 - специфікація мови JavaScript шостої версії (2015)

FK - Foreign Key - зовнішній ключ у реляційній базі даних

HTTP - HyperText Transfer Protocol - протокол передачі гіпертексту

HTML / HTML5 - HyperText Markup Language - мова розмітки гіпертексту

IEC - International Electrotechnical Commission - Міжнародна електротехнічна комісія

ІС - інформаційна система

ISO - International Organization for Standardization - Міжнародна організація зі стандартизації

ІТ - інформаційні технології

JSON - JavaScript Object Notation - текстовий формат обміну даними

KPI - Key Performance Indicators - ключові показники ефективності

ЛМІ - людино-машинний інтерфейс

MySQL - My Structured Query Language - система керування реляційними базами даних

PHP - PHP: Hypertext Preprocessor - мова серверного програмування для веб

PostgreSQL - Post Structured Query Language - об'єктно-реляційна СУБД

REST - Representational State Transfer - архітектурний стиль побудови веб-сервісів

SaaS - Software as a Service - програмне забезпечення як послуга

SKU - Stock Keeping Unit - артикул (одиниця складського обліку)

SPA - Single Page Application - односторінковий застосунок

SQL - Structured Query Language - мова структурованих запитів

СУБД - система управління базами даних

SVG - Scalable Vector Graphics - масштабована векторна графіка

ТЗ - технічне завдання

UI - User Interface - інтерфейс користувача

URL - Uniform Resource Locator - уніфікований локатор ресурсів

UX - User Experience - досвід користувача

ВСТУП

У світі, що швидко змінюється, компанії все частіше мають пристосовуватися до нових умов. Ринок росте, тому потрібно більше уваги приділяти тому, як працює бізнес і що чекають клієнти. Лише ті, хто швидко реагує на зміни, можуть залишати конкурентами. Техніка розвивається дуже швидко, з'являються нові шляхи для роботи, але і нові труднощі, які треба вчасно вирішувати. [12] Щоб добре діяти, підприємства мають думати не лише про сьогоднішній день, а й про майбутнє. Потрібно правильно управляти, використовувати те, що є під рукою, і ухвалювати продумані рішення.

Однак, багато фірм стикаються з тим, що різні відділи не завжди добре розуміють одне одного, процеси в середині компанії не працюють злагоджено, інформація іноді губиться, і на ринок реагують повільно. Через це важливо переглянути, як організована робота, і знайти нові шляхи розвитку. Зараз необхідність змін настільки значна, що потрібно розглянути усе комплексно – від процесів всередині до зовнішніх зв'язків.

Саме тому тема цієї роботи має особливу вагу. Слід знайти прості і чіткі підходи, які зроблять ведення справ більш гнучким, допоможуть адаптуватися і тримати стабільний темп розвитку. Нові технології роботи, моделювання можливих дій і оцінка ризиків стають основою для того, щоб компанія впевнено рухалася вперед і робила сильні кроки. [1, 2]

Період невпевненої економіки змушує фірми задумуватися про стратегію і діяти розумно. Треба не лише оцінювати те, що є зараз, а й передбачати різні сценарії, щоб бути готовим до труднощів і швидко зреагувати. Тому у цьому дослідженні увагу приділяють і тому, що вже зроблено на підприємстві, і створенню нових рішень, які можуть привести до тривалої переваги над іншими.

Зараз основна потреба — змінити організацію роботи так, щоб вона дала змогу знаходити внутрішні резерви і підсилювати ефективність. [16] Це вимагає перевірки усіх етапів створення цінності: від фінансів до роботи з

людьми і те, як відділи підтримують зв’язок між собою та з клієнтами. Важливо враховувати не тільки гроші чи чисельні показники, але й такі речі, як культуру в компанії, швидкість впровадження новинок, і здатність змінювати стратегію у відповідь на виклики.

Підсумовуючи, дослідження в межах цієї роботи допоможе побачити слабкі місця в керуванні підприємством і запропонувати конкретні кроки для покращення. Дані та висновки допоможуть побудувати модель розвитку бізнесу, що буде стійкою, дасть змогу зміцнити свої позиції і рухатися вперед у непростих ринкових умовах. Будуть зроблені рекомендації, які дозволять краще працювати, швидше реагувати на зміни і залишати конкурентами. [1, 12, 16]

РОЗДІЛ 1. ДОСЛІДЖЕННЯ ПРЕДМЕТНОЇ ОБЛАСТІ ТА ПОСТАНОВКА ЗАДАЧІ

1.1 Аналіз предметної області, її структура та функціональна особливість

В сучасному інформаційному світі, де обсяги даних зростають щодня, ефективне управління діяльністю малого торговельного підприємства стає надзвичайно важливим. Веб-орієнтовані інформаційні системи, що допомагають автоматизувати облік товарів, продажі та роботу з клієнтами, стають тут у пригоді. Головне, щоб такі інструменти були доступними, функціональними та простими у використанні. Предметна область управління малим торговельним підприємством охоплює сукупність процесів і дій, пов'язаних із закупівлею, зберіганням та реалізацією товарів, веденням фінансового обліку та аналізом результатів діяльності. Особливістю малого бізнесу є обмеженість фінансових, кадрових і технічних ресурсів, що зумовлює необхідність раціональної організації бізнес-процесів та автоматизації ключових операцій. [16]

Організаційна структура такого підприємства зазвичай є спрощеною та може включати керівника, продавців, працівника складу та бухгалтера. У багатьох випадках одна особа виконує декілька функцій одночасно, що підвищує навантаження та потребує оперативного доступу до достовірної інформації. Основні бізнес-процеси охоплюють управління закупівлями, облік товарних запасів, контроль продажів, формування цін, ведення фінансового обліку та аналіз результатів діяльності. [16]

Функціональна модель системи для керування підприємством має кілька важливих компонентів. По-перше, є модуль для обліку товарів і роботи зі складом. Він дає змогу дивитися, скільки товару залишилось, та вести справи з тими, хто постачає товари. Далі - модуль для керування продажами. Тут реєструють усі покупки і стежать, як змінюються обсяги продажів за часом. Ще є частина, де можна працювати з клієнтами: зберігати їхні дані та все, що пов’язано з ними, бачити всю історію продажів і спілкування. Окремий модуль відповідає за підготовку звітів і аналіз. Він сам збирає

інформацію для оцінки, наскільки прибутковий той чи інший товар, і як швидко товари продаються. Останній модуль - це частина для керування користувачами; тут визначають, хто і що може робити у системі, а також працюють з обліковими записами

З технічної сторони предметна область має такі структурні особливості: необхідність централізованого зберігання даних у базі даних, підтримка багатокористувацького режиму роботи, розмежування прав доступу до інформації, забезпечення захисту та резервного копіювання даних, а також можливість масштабування системи відповідно до зростання підприємства.

Щоб бути справді корисною, система управління малим торговельним підприємством має бути не просто функціональною, але й легко масштабуватися, бути доступною з будь-якого пристрою та гнучкою у використанні. Тому, розробляючи такий веб-сайт, важливо зосередитися не тільки на базових облікових операціях, але й на таких аспектах, як пошук інформації, актуальність залишків у реальному часі та зручність прийняття управлінських рішень.

Отже, розробка web-орієнтованої інформаційної системи управління малим торговельним підприємством залишається актуальною, адже такі інструменти допомагають керівникам бути більш ефективними завдяки кращій організації всіх бізнес-процесів.

1.2 Аналіз наявних інформаційних технологій предметної області

В умовах цифрової трансформації бізнесу автоматизація управління діяльністю малого торговельного підприємства набуває особливої актуальності. Сучасні веб-орієнтовані інформаційні системи дозволяють оптимізувати внутрішні процеси, забезпечити централізоване зберігання даних, підвищити прозорість операцій та покращити координацію роботи персоналу. Використання таких інструментів сприяє зменшенню витрат часу

на рутинні операції, мінімізації людського фактору та підвищенню якості прийняття управлінських рішень.

Малі торговельні підприємства, як правило, мають обмежені фінансові та кадрові ресурси, що зумовлює потребу у простих, доступних та функціонально гнучких програмних рішеннях. При цьому інформаційна система повинна забезпечувати не лише облік завдань, а й підтримувати процеси контролю виконання, зберігання документації, взаємодію між співробітниками та адаптацію до змін бізнес-середовища. Саме тому доцільним є аналіз існуючих програмних продуктів з метою визначення їх можливостей та виявлення функціональних прогалин.

Для дослідження було обрано сучасні веб-сервіси організації роботи: Trello, Notion та Google Keep. [3] Дані системи широко використовуються у малому бізнесі та командній роботі завдяки зручному інтерфейсу та доступності. Водночас кожна з них реалізує власну концепцію організації інформації та управління процесами.

Trello базується на канбан-методології, що передбачає візуалізацію процесів у вигляді дошок, списків і карток. Такий підхід є зручним для відображення стану виконання завдань, розподілу відповідальності та контролю термінів. Notion, у свою чергу, пропонує гнучку систему сторінок і вбудованих баз даних, що дозволяє створювати структуровані інформаційні середовища для зберігання документації, планів та аналітичних матеріалів. Google Keep орієнтований на швидке створення нотаток і нагадувань, що робить його корисним для фіксації оперативної інформації, однак його функціональність є обмеженою для комплексного управління підприємством.

Порівняння функціональних можливостей зазначених систем наведено у таблиці 1.1. [3]

Таблиця 1.1

Порівняння функціональних можливостей існуючих систем

Критерій	**Trello**	**Notion**	**Google Keep**
Модель організації роботи	Канбан-дошки	Сторінки та бази даних	Нотатки
Підтримка вкладених файлів	Підтримується	Підтримується	Частково (зображення)
Наявність шаблонів	Підтримуються шаблони дошок	Гнучкі шалони сторінок	Не підтримуються
Спільна робота	Повноцінна командна взаємодія	Гнучке розмежування доступу	Обмежена
Локалізація українською мовою	Часткова	Часткова	Повна
Розмежування прав доступу	Базове	Розширене	Мінімальне

Як видно з таблиці, Trello забезпечує наочне управління процесами та зручний механізм командної взаємодії, проте має обмежені можливості щодо глибокої структуризації даних. Notion надає широкі інструменти для організації інформації та створення внутрішніх баз даних, однак його складність може ускладнювати впровадження для невеликих підприємств. Google Keep, незважаючи на простоту використання, не підтримує повноцінного розмежування прав доступу та структурованого управління бізнес-процесами.

Окрім функціональних характеристик, важливим є аналіз технологічних підходів до реалізації інформаційних систем. Більшість популярних сервісів функціонують за моделлю SaaS (Software as a Service)

[5], що передбачає використання хмарної інфраструктури та доступ через веб-браузер. Такий підхід забезпечує швидке розгортання та мінімальні витрати на технічну підтримку, проте водночас обмежує можливості глибокої кастомізації та повного контролю над даними.

Порівняння загальних технологічних підходів наведено у таблиці 1.2

Таблиця 1.2

Порівняння технологічних підходів до реалізації веб-систем

Критерій	Готові SaaS-рішення	Власна web-орієнтована система
Контроль над даними	Зберігання на сторонніх серверах	Повний контроль над базою даних
Гнучкість налаштування	Обмежена функціоналом сервісу	Повна адаптація під потреби бізнесу
Інтеграція з внутрішніми процесами	Часткова	Можлива повна інтеграція
Масштабованість	Залежить від тарифного плану	Залежить від серверної інфраструктури
Безпека	Стандартні механізми захисту	Налаштовується індивідуально

Таким чином, проведений аналіз засвідчив, що існуючі універсальні веб-сервіси не повністю враховують специфіку діяльності малого торговельного підприємства, зокрема потребу у гнучкому налаштуванні бізнес-процесів, інтеграції з внутрішніми системами та повному контролі над інформаційними ресурсами. Це обґрунтовує доцільність розробки власної web-орієнтованої інформаційної системи, яка буде адаптована до конкретних організаційних та управлінських завдань підприємства.

Для реалізації web-орієнтованої інформаційної системи управління малим торговельним підприємством критично важливим є вибір бази даних, яка забезпечує надійне зберігання інформації про завдання, клієнтів, продажі та інші операційні процеси. Основними критеріями при виборі є стабільність роботи, швидкість обробки запитів, масштабованість, підтримка транзакцій та інтеграція з веб-сервером і фронтенд-додатком.

Серед реляційних баз даних найбільш поширеними є **PostgreSQL** та **MySQL** (з використанням MySQL Workbench для моделювання та адміністрування). [4] [6] PostgreSQL відзначається високою надійністю, підтримкою складних SQL-запитів, тригерів, збережених процедур і розширеними механізмами контролю цілісності даних. MySQL забезпечує високу продуктивність, простоту інтеграції з веб-сайту та інтуїтивного зрозуміння системи адміністрування. Обидві бази підтримують ACID-транзакції, що є критично важливим для комерційних процесів малого підприємства, де кожна операція має бути точно зафіксована і незмінна.

Вибір між PostgreSQL і MySQL залежить від конкретних вимог до системи: PostgreSQL краще підходить для складних бізнес-процесів та аналітики, MySQL - для легких або середніх за обсягом застосунків, де важлива швидкість розробки та простота інтеграції. Важливо також враховувати можливість горизонтального або вертикального масштабування, резервного копіювання та підтримку сучасних версій SQL.

Для наочності проведено порівняльний аналіз баз даних за ключовими характеристиками, що важливі для реалізації веб-системи:

Таблиця 1.3

Порівняльний аналіз реляційних баз даних

Критерій	**PostgreSQL**	**MySQL (MySQL Workbench)**
Тип бази даних	Реляційна	Реляційна
Структура даних	Таблиці з чітко визначеною схемою	Таблиці з чітко визначеною схемою

Продовження таблиці 1.3

Критерій	**PostgreSQL**	**MySQL (MySQL Workbench)**
Масштабованість	Вертикальна та частково горизонтальна	Вертикальна та частково горизонтальна
Швидкість обробки	Висока для складних запитів	Висока для простих та середніх запитів
Підтримка транзакцій	Повна (ACID)	Повна (ACID)
Гнучкість структури	Середня, зміни в схемі потребують міграцій	Середня, зміни в схемі потребують міграцій
Інтеграція з веб-серверами	Висока, підтримка Node.js, PHP, Python	Висока, підтримка Node.js, PHP, Python
Продуктивність	Висока при складних операціях і аналітиці	Висока при обробці великої кількості простих запитів
Адміністрування	Складніше, але потужні можливості адміністрування	Просте та інтуїтивне адміністрування через MySQL Workbench
Підтримка транзакцій	Повна (ACID)	Повна (ACID)

Проведений аналіз свідчить, що обидві системи є надійними і достатньо продуктивними для потреб малого торговельного підприємства. Вибір конкретної бази даних слід робити з урахуванням вимог до складності бізнес-процесів, обсягу даних і необхідності подальшого розвитку веб-системи. PostgreSQL доцільно застосовувати у разі, коли планується виконувати складні аналітичні запити та забезпечувати високий рівень

контролю цілісності даних, тоді як MySQL більше підходить для швидкого розгортання та стандартного ведення бізнес-процесів.

Таким чином, обидві бази даних дозволяють створити надійну і стабільну платформу для web-орієнтованої інформаційної системи управління підприємством, що стане фундаментом для інтеграції серверної частини, фронтенд-інтерфейсу та бізнес-логіки системи.

1.3 Визначення вимог до інструментальних засобів створення і використання інформаційних систем та технологій, розробка технічного завдання

Функціональні вимоги

Функціональні вимоги визначають основні можливості, які має забезпечувати web-орієнтована інформаційна система управління малим торговельним підприємством. Система повинна автоматизувати облік товарів, управління замовленнями, клієнтською базою та формування звітності, забезпечуючи зручну взаємодію користувача з усіма бізнес-процесами підприємства. [2, 9]

Автентифікація та авторизація користувачів

Автентифікація нового користувача передбачає введення імені, електронної пошти, пароля та визначення ролі (адміністратор, менеджер, продавець). Авторизація здійснюється за допомогою електронної пошти та пароля. Передбачено можливість зміни пароля авторизованим користувачем. Реалізовано механізм завершення сесії з безпечним виходом із системи. Система повинна підтримувати розмежування прав доступу відповідно до ролі користувача.

Управління товарами

Система забезпечує створення, редагування, видалення та перегляд інформації про товари. Для кожного товару передбачено збереження таких атрибутів: назва, категорія, артикул, ціна, кількість на складі, постачальник, опис, зображення. Реалізовано можливість зміни ціни та автоматичного

оновлення залишків після здійснення продажу. Передбачено пошук товарів за назвою, категорією або артикулом. Система підтримує фільтрацію товарів за наявністю на складі.

Управління замовленнями

Створення нового замовлення із вибором товарів та автоматичним розрахунком загальної суми. Редагування, перегляд і видалення замовлень. Зміна статусу замовлення (створено, підтверджено, виконано, скасовано). Автоматичне списання товарів зі складу після підтвердження продажу.

Управління клієнтами та постачальниками

Створення, редагування та перегляд даних клієнтів і постачальників. Збереження контактної інформації, історії замовлень та фінансових операцій. Пошук клієнтів за ім’ям або контактними даними.

Формування звітності

Генерація звітів щодо продажів за обраний період (день, тиждень, місяць). Формування звітів про залишки товарів на складі. Відображення фінансових показників діяльності підприємства. Можливість експорту звітів у файл для подальшого використання.

Прикріплення файлів

Додавання до товарів або замовлень супровідних документів (наприклад, накладних чи договорів) у форматах JPEG, PNG, PDF, DOC, DOCX. Перегляд і завантаження прикріплених файлів. Видалення або оновлення файлів за потреби.

Локалізація

Підтримка української та англійської мов інтерфейсу. Коректне форматування дат, часу та грошових значень відповідно до обраної мови. Можливість ручного вибору мови користувачем.

Тема інтерфейсу

Перемикання між світлою та темною темами оформлення. Збереження вибраної теми в локальному сховищі браузера для подальших сеансів роботи.

Для наочності взаємодії користувача з системою наведено діаграму прецедентів (Рис 1.1), яка відображає основні функціональні можливості інформаційної системи управління малим торговельним підприємством.

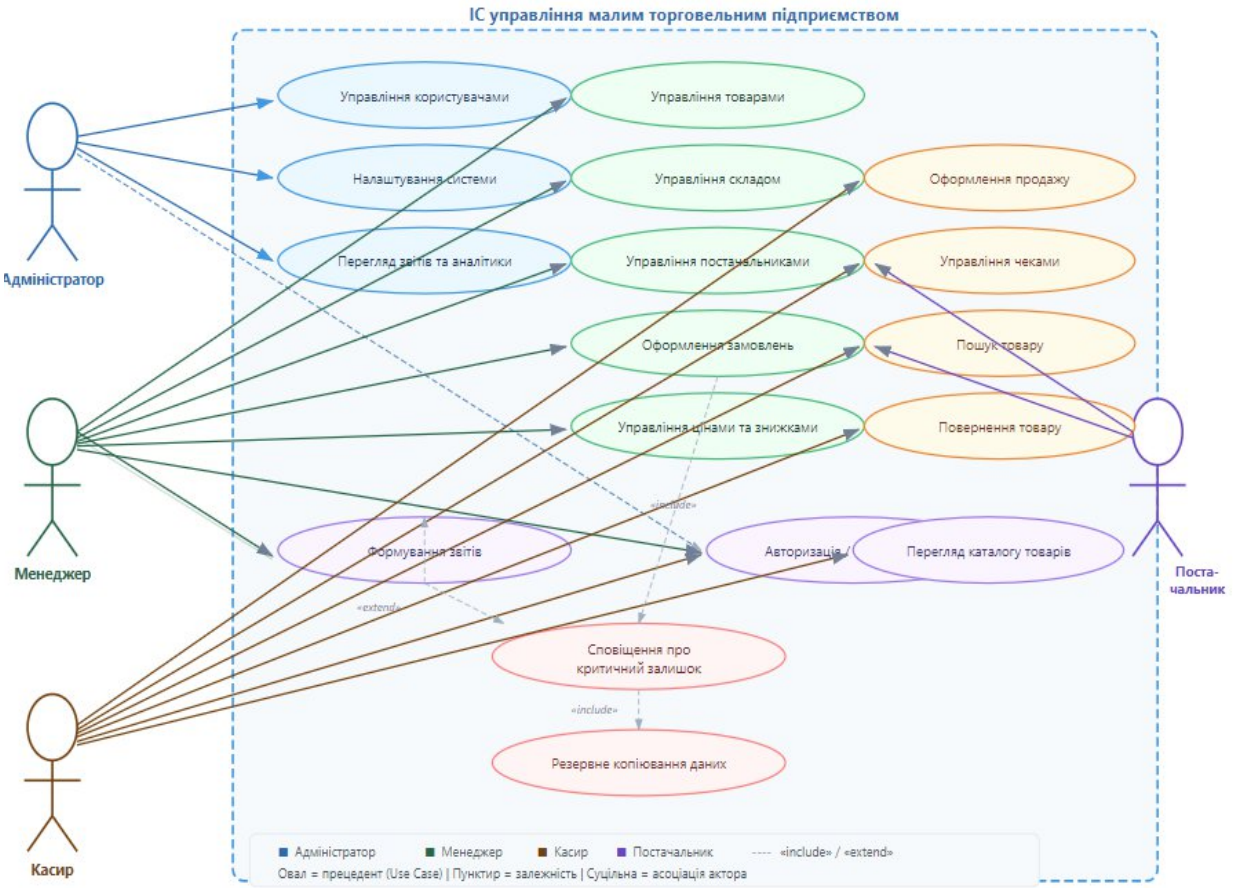


Рис 1.1 Діаграма прецедентів

Діаграма прецедентів демонструє взаємодію актора «Користувач» із системою. Залежно від ролі користувач може виконувати такі прецеденти: «Управління товарами», що включає створення, редагування, перегляд і видалення товарів, оновлення залишків та цін; «Обробка замовлень», що передбачає формування замовлень, зміну їх статусу та контроль продажів; «Управління контрагентами», що забезпечує роботу з даними клієнтів і постачальників; «Формування звітів», що дозволяє переглядати аналітичну інформацію про діяльність підприємства; а також «Керування обліковим записом», що охоплює реєстрацію, авторизацію, зміну пароля та вихід із системи.

Таким чином, система забезпечує комплексну автоматизацію ключових процесів малого торговельного підприємства, підвищуючи ефективність управління ресурсами та якість обслуговування клієнтів.

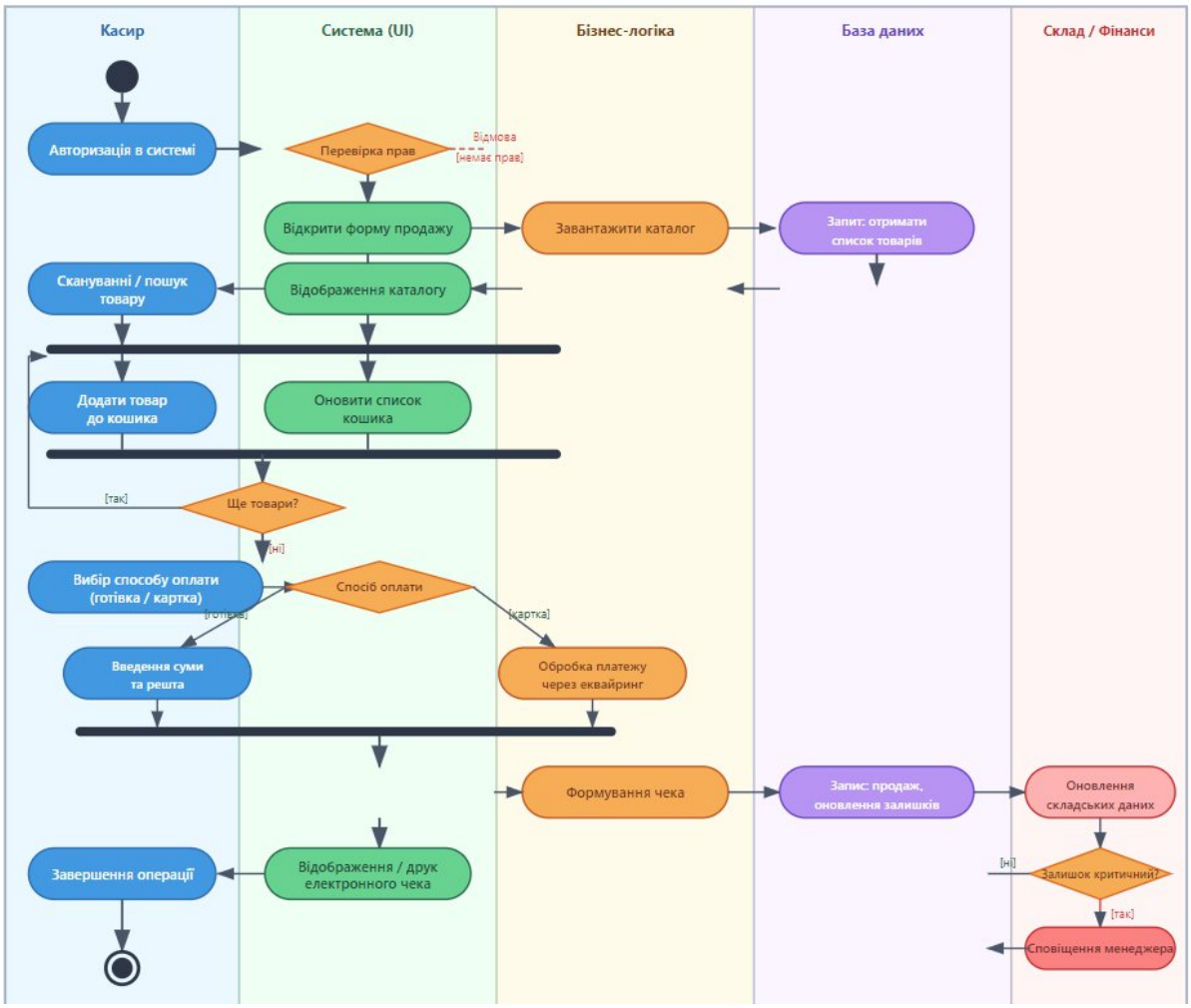


Рис 1.2 Діаграма активності процесу оформлення продажу

На (Рис. 1.2) зображено діаграму активності, яка демонструє повний цикл оформлення продажу в web-орієнтованій інформаційній системі управління малим торговельним підприємством.

Процес розпочинається з авторизації касира в системі. Після успішної перевірки облікових даних система відкриває форму продажу та завантажує актуальний каталог товарів із бази даних. Касир здійснює пошук товару шляхом сканування штрих-коду або ручного введення назви. Система відображає результати пошуку та дозволяє додати обраний товар до кошика.

Після кожного додавання система автоматично оновлює список товарів у кошику та перераховує загальну суму покупки. Якщо потрібно додати ще

товар, процес повторюється. Після завершення формування кошика касир обирає спосіб оплати - готівкою або банківською карткою.

У разі оплати карткою система передає дані до платіжного модуля для обробки транзакції. Якщо обрано оплату готівкою, касир вводить отриману суму, після чого система обчислює решту.

Після підтвердження оплати виконується формування електронного чека, запис інформації про продаж до бази даних та автоматичне оновлення складських залишків. Якщо залишок товару досягає критичного рівня, система генерує сповіщення для менеджера.

У разі виникнення помилки (наприклад, недостатня кількість товару або помилка транзакції) система повертає відповідне повідомлення користувачеві та пропонує повторити операцію або скоригувати дані.

Нефункціональні вимоги

Нефункціональні вимоги визначають якісні характеристики системи, що впливають на її стабільність, продуктивність, безпеку та зручність використання. [20]

Продуктивність

Час відгуку сервера на запит користувача не повинен перевищувати 2 секунди за стандартного навантаження. Система має підтримувати одночасну роботу до 100 користувачів без істотного зниження швидкодії, а SQL-запити для формування звітів та обробки транзакцій мають бути оптимізовані.

Безпека

Перевірка даних, які вводить користувач, робиться за допомогою простих атрибутів, як required, а ще дивляться, щоб типи полів були потрібними. Це допомагає не дати ввести щось не те. Друга перевірка, вже на рівні логіки роботи сайту, виконана у функції openAddOrder(). Там система дивиться, чи обрав користувач клієнта та потрібний товар, і тільки тоді можна зберегти нове замовлення. Дані ізольовані в сесії браузера і не виходять за межі роботи на сторінці - для цього є глобальний об'єкт state.

Якщо щось і зберігається в localStorage, то це лише те, що стосується теми сайту. В localStorage немає особистих чи важливих даних про людину або її покупки.

Масштабованість

1. Використання реляційної бази даних PostgreSQL для забезпечення цілісності транзакцій.
2. Можливість розширення функціоналу (інтеграція з бухгалтерськими системами, CRM тощо).
3. Підтримка зростання обсягу даних без втрати продуктивності.

Доступність та адаптивність

1. Адаптивний інтерфейс, сумісний із настільними та мобільними пристроями.
2. Використання стандартного CSS для створення гнучкого та зрозумілого дизайну.
3. Підтримка сучасних браузерів (Chrome, Edge, Opera, Safari).

Зручність використання

1. Інтуїтивно зрозумілий інтерфейс із логічною структурою навігації.
2. Відображення чітких повідомлень про успішні операції та помилки.
3. Мінімалістичний дизайн без перевантаження інтерфейсу зайвими елементами.

Технічне завдання [1] [3]

Назва проєкту:

Web-орієнтована інформаційна система управління малим торговельним підприємством.

Мета проєкту:

Розробка веб-системи для автоматизації процесів обліку товарів, продажів, складських операцій та формування звітності з метою підвищення ефективності управління підприємством.

Основні компоненти системи

Backend. Серверна частина реалізована на технологіях Node.js, Express.js, PostgreSQL із бібліотеками для роботи з файлами та транзакціями. Функціонал охоплює REST API для управління користувачами, товарами, замовленнями та постачальниками, реалізацію бізнес-логіки продажів, обробку транзакцій, формування звітів та оновлення складських залишків у режимі реального часу.

Структура бази даних:

a) Таблиця Users: id, email, password_hash, first_name, last_name, role, phone, is_active, last_login, created_at.

b) Таблиця Customers: id, first_name, last_name, email, phone, address, city, country, customer_type, discount_percent, is_active, registered_at, created_at.

c) Таблиця Categories: id, name, description, parent_id, is_active, created_at.

d) Таблиця Suppliers: id, company_name, contact_person, email, phone, address, country, payment_terms, is_active, created_at.

e) Таблиця Products: id, name, sku, category_id, supplier_id, description, unit_price, cost_price, unit, is_active, created_at.

f) Таблиця Orders: id, customer_id, user_id, order_date, status, total_amount, discount_amount, final_amount, notes, created_at.

g) Таблиця Order_Items: id, order_id, product_id, quantity, unit_price, discount_percent, total_price, created_at.

h) Таблиця Sales: id, order_id, sale_date, amount, payment_method, sales_channel, user_id, created_at.

i) Таблиця Payments: id, order_id, payment_date, amount, payment_method, status, transaction_reference, currency, created_at.

j) Таблиця Delivery: id, order_id, delivery_method, tracking_number, status, recipient_name, recipient_phone, city, estimated_date, actual_date, delivery_cost, created_at.

k) Таблиця Inventory: id, product_id, quantity_in_stock, quantity_reserved, quantity_available, warehouse_location, min_stock_level, max_stock_level, last_restock_date, created_at.

l) Таблиця Transactions: id, transaction_date, transaction_type, category, amount, currency, description, reference_type, reference_id, balance_after, user_id, created_at.

Frontend

Клієнтська частина побудована на технологіях HTML, CSS та JavaScript і включає сторінку авторизації, панель адміністратора, модуль управління товарами, модуль оформлення продажів, модуль формування звітності та адаптивний дизайн із використанням стандартного CSS.

Інтеграція

Взаємодія між клієнтською та серверною частинами здійснюється через REST API із використанням HTTP-запитів. Для передачі токена автентифікації використовується заголовок Authorization. Обробка відповідей сервера реалізована засобами JavaScript із відображенням результатів у користувацькому інтерфейсі.

Інструментальні засоби

Для серверної частини використовуються Node.js, Express.js та PostgreSQL, для клієнтської - HTML5, CSS3 та JavaScript, середовище розробки - Visual Studio Code, тестування - Thunder Client та інструменти розробника браузера.

Вимоги до розгортання

Сервер API розгортається на localhost:8000, базою даних виступає PostgreSQL та MySQL.

Обмеження

1. Доступ до системи лише для зареєстрованих користувачів.
2. Розмежування прав відповідно до ролей.
3. Підтримка сучасних браузерів.
4. Регулярне резервне копіювання бази даних.

На основі аналізу предметної області сформовано функціональні та нефункціональні вимоги до web-орієнтованої інформаційної системи управління малим торговельним підприємством. Розроблено технічне завдання, що передбачає використання сучасних веб-технологій, реляційної бази даних та стандартних засобів побудови інтерфейсу. Обрана архітектура забезпечує надійність, масштабованість і відповідність вимогам автоматизації бізнес-процесів малого підприємства.

1.4 Аналіз методів реалізації

Щоб web-орієнтована інформаційна система управління малим торговельним підприємством повністю відповідати визначеними функціональним і нефункціональним вимогам, було необхідно обґрунтовано підійти до вибору методів її реалізації. У цьому розділі здійснено аналіз підходів до побудови основних компонентів системи, зокрема клієнтської та серверної частин, вибору бази даних, організації обробки транзакцій, реалізації аутентифікації користувачів, роботи з файлами та забезпечення інтеграції між модулями. Додатково проаналізовано альтернативні рішення щодо вибору системи керування базами даних і серверної платформи. Аналіз проведено з урахуванням критеріїв продуктивності, масштабованості, безпеки, цілісності даних і зручності використання, що є ключовими для автоматизації діяльності малого торговельного підприємства.

Вибір технологій для клієнтської частини

Для реалізації клієнтської частини системи використано стандартні веб-технології: HTML5, CSS3 та JavaScript. [3, 8] Такий підхід дозволяє створити легкий, швидкий та універсальний інтерфейс без залежності від складних SPA-фреймворків. Структура інтерфейсу побудована за модульним принципом: окремо реалізовано сторінки авторизації, управління товарами, оформлення продажів, управління постачальниками та формування звітності.

Використання чистого JavaScript забезпечує асинхронну взаємодію з сервером через HTTP-запити без перезавантаження сторінки. Це дозволяє

оперативно оновлювати інформацію про товари, кошик або звіти, підвищуючи швидкість роботи касира та менеджера.

Стилізація реалізована за допомогою стандартного CSS із використанням адаптивної верстки. Це забезпечує коректне відображення інтерфейсу на різних пристроях - настільних комп'ютерах, планшетах і смартфонах. Передбачено мінімалістичний дизайн із логічною структурою меню та зрозумілою навігацією, що спрощує навчання персоналу.

Маршрутизація між сторінками організована через класичний підхід із використанням серверних маршрутів або динамічного підвантаження вмісту. Це дозволяє зберегти стабільність системи та зменшити навантаження на клієнтську частину.

Реалізація серверної частини

Серверна частина реалізована на платформі Node.js із використанням Express.js для створення REST API. [7, 15] Такий підхід забезпечує асинхронну обробку запитів, що є важливим при одночасній роботі декількох користувачів (касирів, менеджерів, адміністраторів).

Для зберігання даних обрано реляційну систему керування базами даних PostgreSQL. [4, 6] Її використання обґрунтоване необхідністю забезпечення транзакційності, підтримки зовнішніх ключів, складних SQL-запитів і формування фінансової звітності. Реляційна модель дозволяє підтримувати логічні зв'язки між таблицями «Товари», «Замовлення», «Клієнти», «Постачальники» та «Працівники», що є критично важливим для торговельного підприємства.

Поточна версія системи реалізована як клієнтський web-застосунок без серверної автентифікації, що відповідає архітектурному рішенню про розгортання системи у межах локальної мережі підприємства з обмеженим фізичним доступом. Розмежування доступу забезпечується на рівні організаційних заходів підприємства. Подальший розвиток системи передбачає впровадження серверної частини на Node.js з хешуванням паролів

та розмежуванням прав доступу відповідно до ролей персоналу - адміністратор, менеджер з продажу, працівник складу.

Обробка файлів та даних

Система передбачає можливість збереження зображень товарів і супровідних документів (накладних, договорів). Обробка файлів реалізована через серверні механізми приймання multipart-запитів із перевіркою формату та обмеженням розміру. Файли зберігаються в окремій директорії на сервері, а їхні метадані - у базі даних.

Обробка транзакцій продажу здійснюється з використанням механізмів транзакцій PostgreSQL, що гарантує цілісність даних. У разі помилки всі зміни автоматично скасовуються, що запобігає некоректному списанню товарів або втраті фінансової інформації.

Інтеграція клієнтської та серверної частин

Взаємодія між клієнтською та серверною частинами реалізована через REST API з використанням асинхронних HTTP-запитів. Дані передаються у форматі JSON. Кожен запит перевіряється на наявність токена автентифікації та коректність вхідних параметрів.

У разі виникнення помилок сервер повертає відповідні HTTP-коди стану, що дозволяє клієнтській частині відобразити інформативні повідомлення для користувача. Такий підхід забезпечує зрозумілий зворотний зв’язок і підвищує зручність роботи персоналу. [5, 17]

Аналіз ефективності обраних рішень

Проведений аналіз показав, що використання Node.js і Express.js дозволяє забезпечити швидку обробку запитів завдяки асинхронній архітектурі. PostgreSQL гарантує надійність транзакцій і підтримку складних запитів для формування звітності.

З точки зору безпеки реалізовано клієнтську валідацію форм із використанням атрибутів required та перевіркою типів вхідних даних через FormData API, що мінімізує ризик введення некоректних або неповних даних при управлінні товарами, замовленнями та клієнтами. Архітектурна ізоляція

даних у межах сесії браузера через глобальний об'єкт state унеможливлює несанкціонований зовнішній доступ до інформації підприємства.

Використання стандартних веб-технологій (HTML, CSS, JavaScript) забезпечує простоту підтримки та зменшує залежність від сторонніх бібліотек, що позитивно впливає на стабільність системи в довгостроковій перспективі.

Отже, проаналізовані технології та підходи дозволяють ефективно реалізувати всі функціональні й нефункціональні вимоги web-орієнтованої інформаційної системи управління малим торговельним підприємством. Обрана архітектура забезпечує продуктивність, масштабованість, безпеку та надійну основу для подальшого розвитку системи.

РОЗДІЛ 2. ПРОЄКТУВАННЯ ІНСТРУМЕНТАЛЬНИХ ЗАСОБІВ СТВОРЕННЯ ТА ВИКОРИСТАННЯ ІНФОРМАЦІЙНИХ СИСТЕМ І ТЕХНОЛОГІЙ

2.1. Визначення проектних рішень

Для реалізації web-орієнтованої інформаційної системи управління малим торговельним підприємством було прийнято ряд ключових проєктних рішень, спрямованих на забезпечення ефективної роботи системи, надійного зберігання даних та зручної взаємодії користувача з програмним середовищем. Основною метою створення системи є автоматизація основних бізнес-процесів підприємства, таких як облік товарів, управління замовленнями, робота з клієнтами, контроль складських залишків, облік фінансових операцій та формування звітності.

Розроблена система дозволяє централізовано зберігати інформацію про всі ключові процеси діяльності підприємства, що значно спрощує управління та підвищує ефективність роботи персоналу. Завдяки використанню web-орієнтованої архітектури система може використовуватися з будь-якого пристрою, який має доступ до мережі Інтернет.

Архітектура системи

Для реалізації інформаційної системи було обрано клієнт-серверну архітектуру [5], яка є найбільш поширеною при створенні сучасних веб-систем управління підприємствами. У такій архітектурі система поділяється на дві основні частини.

Клієнтська частина (Frontend) відповідає за відображення інтерфейсу користувача та взаємодію з системою. Через веб-інтерфейс користувач може виконувати основні операції: додавання та редагування товарів, оформлення замовлень, перегляд інформації про клієнтів, контроль залишків на складі та перегляд статистики продажів. Клієнтська частина реалізована за допомогою бібліотеки React, що забезпечує компонентну архітектуру та високу швидкість розробки; інструментом збірки є Vite, що дозволяє швидко

перезапускати застосунок під час розробки, а адаптивний інтерфейс стилізовано з використанням CSS.

Серверна частина (Backend) обробляє запити від клієнта, виконує бізнес-логіку системи та забезпечує взаємодію з базою даних. Вона відповідає за перевірку даних, їх обробку, збереження та формування відповідей для клієнтської частини. Такий підхід дозволяє забезпечити високу масштабованість системи, зручність її підтримки та можливість подальшого розширення функціональності.

Використані технології

Для створення web-орієнтованої інформаційної системи було використано сучасні технології веб-розробки, які забезпечують швидкість роботи, надійність та зручність використання системи.

Таблиця 2.1

Використані технології [3, 7, 8, 15]

Компонент системи	**Використана технологія**	**Переваги**
Клієнтська частина	React	Компонентна архітектура, швидкість роботи, повторне використання компонентів
Інструмент збірки	Vite	Швидка компіляція та оптимізація застосунку
Стилізація інтерфейсу	CSS	Адаптивний дизайн та зручна система стилів
Серверна частина	Node.js	Асинхронна обробка запитів, висока продуктивність
Серверний фреймворк	Express.js	Простота створення REST API

Продовження таблиці 2.1

Компонент системи	**Використана технологія**	**Переваги**
База даних	PostgreSQL, MySQL Workbench	Надійність, підтримка складних SQL-запитів
Проєктування БД	MySQL Workbench	Зручне створення та моделювання структури бази даних
Середовище розробки	Visual Studio Code	Потужний редактор з великою кількістю розширень

Використання цих технологій дозволяє створити ефективну інформаційну систему, яка може обробляти значні обсяги даних та забезпечувати стабільну роботу користувачів.

Визначення функціональних компонентів

Діаграма компонентів (Рис. 2.1) ілюструє основні модулі веб-сайту для управління продажами.

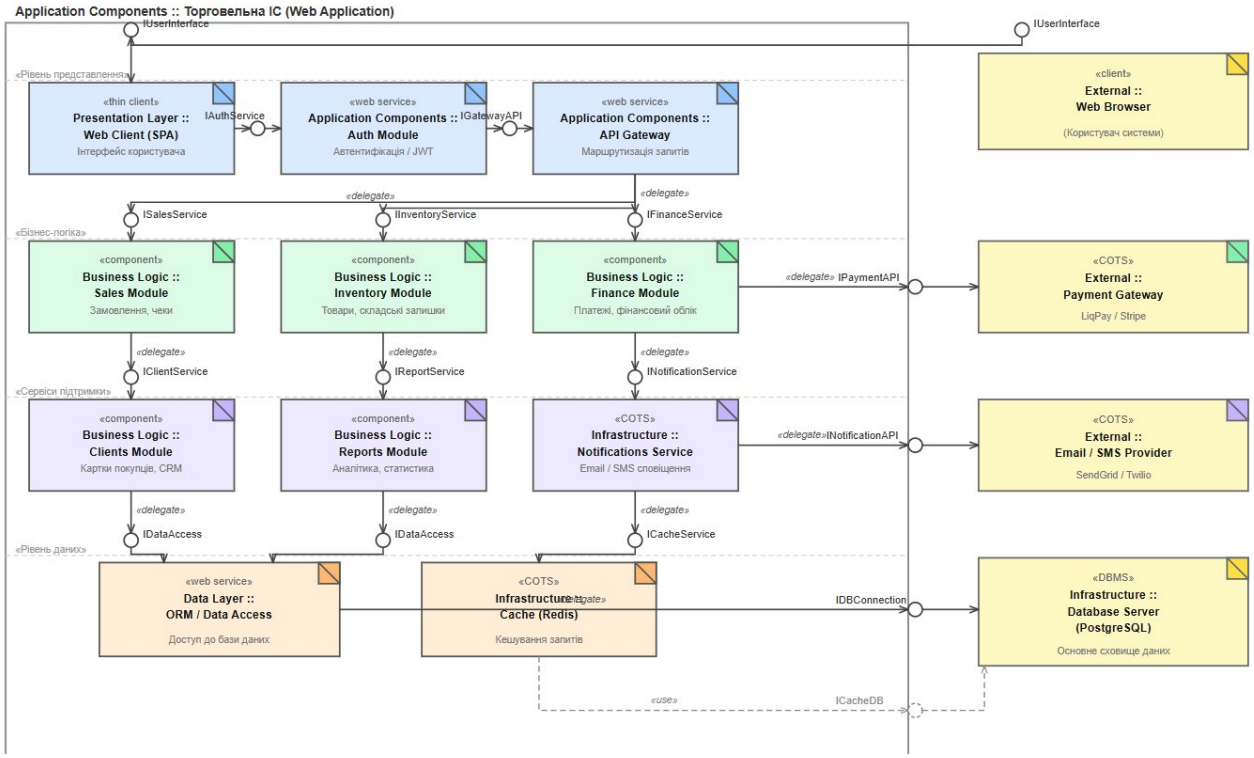


Рис 2.1 Діаграма компонентів

Структура бази даних системи [4] [6]

Одним із ключових елементів інформаційної системи є база даних, яка забезпечує зберігання та обробку всієї інформації підприємства.

У системі використовується реляційна база даних, структура якої складається з декількох взаємопов’язаних таблиць.

Основні таблиці бази даних:

Таблиця 2.2

База даних

Таблиця	Призначення
Users	Зберігає інформацію про користувачів системи (адміністраторів, менеджерів тощо)
Customers	Містить інформацію про клієнтів підприємства
Products	Зберігає інформацію про товари
Categories	Містить категорії товарів
Orders	Дані про оформлені замовлення
Order_Items	Інформація про товари, що входять до конкретного замовлення
Sales	Дані про здійснені продажі
Payments	Інформація про оплату замовлень
Delivery	Дані про доставку товарів клієнтам
Suppliers	Інформація про постачальників товарів
Inventory	Дані про складські залишки товарів
Accounting (Transactions)	Фінансові операції підприємства

Функціональні модулі системи

Інформаційна система складається з кількох основних функціональних модулів, які забезпечують управління різними аспектами діяльності підприємства. Модуль управління користувачами забезпечує реєстрацію, авторизацію та управління правами доступу. Модуль управління товарами дозволяє додавати нові товари, редагувати їх характеристики, призначати категорії та контролювати наявність на складі. Модуль управління клієнтами містить інформацію про клієнтів підприємства та дозволяє вести історію взаємодії з ними. Модуль управління замовленнями дозволяє створювати замовлення, додавати товари та контролювати статус їх виконання. Модуль управління продажами фіксує операції продажу товарів та дозволяє аналізувати діяльність підприємства. Модуль фінансового обліку забезпечує облік фінансових операцій, включаючи платежі, надходження та витрати. Модуль управління складом контролює наявність товарів та відстежує їх рух.

Взаємодія компонентів системи

Взаємодія компонентів інформаційної системи відбувається наступним чином: користувач взаємодіє з системою через веб-інтерфейс, клієнтська частина надсилає запити до серверної частини через REST API, серверна частина обробляє отримані запити та виконує відповідні бізнес-операції, сервер звертається до бази даних для отримання або збереження інформації, після чого повертає результат клієнтській частині, яка відображає отримані дані користувачу.

Особливості реалізації системи

Під час розробки системи було враховано ряд важливих аспектів: забезпечення захищеного доступу користувачів, використання сучасних технологій веб-розробки, створення зручного та інтуїтивно зрозумілого інтерфейсу, забезпечення цілісності даних у базі даних, а також можливість подальшого розширення функціональності системи..

Таким чином, прийняті проєктні рішення дозволяють створити ефективну web-орієнтовану інформаційну систему управління малим

торговельним підприємством, яка автоматизує основні бізнес-процеси підприємства та підвищує ефективність його діяльності.

2.2.Розробка логічної та фізичної структури

Розробка логічної та фізичної структури веб-сайту для управління малим торговельним підприємством є ключовим етапом проєктування, що забезпечує реалізацію вимог технічного завдання, сумісність компонентів і ефективну взаємодію між підсистемами. Логічна структура визначає абстрактне представлення системи, її основні модулі, інформаційні ресурси та їх взаємозв’язки, тоді як фізична структура описує апаратне та програмне забезпечення, необхідне для реалізації системи.

Логічна структура системи

Логічна структура системи відображає функціональні компоненти, їх взаємозв’язки та інформаційні потоки. Вона побудована на основі клієнт-серверної архітектури, де клієнтська частина відповідає за взаємодію з користувачем, а серверна - за обробку даних, автентифікацію та зберігання інформації. Основні модулі системи розділені на клієнтську та серверну частини, а також базу даних.

Основні модулі логічної структури

Клієнтська частина включає такі модулі: модуль автентифікації користувача, що забезпечує реєстрацію, авторизацію, зміну пароля та вихід із системи; модуль управління товарами, що дозволяє створювати, редагувати, видаляти та переглядати товари, а також керувати їх характеристиками (ціна, категорія, постачальник); модуль управління категоріями, що забезпечує створення та редагування категорій товарів для їх структуризації; модуль управління клієнтами, що дозволяє додавати, редагувати та переглядати інформацію про клієнтів; модуль управління замовленнями, що забезпечує створення, редагування та перегляд замовлень, а також додавання товарів до замовлення; модуль управління продажами, що фіксує продажі та дозволяє аналізувати історію реалізації товарів; модуль

платежів, що відповідає за обробку та облік оплат за замовленнями; модуль доставки, що забезпечує відстеження стану доставки товарів клієнтам; модуль складу (Inventory), що дозволяє контролювати залишки товарів та їх оновлення; модуль постачальників, що забезпечує облік інформації про постачальників; а також модуль інтерфейсу користувача (UI/UX), що забезпечує адаптивний інтерфейс, підтримку світлої/темної теми і локалізацію.

Серверна частина включає такі модулі: модуль обробки даних - глобальний об'єкт state із масивами products, orders та customers, що виконує роль сховища даних у межах сесії браузера та забезпечує єдину точку доступу для всіх компонентів системи; модуль CRUD для сутностей системи, що реалізує операції створення, читання та видалення даних для товарів, замовлень і клієнтів через функції openAddProduct(), openAddOrder(), openAddCustomer() та відповідні обробники подій форм; модуль бізнес-логіки, що реалізує автоматичний розрахунок суми замовлення, визначення статусу складських залишків, формування KPI-показників дашборду та фінансових звітів; а також модуль звітності - функції renderReports() та renderInventory(), що формують зведені дані щодо продажів, залишків товарів і фінансових операцій у реальному часі на основі поточного стану масивів.

3. База даних:

Нормалізація даних [4]

База даних розроблена згідно з принципами нормалізації до третьої нормальної форми, що забезпечує мінімізацію дублювання даних і збереження їхньої цілісності. (сам код буде в Додатку Б) [21]

Структура бази даних

1. **Таблиця Users**: Зберігає дані користувачів. Первинний ключ: id. Унікальні поля: email.
2. **Таблиця Customers**: Містить інформацію про клієнтів.
3. **Таблиця Products**: Містить інформацію про товари.
4. **Таблиця Categories**: Зберігає категорії товарів.

5. **Таблиця Suppliers**: Містить інформацію про постачальників.

6. **Таблиця Orders**: Містить дані про замовлення. Первинний ключ: id. Зв’язок "один до багатьох" із Customers.

7. **Таблиця Order_Items**: Зв’язкова таблиця для товарів у замовленні.

8. **Таблиця Sales**: Містить інформацію про продажі.

9. **Таблиця Payments**: Зберігає дані про оплату замовлень.

10. **Таблиця Delivery**: Містить інформацію про доставку.

11. **Таблиця Inventory**: Відповідає за облік залишків товарів.

12. **Таблиця Transactions (Accounting)**: Містить фінансові операції підприємства.

Діаграма на (Рис. 2.2) демонструє зв’язки між основними таблицями системи:

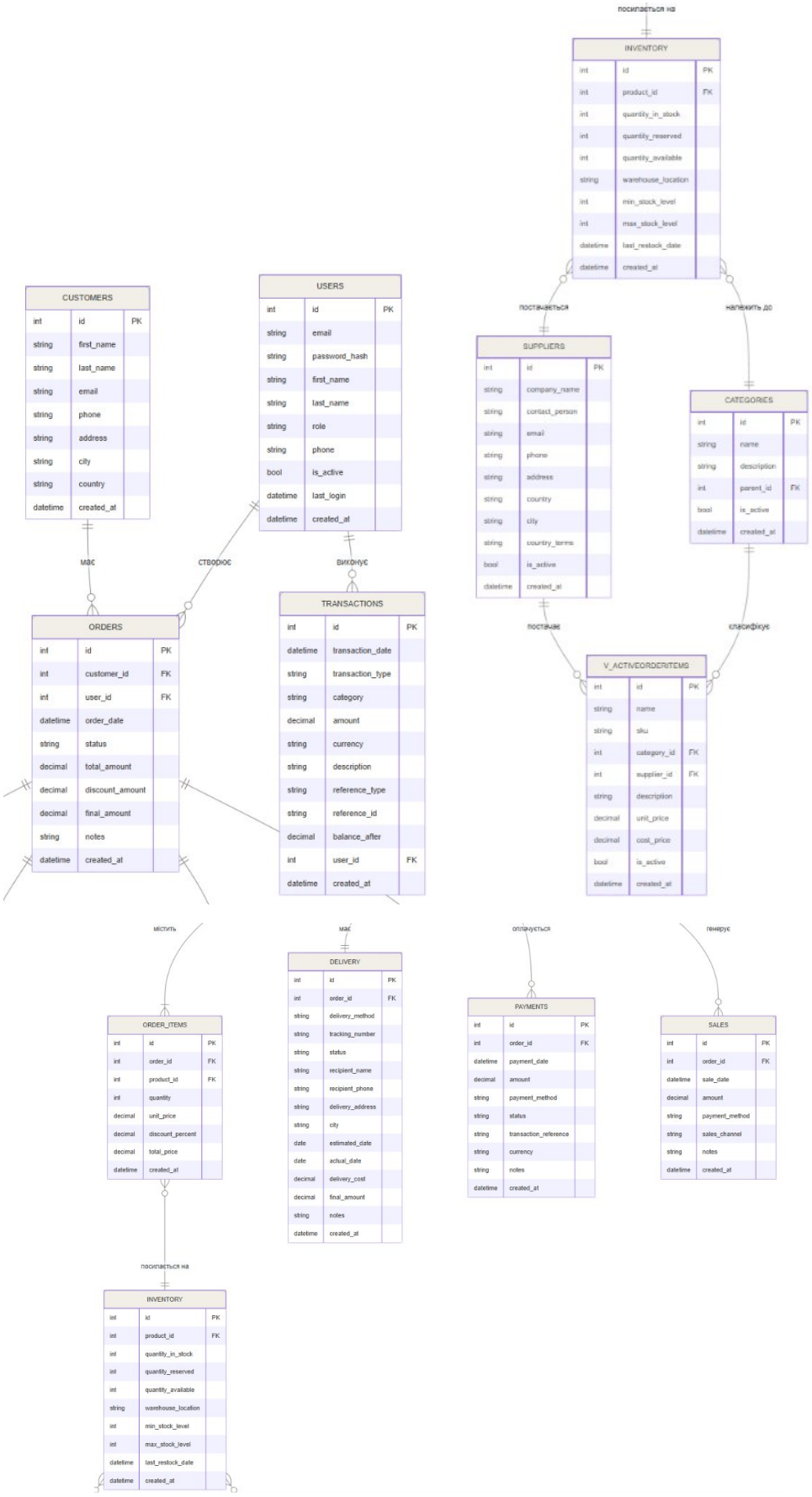


Рис 2.2 Діаграма сутність-зв’язок [2][4]

Пояснення зв’язків і компонентів

1. **Users**: Користувачі працюють із системою та можуть керувати всіма сутностями.
2. **Customers**: Один клієнт може мати багато замовлень.
3. **Orders**: Замовлення складається з кількох товарів через таблицю Order_Items.
4. **Products**: Кожен товар належить до категорії та може бути пов’язаний із постачальником.
5. **Inventory**: Відображає кількість товарів на складі.
6. **Sales**: Фіксують факт продажу товарів.
7. **Payments**: Пов’язані із замовленнями або продажами.
8. **Delivery**: Містить інформацію про доставку замовлень.
9. **Transactions**: Відображають фінансові операції підприємства.

Інформаційні потоки:

Клієнтська частина відправляє HTTP-запити (GET, POST, PUT, DELETE) до API Gateway через Axios, сервер обробляє запити, взаємодіє з реляційною базою даних (PostgreSQL або MySQL) та повертає відповідь клієнту. Дані про товари, замовлення, клієнтів та операції синхронізуються між клієнтом і сервером через REST API, а локалізація інтерфейсу та повідомлень забезпечується через i18next.

Фізична структура системи [7, 17]

Фізична структура показує, яке апаратне та програмне забезпечення необхідне для реалізації веб-сайту, а також описує розгортання системи на сервері та клієнтських пристроях.

Сервер потребує такого апаратного забезпечення: процесор з двома ядрами на частоті 2.4 ГГц або вище, оперативна пам'ять - 4 ГБ для локального сервера та 8 ГБ для продакшену, дисковий простір - 20 ГБ для зберігання бази даних, операційна система - Windows Server або Linux. Клієнтські пристрої мають бути сумісними з ПК, ноутбуками, планшетами та смартфонами,

підтримувати браузери Chrome, Opera, Safari та Edge, а мінімальна роздільна здатність екрана - 320×480.

Програмне забезпечення серверної частини включає Node.js, Express.js та PostgreSQL/MySQL. Клієнтська частина реалізована на React, Vite та CSS із бібліотеками react-router-dom, Axios та i18next. Середовище розробки складається з Visual Studio Code, Thunder Client, інструментів розробника браузера та MySQL Workbench для проєктування бази даних. Розгортання системи передбачає локальний сервер із API на localhost:8000 та клієнтом на localhost:5173, хмарну базу даних або сервер (PostgreSQL/MySQL), а також статичний хостинг для клієнтської частини.

Розроблена логічна структура системи чітко окреслює функціональні модулі - управління товарами, замовленнями, клієнтами, складом і фінансовими операціями - а також їх інформаційні потоки, що забезпечує відповідність функціональним вимогам. Фізична структура, побудована на сучасних технологіях - React, Node.js і реляційних базах даних - гарантує продуктивність, безпеку та адаптивність системи.

2.3. Способи взаємодії між підсистемами

Для забезпечення стабільної та ефективної взаємодії між модулями системи використовується єдиний механізм обміну даними через спільний об'єкт state.

HTTP-запити

Обмін інформацією між фронтендом і бекендом реалізується через REST API [3] [15] із застосуванням стандартних HTTP-методів:

1. **GET** - використовується для отримання даних із сервера;
2. **POST** - застосовується для реєстрації користувачів, а також створення нових завдань і шаблонів;
3. **PUT** - призначений для оновлення вже існуючих записів;
4. **DELETE** - використовується для видалення даних.

Такий підхід забезпечує консистентність відображення даних у всіх розділах системи без перезавантаження сторінки.

Взаємодія з браузерним API

Система використовує стандартні браузерні API для роботи з даними та інтерфейсом: FormData API для збору даних із форм, localStorage API для збереження налаштувань теми, DOM API для динамічного оновлення таблиць і карток, атрибут data-price для передачі ціни товару між елементами форми замовлення.

Реєстрація та авторизація

Поточна версія системи не потребує реєстрації та авторизації користувачів, оскільки розрахована на розгортання у межах локальної мережі торговельного підприємства з обмеженим колом працівників. Доступ до системи здійснюється безпосередньо через браузер без введення облікових даних. У localStorage зберігаються виключно налаштування теми інтерфейсу (shopflow-theme) для збереження комфортних умов роботи між сесіями.

Сповіщення у реальному часі

Для інформування користувача про результати виконаних операцій реалізовано систему візуальних сповіщень через функцію notify(). Після кожної успішної дії - додавання товару, створення замовлення або клієнта - у правому верхньому куті екрана автоматично з'являється анімоване повідомлення із відповідним текстом (наприклад, «Товар успішно додано!» або «Замовлення успішно створено!»). Сповіщення відображається протягом фіксованого часу та зникає автоматично без додаткових дій користувача. У разі помилки - наприклад, при спробі створити замовлення без вибору клієнта або товару - функція notify() викликається з параметром 'error' і відображає повідомлення червоного кольору, що дозволяє оперативно реагувати на некоректні дії.

Завантаження файлів

У поточній версії системи завантаження файлів не передбачено, оскільки система реалізована як клієнтський web-застосунок без серверної

частини. Усі дані про товари, замовлення та клієнтів зберігаються у глобальному об'єкті state в оперативній пам'яті браузера. Подальший розвиток системи передбачає реалізацію завантаження файлів - наприклад, фотографій товарів або документів - із використанням формату multipart/form-data та серверної обробки на Node.js.

Робота з базою даних

Серверна частина взаємодіє з базою даних PostgreSQL, у якій зберігається вся необхідна інформація: облікові записи користувачів, завдання, шаблони та інші дані.

Для роботи з базою даних використовується реляційна СУБД MySQL, яка забезпечує структуроване зберігання даних із підтримкою зовнішніх ключів, індексів та транзакцій.

Обробка помилок [9]

У системі передбачено механізм обробки помилок на серверному рівні. У разі виникнення проблеми сервер повертає відповідний HTTP-статус:

1. **400** - помилка у запиті;
2. **401** - відсутність авторизації;
3. інші коди - залежно від ситуації.

Це дозволяє клієнтській частині коректно реагувати на помилки та інформувати користувача.

2.4. Декомпозиція системи на модулі

Ми виконали декомпозицію веб-сайту на окремі функціональні модулі з метою забезпечення гнучкості системи, можливості повторного використання програмного коду та спрощення її подальшого супроводу. В якості архітектурного підходу було обрано клієнт-серверну модель, де серверна частина реалізована на основі технологій Node.js та Express, а клієнтська - за допомогою React із використанням Vite. У даному підрозділі наведено детальну декомпозицію системи управління малим торговельним підприємством із описом основних модулів, їх функціонального призначення

та обґрунтування вибору, а також представлено ER-діаграму структури бази даних.

Обґрунтування декомпозиції

Поділ системи на модулі здійснено з урахуванням ключових бізнес-процесів малого торговельного підприємства: облік товарів, управління продажами, робота з клієнтами, облік замовлень і забезпечення безпечного доступу користувачів. Такий підхід дозволяє підвищити масштабованість системи, спростити її модернізацію та забезпечити незалежність окремих компонентів. Кожен модуль виконує чітко визначену функцію, що також полегшує тестування та підтримку.

Основні модулі системи [2, 5]

Бекенд (серверна частина)

Серверна частина відповідає за обробку запитів, бізнес-логіку, взаємодію з базою даних та забезпечення безпеки доступу і включає такі модулі.

Модуль автентифікації та авторизації забезпечує реєстрацію користувачів (адміністраторів, менеджерів), вхід у систему, зміну пароля та контроль доступу до функцій; використовує бібліотеки jsonwebtoken та bcrypt і гарантує захист комерційної інформації підприємства та розмежування прав доступу.

Модуль управління товарами забезпечує додавання, редагування, видалення та перегляд товарів, включаючи їх назву, категорію, ціну, кількість на складі, а також містить підмодулі обліку залишків, категоризації та пошуку і фільтрації товарів; використовує бібліотеку mysql і є ключовим модулем для ведення складського обліку.

Модуль управління замовленнями дозволяє створювати, редагувати та відслідковувати замовлення клієнтів, включає підмодулі оформлення замовлення, зміни статусу (нове, обробляється, виконане) та обліку товарів у замовленні, що забезпечує автоматизацію процесу продажів.

Модуль управління клієнтами зберігає інформацію про клієнтів (ПІБ, контакти, історія покупок) і дозволяє аналізувати клієнтську базу та підвищувати якість обслуговування.

Модуль управління файлами забезпечує завантаження та зберігання файлів (наприклад, зображень товарів) із використанням бібліотек multer та fs, що реалізує централізоване управління медіаданими.

Модуль бази даних реалізує доступ до бази даних PostgreSQL та MySQL Workbench, управління схемами та виконання CRUD-операцій через бібліотеку mysql і забезпечує єдину точку доступу до даних.

Фронтенд (клієнтська частина) [2, 5]

Клієнтська частина забезпечує взаємодію користувача із системою та відображення інформації і включає такі модулі.

Модуль навігації та маршрутизації реалізує переходи між сторінками (товари, замовлення, клієнти, вхід у систему) за допомогою react-router-dom, що забезпечує зручну навігацію.

Модуль автентифікації надає інтерфейс для входу та реєстрації користувачів із використанням axios та інтеграцією із серверною частиною для безпечного доступу.

Модуль управління товарами дозволяє переглядати, додавати та редагувати товари і є основним інтерфейсом для роботи зі складом.

Модуль управління замовленнями відображає список замовлень, їх статус та деталі, спрощуючи контроль продажів.

Модуль управління клієнтами дозволяє переглядати та редагувати інформацію про клієнтів, підтримуючи взаємодію з ними.

Модуль локалізації забезпечує підтримку кількох мов інтерфейсу за допомогою бібліотеки i18next, розширюючи можливості використання системи.

Модуль теми дозволяє перемикати світлу та темну тему, покращуючи зручність роботи.

Модуль стилів та UI відповідає за зовнішній вигляд і адаптивність інтерфейсу з використанням CSS, що забезпечує швидку розробку інтерфейсу.

Взаємодія модулів

Модулі системи взаємодіють через спільний об'єкт state, що виступає єдиним джерелом даних. Після кожної операції зміни (додавання товару, створення замовлення, додавання клієнта) автоматично викликаються функції оновлення відповідних компонентів інтерфейсу - renderStats(), renderProductsTable(), renderOrdersTable(), - що забезпечує консистентність відображення даних у всіх розділах системи без перезавантаження сторінки.

ER-діаграма структури бази даних

Для відображення структури даних використовується ER-діаграма (Рис. 2.3), яка включає основні сутності системи:

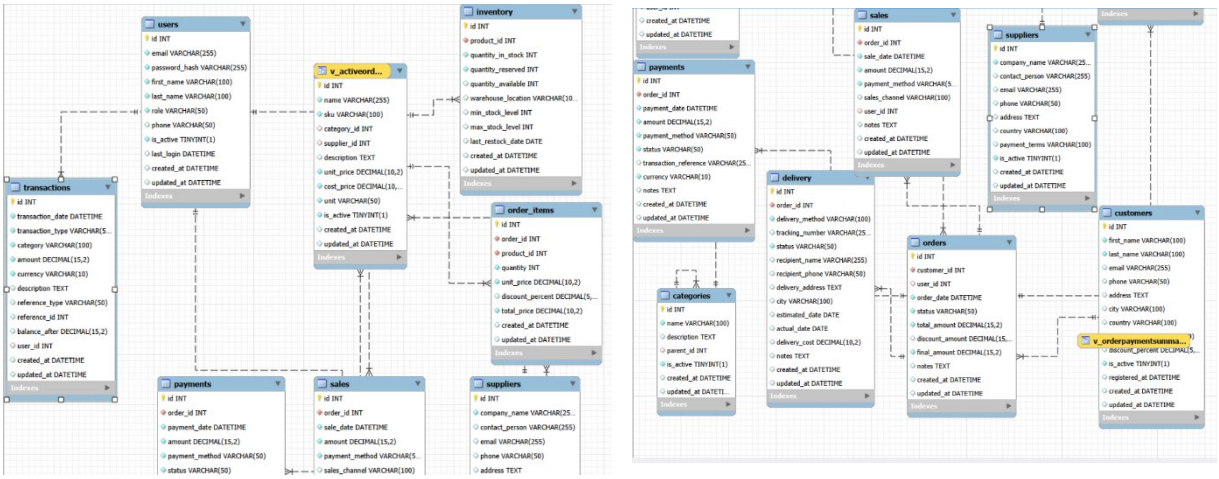


Рис 2.3 ER-діаграма бази даних [4]

1. **USERS** (id, email, password_hash, first_name, last_name, role, phone, is_active, last_login, created_at, updated_at)
2. **INVENTORY** (id, product_id, quantity_in_stock, quantity_reserved, quantity_available, warehouse_location, min_stock_level, max_stock_level, last_restock_date, created_at, updated_at)

3. **V_ACTIVEORDERITEMS** (id, name, sku, category_id, supplier_id, description, unit_price, cost_price, is_active, created_at, updated_at) *(вигляд "v_activeord..." - це view)*

4. **TRANSACTIONS** (id, transaction_date, transaction_type, category, amount, currency, description, reference_type, reference_id, balance_after, user_id, created_at, updated_at)

5. **PAYMENTS** (id, order_id, payment_date, amount, payment_method, status, transaction_reference, currency, notes, created_at, updated_at)

6. **SALES** (id, order_id, sale_date, amount, payment_method, sales_channel, notes, created_at, updated_at)

7. **SUPPLIERS** (id, company_name, contact_person, email, phone, address, country, city, country_terms, is_active, created_at, updated_at)

8. **ORDER_ITEMS** (id, order_id, product_id, quantity, unit_price, discount_percent, total_price, created_at, updated_at)

9. **DELIVERY** (id, order_id, delivery_method, tracking_number, status, recipient_name, recipient_phone, delivery_address, city, estimated_date, actual_date, delivery_cost, final_amount, notes, created_at, updated_at)

10. **ORDERS** (id, customer_id, user_id, order_date, status, total_amount, discount_amount, final_amount, notes, created_at, updated_at)

11. **CUSTOMERS** (id, first_name, last_name, email, phone, address, city, country, created_at, updated_at)

12. **CATEGORIES** (id, name, description, parent_id, is_active, created_at, updated_at)

Зв’язки:

1. Один користувач (**USERS**) може створювати багато замовлень (**ORDERS**). *(один-до-багатьох: один користувач - багато замовлень)*

2. Одне замовлення (**ORDERS**) може містити багато товарів (**ORDER_ITEMS**). *(один-до-багатьох: одне замовлення - багато позицій товарів)*

3. Один клієнт (**CUSTOMERS**) може мати багато замовлень (**ORDERS**). *(один-до-багатьох: один клієнт - багато замовлень)*

4. Один товар з інвентарю (**INVENTORY**) може бути присутнім у багатьох позиціях замовлень (**ORDER_ITEMS**). *(один-до-багатьох)*

5. Один постачальник (**SUPPLIERS**) може постачати багато товарів (**INVENTORY**). *(один-до-багатьох)*

6. Одне замовлення (**ORDERS**) може мати тільки одну доставку (**DELIVERY**). *(один-до-одного)*

7. Одне замовлення (**ORDERS**) може мати багато платежів (**PAYMENTS**). *(один-до-багатьох)*

8. Одне замовлення (**ORDERS**) може мати багато продажів/транзакцій (**SALES**). *(один-до-багатьох)*

9. Одна категорія (**CATEGORIES**) може містити багато товарів (**INVENTORY**). *(один-до-багатьох)*

10. Один користувач (**USERS**) може виконувати багато транзакцій (**TRANSACTIONS**). (один-до-багатьох)

Ця структура забезпечує узгодженість даних і підтримує всі бізнес-процеси системи.

Висновок

Декомпозиція системи на модулі дозволяє досягти високої гнучкості, масштабованості та зручності супроводу. Вибір модулів обумовлений функціональними потребами малого торговельного підприємства, а ER-

діаграма відображає логічну структуру даних, необхідну для ефективної роботи системи.

2.5. Розробка людино машинного інтерфейсу

Створюючи web-орієнтовану інформаційну систему управління малим торговельним підприємством, ми приділили особливу увагу розробці людино-машинного інтерфейсу. Нашою метою було зробити взаємодію користувачів із системою максимально ефективною та зручною. Ми розробили інтерфейс, керуючись принципами ергономіки, інтуїтивності та адаптивності, щоб він був зручним для широкого кола користувачів, незалежно від їхнього рівня технічної підготовки. Далі ми детальніше розглянемо ключові елементи інтерфейсу, як вони взаємодіють, та на яких принципах побудована їх реалізація. Візуалізація основних екранних форм допоможе краще зрозуміти структуру ЛМІ. В основі нашого підходу лежать сучасні методології UX/UI дизайну. [8, 19]

Елементи людино-машинного інтерфейсу

Визначено ключові елементи інтерфейсу, які реалізують основні функції системи. До них належать:

1. **Навігаційна панель:** Забезпечує доступ до основних розділів застосунку, таких як "Головна", "Товари", "Замовлення", "Склад", "Аналітика", "Вихід" та "Профіль". Реалізовано через компонент Navbar, інтегрований у головний файл застосунку для маршрутизації, що дозволяє користувачам легко переключатися між функціональними модулями.

2. **Карти товарів:** Відображають інформацію про товари у вигляді інтерактивних карток, які підтримують перегляд, редагування та оновлення даних, сприяючи зручній візуальній організації асортименту.

3. **Модальні вікна:** Використовуються для детального перегляду інформації про товар або замовлення, а також для створення нових записів, забезпечуючи зручний доступ до додаткових функцій без перезавантаження сторінки.

4. **Форми введення:** Включають поля для створення та редагування товарів, замовлень і клієнтів, а також інструменти для введення характеристик товару (ціна, кількість, категорія), що полегшує введення даних.

5. **Тост-повідомлення:** Надають зворотний зв’язок користувачу про успішні або невдалі операції, підвищуючи інтерактивність і зручність використання.

6. **Аналітична панель:** Інтегрований елемент, який відображає статистику продажів, залишків товарів та інші показники у вигляді графіків і діаграм, сприяючи ефективному управлінню підприємством.

7. **Кнопки дій:** Інтерактивні елементи, такі як кнопка "Додати товар" на головній сторінці та кнопки "Зберегти" та "Очистити" у модальних вікнах, забезпечують швидкий доступ до основних операцій.

Взаємозв’язок елементів інтерфейсу

Елементи інтерфейсу пов’язані між собою для забезпечення логічної послідовності дій користувача. Навігаційна панель слугує центральним елементом, який спрямовує користувача до відповідних сторінок, де компоненти, такі як карти товарів і аналітична панель, відображають дані, отримані з бекенду через REST API. Модальні вікна активуються з карток товарів або кнопок дій (наприклад, "Додати товар"), забезпечуючи доступ до форм введення без зміни основного екрану. Тост-повідомлення інтегровано з усіма інтерактивними елементами для миттєвого інформування про результати дій, що підвищує зручність і зрозумілість інтерфейсу.

Принципи реалізації

При розробці застосовано сучасні методи UI/UX дизайну. Адаптивність інтерфейсу забезпечується використанням CSS, що гарантує коректне відображення на мобільних і настільних пристроях. [3] Локалізація реалізована через відповідні бібліотеки, що дозволяє перемикатися між мовами без втрати функціональності. Механізм перемикання темної та світлої теми зберігає вибір користувача, забезпечуючи персоналізацію та

комфорт. Використання сучасних технологій для динамічних елементів підвищує інтерактивність і швидкість реагування системи.

Візуалізація інтерфейсу

Для ілюстрації структури людино-машинного інтерфейсу наведено зображення ключових екранних форм. На (Рис. 2.4) представлено вигляд головної сторінки застосунку, яка включає навігаційну панель із розділами "Головна", "Товари", "Замовлення", поле пошуку, кнопку "Додати товар", іконку перемикання теми та виходу чи переходу до профілю при натисканні на аватар. Центральна частина сторінки відображає інформаційні блоки з товарами та коротку статистику, що підкреслює інтуїтивність інтерфейсу.

На (Рис. 2.5) показано модальне вікно створення товару, яке містить поля для введення назви, опису, ціни, кількості, категорії, а також кнопки "Зберегти" та "Очистити". Ці зображення демонструють логічну організацію елементів і їхню інтеграцію з функціоналом системи. (самі коди будуть в Додаток А)

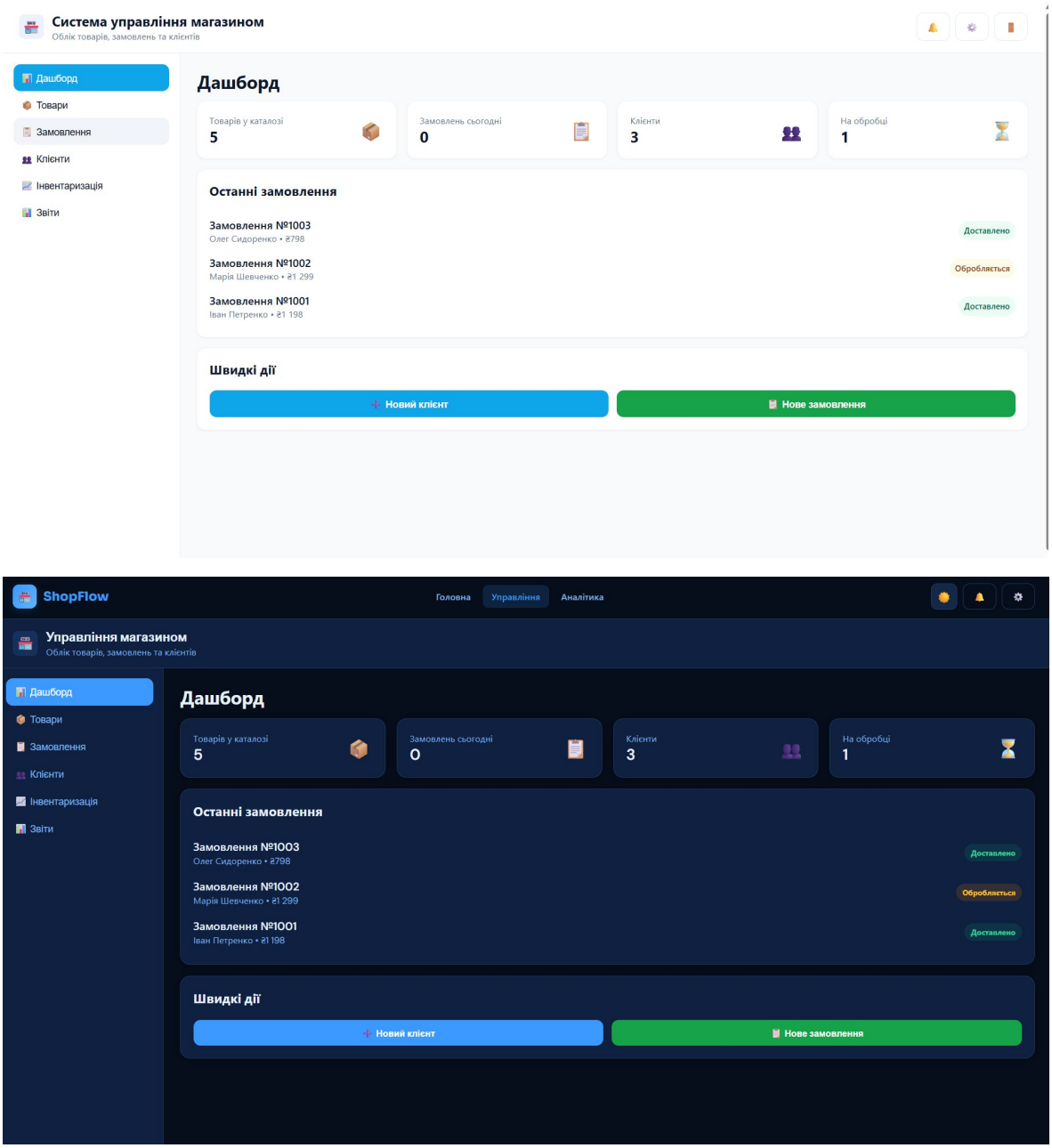


Рис 2.4 Вигляд головної сторінки (а – демо версія, б – остаточна версія)

Опис: Головна сторінка відображає навігаційну панель, поле пошуку, кнопку "Нове замовлення" та інформаційні блоки з товарами, що спрощує перше знайомство з системою. (Демо версія буде описана в Додатку Г)

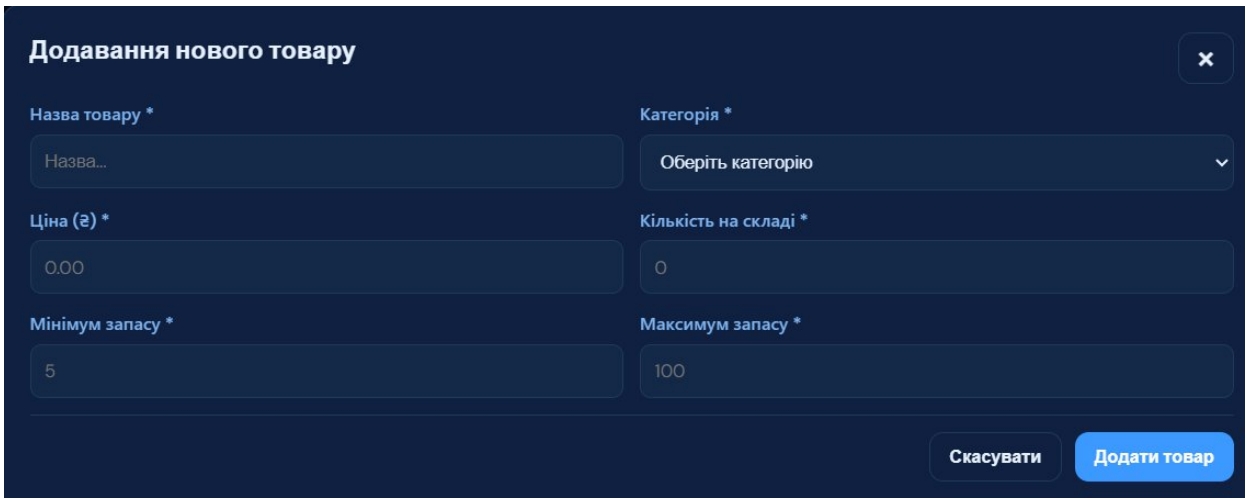


Рис 2.5 Модальне вікно для додавання товару

Опис: Модальне вікно підсистеми управління товарним каталогом включає поля для введення даних (назва товару, категорія, ціна, кількість на складі, мінімум та максимум запасу) та кнопки дій, забезпечуючи зручність додавання нових позицій до бази даних малого торговельного підприємства.

Розробка web-орієнтованої інформаційної системи управління малим торговельним підприємством передбачає реалізацію інтуїтивного інтерфейсу для оперативного введення та редагування товарних даних. Модальні вікна як елемент сучасного UI/UX-дизайну забезпечують фокусування уваги користувача на конкретній задачі без переходу між сторінками, що підвищує ефективність роботи персоналу підприємства та мінімізує ймовірність помилок при введенні інформації. [3]

РОЗДІЛ 3. ДОСЛІДЖЕННЯ ПРЕДМЕТНОЇ ОБЛАСТІ ТА ПОСТАНОВКА ЗАДАЧІ

3.1 Опис модулів системи

Інформаційна система ShopFlow для управління малим торговельним підприємством складається з трьох основних модулів, реалізованих у вигляді окремих HTML-сторінок із вбудованим JavaScript. [3, 21] Кожен модуль відповідає за окремі функції: відображення загальної інформації про систему, оперативне управління торговельними процесами та аналітику. Нижче наведено детальний опис модулів системи, їх функціонального призначення та взаємодії.

Модуль головної сторінки (index.html)

Функціонал: Представлення системи, навігація між розділами, відображення ключових можливостей платформи.

Реалізація: Статична HTML-сторінка з навігаційною панеллю, що містить посилання на розділи «Головна», «Управління», «Аналітика» та кнопку «Відкрити систему». Секція Hero відображає назву системи та основний опис. Панель статистики (stats-bar) містить чотири показники: кількість магазинів, оброблений обсяг замовлень, час роботи системи та середній час налаштування. Блок можливостей (features-grid) представляє шість функціональних карток: управління товарами, замовлення та доставка, база клієнтів, інвентаризація, звіти та аналітика, підтримка темної та світлої теми. Перемикання теми реалізовано через атрибут data-theme та збереження вибору у localStorage.

Модуль управління магазином (management.html)

Функціонал: Оперативне управління товарами, замовленнями, клієнтами, складськими залишками та формування звітів.

Реалізація: Односторінковий застосунок з бічною панеллю навігації (aside) та динамічною зміною вкладок у головній області (main). Усі дані зберігаються у глобальному об'єкті state із масивами products, orders та

customers, що ініціалізуються тестовими значеннями при завантаженні сторінки. Модуль включає шість підрозділів:

1. Дашборд (tab-dashboard): чотири KPI-картки із загальним доходом, кількістю активних замовлень, товарів та клієнтів; таблиця останніх замовлень; блок «Товари з низьким запасом» та блок «Швидкі дії» з кнопками відкриття модальних вікон.

2. Управління товарами (tab-products): таблиця каталогу товарів; функція openAddProduct() відкриває модальне вікно з полями назва, категорія, ціна, кількість на складі, мінімум та максимум запасу. Після збереження товар додається до масиву state.products із оновленням KPI та сторінки інвентаризації.

3. Управління замовленнями (tab-orders): таблиця замовлень зі статусами (Обробляється / Доставлено / Скасовано) у вигляді кольорових pill-міток; функція openAddOrder() відкриває модальне вікно з динамічним заповненням списків клієнтів і товарів та автоматичною підстановкою ціни через атрибут data-price.

4. Управління клієнтами (tab-customers): таблиця бази клієнтів із контактними даними; функція openAddCustomer() відкриває форму додавання нового клієнта з полями повне ім'я, телефон, email та місто.

5. Інвентаризація (tab-inventory): таблиця складських залишків із автоматичним визначенням статусу: «Критичний» (червона мітка) якщо кількість менша за minStock, «Низький запас» (жовта) якщо менша за minStock × 1.5, «Достатньо» (зелена) в інших випадках.

6. Звіти (tab-reports): зведений фінансовий звіт із загальним доходом від завершених замовлень, кількістю замовлень, середнім чеком та вартістю складських запасів, що розраховується як сума добутків ціни та кількості по кожній товарній позиції.

Повідомлення про результати операцій відображаються функцією notify(), яка створює анімовану нотифікацію у правому верхньому куті екрана. [3, 8]

Модуль аналітики (analytics.html)

Функціонал: Відображення детальних аналітичних звітів, графіків динаміки продажів, розподілу по категоріях та топу товарів за обраний часовий період.

Реалізація: Сторінка з перемикачем часового періоду (7 днів, 30 днів, 90 днів), чотирма KPI-картками (дохід, замовлення, середній чек, нові клієнти) із кольоровими індикаторами зміни показників. Стовпчикова діаграма доходів (renderRevenueChart()) реалізована на чистому SVG без зовнішніх бібліотек і відображає 30 точок даних із градієнтним заливанням. Кругова діаграма (renderDonut()) показує розподіл продажів за категоріями: Одяг (38%), Взуття (24%), Аксесуари (19%), Електроніка (12%), Інше (7%). Блок топ товарів (renderTopProducts()) відображає п'ять найпопулярніших позицій із горизонтальними прогрес-барами. Таблиця останніх замовлень містить номер, клієнта, товар, суму, дату та статус. Усі графіки динамічно зчитують кольори з CSS-змінних, що забезпечує коректне відображення в обох темах.

Побудова графіків реалізована засобами SVG без - функція renderRevenueChart() динамічно створює елементи rect, line та text на основі масиву даних, а кольори зчитуються з CSS-змінних через getComputedStyle(), що забезпечує коректне відображення в обох темах інтерфейсу.

Спільний модуль навігації та теми

Функціонал: Підтримка темного/світлого режимів, навігація між розділами системи.

Реалізація: Усі три сторінки містять єдину навігаційну панель із логотипом ShopFlow та посиланнями на розділи. Логіка перемикання теми реалізована через атрибут data-theme на елементі html та CSS-змінні (:root для світлої, [data-theme="dark"] для темної). Вибрана тема зберігається у localStorage під ключем shopflow-theme та відновлюється при кожному завантаженні сторінки. Активний розділ виділяється класом active.

3.2 Людино-машинний інтерфейс та інструкції для користувачів

Людино-машинний інтерфейс системи ShopFlow розроблено з акцентом на інтуїтивність, адаптивність і підтримку різних категорій персоналу торговельного підприємства. Інтерфейс побудовано з використанням HTML, CSS та JavaScript із застосуванням сучасних принципів UI/UX-дизайну, що забезпечує адаптивне відображення для десктопних і мобільних пристроїв. Система підтримує темний і світлий режими для комфортної роботи в різних умовах освітлення.

Основні елементи інтерфейсу

Навігаційна панель

Містить логотип ShopFlow, меню для переходу між розділами системи (Головна, Управління, Аналітика), перемикач теми (темна/світла). Навігаційна панель у світлому та темному режимах показана на рис. 3.1. Відображено панель із логотипом, посиланнями на розділи та кнопкою перемикання теми.

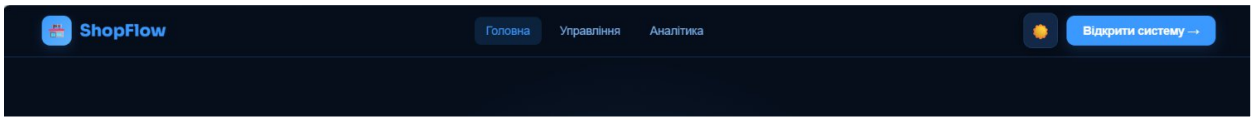


Рис 3.1 Навігаційна панель

Головна сторінка

Відображає публічний лендінг системи з описом можливостей платформи, панеллю статистики та блоком навігації по розділах. Головна сторінка у світлому та темному режимах показана на рис. 3.2 та рис. 3.3. Відображено секцію Hero із заголовком, картки можливостей та блок навігації.

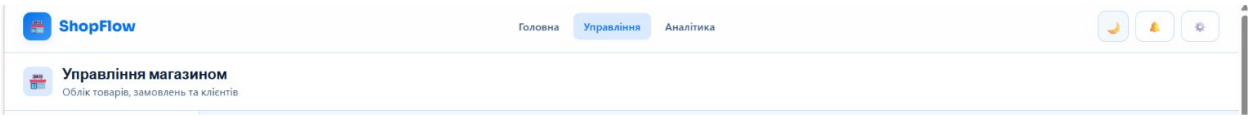


Рис 3.2 Головна сторінка (світла тема)

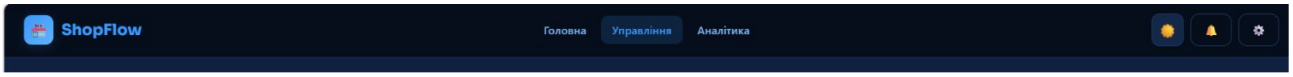


Рис 3.3 Головна сторінка (темна тема)

Дашборд управління магазином

Відображає чотири KPI-картки із ключовими показниками діяльності підприємства, таблицю останніх замовлень та блок товарів із низьким запасом. Дашборд модуля управління показано на рис. 3.4. Відображено KPI-картки, таблицю замовлень та блок швидких дій.

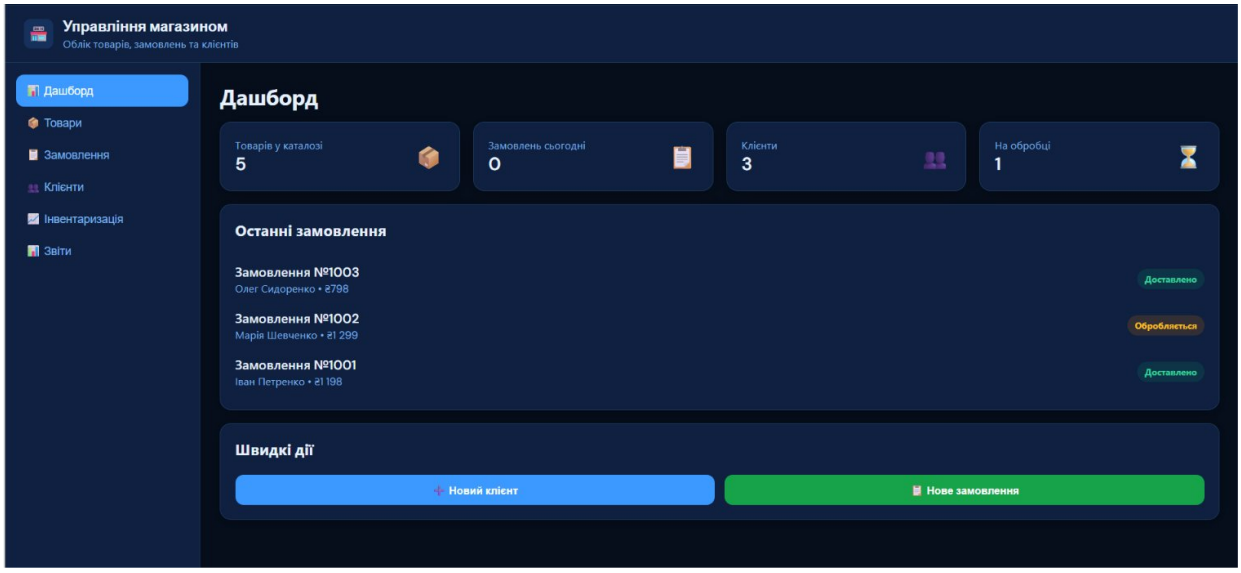


Рис 3.4 Дашборд модуля управління магазином

Модальне вікно додавання товару

Містить поля для введення даних (назва товару, категорія, ціна, кількість на складі, мінімум та максимум запасу) та кнопки дій, забезпечуючи зручність додавання нових позицій до каталогу. Модальне вікно показано на рис. 3.5.

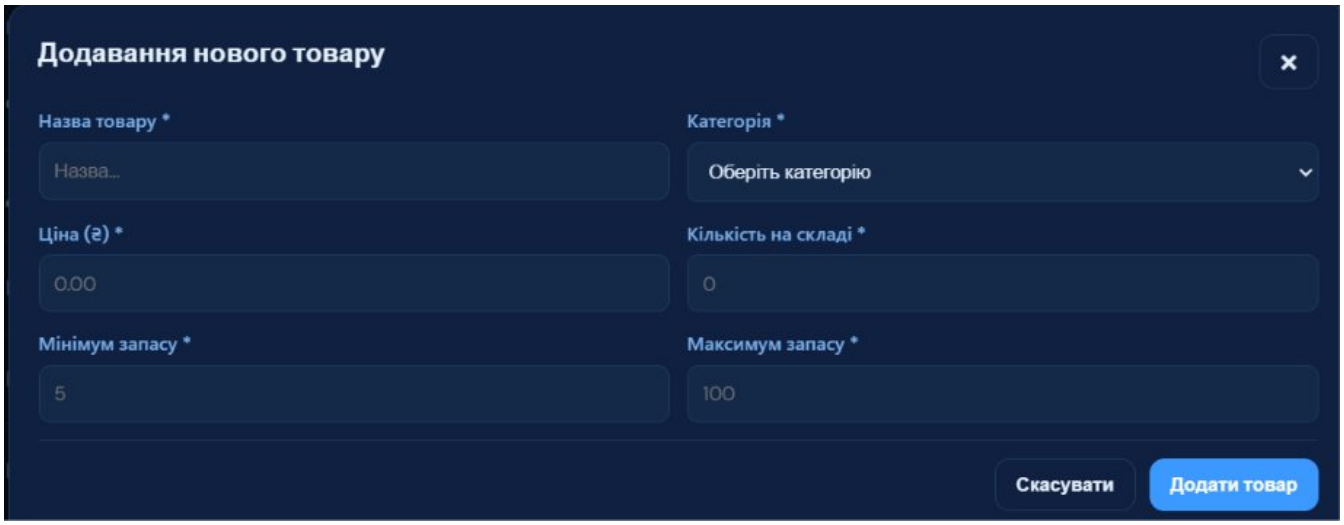


Рис 3.5 Модальне вікно додавання нового товару

Підмодуль управління замовленнями

Відображає таблицю замовлень із кольоровими статусами та форму створення нового замовлення з автоматичною підстановкою ціни. Підмодуль замовлень показано на рис. 3.6.

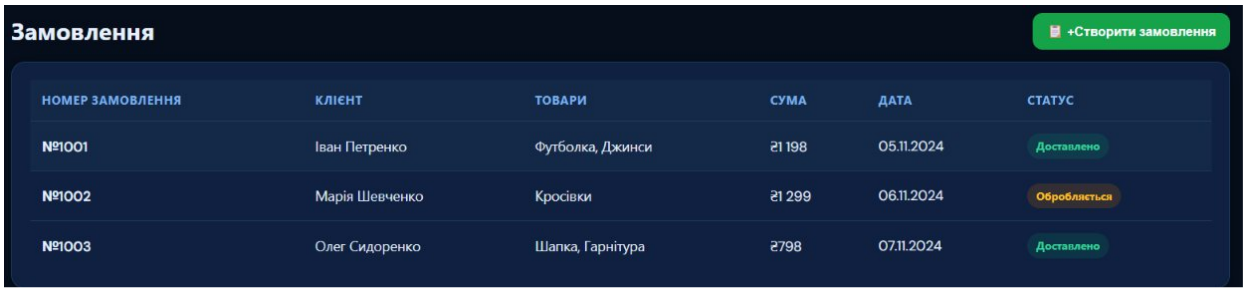


Рис 3.6 Підмодуль управління замовленнями

Підмодуль інвентаризації

Відображає складські залишки товарів із автоматичним визначенням статусу запасу на основі порівняння поточного залишку з мінімальним рівнем. Підмодуль інвентаризації показано на рис. 3.7.

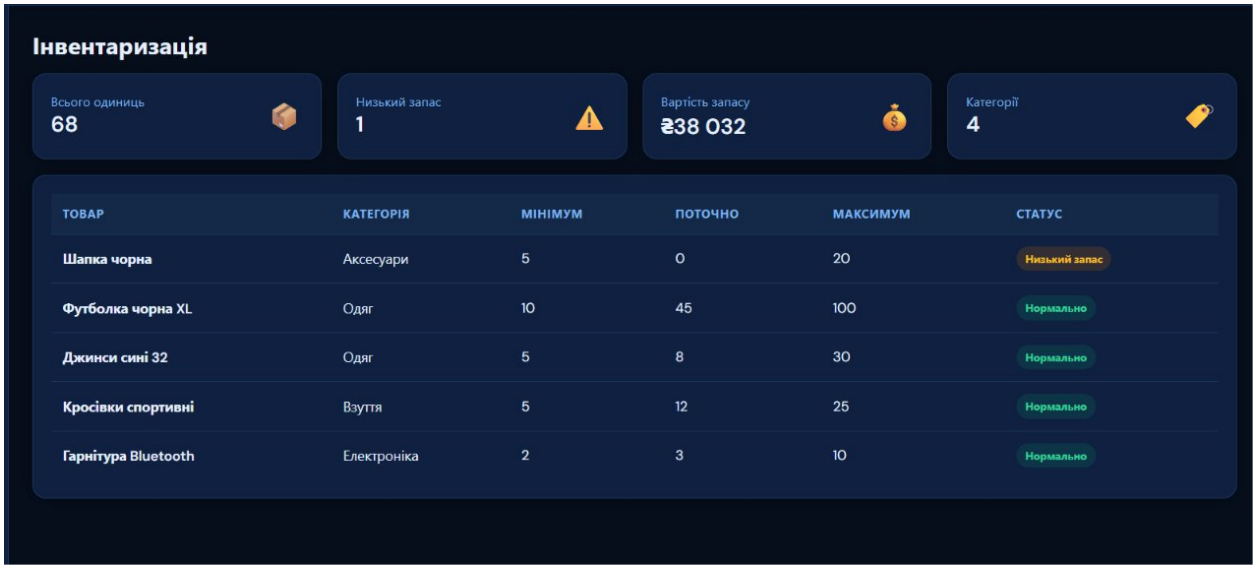


Рис 3.7 Підмодуль інвентаризації

Сторінка аналітики

Відображає KPI-картки з перемикачем часового періоду, стовпчикову діаграму доходів, кругову діаграму розподілу по категоріях та топ товарів. Сторінка аналітики показана на рис. 3.8 та рис. 3.9.

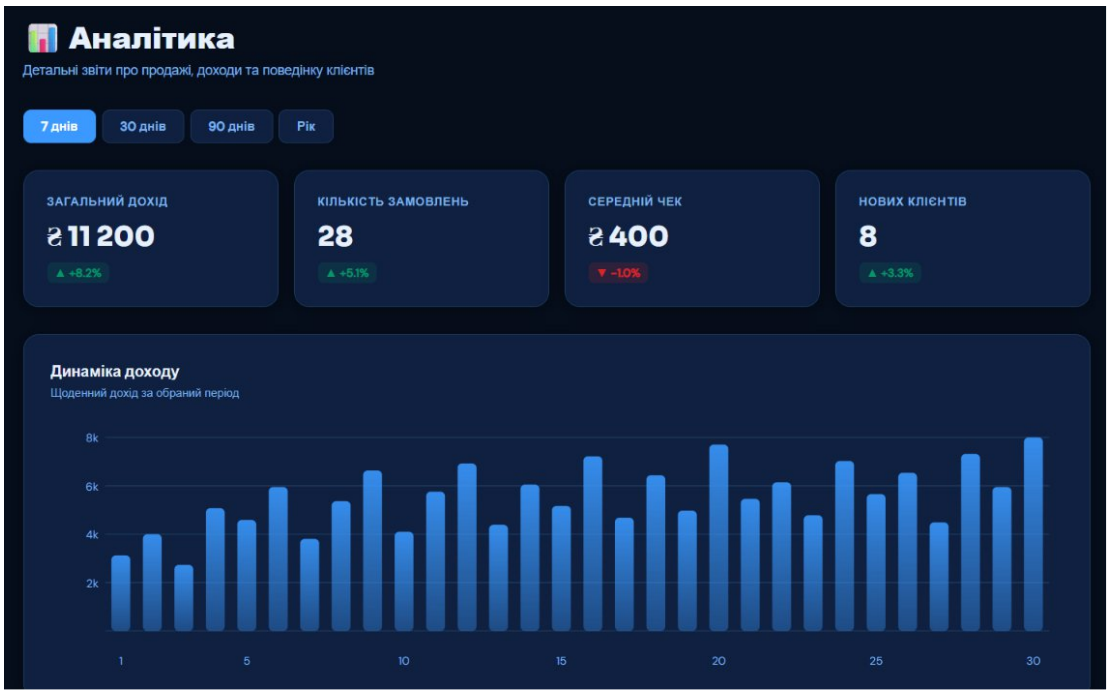


Рис 3.8 Сторінка аналітики: KPI-картки та діаграма доходів

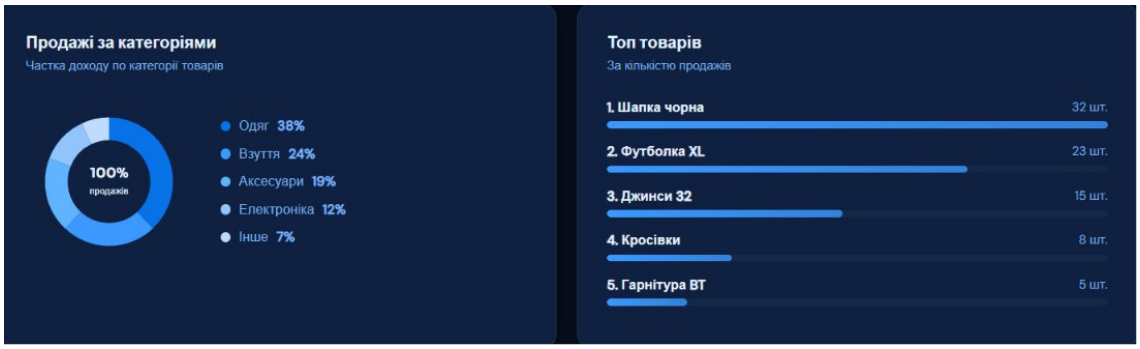


Рис 3.9 Сторінка аналітики: кругова діаграма та топ товарів [8][19]

Інструкції для користувачів

Для забезпечення зручності використання системи розроблено інструкції, які охоплюють основні сценарії роботи:

Відкриття системи:

1. Перейдіть на головну сторінку системи ShopFlow.
2. Натисніть кнопку «Відкрити систему» або перейдіть до розділу «Управління» через навігаційну панель.

Додавання нового товару:

1. У розділі «Управління» перейдіть на вкладку «Товари».
2. Натисніть кнопку «Додати товар».

3. Заповніть поля: назва, категорія, ціна, кількість на складі, мінімум та максимум запасу.

4. Натисніть «Додати товар» для збереження позиції в каталозі.

Оформлення замовлення:

1. Перейдіть на вкладку «Замовлення» та натисніть «Нове замовлення».

2. Оберіть клієнта зі списку, потім товар - ціна підставиться автоматично.

3. Вкажіть кількість, дату та статус замовлення, натисніть «Створити замовлення».

Контроль складських залишків:

1. Перейдіть на вкладку «Інвентаризація».

2. Перегляньте статуси товарів: зелений - «Достатньо», жовтий - «Низький запас», червоний - «Критичний».

3. На дашборді у блоці «Товари з низьким запасом» відображаються позиції, що потребують термінового поповнення.

Перегляд аналітики:

1. Перейдіть до розділу «Аналітика» через навігаційну панель.

2. Оберіть часовий період (7, 30 або 90 днів) для оновлення показників.

3. Перегляньте KPI-картки, графік доходів, розподіл по категоріях та топ товарів.

Налаштування інтерфейсу:

1. У навігаційній панелі натисніть кнопку перемикання теми для зміни між темним і світлим режимом.

2. Вибір теми зберігається автоматично та відновлюється при наступному відкритті системи.

3.3 Технічні характеристики системи

Система ShopFlow розроблена з урахуванням сучасних вимог до web-орієнтованих інформаційних систем, що забезпечує її продуктивність та зручність розгортання. Нижче наведено технічні характеристики системи, включаючи апаратні, програмні та архітектурні аспекти. [3, 6, 7, 15]

Апаратні вимоги

Сервер:

1. Процесор: 2-ядерний, 2 ГГц або вище.
2. Оперативна пам'ять: 2 ГБ.
3. Дисковий простір: 5 ГБ для файлів системи.
4. Операційна система: Windows Server або Linux.

Клієнт:

1. Пристрій: Персональний комп'ютер, планшет або смартфон.
2. Браузер: Chrome, Firefox, Safari, Edge (актуальні версії з підтримкою ES6+).
3. Інтернет-з'єднання: 1 Мбіт/с або вище.

Програмне забезпечення

Клієнтська частина:

1. HTML5, CSS3 із CSS-змінними для підтримки тематизації.
2. JavaScript (ES6+) для реалізації бізнес-логіки та DOM-маніпуляцій.
3. SVG для побудови аналітичних діаграм без зовнішніх бібліотек.
4. Google Fonts (Sora, DM Sans) для типографіки інтерфейсу.
5. localStorage API для збереження налаштувань теми.

Архітектура системи

Система реалізована як клієнтський web-застосунок без серверної частини:

1. Фронтенд: три HTML-сторінки (index.html, management.html, analytics.html) із вбудованим CSS та JavaScript.

2. Дані: зберігаються у глобальному об'єкті state в оперативній пам'яті браузера; налаштування теми зберігаються у localStorage.

3. Архітектурний підхід: модульна організація з розподілом відповідальності між окремими сторінками та функціями JavaScript.

3.4 Стратегія та методика тестування

Якість програмного забезпечення є ключовим аспектом розробки web-орієнтованої інформаційної системи управління малим торговельним підприємством, що прямо визначає, наскільки продукт буде функціональним, надійним та зручним у використанні персоналом підприємства. Для забезпечення відповідності системи всім вимогам і мінімізації дефектів розроблено та застосовано комплексну стратегію тестування.

Визначення стратегій тестування

В основі стратегії тестування лежать принципи ітеративного та інкрементального підходів, інтегрованих безпосередньо в процес розробки. Компоненти системи тестувалися поетапно: від окремих функцій до повних бізнес-сценаріїв. Головне завдання - переконатися, що розроблений продукт працює правильно (верифікація) і відповідає очікуванням персоналу підприємства (валідація), а також виявити та виправити можливі дефекти якомога раніше.

Рівні тестування: [9, 17]

Модульне тестування - тестування окремих ізольованих функцій та компонентів застосунку:

1. Перевірка коректності роботи функцій renderStats(), renderProductsTable(), renderOrdersTable(), renderCustomersTable(), renderInventory(), renderReports().

2. Перевірка функцій openAddProduct(), openAddOrder(), openAddCustomer() на коректне відкриття модальних вікон та валідацію введених даних.

3. Перевірка функцій renderRevenueChart(), renderDonut(), renderTopProducts() модуля аналітики на коректність побудови SVG-графіків.

Інтеграційне тестування - перевірка взаємодії між компонентами системи:

1. Перевірка оновлення KPI-карток після додавання нового товару, замовлення або клієнта.

2. Перевірка коректного відображення змін у підмодулі інвентаризації після операцій з товарами.

3. Перевірка коректного переключення вкладок бічної панелі та відображення відповідного контенту.

Системне тестування - комплексна перевірка всієї системи як єдиного цілого:

1. Тестування повних бізнес-сценаріїв: додавання товару, оформлення замовлення, додавання клієнта, перегляд інвентаризації, формування звіту, перегляд аналітики за різні періоди.

2. Перевірка коректного перемикання між темною та світлою темами на всіх трьох сторінках системи.

3. Перевірка збереження теми у localStorage та її відновлення при перезавантаженні сторінки.

Приймальне тестування - підтвердження відповідності системи вимогам і готовності до розгортання на підприємстві. На цьому етапі перевіряється відповідність системи очікуванням кінцевих користувачів - менеджерів і працівників торговельного підприємства.

Типи тестування:

1. Функціональне тестування: перевірка відповідності всіх функціональних вимог, включаючи CRUD-операції для товарів, замовлень та

клієнтів, автоматичний розрахунок KPI-показників, визначення статусу складських залишків та побудову аналітичних графіків.

2. Тестування зручності використання: оцінка інтуїтивності інтерфейсу для різних категорій персоналу підприємства. Перевіряється зрозумілість навігації між розділами, зручність роботи з модальними вікнами та таблицями, читабельність статусних міток.

3. Кросбраузерне тестування: перевірка коректного відображення та функціональності системи у браузерах Chrome, Firefox, Edge та Safari для забезпечення консистентного користувацького досвіду.

4. Тестування темної та світлої теми: перевірка коректності застосування CSS-змінних та читабельності всіх елементів інтерфейсу при перемиканні між темами на всіх сторінках системи.

5. Тестування адаптивності: перевірка коректного відображення інтерфейсу на різних розмірах екрану - десктоп, планшет та мобільний пристрій із використанням CSS media queries.

Методика тестування [20]

Для реалізації вищезгаданої стратегії обрано методику ручного тестування як основний метод верифікації функціоналу. Тестування застосовувалось на всіх рівнях, особливо на етапах системного та приймального тестування, для перевірки загального користувацького досвіду та візуального оформлення. Це включає:

1. Позитивні сценарії: перевірка очікуваної поведінки системи при коректних вхідних даних та стандартних діях користувача.

2. Негативні сценарії: перевірка коректної обробки помилок при некоректному введенні даних, незаповнених обов'язкових полях та невалідних значеннях.

Вимоги до проведення експериментів

Для забезпечення об'єктивності та відтворюваності результатів тестування сформовано такі вимоги:

1. Ізольоване середовище: тестування проводиться у браузері з початковим станом системи (очищений localStorage) для виключення впливу попередніх сесій.

2. Тестові дані: набір даних охоплює типові бізнес-сценарії торговельного підприємства з товарами різних категорій, замовленнями з різними статусами та клієнтами різних типів.

3. Фіксація результатів: результати кожного тест-кейсу документуються із зазначенням ідентифікатора, опису, вхідних даних, очікуваного та фактичного результату і статусу виконання.

4. Відтворюваність: тест-кейси спроектовані для багаторазового повторення з однаковими вхідними даними, що забезпечує стабільність результатів та можливість верифікації виправлених дефектів.

3.5 Результати тестування

Після реалізації програмно-технічних засобів та формування стратегії і методики тестування, наступним етапом є безпосереднє проведення експериментів та фіксація їх результатів. Цей підрозділ містить опис процесу виконання тест-кейсів, аналіз отриманих даних та їх зіставлення з очікуваними результатами, що дозволяє оцінити ступінь працездатності та відповідності розробленої системи вимогам технічного завдання.

Організація проведення тестування

Тестування розробленої інформаційної системи управління малим торговельним підприємством проводилось у спеціально підготовленому тестовому середовищі, що імітує умови реальної експлуатації. Це середовище включало:

1. Серверна частина: Node.js з Express.js, запущений локально на порту 8000.

2. База даних: MySQL з тестовою базою даних business, що містить 12 таблиць із тестовими записами.

3. Клієнтська частина: веб-сайт ShopFlow, доступний через веб-браузер за адресою http://localhost:5500.

Розроблені тест-кейси та їх виконання

DB-001. Перевірка первинних ключів таблиць.

Виконано запит до INFORMATION_SCHEMA з фільтром CONSTRAINT_NAME = 'PRIMARY'. Усі 12 таблиць містять первинний ключ по колонці id.

Статус: Успіх (див. рис. В.1 Додаток В, лістинг Б.1 Додаток Б).

DB-002. Перевірка зовнішніх ключів між таблицями.

Виконано запит до INFORMATION_SCHEMA з фільтром REFERENCED_TABLE_NAME IS NOT NULL. Підтверджено 13 зовнішніх ключів між усіма таблицями системи.

Статус: Успіх (див. рис. В.2 Додаток В, лістинг Б.2 Додаток Б).

DB-003. Аналіз замовлень та суми продажів по клієнтах.

Виконано запит із LEFT JOIN між Customers та Orders з групуванням по клієнту. Лідером є ТОВ «Бізнес Плюс Корпоративний» - 44 998.00 грн, далі Андрій Степаненко - 37 999.00 грн, Олена Ковальчук - 14 999.00 грн. Логіку формування замовлень реалізовано у функції openAddOrder() (див. Додаток А, лістинг А.3).

Статус: Успіх (див. рис. В.3 Додаток В, лістинг Б.3 Додаток Б).

DB-004. Залишки товарів на складі зі статусом поповнення.

Виконано запит із JOIN між Inventory, Products та Categories з оператором CASE для визначення статусу. Усі 6 позицій мають статус «OK». Аналогічну логіку на стороні клієнта реалізовано через if/else (див. Додаток А, лістинг А.5).

Статус: Успіх (див. рис. В.4 Додаток В, лістинг Б.4 Додаток Б).

DB-005. Аналіз продажів по каналах та способах оплати.

Виконано запит із GROUP BY по sales_channel та payment_method. Найбільший обсяг - корпоративний канал із безготівковою оплатою (44 998.00 грн), офлайн із готівкою - 38 188.00 грн, онлайн із карткою - 20 697.00 грн.

Статус: Успіх (див. рис. В.5 Додаток В, лістинг Б.5 Додаток Б).

DB-006. Статус доставки замовлень із деталями.

Виконано запит із JOIN між Orders, Customers та Delivery. Замовлення 1, 2, 5 - «Доставлено», замовлення 3, 4 - «В дорозі». Для замовлень 3 та 5 трекінг-номер відсутній (NULL).

Статус: Успіх (див. рис. В.6 Додаток В, лістинг Б.6 Додаток Б).

DB-007. Фінансовий звіт по доходах і витратах.

Виконано запит із GROUP BY по місяцю, типу операції та категорії. За 2024-01: витрати - закупівля товарів 55 000 грн, зарплата 80 000 грн, оренда 25 000 грн; дохід від продажів - 78 696 грн. Відображення показників реалізовано у підмодулі звітності (див. Додаток А, лістинг А.1).

Статус: Успіх (див. рис. В.7 Додаток В, лістинг Б.7 Додаток Б).

DB-008. Топ товарів за кількістю продажів та валовим прибутком.

Виконано запит із JOIN між Order_Items, Products та Categories з розрахунком валового прибутку. Лідер - ноутбук Lenovo IdeaPad 3 (2 шт, виручка 44 998.20 грн, прибуток 6 998.20 грн). Побудову діаграм реалізовано у функції renderRevenueChart() (див. Додаток А, лістинг А.7).

Статус: Успіх (див. рис. В.8 Додаток В, лістинг Б.8 Додаток Б).

DB-009. Перевірка статистики кількості записів у таблицях.

Виконано запит до information_schema.tables. Найбільше записів у categories та transactions - по 7, найменше у suppliers та users - по 4. Тестові дані на стороні клієнта визначено у глобальному об'єкті state (див. Додаток А, лістинг А.1).

Статус: Успіх (див. рис. В.9 Додаток В, лістинг Б.9 Додаток Б). [17]

3.6 Опис модулю номенклатури та контрагентів

Модуль номенклатури та контрагентів реалізовано у файлі nomenclature.html як окрему сторінку системи ShopFlow із власною бічною панеллю навігації та п'ятьма підрозділами: товари, категорії, постачальники, клієнти та діаграми. Сторінка доступна через навігаційну панель усіх сторінок системи за посиланням «Номенклатура».

Підмодуль номенклатури товарів (вкладка «Товари»)

Відображає повний каталог товарних позицій підприємства у розширеному табличному форматі з колонками: назва товару, артикул SKU, категорія, постачальник, ціна продажу, собівартість, маржа у відсотках, кількість на складі, мінімальний рівень запасу та статус наявності. Маржа розраховується автоматично за формулою (ціна − собівартість) / ціна × 100. Статус визначається через порівняння поточного залишку з мінімальним рівнем: «✓ В наявності» (зелена мітка), «⚠ Мало» (жовта), «✗ Закінчився» (червона). Зверху розміщено чотири KPI-картки: загальна кількість позицій, кількість у наявності, кількість із низьким запасом та загальна вартість складу, що розраховується як сума добутків кількості та собівартості по кожній позиції. Підмодуль показано на рис. 3.10.

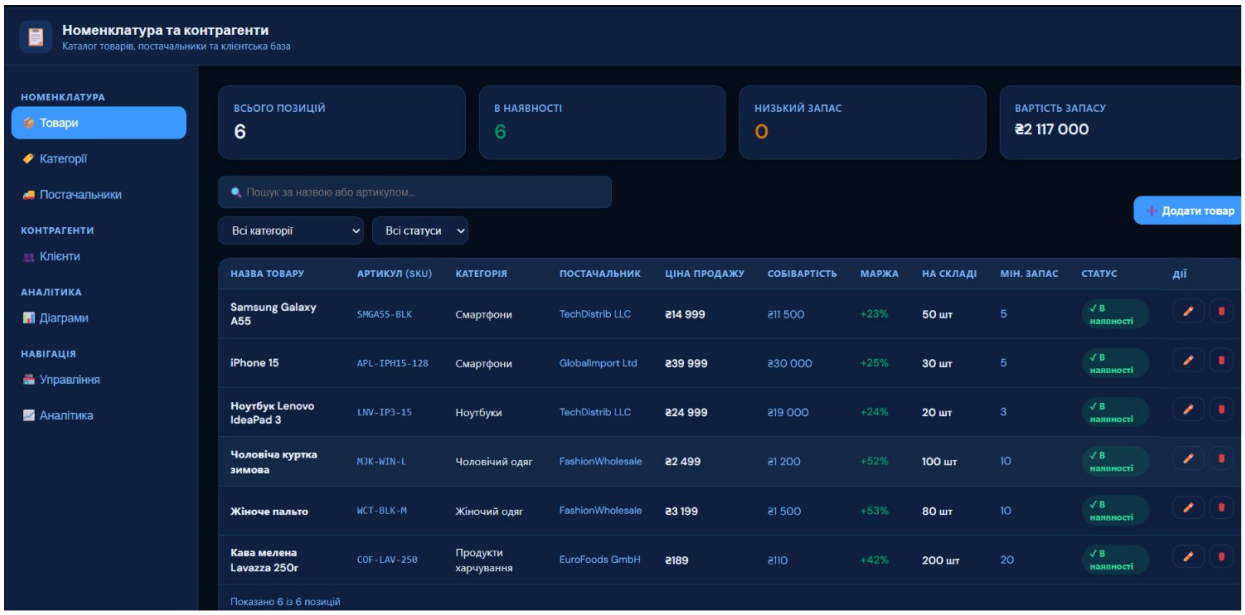


Рис 3.10 Підмодуль номенклатури товарів

Опис: Таблиця номенклатури відображає 6 товарних позицій із повними даними - артикулом SKU, постачальником, ціною продажу, собівартістю та автоматично розрахованою маржею. Пошук виконується за назвою або артикулом, фільтрація - за категорією та статусом запасу.

Підмодуль категорій (вкладка «Категорії»)

Відображає автоматично сформовані картки категорій на основі існуючих товарів у масиві state.products. Для кожної категорії відображається

іконка, кількість позицій, загальний залишок у штуках та вартість складських запасів категорії. Картки формуються динамічно функцією renderCategories() без ручного введення - додавання нового товару автоматично оновлює відповідну картку категорії. Підмодуль показано на рис. 3.11.

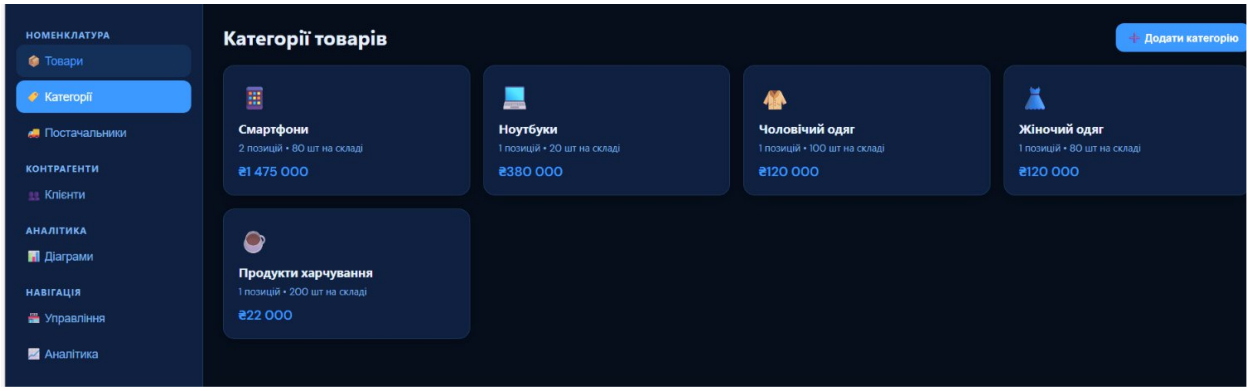


Рис 3.11 Підмодуль категорій товарів

Опис: Відображено п'ять категорій: Смартфони, Ноутбуки, Чоловічий одяг, Жіночий одяг, Продукти харчування - кожна з кількістю позицій та вартістю запасу. Картки перебудовуються автоматично після кожної операції з товарами.

Підмодуль постачальників (вкладка «Постачальники»)

Відображає базу постачальників підприємства у табличному форматі з колонками: назва компанії, контактна особа, email, телефон, країна, умови оплати та кількість пов'язаних товарних позицій. Кількість товарів по кожному постачальнику розраховується автоматично на основі поля supplier у масиві state.products. Підмодуль показано на рис. 3.12.

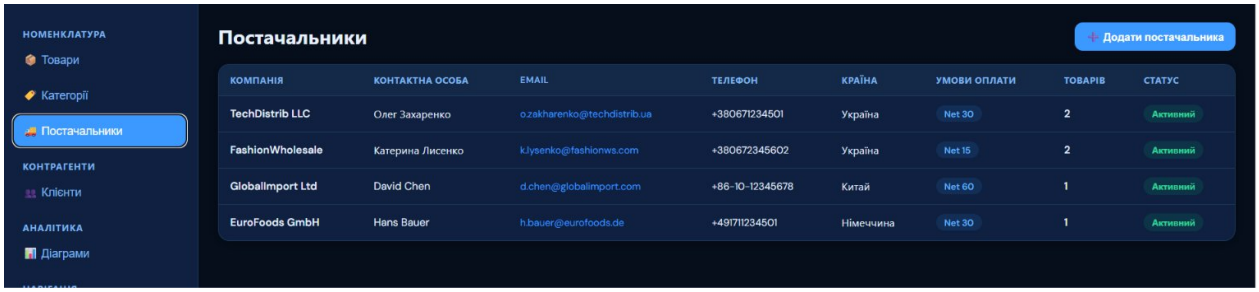


Рис 3.12 Підмодуль управління постачальниками

Опис: Таблиця містить чотири постачальники системи: TechDistrib LLC (Україна, Net 30, 2 товари), FashionWholesale (Україна, Net 15, 2 товари), GlobalImport Ltd (Китай, Net 60, 1 товар), EuroFoods GmbH (Німеччина, Net

30, 1 товар). Форма додавання нового постачальника включає поля: назва компанії, контактна особа, email, телефон, країна та умови оплати.

Підмодуль клієнтів та контрагентів (вкладка «Клієнти»)

Відображає повну клієнтську базу підприємства у розширеному табличному форматі з колонками: повне ім'я або назва організації, тип контрагента, телефон, email, місто, знижка, дата реєстрації, кількість замовлень та загальна сума витрат. Тип контрагента позначається кольоровими мітками: «Фізична особа» (синя), «Юридична особа» (жовта), «ФОП» (фіолетова). Зверху розміщено чотири KPI-картки: загальна кількість контрагентів, кількість фізичних осіб, кількість юридичних осіб та ФОП, загальний оборот по всіх клієнтах. Пошук виконується за ім'ям або email, фільтрація - за типом контрагента та містом. Підмодуль показано на рис. 3.13.

Рис 3.13 Підмодуль клієнтів та контрагентів

Опис: Таблиця контрагентів відображає 5 записів із повними контактними даними, типом, містом, знижкою, датою реєстрації та загальним оборотом. Лідером за обсягом витрат є ТОВ «Бізнес Плюс» - 44 998 грн, далі Андрій Степаненко - 37 999 грн, Олена Ковальчук - 14 999 грн.

Підмодуль діаграм (вкладка «Діаграми»)

Реалізує візуалізацію аналітичних даних номенклатури та клієнтської бази за допомогою п'яти SVG-діаграм, побудованих без використання сторонніх бібліотек. Діаграми формуються функцією renderCharts() при переході на вкладку та автоматично перемальовуються при зміні теми інтерфейсу через getComputedStyle() для зчитування CSS-змінних. Підмодуль показано на рис. 3.14 та рис. 3.15.

Рис 3.14 Діаграми маржинальності та розподілу за категоріями

Опис: Стовпчикова діаграма маржинальності відображає відсоток маржі для кожної товарної позиції з кольоровим кодуванням - зелений (маржа ≥ 40%), синій (≥ 25%), жовтий (нижче 25%). Кругова donut-діаграма відображає розподіл товарів за категоріями з легендою та відсотками.

Рис 3.15 Діаграми залишків, обороту контрагентів та scatter-графік

Опис: Стовпчикова діаграма залишків відображає поточну кількість кожного товару на складі, червона пунктирна лінія позначає мінімальний рівень запасу. Горизонтальні смуги обороту відображають загальну суму витрат кожного контрагента. Scatter-діаграма «ціна vs собівартість» показує кожен товар як кружок із відсотком маржі, пунктирна діагональ відповідає лінії «ціна = собівартість» - усі точки вище неї означають прибуткові товари.

Модальні вікна підмодулю

Модальне вікно додавання нової товарної позиції включає розширений набір полів порівняно зі стандартною формою модуля управління: назва

товару, артикул SKU, категорія, постачальник із вибором із зареєстрованих постачальників, ціна продажу, собівартість, кількість на складі, мінімум та максимум запасу. Модальне вікно показано на рис. 3.16.

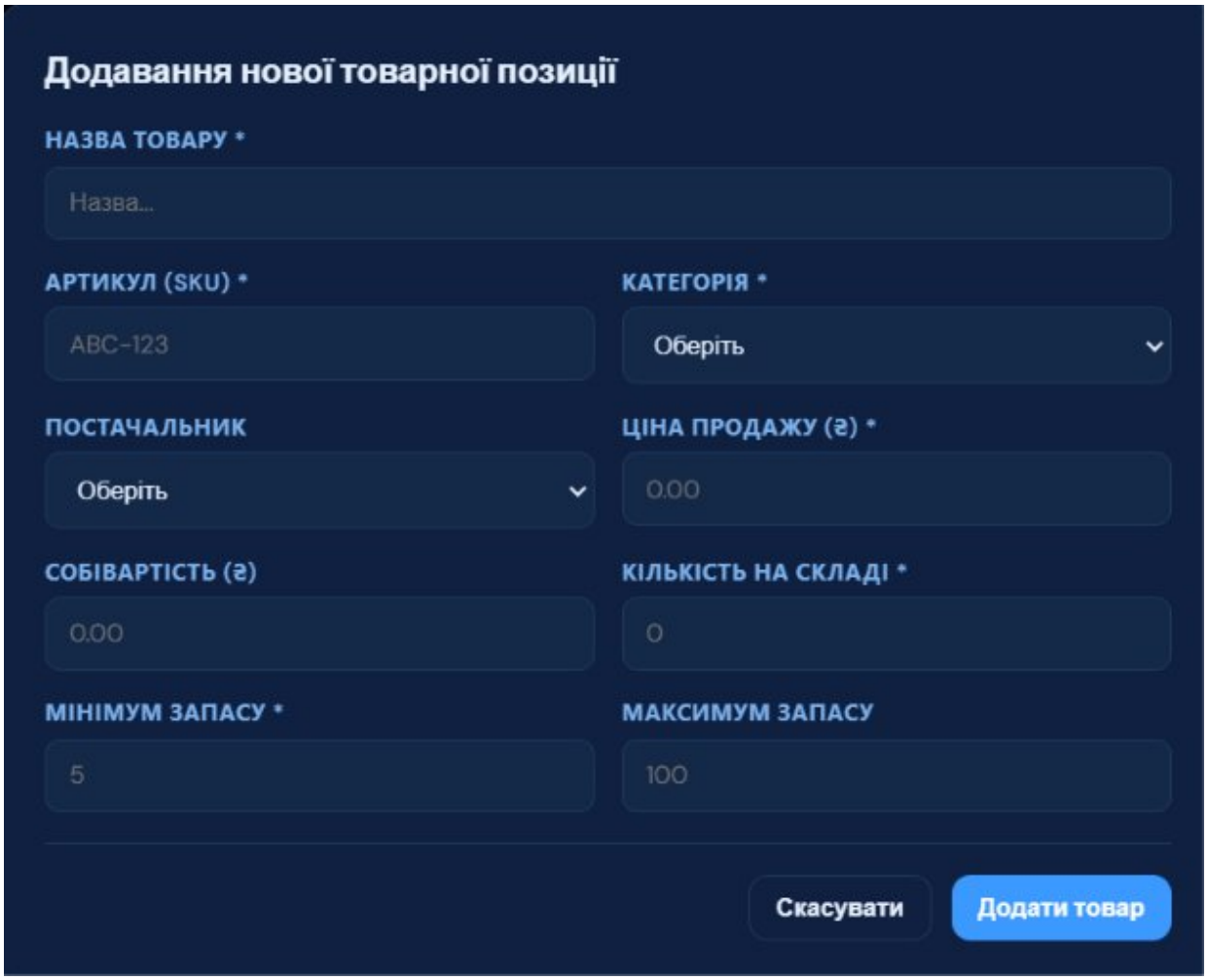


Рис 3.16 Модальне вікно додавання нової товарної позиції

Опис: Форма містить дев'ять полів введення, постачальник обирається зі списку зареєстрованих у системі постачальників. Після збереження товар з'являється у таблиці номенклатури, картки категорій та діаграми оновлюються автоматично.

Модальне вікно реєстрації нового контрагента включає поля: ПІБ або назва організації, тип контрагента з вибором із трьох варіантів, телефон, email, місто та відсоток індивідуальної знижки. Модальне вікно показано на рис. 3.17.

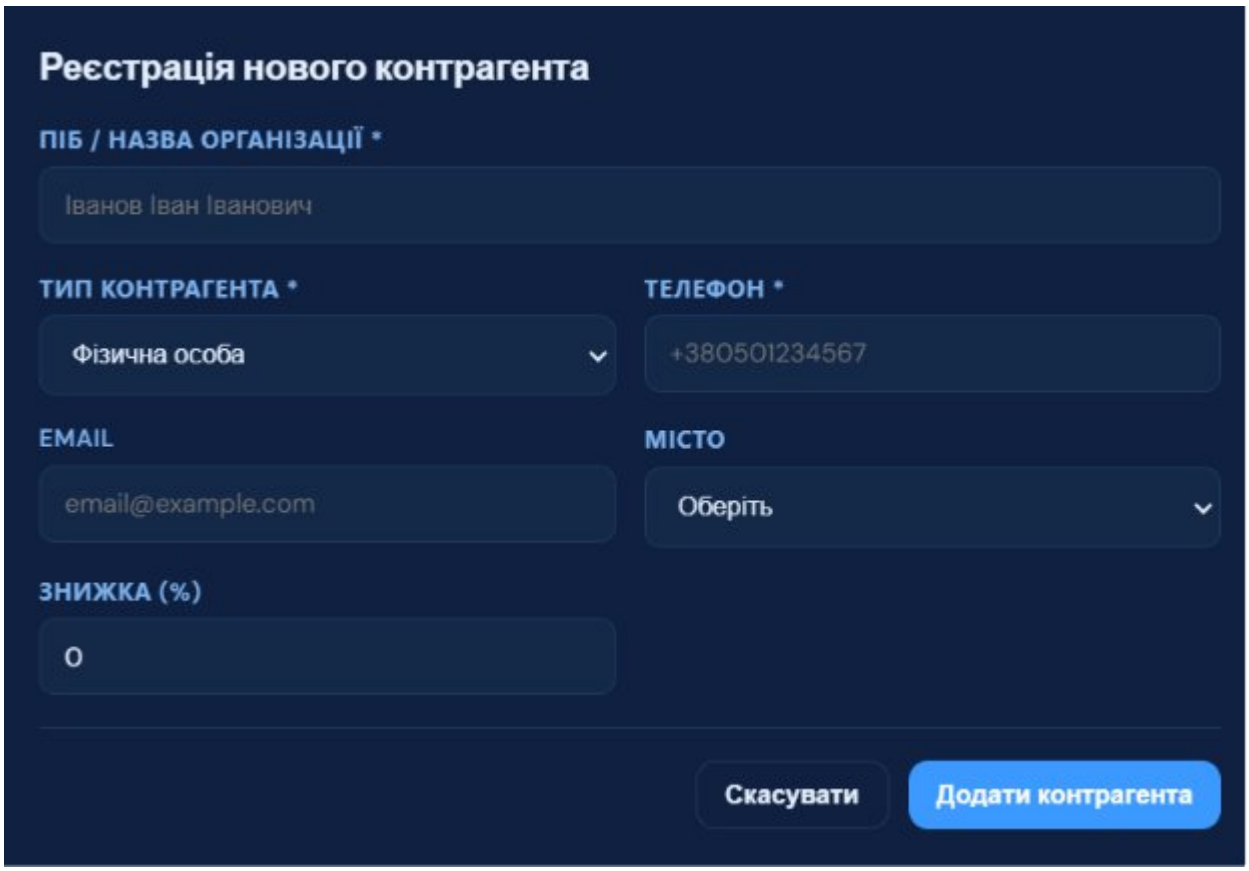


Рис 3.17 Модальне вікно реєстрації нового контрагента

Опис: Форма реєстрації контрагента забезпечує введення повних даних із визначенням типу через випадний список. Після збереження запис додається до масиву state.customers, таблиця контрагентів та KPI-картки оновлюються автоматично.

ВИСНОВОК

В результаті виконання цієї бакалаврської кваліфікаційної роботи успішно розроблено та реалізовано web-орієнтовану інформаційну систему управління малим торговельним підприємством ShopFlow. Досягнуто поставленої мети, створивши зручний, функціональний та практично застосовний інструмент, який забезпечує ефективну цифрову організацію торговельних процесів і сприяє підвищенню операційної ефективності підприємства. [1, 3]

У процесі дослідження та розробки було вирішено такі ключові завдання:

Проведено ґрунтовний аналіз предметної області управління малим торговельним підприємством та здійснено огляд існуючих програмних рішень. Це дозволило визначити функціональні переваги розроблюваної системи та сформувати вимоги до неї, зокрема акцентуючи на комплексному охопленні бізнес-процесів підприємства - від управління товарним каталогом і складськими залишками до обліку замовлень, клієнтської бази та фінансової аналітики.

Сформульовано детальні функціональні та нефункціональні вимоги, на основі яких розроблено технічне завдання. Воно охопило розширений набір можливостей: управління товарами та категоріями, оформлення та відстеження замовлень, ведення бази клієнтів, контроль складських залишків із автоматичним визначенням критичного рівня запасу, формування фінансових звітів та побудову аналітичних діаграм. [16]

Розроблено та реалізовано архітектуру системи у вигляді web-орієнтованого застосунку на основі HTML, CSS та JavaScript із серверною частиною на Node.js та Express.js і реляційною базою даних MySQL. Клієнтська частина забезпечує адаптивний та інтуїтивно зрозумілий інтерфейс із підтримкою темного і світлого режимів роботи.

Здійснено декомпозицію системи на три функціональні модулі - головна сторінка, модуль управління магазином та модуль аналітики, - що

підвищило її структурованість, полегшило розробку, тестування та подальший супровід.

Розроблено та наповнено реляційну базу даних MySQL із 12 таблицями, визначено первинні та зовнішні ключі, індекси, представлення та аналітичні запити. Проведено перевірку цілісності структури бази даних та коректності виконання запитів, що підтверджено результатами тестування.

Реалізовано аналітичний модуль із інтерактивними SVG-діаграмами, побудованими без використання сторонніх бібліотек, що забезпечило швидкодію системи та незалежність від зовнішніх залежностей.

Проведено комплексне тестування розробленої системи, яке підтвердило її функціональну придатність, коректність роботи бази даних та зручність використання персоналом підприємства.

Розроблена система надає підприємству такі переваги:

Скорочення часу на оперативне управління товарами, замовленнями та клієнтами завдяки єдиному інтерфейсу з інтуїтивною навігацією.

Підвищення ефективності контролю складських залишків завдяки автоматичному визначенню критичного рівня запасів із кольоровою індикацією статусу.

Оперативний доступ до аналітики продажів, фінансових показників та топу товарів для прийняття обґрунтованих управлінських рішень.

Гнучкість та персоналізація користувацького досвіду через адаптивний дизайн та підтримку темного і світлого режимів інтерфейсу.

Практичне значення роботи полягає у створенні готового до використання програмного продукту для автоматизації торговельних процесів малого підприємства. Функціонал системи є гнучким і може бути розширений, наприклад, шляхом підключення повноцінної серверної автентифікації з розмежуванням прав доступу для різних категорій персоналу, інтеграції з платіжними системами та сервісами доставки, впровадження модуля штрихкодування для автоматизації інвентаризації або

розгортання системи у хмарному середовищі для забезпечення віддаленого доступу. [12]

Таким чином, усі поставлені у вступі завдання бакалаврської кваліфікаційної роботи виконані в повному обсязі, а мета дослідження досягнута. Отримані результати демонструють практичну цінність розробленої системи та можуть слугувати основою для подальших наукових досліджень і комерційних розробок у галузі інформаційних технологій для автоматизації управління торговельними підприємствами.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Смолій В. М. Інформаційні системи та технології в управлінні підприємством : навчальний посібник. Київ : НУБіП України, 2024. 285 с.

2. Кравець О. В. Проектування інформаційних систем : підручник. Львів : Новий Світ-2000, 2023. 412 с.

3. Гавриленко В. В., Литвиненко О. В. Web-технології та розробка сучасних веб-додатків. Київ : Видавничий дім «Слово», 2024. 368 с.

4. Бойко А. С. Бази даних. Проектування та використання : навчальний посібник. Київ : КНЕУ, 2022. 456 с.

5. Fowler M. Patterns of Enterprise Application Architecture. Addison-Wesley, 2002. 560 p.

6. PostgreSQL Documentation. Version 16. URL: https://www.postgresql.org/docs/

7. Node.js Documentation. URL: https://nodejs.org/en/docs/

8. React – A JavaScript library for building user interfaces. URL: https://react.dev/

9. ISO/IEC/IEEE 12207:2017. Systems and software engineering - Software life cycle processes.

10. ДСТУ 4163-2020. Діловодство та архівна справа. Уніфікована система організаційно-розпорядчої документації. Вимоги до оформлювання документів. Київ : Держспоживстандарт України, 2020.

11. Закон України «Про електронні довірчі послуги» від 05.10.2017 № 2155-VIII.

12. Міністерство цифрової трансформації України. Концепція розвитку цифрової економіки та суспільства України 2030. URL:

13. Martin Fowler. Refactoring: Improving the Design of Existing Code. 2nd Edition. Addison-Wesley, 2018.

14. Ерік Еванс. Предметно-орієнтоване проєктування (DDD). Київ : https://thedigital.gov.ua/ Видавництво «Науковий світ», 2021.

15. Офіційна документація Express.js. URL: https://expressjs.com/ (дата звернення: 08.05.2026).

16. Савчук О. П. Автоматизація бізнес-процесів торговельних підприємств : монографія. Київ : ЦУЛ, 2023. 312 с.

17. Pressman R. S., Maxim B. R. Software Engineering: A Practitioner's Approach. 9th Edition. McGraw-Hill Education, 2020.

18. Django Documentation. URL: https://docs.djangoproject.com/ (дата звернення: 08.05.2026). [альтернатива для порівняння]

19. Nielsen J. Usability Engineering. Morgan Kaufmann, 1993. (класика UX/UI).

20. Державний стандарт України ДСТУ ISO/IEC 25010:2015. Системна та програмна інженерія. Вимоги до якості систем та програмного забезпечення.

21. https://github.com/vikayesip04/thesis

ДОДАТКИ

ДОДАТОК А (САЙТ) [21]

Лістинг А.1 – Глобальний об'єкт стану системи (state)

```
// -------------------- ДЕМО-ДАНІ --------------------
const products = [
  { id: 1, name: 'Футболка чорна XL', category: 'Одяг',
    price: 299, quantity: 45, minStock: 10, maxStock: 100, soldCount: 23 },
  { id: 2, name: 'Джинси сині 32', category: 'Одяг',
    price: 899, quantity: 8, minStock: 5, maxStock: 30, soldCount: 15 },
  { id: 3, name: 'Кросівки спортивні', category: 'Взуття',
    price: 1299, quantity: 12, minStock: 5, maxStock: 25, soldCount: 8 },
  { id: 4, name: 'Шапка чорна', category: 'Аксесуари',
    price: 199, quantity: 0, minStock: 5, maxStock: 20, soldCount: 32 },
  { id: 5, name: 'Гарнітура Bluetooth', category: 'Електроніка',
    price: 599, quantity: 3, minStock: 2, maxStock: 10, soldCount: 5 },
];

const customers = [
  { id: 1, fullName: 'Іван Петренко', phone: '+380501234567',
    email: 'ivan@example.com', joinDate: '2024-09-15', totalSpent: 4590 },
  { id: 2, fullName: 'Марія Шевченко', phone: '+380502345678',
    email: 'maria@example.com', joinDate: '2024-08-20', totalSpent: 2890 },
];

const orders = [
  { id: 1001, customerId: 1, customerName: 'Іван Петренко',
    items: 'Футболка, Джинси', amount: 1198,
    date: '2024-11-05', status: 'completed' },
  { id: 1002, customerId: 2, customerName: 'Марія Шевченко',
    items: 'Кросівки', amount: 1299,
    date: '2024-11-06', status: 'pending' },
];

// -------------------- STATE --------------------
let state = {
  products: [...products],
  customers: [...customers],
  orders: [...orders]
};
```

Лістинг А.2 - Функція додавання товару

```
modal.querySelector('#addProductForm')
  .addEventListener('submit', (e) => {
    e.preventDefault();
    const fd = new FormData(e.target);
    const newP = {
      id: state.products.length + 1,
      name: fd.get('name'),
      category: fd.get('category'),
      price: parseFloat(fd.get('price')),
      quantity: parseInt(fd.get('quantity')),
      minStock: parseInt(fd.get('minStock')),
      maxStock: parseInt(fd.get('maxStock')),
      soldCount: 0
    };
    state.products.push(newP);
    renderProductsTable();
    renderStats();
    renderInventory();
    modal.remove();
    notify('Товар успішно додано!');
  });
```

Лістинг А.3 - Функція створення замовлення з валідацією

```
form.addEventListener('submit', (e) => {
  e.preventDefault();
  const fd = new FormData(form);
  const customerId = parseInt(fd.get('customerId'));
  const productId = parseInt(fd.get('productId'));
  const customer = state.customers.find(c => c.id === customerId);
  const product = state.products.find(p => p.id === productId);
  const quantity = parseInt(fd.get('quantity'));
  const unitPrice = parseFloat(fd.get('unitPrice'));

  if (!customer || !product) {
    notify('Будь ласка, оберіть клієнта та товар', 'error');
    return;
  }

  const newO = {
    id: 1000 + state.orders.length + 1,
```

```
    customerId: customer.id,
    customerName: customer.fullName,
    items: product.name,
    amount: quantity * unitPrice,
    date: fd.get('date'),
    status: fd.get('status')
  };
  state.orders.push(newO);
  renderOrdersTable();
  renderStats();
  modal.remove();
  notify('Замовлення успішно створено!');
});
```

Лістинг А.4 - Автоматична підстановка ціни при виборі товару

```
productSelect.addEventListener('change', () => {
  const selectedOption =
    productSelect.options[productSelect.selectedIndex];
  priceInput.value =
    selectedOption.getAttribute('data-price') || '';
});
```

Лістинг А.5 - Визначення статусу складських залишків

```
state.products.forEach(p => {
  const tr = document.createElement('tr');
  const statusClass = p.quantity > p.minStock
    ? 'green' : 'yellow';
  const statusText = p.quantity > p.minStock
    ? 'Нормально' : 'Низький запас';
  tr.innerHTML = `
    <td><strong>${p.name}</strong></td>
    <td>${p.category}</td>
    <td>${p.minStock}</td>
    <td>${p.quantity}</td>
    <td>${p.maxStock}</td>
    <td><span class='pill ${statusClass}'>
      ${statusText}
    </span></td>`;
  tbody.appendChild(tr);
});
```

Лістинг А.6 - Перемикання теми та збереження у localStorage

```
(function() {
  const html = document.documentElement;
  const btn = document.getElementById('themeToggle');
  const saved = localStorage.getItem('shopflow-theme') || 'light';
  html.setAttribute('data-theme', saved);
  btn.textContent = saved === 'dark' ? ';

  btn.addEventListener('click', () => {
    const c = html.getAttribute('data-theme');
    const n = c === 'light' ? 'dark' : 'light';
    html.setAttribute('data-theme', n);
    localStorage.setItem('shopflow-theme', n);
    btn.textContent = n === 'dark' ? ';
  });
})();
```

Лістинг А.7 - Побудова SVG-діаграми доходів

```
function renderRevenueChart() {
  const svg = document.getElementById('revenueChart');
  const accent = getComputedStyle(document.documentElement)
    .getPropertyValue('--accent').trim();
  const text2 = getComputedStyle(document.documentElement)
    .getPropertyValue('--text2').trim();
  const border = getComputedStyle(document.documentElement)
    .getPropertyValue('--border').trim();

  const data = [3200,4100,2800,5200,4700,6100,3900,5500,
    6800,4200,5900,7100,4500,6200,5300,7400,4800,6600,
    5100,7900,5600,6300,4900,7200,5800,6700,4600,7500,6100,8200];

  const W = 900, H = 220, padL = 48, padR = 12,
        padT = 12, padB = 36;
  const cW = W - padL - padR, cH = H - padT - padB;
  const max = Math.max(...data);
  const barW = (cW / data.length) * 0.6;
  const gap = cW / data.length;
```

```
  let content = `<defs><linearGradient id="barGrad"
    x1="0" y1="0" x2="0" y2="1">
    <stop offset="0%" stop-color="${accent}" stop-opacity="1"/>
    <stop offset="100%" stop-color="${accent}" stop-opacity="0.4"/>
  </linearGradient></defs>`;

  // Лінії сітки
  [0, 0.25, 0.5, 0.75, 1].forEach(t => {
    const y = padT + cH * (1 - t);
    content += `<line x1="${padL}" y1="${y}"
      x2="${W - padR}" y2="${y}"
      stroke="${border}" stroke-width="1"/>`;
    if (t > 0) content += `<text x="${padL - 6}" y="${y + 4}"
      text-anchor="end" fill="${text2}" font-size="10">
      ${Math.round(max * t / 1000)}k</text>`;
  });

  // Стовпці
  data.forEach((v, i) => {
    const barH = (v / max) * cH;
    const x = padL + i * gap + (gap - barW) / 2;
    const y = padT + cH - barH;
    content += `<rect x="${x}" y="${y}" width="${barW}"
      height="${barH}" rx="4"
      fill="url(#barGrad)" opacity="0.9">
      <title>₴ ${v.toLocaleString()}</title>
    </rect>`;
  });

  svg.innerHTML = content;
}
```

ДОДАТОК Б

База даних [21]

Лістинг Б

Лістинг Б.1 - Перевірка первинних ключів таблиць (рис. В.1)

```
SELECT
   TABLE_NAME AS 'Таблиця',
   COLUMN_NAME AS 'Колонка',
   CONSTRAINT_NAME AS 'Обмеження'
FROM INFORMATION_SCHEMA.KEY_COLUMN_USAGE
WHERE CONSTRAINT_SCHEMA = DATABASE()
  AND CONSTRAINT_NAME = 'PRIMARY'
ORDER BY TABLE_NAME;
```

Лістинг Б.2 - Перевірка зовнішніх ключів між таблицями (рис. В.2)

```
SELECT
   TABLE_NAME AS 'Таблиця',
   COLUMN_NAME AS 'Колонка FK',
   CONSTRAINT_NAME AS 'Назва FK',
   REFERENCED_TABLE_NAME AS 'Батьківська таблиця',
   REFERENCED_COLUMN_NAME AS 'Батьківська колонка'
FROM INFORMATION_SCHEMA.KEY_COLUMN_USAGE
WHERE CONSTRAINT_SCHEMA = DATABASE()
  AND REFERENCED_TABLE_NAME IS NOT NULL
ORDER BY TABLE_NAME;
```

Лістинг Б.3 - Аналіз замовлень та суми продажів по клієнтах (рис. В.3)

```
SELECT
   CONCAT(c.first_name, ' ', c.last_name) AS 'Клієнт',
   c.city AS 'Місто',
   c.customer_type AS 'Тип',
   COUNT(o.id) AS 'Кількість замовлень',
   SUM(o.final_amount) AS 'Загальна сума',
   ROUND(AVG(o.final_amount), 2) AS 'Середній чек'
FROM Customers c
LEFT JOIN Orders o ON c.id = o.customer_id
GROUP BY c.id, c.first_name, c.last_name,
       c.city, c.customer_type
ORDER BY SUM(o.final_amount) DESC;
```

Лістинг Б.4 - Залишки товарів на складі зі статусом поповнення (рис. В.4)

```
SELECT
```

```
    p.name AS 'Товар',
    p.sku AS 'Артикул',
    cat.name AS 'Категорія',
    i.quantity_in_stock AS 'На складі',
    i.quantity_reserved AS 'Зарезервовано',
    i.quantity_available AS 'Доступно',
    i.min_stock_level AS 'Мінімум',
    i.warehouse_location AS 'Місце',
    CASE
        WHEN i.quantity_available <= i.min_stock_level
            THEN '⚠ Потрібне поповнення'
        ELSE 'OK'
    END AS 'Статус'
FROM Inventory i
JOIN Products p ON i.product_id = p.id
LEFT JOIN Categories cat ON p.category_id = cat.id
ORDER BY i.quantity_available ASC;
```

Лістинг Б.5 - Аналіз продажів по каналах та способах оплати (рис. В.5)

```
SELECT
    s.sales_channel AS 'Канал продажів',
    s.payment_method AS 'Спосіб оплати',
    COUNT(s.id) AS 'Кількість продажів',
    SUM(s.amount) AS 'Загальна сума',
    ROUND(AVG(s.amount), 2) AS 'Середня сума'
FROM Sales s
GROUP BY s.sales_channel, s.payment_method
ORDER BY SUM(s.amount) DESC;
```

Лістинг Б.6 - Статус доставки замовлень із деталями (рис. В.6)

```
SELECT
    o.id AS 'Замовлення №',
    CONCAT(c.first_name, ' ', c.last_name) AS 'Клієнт',
    o.final_amount AS 'Сума',
    d.delivery_method AS 'Метод доставки',
    d.tracking_number AS 'Трекінг',
    d.status AS 'Статус доставки',
    d.city AS 'Місто',
    d.estimated_date AS 'Очікувана дата',
    d.delivery_cost AS 'Вартість доставки'
FROM Orders o
```

```
JOIN Customers c ON o.customer_id = c.id
LEFT JOIN Delivery d ON o.id = d.order_id
ORDER BY o.id;
```

Лістинг Б.7 - Фінансовий звіт по доходах і витратах (рис. В.7)

```
SELECT
   DATE_FORMAT(transaction_date, '%Y-%m') AS 'Місяць',
   transaction_type AS 'Тип операції',
   category AS 'Категорія',
   COUNT(*) AS 'Кількість операцій',
   SUM(amount) AS 'Загальна сума'
FROM Transactions
GROUP BY DATE_FORMAT(transaction_date, '%Y-%m'),
       transaction_type, category
ORDER BY `Місяць`, transaction_type;
```

Лістинг Б.8 - Топ товарів за кількістю продажів та валовим прибутком (рис. В.8)

```
SELECT
   p.name AS 'Товар',
   p.sku AS 'Артикул',
   cat.name AS 'Категорія',
   SUM(oi.quantity) AS 'Продано (шт)',
   SUM(oi.total_price) AS 'Виручка',
   ROUND(SUM(oi.total_price)
      - SUM(oi.quantity * p.cost_price), 2)
      AS 'Валовий прибуток'
FROM Order_Items oi
JOIN Products p ON oi.product_id = p.id
LEFT JOIN Categories cat ON p.category_id = cat.id
GROUP BY p.id, p.name, p.sku, cat.name
ORDER BY SUM(oi.quantity) DESC;
```

Лістинг Б.9 - Статистика кількості записів у таблицях (рис. В.9)

```
SELECT
   table_name AS 'Таблиця',
   table_rows AS 'Кількість записів'
FROM information_schema.tables
WHERE table_schema = DATABASE()
  AND table_type = 'BASE TABLE'
ORDER BY table_rows DESC;
```

ДОДАТОК В (ФОТО З РЕЗУЛЬТАТУ БД)

Таблиця	Колонка	Обмеження
categories	id	PRIMARY
customers	id	PRIMARY
delivery	id	PRIMARY
inventory	id	PRIMARY
order_items	id	PRIMARY
orders	id	PRIMARY
payments	id	PRIMARY
products	id	PRIMARY
sales	id	PRIMARY
suppliers	id	PRIMARY
transactions	id	PRIMARY
users	id	PRIMARY

Рис В.1 Результат перевірки первинних ключів таблиць бази даних

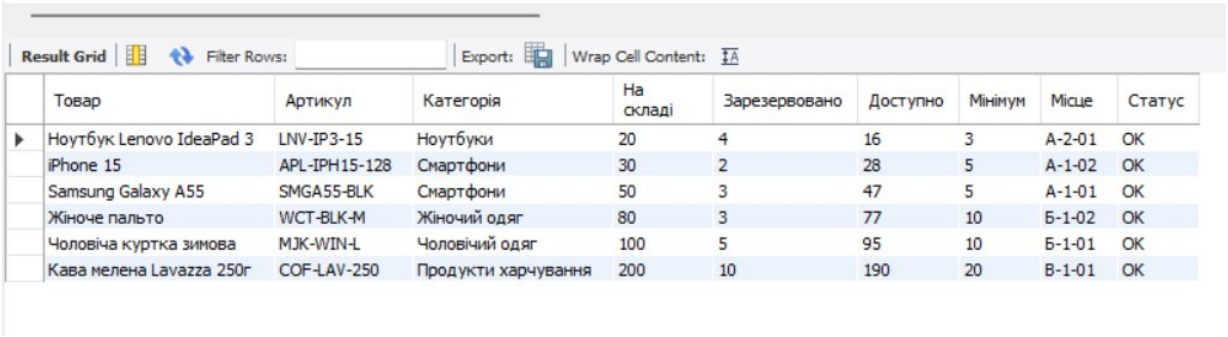
Таблиця	Колонка FK	Назва FK	Батьківська таблиця	Батьківська колонка
categories	parent_id	categories_ibfk_1	categories	id
delivery	order_id	delivery_ibfk_1	orders	id
inventory	product_id	inventory_ibfk_1	products	id
order_items	order_id	order_items_ibfk_1	orders	id
order_items	product_id	order_items_ibfk_2	products	id
orders	customer_id	orders_ibfk_1	customers	id
orders	user_id	orders_ibfk_2	users	id
payments	order_id	payments_ibfk_1	orders	id
products	category_id	products_ibfk_1	categories	id
products	supplier_id	products_ibfk_2	suppliers	id
sales	order_id	sales_ibfk_1	orders	id
sales	user_id	sales_ibfk_2	users	id
transactions	user_id	transactions_ibfk_1	users	id

Рис В.2 Результат перевірки зовнішніх ключів між таблицями бази даних

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Клієнт	Місто	Тип	Кількість замовлень	Загальна сума	Середній чек
ТОВ Бізнес Плюс Корпоративний	Дніпро	Юридична особа	1	44998.00	44998.00
Андрій Степаненко	Харків	Фізична особа	1	37999.00	37999.00
Олена Ковальчук	Київ	Фізична особа	1	14999.00	14999.00
Наталія Кравець	Львів	Фізична особа	1	5698.00	5698.00
ФОП Іваненко Сервіс	Одеса	ФОП	1	189.00	189.00

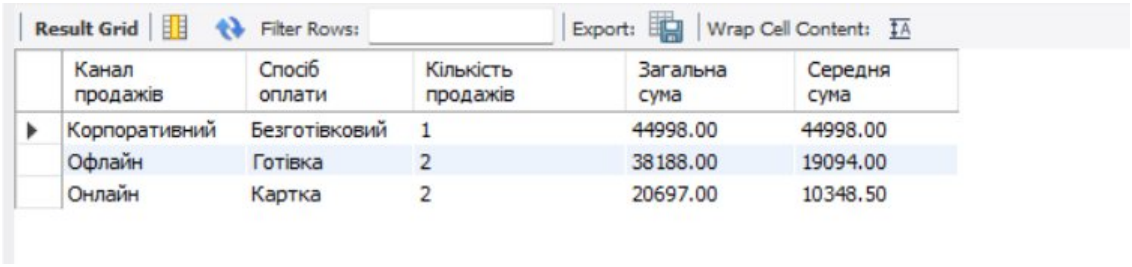
Рис В.3 Результат запиту аналізу замовлень та суми продажів по клієнтах



Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Товар	Артикул	Категорія	На складі	Зарезервовано	Доступно	Мінімум	Місце	Статус
Ноутбук Lenovo IdeaPad 3	LNV-IP3-15	Ноутбуки	20	4	16	3	А-2-01	OK
iPhone 15	APL-IPH15-128	Смартфони	30	2	28	5	А-1-02	OK
Samsung Galaxy A55	SMGA55-BLK	Смартфони	50	3	47	5	А-1-01	OK
Жіноче пальто	WCT-BLK-M	Жіночий одяг	80	3	77	10	Б-1-02	OK
Чоловіча куртка зимова	MJK-WIN-L	Чоловічий одяг	100	5	95	10	Б-1-01	OK
Кава мелена Lavazza 250г	COF-LAV-250	Продукти харчування	200	10	190	20	В-1-01	OK

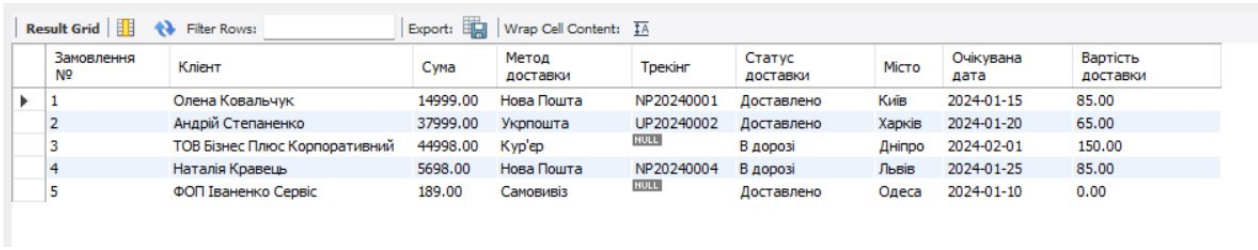
Рис В.4 Результат запиту залишків товарів на складі зі статусом поповнення



Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Канал продажів	Спосіб оплати	Кількість продажів	Загальна сума	Середня сума
Корпоративний	Безготівковий	1	44998.00	44998.00
Офлайн	Готівка	2	38188.00	19094.00
Онлайн	Картка	2	20697.00	10348.50

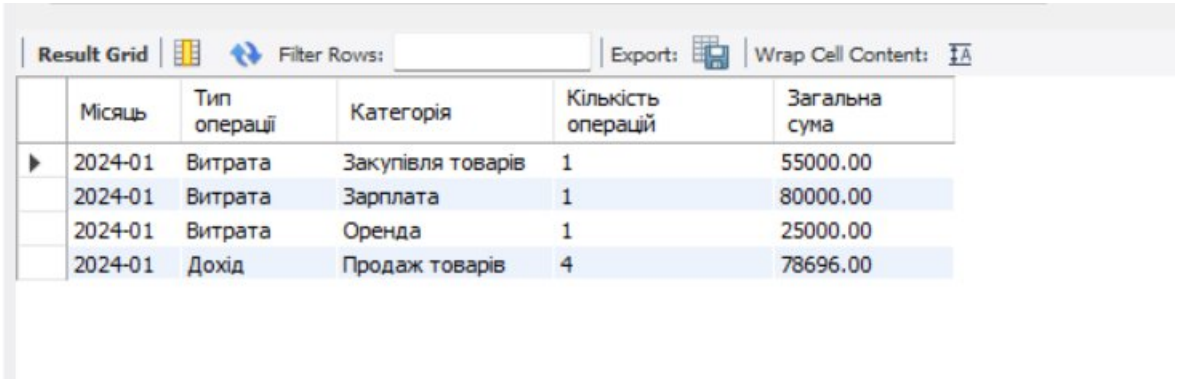
Рис В.5 Результат запиту аналізу продажів по каналах та способах оплати



Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Замовлення №	Клієнт	Сума	Метод доставки	Трекінг	Статус доставки	Місто	Очікувана дата	Вартість доставки
1	Олена Ковальчук	14999.00	Нова Пошта	NP20240001	Доставлено	Київ	2024-01-15	85.00
2	Андрій Степаненко	37999.00	Укрпошта	UP20240002	Доставлено	Харків	2024-01-20	65.00
3	ТОВ Бізнес Плюс Корпоративний	44998.00	Кур'єр	NULL	В дорозі	Дніпро	2024-02-01	150.00
4	Наталія Кравець	5698.00	Нова Пошта	NP20240004	В дорозі	Львів	2024-01-25	85.00
5	ФОП Іваненко Сервіс	189.00	Самовивіз	NULL	Доставлено	Одеса	2024-01-10	0.00

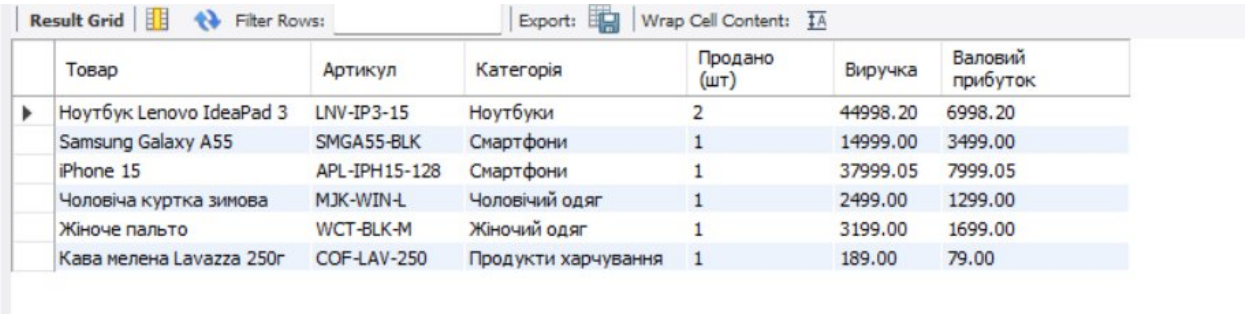
Рис В.6 Результат запиту статусу доставки замовлень із деталями



Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Місяць	Тип операції	Категорія	Кількість операцій	Загальна сума
2024-01	Витрата	Закупівля товарів	1	55000.00
2024-01	Витрата	Зарплата	1	80000.00
2024-01	Витрата	Оренда	1	25000.00
2024-01	Дохід	Продаж товарів	4	78696.00

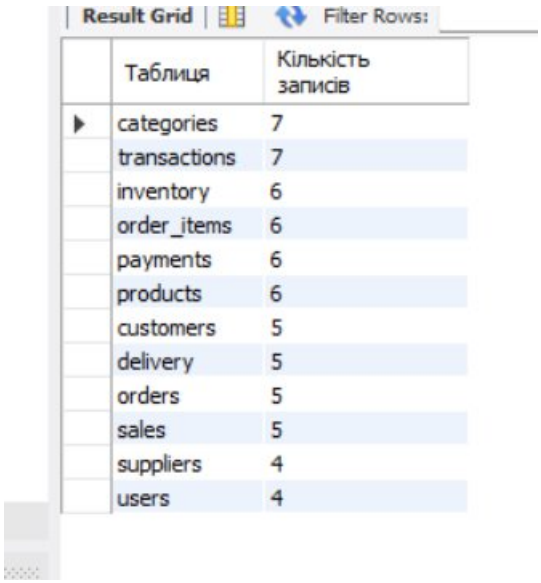
Рис В.7 Результат фінансового звіту по доходах і витратах за січень 2024 року



Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Товар	Артикул	Категорія	Продано (шт)	Виручка	Валовий прибуток
Ноутбук Lenovo IdeaPad 3	LNV-IP3-15	Ноутбуки	2	44998.20	6998.20
Samsung Galaxy A55	SMGA55-BLK	Смартфони	1	14999.00	3499.00
iPhone 15	APL-IPH15-128	Смартфони	1	37999.05	7999.05
Чоловіча куртка зимова	MJK-WIN-L	Чоловічий одяг	1	2499.00	1299.00
Жіноче пальто	WCT-BLK-M	Жіночий одяг	1	3199.00	1699.00
Кава мелена Lavazza 250г	COF-LAV-250	Продукти харчування	1	189.00	79.00

Рис В.8 Результат запиту топу товарів за кількістю продажів та валовим прибутком



Result Grid | Filter Rows:

Таблиця	Кількість записів
categories	7
transactions	7
inventory	6
order_items	6
payments	6
products	6
customers	5
delivery	5
orders	5
sales	5
suppliers	4
users	4

Рис В.9 Статистика кількості записів у таблицях бази даних

ДОДАТОК Г (ІНФОРМАЦІЯ ПРО ДЕМО)

Демонстраційна версія системи управління малим торговельним підприємством ShopFlow

Демонстраційна версія системи реалізована у файлі demo.html та є повнофункціональним прототипом системи управління магазином, що працює повністю у браузері без необхідності встановлення додаткового програмного забезпечення.

Технічні відомості

Файл: demo.html Технології: HTML5, CSS3, JavaScript (ES6+) Запуск: відкрити файл demo.html у будь-якому сучасному браузері (Chrome, Firefox, Edge) Залежності: відсутні - жодних зовнішніх бібліотек не потрібно

Структура демо-версії

Демо містить повний набір функцій системи, розподілених по шести вкладках бічної панелі навігації:

Дашборд - відображає чотири KPI-картки у реальному часі: товарів у каталозі, замовлень сьогодні, кількість клієнтів та кількість замовлень на обробці. Показники оновлюються автоматично після кожної операції. Містить таблицю останніх замовлень та блок швидких дій із кнопками «Новий клієнт» та «Нове замовлення».

Товари - таблиця каталогу товарів із колонками назва, категорія, ціна та запас. Кнопка «Додати товар» відкриває модальне вікно з формою, що містить поля: назва товару, категорія (Одяг, Взуття, Аксесуари, Електроніка, Інше), ціна, кількість на складі, мінімум та максимум запасу. Після збереження товар миттєво з'являється у таблиці, а KPI-картка оновлюється.

Замовлення - таблиця замовлень із колонками: номер, клієнт, товари, сума, дата та статус із кольоровими мітками (зелена - «Доставлено», жовта - «Обробляється», червона - «Скасовано»). Кнопка «Нове замовлення» відкриває форму з динамічними списками клієнтів і товарів та автоматичною підстановкою ціни при виборі товару.

Клієнти - таблиця бази клієнтів із контактними даними та датою реєстрації. Кнопка «Додати клієнта» відкриває форму з полями: повне ім'я, телефон, email та місто.

Інвентаризація - таблиця складських залишків із автоматичним визначенням статусу для кожного товару: «Достатньо» (зелена мітка) якщо запас вище норми, «Низький запас» (жовта) якщо наближається до мінімуму, «Критичний» (червона) якщо запас на або нижче мінімального рівня.

Звіти - зведений фінансовий звіт із чотирма показниками: загальний дохід від завершених замовлень, загальна кількість замовлень, середній чек та вартість складських запасів (розраховується як сума добутків ціни та кількості по кожному товару). Розбивка продажів за категоріями товарів.

Початкові тестові дані

Демо-версія завантажується з попередньо заповненими тестовими даними:

Товари - 5 позицій: футболка базова (Одяг, 599 грн, 45 шт), кросівки Nike Air (Взуття, 2 999 грн, 12 шт), рюкзак міський (Аксесуари, 1 299 грн, 8 шт), навушники Sony (Електроніка, 3 499 грн, 15 шт), куртка зимова (Одяг, 4 299 грн, 6 шт).

Замовлення - 3 записи з різними статусами: доставлено, обробляється, скасовано.

Клієнти - 3 записи: Олена Ковальчук (Київ), Максим Дяченко (Харків), Ірина Бондаренко (Одеса).

Сповіщення

Після кожної успішної операції у правому верхньому куті екрана з'являється анімоване сповіщення: зелене - при успішному додаванні («Товар успішно додано!», «Замовлення успішно створено!», «Клієнта успішно додано!»), червоне - при помилці валідації («Будь ласка, оберіть клієнта та товар»). Сповіщення зникає автоматично через 3 секунди.

Обмеження демо-версії

Дані зберігаються лише у межах поточної сесії браузера - після перезавантаження сторінки повертаються початкові тестові дані. Для повноцінного збереження даних між сесіями необхідне підключення серверної частини на Node.js із базою даних MySQL, що є наступним етапом розвитку системи.

НАЦІОНАЛЬНИЙ УНІВЕРСТИТЕТ БІОРЕСУРСІВ І ПРИРОДОКОРИСТУВАННЯ УКРАЇНИ
ФАКУЛЬТЕТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

Декларація про академічну доброчесність та використання ШІ

ДЕКЛАРАЦІЯ

Я, Єсіп Вікторія Іванівна, здобувач(ка) НУБіП України, усвідомлюючи відповідальність за порушення академічної доброчесності, цією заявою підтверджую, що:

1. Моя бакалаврська кваліфікаційна робота (проєкт) на тему

«Розробка web орієнтованої інформаційної системи управління малим торговельним підприємством» є результатом моєї особистої праці.

2. Усі запозичення з текстів інших авторів мають відповідні посилання згідно з чинними стандартами.

3. Щодо використання інструментів ШІ зазначаю, що (обрати необхідне):

- [0] інструменти генеративного ШІ не використовувалися.
- [0] інструменти ШІ (ChatGPT, Claude) були використані для:
  - [0] перекладу іншомовних джерел;
  - [ ] стилістичного редагування та перевірки граматики;
  - [ ] допомоги у написанні/відлагодженні програмного коду;
  - [ ] інше: _____.

Я розумію, що надання недостовірної інформації може призвести до скасування результатів захисту та відрахування з числа здобувачів НУБіП України.

«__» _____2026**р.** _____ ЄСІП Вікторія
(підпис) **(Ім'я та ПРІЗВИЩЕ)**