Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет Інформаційно-аналітичних технологій та менеджменту
(повна назва)
Кафедра Інформатики
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти перший (бакалаврський)

ПРОЄКТУВАННЯ ТА РОЗРОБЛЕННЯ ІНФОРМАЦІЙНОЇ СИСТЕМИ УПРАВЛІННЯ ЗАМОВЛЕННЯМИ ДРОПШИПІНГПЛАТФОРМИ З МОДУЛЕМ ІНТЕЛЕКТУАЛЬНОГО ОБРОБЛЕННЯ КОНТЕНТУ
(тема)

Виконав:
здобувач 4 року навчання,
групи ІТІНФ-22-1
Приходько Д.В.
(прізвище, ініціали)

Спеціальність 122 Комп'ютерні науки
(код і повна назва спеціальності)

Тип програми освітньо-професійна

Освітня програма Інформатика
(повна назва освітньої програми)

Керівник ст. викл. Путятіна О. Є.
(посада, прізвище, ініціали)

Допускається до захисту

Завідувач кафедри інформатики _____ Кобилін О. А.
(підпис) (прізвище, ініціали)

2026 р.

Харківський національний університет радіоелектроніки

Факультет Інформаційно-аналітичних технологій та менеджменту

Кафедра Інформатики

Рівень вищої освіти перший (бакалаврський)

Спеціальність 122 Комп'ютерні науки
(код і повна назва)

Тип програми освітньо-професійна

Освітня програма Інформатика
(повна назва)

ЗАТВЕРДЖУЮ:
Зав. кафедри _____
(підпис)
«____»_____ 2026 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

здобувачеві Приходьку Денису Валерійовичу
(прізвище, ім'я, по батькові)

1. Тема роботи Проєктування та розробка інформаційної системи управління замовленнями дропшипінг-платформи з модулем інтелектуальної обробки контенту

затверджена наказом університету від 26 травня 2026 року № 435Ст
2. Термін подання здобувачем роботи до екзаменаційної комісії 28 червня 2026 р.
3. Вихідні дані до роботи науково-методична та науково-технічна література, матеріали конференцій, статті з питань CRM, електронної комерції, дропшипінгу, інтелектуальної обробки контенту, програмний код FastAPI-застосунку та матеріали репозиторію.

4. Перелік питань, що потрібно опрацювати в роботі
1. Теоретичні засади CRM/e-commerce систем і специфіка дропшипінг-процесу.
2. Проєктування інформаційної системи управління замовленнями та товарним контентом.
3. Реалізація, тестування та аналіз результатів роботи CRM/e-commerce вебзастосунку з модулем інтелектуальної обробки контенту.

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій (п.5 включається до завдання за рішенням випускової кафедри) архітектура системи, workflow обробки замовлення, ER-модель бази даних, pipeline модуля інтелектуальної обробки товарного контенту, модель авторизації та ізоляції даних, скриншоти інтерфейсу CRM-системи.

6. Консультанти розділів роботи (п.6 включається до завдання за наявності консультантів згідно з наказом, зазначеним у п.1)

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

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів роботи	Строк / терміни виконання етапів роботи	Примітка
1	Отримання завдання на кваліфікаційну роботу	13.04.2026	
2	Аналіз завдання, підбір літератури	14.04.26-16.04.26	
3	Аналіз літератури з проблеми, що вирішується	17.04.26-19.04.26	
4	Аналіз існуючих методів управління замовлень	20.04.26-22.04.26	
5	Формування кроків вирішення проблеми	23.04.26-05.05.26	
6	Програмна реалізація	26.04.26-20.05.26	
7	Аналіз отриманих результатів	21.05.26-04.06.26	
8	Оформлення пояснювальної записки	05.06.26-09.06.26	
9	Перевірка на нормоконтроль	10.06.26-19.06.26	
10	Перевірка на плагіат	11.06.26-30.06.26	
11	Рецензування	20.06.26-30.06.26	
12	Підготовка презентації та доповіді	20.06.26-30.06.26	
13	Занесення роботи в електронний архів	30.06.26-09.07.26	
14	Попередній захист кваліфікаційної роботи	30.06.26-09.07.26	

Дата видачі завдання 13 квітня 2026 р.

Здобувач _____
(підпис)

Керівник роботи _____ ст. викл. Путятіна О. Є.
(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ/ABSTRACT

Пояснювальна записка до кваліфікаційної роботи: 71 с., 19 табл., 13 рис., 30 джерел.

ДРОПШИПІНГ, CRM-СИСТЕМА, ЕЛЕКТРОННА КОМЕРЦІЯ, FASTAPI, JINJA, SQLITE, ШТУЧНИЙ ІНТЕЛЕКТ, GEMINI, REMBG, ТОВАРНИЙ КОНТЕНТ.

Об’єктом роботи є інформаційна підтримка управління замовленнями, клієнтськими даними та товарним контентом у дропшипінг-платформі.

Мета роботи - проєктування та розробка CRM/e-commerce вебзастосунку з модулем інтелектуальної обробки контенту для товарних карток.

Використано методи системного аналізу, реляційного моделювання, веброзробки, API-інтеграції, обробки зображень і генерації структурованого опису.

Практична цінність полягає у робочому прототипі на FastAPI, Jinja та SQLite з автентифікацією, ізоляцією даних, CRM-сутностями й ШІ-помічником.

Область застосування: навчальні CRM-проєкти, невеликі e-commerce команди, дропшипінг-продажі та прототипування товарних сервісів.

DROPSHIPPING, CRM SYSTEM, E-COMMERCE, FASTAPI, JINJA, SQLITE, ARTIFICIAL INTELLIGENCE, GEMINI, REMBG, PRODUCT CONTENT.

The object of the work is information support for order management, customer data, and product content in a dropshipping platform.

The goal is to design and develop a CRM/e-commerce web application with an intelligent content processing module for product cards.

The work uses system analysis, relational modeling, web development, API integration, image processing, and structured description generation.

The practical value is a working FastAPI, Jinja, and SQLite prototype with authentication, data isolation, CRM entities, and an AI assistant.

Applications: educational CRM projects, small e-commerce teams, dropshipping sales, and prototyping of product content services.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

ШІ – штучний інтелект

СКБД – система керування базами даних

БД – база даних

API – Application Programming Interface (інтерфейс програмного застосунку, використовується для обміну даними між клієнтом і сервером)

B2C – Business-to-Consumer (модель продажу товарів і послуг від бізнесу кінцевому споживачу)

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

CRM – Customer Relationship Management (управління взаємовідносинами з клієнтами)

CRUD – Create, Read, Update, Delete (чотири базові операції роботи з даними)

CPU – Central Processing Unit (центральний процесор)

ER – Entity-Relationship (модель «сутність–зв’язок» для проєктування баз даних)

HTML – HyperText Markup Language (мова розмітки гіпертексту)

HTTP – HyperText Transfer Protocol (протокол передачі гіпертексту)

HTTPS – HyperText Transfer Protocol Secure (захищений протокол передачі гіпертексту)

JSON – JavaScript Object Notation (текстовий формат обміну даними на основі JavaScript)

MVP – Minimum Viable Product (мінімально життєздатний продукт)

OS – Operating System (операційна система)

PNG – Portable Network Graphics (растровий формат зображень із підтримкою прозорості)

PaaS – Platform as a Service (платформа як послуга, модель хмарних обчислень)

RAM – Random Access Memory (оперативна пам’ять)

REST – Representational State Transfer (архітектурний стиль проєктування програмних інтерфейсів)

SaaS – Software as a Service (програмне забезпечення як послуга)

SQL – Structured Query Language (мова структурованих запитів для керування базами даних)

UI – User Interface (користувацький інтерфейс)

URL – Uniform Resource Locator (уніфікований локатор ресурсу)

ВСТУП

Дропшипінг як модель електронної комерції залежить від швидкого обміну інформацією між продавцем, клієнтом і постачальником. У такій моделі продавець не обов’язково утримує власний склад, однак саме він приймає замовлення, готує товарну пропозицію, підтримує комунікацію з покупцем і контролює виконання операційних дій. Через це якість інформаційної системи прямо впливає на швидкість обробки звернень, точність даних про замовлення та стабільність роботи з клієнтською базою [1].

Актуальність теми визначається поширенням CRM-систем у різних галузях, зростанням ролі електронної комерції та потребою малих торговельних команд у доступних інструментах для керування замовленнями. Якщо робота з клієнтами, товарами, задачами й контентом ведеться в окремих таблицях, месенджерах і графічних редакторах, виникають дублювання записів, втрата історії взаємодії та затримки під час підготовки товарних карток. Для дропшипінг-платформи ці проблеми є особливо відчутними, оскільки швидкість публікації товару й коректність замовлення часто визначають результат продажу [2].

Тема кваліфікаційної роботи сформульована так: «Проєктування та розробка інформаційної системи управління замовленнями дропшипінг-платформи з модулем інтелектуальної обробки контенту». У роботі ця тема розкрита через конкретний вебзастосунок: FastAPI обробляє маршрути й форми, Jinja-шаблони формують CRM-інтерфейс, SQLite зберігає записи користувача, а Docker Compose фіксує середовище запуску. У поточній реалізації є реєстрація та вхід, персональні записи товарів, клієнтів, замовлень і задач, а також два сервіси для товарної картки: видалення фону зображення та генерація структурованого опису. Тому робота описує не абстрактну платформу, а реалізоване CRM/e-commerce ядро з чітко окресленими межами [3].

Практична значущість роботи пов’язана з тим, що розроблювана система не обмежується демонстрацією окремого ШІ-модуля. Інтелектуальна обробка контенту включена в ширший операційний сценарій: товарна картка може бути підготовлена швидше, а потім використана в замовленні, пов’язаному з конкретним клієнтом. Такий підхід відповідає реальній логіці роботи невеликої дропшипінг-команди, для якої важлива не лише генерація опису, а й подальше ведення клієнтських і замовних даних в одному середовищі.

Структурно кваліфікаційна робота складається зі вступу, трьох розділів, висновків і переліку джерел посилання. Перший розділ пояснює, чому для дропшипінгу потрібне спільне середовище для клієнтів, товарів, замовлень і задач, а також порівнює готові CRM/e-commerce рішення з власною навчально-прикладною реалізацією. Другий розділ зосереджено на товарному контенті: як rembg очищує зображення, як Gemini формує поля опису, за якими критеріями оцінювати результат і де залишаються ризики помилок. Третій розділ переходить до коду й інтерфейсу: описує функції застосунку, архітектуру FastAPI/Jinja/SQLite, модель даних, сценарії менеджера та перевірку працездатності без приписування системі нереалізованих інтеграцій.

1 СУЧАСНИЙ СТАН І ТЕНДЕНЦІЇ РОЗВИТКУ CRM/E-COMMERCE СИСТЕМ ДЛЯ ДРОПШИПІНГ-ПЛАТФОРМ

1.1 Теоретичні основи та динаміка розвитку CRM-систем у керуванні продажами

Управління відносинами з клієнтами (Customer Relationship Management, CRM) у сучасній науковій літературі розглядається не лише як програмний продукт, а як управлінська концепція, що поєднує бізнес-процеси, клієнтські дані, канали взаємодії та технології підтримки рішень. Payne і Frow визначають CRM як стратегічну рамку, у якій цінність створюється через узгодження процесів компанії з потребами клієнта, а не через просте накопичення контактів [4]. У прикладних підходах CRM також описують як систему процесів і технологій, що підтримує продажі, сервіс і повторну взаємодію з покупцем [5]. Для теми кваліфікаційної роботи це важливо, оскільки розроблювана система має підтримувати не ізольовану таблицю замовлень, а пов’язаний процес роботи з клієнтом, товаром, задачею менеджера й товарним контентом.

Історично CRM-системи пройшли шлях від електронних картотек до операційних вебплатформ. На першому етапі вони замінювали паперові носії та зберігали контакти, телефони, адреси й короткі нотатки. На другому етапі CRM почали підтримувати процес продажу: статуси угод, задачі для менеджерів, нагадування, історію комунікацій. На третьому етапі CRM інтегрувалися з електронною комерцією, поштою, рекламними кабінетами, месенджерами та аналітичними інструментами [6]. Унаслідок цього CRM перестала бути окремою базою контактів і стала середовищем, де фіксується поточний стан взаємодії з покупцем.

Електронна комерція посилила значення CRM, бо в онлайн-продажах покупець оцінює компанію через швидкість відповіді, точність опису, наявність товару, прозорість статусу замовлення та якість післяпродажної комунікації. У цьому середовищі товар стає такою самою базовою сутністю, як клієнт або

замовлення. Класична CRM може концентруватися на контакті чи угоді, але CRM/e-commerce система повинна додатково підтримувати каталог, товарні фото, описи, ціни, канали продажу й дані для повторного використання в замовленнях [7].

З теоретичного погляду така система поєднує три рівні. Операційний рівень відповідає за щоденні дії менеджера: створення клієнта, додавання товару, оформлення замовлення, зміну статусу, фіксацію задачі. Інформаційний рівень забезпечує цілісність даних: зв’язки між сутностями, ізоляцію записів користувача, перевірку обов’язкових полів. Аналітичний рівень використовує накопичені дані для звітів, прогнозування та оцінки ефективності, але в межах поточного прототипу представлений тільки базовими показниками на інформаційній панелі. Такий поділ дозволяє чесно відокремити реалізоване операційне ядро від можливих майбутніх розширень.

Отже, розвиток CRM-систем можна описати як рух від простого обліку клієнтів до інтегрованого середовища керування даними, процесами й товарним контентом. Для дропшипінг-платформи особливо важливим є саме операційний варіант CRM/e-commerce системи, оскільки він зменшує розрив між клієнтською базою, каталогом, замовленнями й задачами менеджера.

1.2 Специфіка дропшипінгу та роль CRM в електронній комерції

Дропшипінг є моделлю електронної комерції, у якій продавець приймає замовлення від покупця, але не обов’язково зберігає товар на власному складі. Виконання замовлення може передаватися постачальнику або виробнику, який відправляє товар кінцевому споживачу [8]. Теоретична перевага цієї моделі полягає в нижчому порозі входу: продавець може тестувати нішу без значних складських витрат. Водночас така гнучкість створює підвищені вимоги до інформаційної дисципліни.

Інформаційна складність дропшипінгу пов’язана з розподіленою відповідальністю. Продавець не контролює весь фізичний ланцюг, але саме він відповідає перед покупцем за коректність опису, актуальність ціни, точність адреси, швидкість уточнення деталей і фіксацію статусу. Дослідження fulfilment-процесів показують, що передавання виконання замовлення на сторону постачальника змінює не тільки логістику, а й вимоги до обміну даними між учасниками ланцюга [9]. Для онлайн-продажів також суттєвою є організація останньої милі доставки, оскільки саме вона впливає на очікування покупця та якість сервісу після оформлення замовлення [10].

Для стабільного дропшипінг-процесу потрібно підтримувати зв’язок між чотирма сутностями. Клієнтський запис відповідає на питання, кому продається товар; товарний каталог - що саме продається; замовлення - яка операція виконується; задача - яку дію повинен зробити менеджер. Якщо ці сутності зберігаються в різних таблицях, месенджерах або нотатках, менеджер витрачає час на відновлення контексту, а ризик помилки зростає.

Окреме значення має товарний контент. У дропшипінгу продавець часто отримує від постачальника короткий технічний опис або фото зі складним фоном. Покупець не бачить товар фізично, тому текст, фото й ключові характеристики стають основою довіри. У цьому сенсі товарна картка має подвійну природу: вона є і операційним записом у системі, і маркетинговим матеріалом для майбутньої публікації.

Таблиця 1.1 подає основні інформаційні потреби дропшипінг-процесу.

Таблиця 1.1 – Інформаційні потреби дропшипінг-процесу

Потреба процесу	Дані, які необхідно зберігати	Реалізований або модельований рівень у системі
1	**2**	**3**
Підготовка товару до продажу	Назва, ціна, артикул, постачальник, опис, зображення	Реалізовано в товарному каталозі та ШІ-модулі

Продовження таблиці 1.1

1	2	3
Робота з клієнтом	Ім’я, телефон, місто, джерело, нотатка	Реалізовано в клієнтських записах
Оформлення замовлення	Клієнт, товар, кількість, сума, статус, адреса	Реалізовано базову модель замовлення
Контроль дій менеджера	Тип задачі, строк, статус, зв’язок із сутностями	Реалізовано базову модель задач
Ізоляція робочого простору	Ідентифікатор користувача, сесія	Реалізовано через `user_id` і сесію
Логістична й платіжна інтеграція	Перевізник, транзакція, повернення	Напрям подальшого розвитку

Таким чином, CRM у дропшипінгу виконує не тільки функцію обліку клієнтів. Вона централізує дані про продаж, підтримує контроль поточних дій і створює основу для майбутньої інтеграції з постачальниками, оплатою чи доставкою без приписування цих функцій поточній реалізації.

1.3 Можливості штучного інтелекту для автоматизації підготовки товарного контенту

В електронній комерції якість товарного контенту безпосередньо впливає на рівень довіри покупця. Опис, фото, перелік переваг і пошукові ключі виконують роль замінника фізичного огляду товару. Через це автоматизація підготовки контенту має практичну цінність, якщо вона скорочує рутинні дії, але не знімає з менеджера відповідальність за точність інформації.

Генеративний штучний інтелект у маркетингових і комерційних процесах доцільно розглядати як інструмент підготовки першої версії тексту, а не як автономного автора, що самостійно приймає рішення про публікацію [11]. Великі мовні моделі можуть перетворювати короткі характеристики на

структурований опис, однак вони здатні додавати непідтверджені властивості, змінювати тон або повторювати загальні маркетингові фрази [12]. Тому результат потребує перевірки людиною.

У межах розробленої системи ШІ-модуль має вузьке й контрольоване призначення. Текстова частина формує поля title, short_summary, key_features і seo_keywords, тобто не повертає один довільний абзац, а створює структуру, придатну для збереження в базі даних і редагування в інтерфейсі. Такий підхід узгоджується з практикою інженерії запитів до мовних моделей, де формат відповіді обмежується завданням і подальшою обробкою [13].

Візуальна частина пов’язана з комп’ютерним зором і сегментацією зображень. Автоматичне видалення фону спирається на ідею відокремлення об’єкта від навколишнього середовища, що має довгу історію в методах інтерактивної сегментації [14]. Подальший розвиток нейромережевих методів сегментації показав, що контур об’єкта можна уточнювати автоматизовано навіть за складнішої структури зображення [15]. Для товарної картки це означає не створення нового зображення, а приведення наявного фото до чистішого й придатнішого для каталогу вигляду.

Поєднання генерації тексту та обробки зображення в одному CRM-сценарії є важливим саме для дропшипінгу. Менеджер не переходить між окремими редакторами й сервісами, а працює у формі товарної картки. ШІ-модуль готує чернетку контенту, після чого користувач уточнює текст, перевіряє фото й вирішує, чи можна використовувати результат у продажах.

1.4 Огляд існуючих CRM/e-commerce рішень для малого бізнесу

Ринок CRM/e-commerce рішень є сегментованим. Платформи електронної комерції на зразок Shopify або WooCommerce насамперед орієнтовані на вітрину, кошик, оплату, каталог і публікацію товарів. CRM-системи на зразок HubSpot CRM чи Zoho CRM концентруються на контактах, угодах,

комунікаціях і задачах менеджерів. Модульні ERP/CRM-рішення, зокрема Odoo, поєднують багато бізнес-підсистем, але потребують складнішого налаштування й супроводу.

Для малого дропшипінгу готові системи можуть бути корисними, але не завжди відповідають задачі навчально-прикладної розробки. Вони приховують частину архітектури, залежать від плагінів і не завжди дозволяють прозоро показати, як побудовано модель даних, маршрути, ізоляцію користувача та інтеграцію ШІ-модуля. Для бакалаврської роботи важливо продемонструвати не тільки користувацький результат, а й внутрішню логіку побудови інформаційної системи.

Таблиця 1.2 подає узагальнене порівняння типових рішень.

Таблиця 1.2 – Порівняння CRM/e-commerce рішень для малого бізнесу та дропшипінгу

Рішення	Клієнти	Замовлення	Товари	ШІ-контент	Придатність для малого дропшипінгу
Shopify	Є	Сильна підтримка	Розвинений каталог	Інструменти платформи	Висока для магазину, нижча для власного CRM-проєкту
HubSpot CRM	Сильна підтримка	Через угоди	Не є центральними	В екосистемі	Потребує інтеграції e-commerce рівня
Zoho CRM	Сильна CRM-модель	Модулі	Налаштовується	ШІ Zia	Добра для гнучкого CRM, потребує налаштування
Odoo	Є модулі	Сильна підтримка	Розвинена модель	Модульно	Потужна, але надмірна для прототипу
WooCommerce	У межах WordPress	Сильна підтримка	Сильна підтримка	Плагіни	Добра для магазину, слабша як CRM

Власна реалізація прототипу не конкурує з промисловими платформами за повнотою функцій. Її цінність полягає в прозорій побудові операційного ядра: користувач, клієнти, товари, замовлення, задачі, сесійна автентифікація, ізоляція даних і контрольований модуль інтелектуальної обробки контенту.

1.5 Обґрунтування вибраного стеку технологій та постановка задачі

Технологічний стек CRM/e-commerce системи має забезпечувати швидку розробку, зрозумілу архітектуру, збереження даних і можливість підключення ШІ-компонентів. Для серверної частини обрано FastAPI, оскільки Python-екосистема зручна для поєднання веблогіки, обробки зображень і викликів генеративних сервісів. Для інтерфейсу використано Jinja-шаблони й HTMX-поведінку, що дає змогу оновлювати частини сторінки без створення окремого важкого клієнтського застосунку. Як СКБД використано SQLite, достатню для навчально-прикладного прототипу й відтворюваного запуску через Docker Compose.

Об’єктом дослідження є процес інформаційної підтримки управління замовленнями, клієнтськими даними та товарним контентом у дропшипінг-платформі. Предметом дослідження є методи проєктування та розробки веборієнтованої CRM/e-commerce системи з модулем інтелектуальної обробки товарного контенту.

Метою роботи є проєктування та розробка інформаційної системи управління замовленнями дропшипінг-платформи, що забезпечує централізовану роботу з товарами, клієнтами, замовленнями й задачами, а також інтегрує модуль інтелектуальної обробки контенту для скорочення ручної рутини.

Для досягнення мети необхідно виконати такі завдання: проаналізувати теоретичні засади CRM/e-commerce систем і специфіку дропшипінгу; визначити інформаційні потреби процесу; спроєктувати модель даних і

користувацькі сценарії; реалізувати серверну логіку на FastAPI та SQLite; інтегрувати модулі видалення фону й генерації структурованого опису; перевірити роботу автентифікації, ізоляції даних, CRUD-операцій і ШІ-сценарію.

Межі реалізації зосереджені на операційному ядрі: каталогах, замовленнях, клієнтах, задачах і підготовці товарного контенту. Платіжні шлюзи, складська логістика, інтеграції зі службами доставки, розширені ролі та поглиблена аналітика залишаються напрямами подальшого розвитку.

2 ДОСЛІДЖЕННЯ МОДУЛЯ ІНТЕЛЕКТУАЛЬНОЇ ОБРОБКИ ТОВАРНОГО КОНТЕНТУ ДЛЯ CRM-СИСТЕМИ ДРОПШИПІНГ-ПЛАТФОРМИ

2.1 Мета використання ШІ в товарній картці

Товарна картка в CRM/e-commerce системі дропшипінг-платформи є не лише записом у каталозі. Вона поєднує комерційні атрибути, зображення, короткий опис, пошукові ознаки й матеріал, який менеджер може використати для публікації на торговельному майданчику. Для малого дропшипінг-бізнесу це має практичний вимір: менеджер регулярно працює з товарами від різних постачальників, а вихідні матеріали часто різняться за якістю, повнотою і стилем [16].

У межах цієї роботи модуль інтелектуальної обробки товарного контенту розглядається як допоміжний компонент CRM-системи. Він не замінює менеджера і не виконує автоматичну публікацію товару без перевірки. Його призначення – скоротити первинну підготовку картки: очистити зображення від фону, запропонувати комерційний заголовок, сформувати короткий опис, виділити ключові переваги та підібрати пошукові ключі. Остаточне рішення залишається за користувачем, який перевіряє зміст, уточнює характеристики й редагує результат перед використанням у продажах.

Фактична реалізація складається з двох сервісних частин. У файлі app/services/background_removal.py функція remove_background() приймає байти зображення, відкриває файл через Pillow, передає його до rembg, зберігає результат у форматі PNG і повертає його як рядок base64. У файлі app/services/description_generator.py функція generate_description() передає назву товару й додаткові деталі до Gemini та очікує структуровану JSON-відповідь із полями title, short_summary, key_features і seo_keywords.

Таке розділення відповідає природі товарного контенту. Зображення має бути придатним для каталогу: товар повинен залишатися видимим, а фон не

має відволікати від об’єкта. Текстова частина має бути не довільним рекламним абзацом, а набором полів, які можна зберегти, показати в інтерфейсі й оцінити окремо. Через це в модулі використовується структурована відповідь, а не один великий фрагмент тексту.

Проблема, яку вирішує модуль, полягає у зменшенні повторюваних ручних дій. У звичайному сценарії менеджер завантажує фото, готує опис, виділяє переваги, підбирає ключові слова, перевіряє грамотність і приводить зображення до чистішого вигляду. Якщо в каталозі з’являються десятки позицій, навіть часткова автоматизація помітно зменшує рутинне навантаження. Водночас дропшипінгова товарна база часто містить неповні дані: коротку назву, фото на побутовому фоні або технічні характеристики без комерційного опису. Тому модуль має працювати з мінімальним вводом, але не повинен вигадувати факти, яких користувач не надав.

Практичні завдання використання ШІ в товарній картці такі:

– підготувати початковий варіант комерційного заголовка;

– сформувати короткий опис для перегляду менеджером або покупцем;

– виділити 3-5 ключових переваг без надмірної реклами;

– підібрати 3-5 релевантних пошукових ключів;

– очистити зображення товару від фону й повернути його в уніфікованому форматі;

– зберегти результат у структурі, сумісній із карткою CRM-системи.

У цьому розділі ШІ-модуль оцінюється через придатність до створення товарної картки. Потрібно перевірити структурованість відповіді, коректність використання деталей, стійкість до неповного вводу, час підготовки й зрозумілість для користувача.

2.2 Напрями дослідження

Дослідження має встановити, чи достатній обраний підхід для CRM-системи дропшипінг-платформи. Оскільки модуль працює і з текстом, і з візуальним матеріалом, оцінювання не можна звести до одного показника. Воно охоплює якість опису, повноту переваг, релевантність ключових слів, якість видалення фону, час обробки, стійкість до неповних даних і потребу в ручному редагуванні [17].

Перший напрям – якість текстового опису. Текст має бути зрозумілим і комерційно придатним, без неправдивих тверджень. Оцінюються точність, повнота, читабельність і відповідність вхідним деталям.

Другий напрям – повнота key_features. Список із 3-5 пунктів має допомагати швидко зрозуміти цінність товару.

Третій напрям – релевантність seo_keywords. Ключові слова трактуються як пошукові теги для каталогу й основа для подальшої оптимізації.

Четвертий напрям – час підготовки. Оцінюється час обробки зображення (rembg), генерації опису (Gemini) та повний час створення картки.

П’ятий напрям – стійкість до неповних даних. Якісний результат має бути без непідтверджених характеристик за короткої назви.

Шостий напрям – якість видалення фону. Оцінюється збереження об’єкта, прозорість фону та відсутність артефактів.

Сьомий напрям – редагованість. Автоматизація ефективна, якщо результат потребує мінімального уточнення, а не повного переписування.

Таблиця 2.1 узагальнює напрями оцінювання.

Таблиця 2.1 – Напрями оцінювання модуля інтелектуальної обробки товарного контенту

Напрям оцінювання	Що перевіряється	Очікуваний результат для CRM-системи
Якість опису	Точність, грамотність, відповідність деталям	Опис можна використати як основу картки
Ключові переваги	Повнота й корисність `key_features`	Менеджер швидко бачить сильні сторони товару
Пошукові ключі	Релевантність `seo_keywords`	Товар легше знаходити й описувати
Час обробки	Час зображення, тексту й повного запиту	Створення картки не перериває робочий процес
Неповні дані	Поведінка за короткої або загальної назви	Модель не додає непідтверджених характеристик
Видалення фону	Збереження об’єкта й прозорість фону	Фото придатне для каталогу
Редагованість	Обсяг ручних правок	Автоматизація економить час

Усі числові дані в цьому розділі подаються обережно, як дослідницький каркас і методика вимірювання.

2.3 Огляд моделей/сервісів для генерації опису та обробки зображень

Модуль інтелектуальної обробки товарного контенту можна реалізувати різними способами. Для текстової частини потрібна модель, здатна писати короткий комерційний опис і повертати структуровану відповідь. Для візуальної частини потрібен інструмент сегментації або видалення фону, придатний для типових товарних фотографій. У поточній реалізації використано Gemini для генерації опису та rembg для видалення фону. Інші варіанти розглядаються як альтернативи, але не інтегровані в код [18].

Gemini підключено через бібліотеку google.generativeai. У app/services/description_generator.py ключ доступу береться з settings.google_api_key, а назва моделі – з settings.gemini_model. Код не прив’язаний до жорстко зашитого ідентифікатора: модель можна змінити в конфігурації. Для дипломного проєкту це зручно, оскільки зовнішній сервіс може оновлювати модельний ряд, а власна реалізація зберігає спосіб виклику, структуру відповіді й обробку помилок.

Ключова перевага Gemini в цій задачі – підтримка структурованої відповіді. У generation_config задано response_mime_type="application/json« і схему з обов’язковими полями. Після отримання відповіді код виконує json.loads(response.text) і створює об’єкт ProductDescription з app/schemas.py. Отже, результат одразу пов’язаний із типізованою структурою застосунку.

Альтернативами є інші хмарні API (залежать від мережі та квот), локальні моделі (потребують ресурсів), e-commerce сервіси (жорсткі формати) або шаблонна генерація.

Таблиця 2.2 подає порівняння підходів до генерації опису.

Таблиця 2.2 – Порівняння підходів до генерації товарного опису

Підхід	Переваги	Обмеження	Придатність для поточного проєкту
1	**2**	**3**	**4**
Gemini через API	Структурована відповідь, мовна гнучкість, швидка інтеграція	Залежність від ключа, мережі, квот і сервісу	Реалізований варіант
Інші хмарні ВММ	Подібні можливості генерації й інструкцій	Потрібна окрема інтеграція	Альтернатива для майбутнього порівняння
Локальна ВММ	Більший контроль даних	Вимоги до ресурсів і підтримки	Напрям розвитку

Продовження таблиці 2.2

1	2	3	4
Спеціалізований e-commerce сервіс	Орієнтація на комерційні тексти	Менша керованість формату	Можлива майбутня інтеграція
Шаблонна генерація	Передбачуваність, відсутність API-витрат	Низька гнучкість	Недостатньо для різнорідного каталогу

Для обробки зображень використано rembg. У app/services/background_removal.py зображення відкривається через Image.open(), передається у функцію remove(), після чого результат зберігається як PNG. Формат важливий, бо підтримує прозорість, потрібну після видалення фону.

Альтернативи (зовнішні API сегментації або ручне редагування) ускладнюють інфраструктуру або вимагають зайвого часу.

Таблиця 2.3 порівнює підходи до обробки зображення.

Таблиця 2.3 – Порівняння підходів до обробки товарного зображення

Підхід	Переваги	Обмеження	Придатність для поточної системи
`rembg` у серверному коді	Локальна обробка, просте підключення, `PNG`	Якість залежить від фото	Реалізований варіант
Зовнішній API	Потенційно стабільна якість	Вартість, мережа, приватність	Напрям розвитку
Окрема модель сегментації	Гнучкість і контроль	Складність підтримки	Надмірно для прототипу
Ручне редагування	Максимальний контроль	Значні часові витрати	Не відповідає меті автоматизації

Обраний підхід є компромісом між якістю, складністю і прив’язкою до реального CRM-сценарію. Gemini забезпечує структурований текст, а rembg виконує базову візуальну підготовку без винесення зображень до ще одного зовнішнього сервісу. Разом вони не перетворюють систему на платформу дизайну, але закривають ключову задачу дипломного проєкту: допомогти менеджеру швидше отримати заповнену товарну картку [19].

2.4 Інструкція та структура JSON-відповіді

Для генеративної моделі важливий не лише вибір сервісу, а й інструкція. У поточному проєкті вона зберігається в змінній SYSTEM_PROMPT у app/services/description_generator.py. Інструкція задає роль моделі як експерта з e-commerce копірайтингу, просить створити переконливий опис і водночас обмежує результат фіксованим форматом.

Ключова вимога – повернути валідний JSON-об’єкт із такими полями:

```
json
{
"title": «Комерційний заголовок товару»,
"short_summary": «Короткий опис у 2-3 реченнях»,
"key_features": [
«Ключова перевага 1»,
«Ключова перевага 2»,
«Ключова перевага 3»
],
"seo_keywords": [
«ключове слово 1»,
«ключове слово 2»,
«ключове слово 3»
]
```

Ця структура дублюється на рівні response_schema у GenerationConfig. Схема визначає тип об’єкта, властивості, типи масивів і обов’язкові поля. Після отримання відповіді вона перетворюється через json.loads() і перевіряється Pydantic-моделлю ProductDescription. Тому застосунок приймає не просто текст, а структуру, сумісну з товарною карткою.

Поле title призначене для комерційної назви. short_summary містить опис у 2-3 реченнях. key_features (3-5 переваг) зручне для інтерфейсу. seo_keywords слугують пошуковими тегами для каталогу.

Модель має генерувати текст мовою вхідних даних та суворо спиратися на надані користувачем деталі (наприклад, матеріал чи об’єм).

Таблиця 2.4 показує, чому структурований формат кращий за довільний абзац.

Таблиця 2.4 – Порівняння довільного тексту та структурованої відповіді

Ознака	Довільний абзац	Структурований `JSON`
Збереження в БД	Потрібне ручне розділення	Поля відповідають картці
Відображення	Важко окремо показати переваги й ключі	Кожне поле має власний блок
Перевірка формату	Майже немає автоматичної перевірки	Можливі `json.loads()` і Pydantic
Редагування	Менеджер сам структурує текст	Менеджер редагує готові поля
Повторне використання	Обмежене	Придатне для API, картки й каталогу

Технічна валідація формату JSON має завжди поєднуватися зі змістовною перевіркою користувачем.

2.5 Оцінка часу підготовки контенту

Час підготовки контенту є практичним показником для CRM-системи. Якщо автоматизація змушує менеджера довго чекати або потребує тривалого виправлення, її користь зменшується. Тому оцінюється не лише швидкість генерації, а й час до прийнятного результату.

У модулі можна виділити три часові складники. Перший – обробка зображення функцією remove_background(), що залежить від розміру файлу, роздільної здатності, складності фону й ресурсів сервера. Другий – генерація опису функцією generate_description(), на яку впливають Gemini, мережа, модель, довжина деталей і навантаження сервісу. Третій – повний час створення картки: приймання форми, перевірка файлу, виклики сервісів, збереження даних і відповідь інтерфейсу.

Для оцінювання часу підготовки контенту використано контрольний сценарій із фіксацією дій менеджера та результатів автоматизованої обробки. Таблиці в цьому підрозділі подають структуру протоколу, за якою результати можна відтворити під час повторного запуску системи в однаковому середовищі.

Методика вимірювання часу:

Крок 1 Підготувати контрольний набір із п’яти товарів і зображень.

Крок 2 Для кожного товару зафіксувати розмір фото, складність фону й наявність деталей.

Крок 3 Запустити створення товару через вебінтерфейс або POST /api/v1/process-product.

Крок 4 Окремо виміряти час видалення фону, генерації опису й повної відповіді.

Крок 5 Повторити прогін не менше трьох разів для кожного товару.

Крок 6 Обчислити середній час, мінімум і максимум.

Крок 7 Окремо зафіксувати помилки API, файлу або невиявлення об’єкта.

Для точнішого вимірювання в коді можна тимчасово додати службове логування навколо remove_background() і generate_description(). Воно не має змінювати бізнес-логіку, а лише фіксувати часові позначки. Якщо сценарій виконується через API, загальний час HTTP-запиту можна додатково виміряти на стороні клієнта.

Таблиця 2.5 подає формат фіксації часу підготовки контенту.

Таблиця 2.5 – Протокол вимірювання часу підготовки контенту

Контрольний товар	Умови вхідних даних	Видалення фону, с	Генерація опису, с	Загальний час, с	Примітка
Спортивна пляшка 750 мл	Чітке фото, короткі деталі	1,8	3,6	5,9	Методичний приклад
Органайзер для кабелів	Фото на столі, неповні деталі	2,4	3,9	6,8	Методичний приклад
LED-лампа настільна	Технічні характеристики	2,1	4,4	7,1	Методичний приклад
Рюкзак міський	Складний фон	3,2	3,8	7,7	Методичний приклад
Чохол для смартфона	Загальна назва, мало деталей	1,6	3,5	5,5	Методичний приклад

Порівняння з ручною підготовкою також потребує обережності. Реальний час залежить від досвіду менеджера, якості матеріалів постачальника, вимог до тексту й потреби редагувати фото. Тому порівнюються кілька режимів: ручну підготовку тексту, ручну підготовку тексту й фото, автоматизовану підготовку та автоматизовану підготовку з правками.

Таблиця 2.6 показує структуру такого порівняння.

Таблиця 2.6 – Методика порівняння автоматизованої та ручної підготовки товарної картки

Сценарій	Що входить у підготовку	Очікуваний показник
Ручна підготовка тексту	Заголовок, опис, переваги, ключі	Час до готового тексту
Ручна підготовка тексту й фото	Текст плюс очищення зображення	Повний час підготовки картки
Автоматизована підготовка	`remove_background()` плюс `generate_description()`	Час відповіді системи
Автоматизована підготовка з правками	Автоматичний результат і ручне уточнення	Час до прийнятного контенту

Найбільш корисним є показник «час до прийнятного результату». Швидка генерація не дає переваги, якщо менеджер потім виправляє багато неточностей; натомість короткий перегляд і кілька правок уже свідчать про користь автоматизації.

Під час аналізу часу потрібно окремо враховувати помилки Gemini, ліміти, недоступність мережі та якість фото. Такі випадки не слід включати в середнє значення без пояснення. Для поточного проєкту достатньо описати методику й зазначити, що фактичні числа мають бути отримані під час контрольного запуску в одному середовищі.

2.6 Оцінка якості

Мета дослідження якості – перевірити, чи придатний автоматично підготовлений контент для товарної картки CRM-системи. Якість тут не означає лише «гарний текст». Потрібно встановити, чи відповідає опис вхідним даним, чи не містить необґрунтованих характеристик, чи має зручну структуру, чи допомагає менеджеру швидше створити картку і чи можна використати очищене зображення в каталозі.

Оцінювання охоплює два результати: текст із generate_description() і зображення з remove_background(). Для тексту перевіряються title, short_summary, key_features і seo_keywords; для зображення – збереження об’єкта, прозорість фону, якість країв і придатність до показу в картці. Якщо одна частина якісна, а інша непридатна, картка все одно потребує ручного втручання.

Щоб оцінювання не залишалося лише враженням, використовується проста шкала від 0 до 2:

– 0 – результат непридатний або містить критичну помилку;

– 1 – результат частково придатний, але потребує помітного редагування;

– 2 – результат придатний після мінімальної перевірки.

Така шкала не створює ілюзії надмірної точності, але дозволяє порівнювати товари, сценарії й типи помилок. Якщо в майбутньому буде залучено кількох експертів, можна обчислювати середнє значення за кожним критерієм [20].

Таблиця 2.7 містить критерії оцінювання тексту.

Таблиця 2.7 – Критерії оцінювання згенерованого товарного опису

Критерій	Опис оцінювання	Приклад небажаної ситуації
1	**2**	**3**
Точність	Відповідність назві й деталям	Модель додає матеріал або функцію, яких не було
Повнота	Відображення ключових характеристик	Пропущено об’єм, розмір або призначення
Комерційна придатність	Чи можна використати текст у картці	Текст сухий або перебільшений
Читабельність українською	Грамотність і природність	Кальки або змішування мов

Продовження таблиці 2.7

1	2	3
Релевантність ключів	Корисність `seo_keywords`	Ключі загальні й не стосуються товару
Потреба в редагуванні	Обсяг правок	Опис треба переписати повністю

Таблиця 2.8 містить критерії оцінювання зображення.

Таблиця 2.8 – Критерії оцінювання очищеного зображення

Критерій	Опис оцінювання	Приклад небажаної ситуації
Збереження об’єкта	Основний товар залишився видимим	Частина товару обрізана
Якість фону	Фон прозорий або майже прозорий	Залишилися великі фрагменти фону
Краї об’єкта	Контур без грубих артефактів	Рвані краї або втрата деталей
Стійкість до складного фону	Результат прийнятний на неідеальному фото	Система плутає товар із фоном
Придатність для картки	Фото можна показати в каталозі	Результат порожній або нечитабельний

Завдання оцінювання – визначити, чи достатня комбінація Gemini і rembg для поточного рівня CRM-системи, і зафіксувати межі її застосування [21].

Інструменти проведення експерименту.

Інструментальною основою дослідження є наявний FastAPI-застосунок і його сервісні модулі. Окремий демонстраційний застосунок для експерименту не потрібен, бо перевіряти слід саме той шлях, яким користується CRM-система.

До інструментів контрольного сценарію належать:

– remove_background() з app/services/background_removal.py;

– generate_description() з app/services/description_generator.py;

– Pydantic-схема ProductDescription з app/schemas.py;

– API-маршрут POST /api/v1/process-product, якщо перевірка виконується програмно;

– вебформа створення товару, якщо перевірка виконується через інтерфейс;

– контрольний набір товарів із назвами, деталями й зображеннями;

– таблиця фіксації часу, помилок і якісних оцінок.

Перевага такого підходу в тому, що оцінюється не ізольована модель, а повний шлях обробки: валідація, інтеграція з карткою, зображення й зручність для менеджера [22].

Для API-сценарію в запит передаються назва товару, додаткові деталі й файл зображення. У відповідь система має повернути статус success, очищене зображення в image_no_bg_base64 і структурований опис у description. Такий формат відповідає схемі ProcessProductResponse і дає змогу програмно перевірити наявність очікуваних полів [23].

Вебсценарій ближчий до реальної роботи менеджера: користувач відкриває форму, вводить дані, завантажує фото й оцінює результат у картці. Він корисний для перевірки редагованості, але менш зручний для точного поділу часу між етапами [24].

Таблиця 2.9 узагальнює інструменти й показники.

Таблиця 2.9 – Інструменти контрольного сценарію та показники

Інструмент	Що дає змогу перевірити	Обмеження
1	**2**	**3**
Прямий виклик `remove_background()`	Час і якість видалення фону	Не перевіряє повну картку
Прямий виклик `generate_description()`	Якість і структура опису	Не враховує інтерфейс
API `POST /api/v1/process-product`	Повний програмний шлях	Потрібне налаштування ключа API

Продовження таблиці 2.9

1	2	3
Вебформа товару	Сценарій менеджера	Складніше відокремити час етапів
Таблиця експертної оцінки	Порівняння якості	Потребує однакових правил оцінювання

Перед реальним експериментом потрібно зафіксувати середовище: спосіб запуску застосунку, версії залежностей, модель Gemini, розмір зображень і кількість повторів. Без цього часові дані важко порівнювати між запусками.

Проєктування сценарію тестування ШІ-модуля.

Сценарій тестування має повторювати процес створення товарної картки. Спочатку готуються назва, додаткові деталі й зображення. Далі дані передаються до застосунку, проходять перевірки, обробляються двома сервісами, а результат оцінюється за визначеними критеріями.

Послідовність сценарію:

Крок 1 Обрати контрольний товар.

Крок 2 Передати до системи назву товару.

Крок 3 За наявності додати матеріал, розмір, призначення, комплектацію або обмеження.

Крок 4 Завантажити зображення у підтримуваному форматі.

Крок 5 Перевірити тип і розмір файлу.

Крок 6 Виконати remove_background().

Крок 7 Виконати generate_description().

Крок 8 Перевірити структуру через ProductDescription.

Крок 9 Зафіксувати час, помилки й отримані поля.

Крок 10 Оцінити результат за шкалою.

Для візуального пояснення сценарію в дипломній роботі використовується схема thesis/report_assets/diagrams/04_ai_product_assistant_pipeline.*. У тексті її

потрібно пов’язувати з реальними файлами background_removal.py і description_generator.py, а не подавати як абстрактну архітектуру.

Помилки мають фіксуватися як окремі результати. Якщо зображення не містить виразного об’єкта, remove_background() може повернути помилку через перевірку частки непрозорих пікселів. Якщо Gemini недоступний або повертає некоректну відповідь, генерація опису не є успішною. Такі випадки показують межі застосування модуля і не повинні приховуватися.

Для тексту зберігаються не тільки бали, а й короткий коментар: наприклад, «додано непідтверджену водонепроникність» або «ключові слова релевантні, але загальні». Для зображення коментарі також потрібні: «залишився фрагмент столу», «обрізано лямку», «товар виділено коректно». Саме коментар пояснює, що стоїть за числовою оцінкою.

Таблиця 2.10 подає шаблон протоколу для одного товару.

Таблиця 2.10 – Шаблон протоколу тестування одного товару

Поле протоколу	Зміст
Ідентифікатор товару	Номер або назва контрольного прикладу
Назва товару	Початковий текст, переданий у систему
Додаткові деталі	Характеристики, які мають пріоритет
Опис зображення	Якість фото, фон, розмір
Результат `title`	Згенерований заголовок
Результат `short_summary`	Згенерований короткий опис
Результат `key_features`	Список переваг
Результат `seo_keywords`	Список ключових слів
Оцінка тексту	Бали за критеріями
Оцінка зображення	Бали за критеріями
Час обробки	Зафіксовані часові показники
Коментар	Помилки, обмеження, потреба в редагуванні

Такий сценарій зберігає дослідницьку дисципліну й не вимагає вигадувати окремий експериментальний інструмент: потрібні функції вже реалізовані в CRM-проєкті.

Опис і підготовка контрольного набору товарів.

Контрольний набір має охоплювати різні типи вхідних даних. Якщо перевіряти модуль лише на простих прикладах із чіткими назвами й якісними фото, результати можуть виглядати кращими, ніж у реальному дропшипінговому процесі. Тому набір повинен містити як сприятливі, так і проблемні випадки.

Для цього дослідження пропонується п’ять типів товарів:

– простий товар із чіткою назвою;

– товар із неповними деталями;

– товар із технічними характеристиками;

– товар із фото на складному фоні;

– товар із загальною назвою, що потребує уточнення.

Перший тип перевіряє базову якість. Прикладом є «Спортивна пляшка для води 750 мл» з фото на нейтральному фоні. Якщо модуль не справляється з таким випадком, його придатність для каталогу сумнівна.

Другий тип перевіряє поведінку за неповних деталей. «Органайзер для кабелів» без матеріалу, розміру й кількості відділень має отримати обережний опис про порядок на робочому місці, але без вигаданих секцій або матеріалу.

Третій тип містить конкретні характеристики. Для «Настільної LED-лампи» з деталями «три режими яскравості, USB-живлення, гнучка ніжка» оцінюється перенесення цих даних у short_summary і key_features. Помилкою буде пропуск суттєвої характеристики або додавання бездротової зарядки, якщо її не було в деталях.

Четвертий тип перевіряє візуальну частину. «Міський рюкзак» на складному побутовому фоні дає змогу оцінити, чи не зникають лямки, чи не залишаються великі фрагменти фону і чи придатне фото для картки.

П’ятий тип перевіряє неоднозначну назву. «Чохол» без моделі смартфона, матеріалу або призначення не повинен перетворюватися на опис конкретної сумісності. Модель може сформувати загальний текст захисного аксесуара, але має залишити місце для уточнення.

Таблиця 2.11 описує контрольний набір.

Таблиця 2.11 – Контрольний набір товарів для оцінювання ШІ-модуля

N	Тип прикладу	Назва товару	Додаткові деталі	Основний ризик
1	Чітка назва	Спортивна пляшка для води 750 мл	Об’єм 750 мл, кришка з поїлкою	Перевірка базової якості
2	Неповні деталі	Органайзер для кабелів	Деталі відсутні або мінімальні	Вигадані характеристики
3	Технічні характеристики	Настільна LED-лампа	3 режими яскравості, USB, гнучка ніжка	Пропуск важливих деталей
4	Складний фон	Міський рюкзак	Об’єм 20 л, кишеня для ноутбука	Артефакти видалення фону
5	Загальна назва	Чохол	Без уточнення моделі	Надмірні припущення

Підготовка набору включає нормалізацію файлів зображень: підтримуваний формат, контроль розміру, стабільні назви файлів у протоколі. Якщо фото навмисно складне, це потрібно вказати в описі прикладу. Для кожного товару бажано виконати кілька прогонів, адже текстова модель може повертати різні формулювання, а час обробки зображень також коливається.

П’ять прикладів достатні для методичної перевірки в межах бакалаврської роботи. За потреби набір можна розширити до 10-15 товарів, але за тією самою логікою категорій, щоб порівняння залишалося зрозумілим.

Результати оцінювання якості згенерованих описів і оброблених зображень.

У цьому підрозділі подано формат подання результатів оцінювання, орієнтований на повторювану перевірку відповідей Gemini та зображень після rembg. Оцінювання використовується як контрольна процедура для перевірки якості модуля в межах CRM-сценарію.

Для кожного товару оцінювання виконується за шкалою 0-2. Окремо підсумовується текст і зображення, щоб не змішувати мовну якість із візуальною. Наприклад, опис для рюкзака може бути добрим, але видалення фону може пошкодити лямки; тоді текстова оцінка буде високою, а візуальна – нижчою.

Таблиця 2.12 показує формат оцінювання текстового результату.

Таблиця 2.12 – Формат оцінювання згенерованих описів

Товар	Точність	Повнота	Комерційна придатність	Читабельність	Ключові слова	Потреба в редагуванні	Коментар
Спортивна пляшка	2	2	2	2	2	2	Методичний приклад; очікується мінімальне редагування
Органайзер для кабелів	1	1	2	2	1	1	Методичний приклад; можливі загальні формулювання
LED-лампа	2	2	2	2	2	2	Методичний приклад; треба не пропустити режими
Міський рюкзак	2	1	2	2	2	1	Методичний приклад; можливий пропуск деталей
Чохол	1	1	1	2	1	1	Методичний приклад; потрібне уточнення моделі

Таблиця 2.13 подає формат оцінювання очищених зображень.

Таблиця 2.13 – Формат оцінювання результату видалення фону

Товар	Збереження об’єкта	Якість фону	Краї об’єкта	Складний фон	Придатність для картки	Коментар
Спортивна пляшка	2	2	2	2	2	Методичний приклад для простого фото
Органайзер для кабелів	2	1	1	1	1	Методичний приклад; дрібні деталі ускладнюють сегментацію
LED-лампа	2	2	1	2	2	Методичний приклад; тонкі елементи потребують перевірки
Міський рюкзак	1	1	1	0	1	Методичний приклад для складного фону
Чохол	2	2	2	2	2	Методичний приклад за простого контуру

Для аналізу якості корисно виділяти типові помилки:

– загальні переваги, які не відрізняють товар від аналогів;

– додавання характеристик, яких немає у вхідних даних;

– повторення тих самих слів у title, short_summary і seo_keywords;

– змішування української та англійської мови;

– втрата дрібних деталей під час видалення фону;

– залишки фону біля складних контурів;

– майже порожній результат для фото, де об’єкт погано відділяється від фону.

Оцінювання виконується за шкалою 0-2 окремо для тексту і зображення. Частину помилок система виявляє автоматично (невалідний JSON, майже

порожнє зображення), проте змістові проблеми потребують людської перевірки. Після експерименту таблиці заповнюються реальними даними.

2.7 Обмеження на складних вхідних даних

Модуль має обмеження, пов’язані із зображеннями, текстовою генерацією та зовнішнім API. Їх потрібно описувати прямо, щоб не створювати враження, ніби ШІ завжди готує готовий до публікації контент.

Серед основних обмежень ШІ-модуля:

– низька якість фото або складний фон (можуть призвести до артефактів чи втрати частини об’єкта);

– неоднозначна назва та відсутність характеристик (призводять до загального, неточного опису);

– залежність від зовнішнього API Gemini;

– маркетингові перебільшення та нестабільність стилю (ризик недостовірного опису);

– неповна автоматична перевірка (валідний JSON не гарантує правильного змісту).

Таблиця 2.14 узагальнює обмеження та способи зменшення ризику.

Таблиця 2.14 – Обмеження ШІ-модуля на складних вхідних даних

Обмеження	Наслідок	Спосіб зменшення ризику
1	**2**	**3**
Низька якість фото	Погане видалення фону	Вимоги до фото, повторне завантаження
Складний фон	Артефакти або втрата частин товару	Перевірка результату користувачем
Неоднозначна назва	Загальний або неточний опис	Поле додаткових деталей

Продовження таблиці 2.14

1	2	3
Відсутність характеристик	Брак конкретики	Підказки для заповнення форми
Недоступність Gemini	Опис не генерується	Повідомлення про помилку, ручне створення
Маркетингове перебільшення	Ризик недостовірного опису	Редагування й стриманіша інструкція
Нестабільний стиль	Різний тон карток	Додаткові правила форматування
Валідний формат, але неточний зміст	Помилки не ловляться автоматично	Людська перевірка перед публікацією

Ці обмеження не заперечують корисності модуля, але визначають режим його використання. ШІ готує першу версію контенту, а менеджер перевіряє фото, прибирає перебільшення й уточнює характеристики. Для дропшипінгової CRM-системи це реалістичний підхід: каталог потрібно наповнювати швидко, але відповідальність за зміст залишається за користувачем.

2.8 Підсумки вибору підходу

Дослідження підтверджує доцільність комбінованого підходу: локальна сегментація rembg для зображень та Gemini для генерації JSON-опису. rembg забезпечує швидке очищення фото (формат PNG), а Gemini генерує структурований текст, сумісний із БД. Разом вони формують ШІ-помічника, що оптимізує підготовку контенту, вимагаючи при цьому фінальної перевірки менеджером.

Таблиця 2.15 подає підсумкове обґрунтування вибору.

Таблиця 2.15 – Обґрунтування вибору підходу інтелектуальної обробки контенту

Компонент	Обраний підхід	Причина вибору	Умова якісного використання
Текстовий опис	Gemini через `generate_description()`	Структурована відповідь, гнучка мовна генерація	Перевірка змісту менеджером
Зображення	`rembg` через `remove_background()`	Локальна обробка й результат у `PNG`	Якісне фото та перегляд результату
Формат даних	`JSON` і `ProductDescription`	Сумісність із полями картки	Валідація й обробка помилок
Контроль якості	Ручне редагування користувачем	Зменшення ризику неточностей	Інтерфейс має дозволяти правки
Оцінювання	Контрольний набір товарів	Перевірка типових і складних випадків	Фіксація реальних результатів після запуску

Для розроблюваної CRM/e-commerce системи обрано контрольований ШІ-помічник, а не автоматичну публікацію без участі людини. Він скорочує первинну підготовку контенту, але залишає менеджеру роль редактора й відповідального користувача. Такий підхід відповідає дропшипінгу: картки потрібно створювати швидко, проте опис, фото й комерційні твердження мають залишатися перевіреними.

3 РОЗРОБКА CRM/E-COMMERCE ЗАСТОСУНКУ ДЛЯ УПРАВЛІННЯ ЗАМОВЛЕННЯМИ ДРОПШИПІНГ-ПЛАТФОРМИ

3.1 Функціональна специфікація застосунку

Розроблений застосунок являє собою комплексну навчально-прикладну CRM/e-commerce систему, яка була спеціально спроєктована та оптимізована для класичного дропшипінг-сценарію. Головна концепція дропшипінгу полягає у тому, що продавець не зберігає товари на власному складі, а передає замовлення та деталі відправки безпосередньо постачальнику. Відповідно, програмне забезпечення для такого типу бізнесу повинно зосереджуватися не на складському обліку, а на ефективному управлінні відносинами з клієнтами (CRM), обробці замовлень, маркетинговому поданні товарів та оптимізації щоденної операційної роботи менеджера. Запропонований застосунок об’єднує роботу з профілями клієнтів, товарними картками, замовленнями, операційними задачами та модулем інтелектуальної підготовки товарного контенту [25].

Призначення цієї програмної платформи полягає не у спробі повної заміни масштабної промислової ERP-системи, яка зазвичай вимагає місяців на впровадження та значних фінансових ресурсів на підтримку. Метою є концептуальна реалізація компактного, максимально зручного, зрозумілого та чуйного робочого середовища менеджера продажів. У такому централізованому середовищі всі критично важливі дані дропшипінг-процесу надійно зберігаються, обробляються та керуються в єдиному уніфікованому вебінтерфейсі. Це суттєво знижує когнітивне навантаження на працівника, дозволяючи йому фокусуватися на продажах, а не на перемиканні між десятками різних таблиць, блокнотів та месенджерів.

Функціональна специфікація системи була ретельно та детально сформована з огляду на фактичний стан розробленого репозиторію та

актуального програмного коду. У застосунку на даний момент повноцінно та безпечно реалізовано:

– систему управління обліковими записами: реєстрацію нового профілю, безпечний вхід за допомогою хешованих паролів і вихід користувача із системи із закриттям сесії;

– інформативну персональну оглядову панель (Dashboard), яка агрегує статистичні показники роботи в режимі реального часу;

– повнофункціональні базові CRUD-операції (від англ. Create, Read, Update, Delete – створення, читання, оновлення, видалення) для записів про клієнтів бази, замовлень, товарних позицій і внутрішніх задач органайзера;

– можливість миттєвої швидкої зміни статусу виконання замовлення без повного перезавантаження сторінки;

– динамічне перемикання стану запланованої задачі (виконана або відкрита) в один клік;

– модуль обробки зображень: завантаження вихідного фотографічного зображення товару та автоматизоване видалення фону за допомогою локальних алгоритмів машинного зору (Computer Vision);

– модуль генерації тексту: створення структурованого та комерційно привабливого опису товарної картки на основі API великої мовної моделі (Large Language Model);

– публічний API-сценарій для віддаленої програмної обробки товару зовнішніми сервісами.

фундаментальною архітектурною особливістю застосунку є те, що всі збережені дані жорстко розмежовані на рівні бази даних через обов’язкове поле user_id. Завдяки цьому рішенню кожен зареєстрований користувач платформи гарантовано і безпечно працює лише зі своїми власними ізольованими записами. Це унеможливлює витік клієнтської бази між різними менеджерами та забезпечує необхідний рівень комерційної конфіденційності.

Блок управління обліковим записом охоплює ключові вебмаршрути /register, /login і /logout. Логіка автентифікації та безпеки зосереджена у двох

спеціалізованих файлах: app/auth.py та app/auth_web.py. Під час процедури реєстрації пароль користувача хешується за допомогою алгоритму bcrypt із застосуванням солі. Після успішної перевірки пароля при вході у сесійне сховище браузера записується ідентифікатор користувача. Надалі сервер використовує цей ідентифікатор у запитах до бази даних, формуючи основу персоналізації робочого простору.

Оглядова панель, яка слугує командним центром менеджера, відкривається на головному маршруті / одразу після успішної автентифікації. Вона наочно, у вигляді великих інформаційних віджетів, показує актуальну кількість збережених клієнтів, загальну кількість доданих у каталог товарів, співвідношення активних (тих, що ще обробляються) і вже успішно завершених замовлень, а також виводить кількість відкритих операційних задач із наближеним або вже простроченим строком виконання. Ці статистичні показники не є статичними – їх динамічно формує оптимізована SQL-функція get_dashboard_summary(user_id) у модулі взаємодії з базою даних app/database.py. Зазначений модуль виконує життєво важливу роль щоденного контрольного екрана, що дозволяє менеджеру з перших секунд після авторизації розуміти поточне навантаження та виявляти «вузькі місця» в роботі.

Клієнтська база платформи підтримує комплексне створення, перегляд, детальне редагування та безпечне видалення профілів покупців. Картка кожного клієнта у системі здатна зберігати: повне ім’я, актуальний номер мобільного телефону, електронну пошту, місто проживання, детальну адресу доставки для поштових служб, маркер джерела звернення (наприклад, «таргетована реклама», «соціальні мережі» чи «органічний пошук») та довільну текстову примітку. Для сценарію дропшипінгу саме такий стандартизований набір полів було визначено як оптимальний. Він дає змогу ефективно фіксувати канал надходження клієнта, що корисно для маркетингу, й швидко використовувати контактні та адресні дані безпосередньо під час оформлення нового замовлення, уникнувши повторного їх введення. На рівні програмного коду цей функціональний блок нерозривно пов’язаний із реляційною таблицею

бази даних customers та обробляється маршрутами контролера /customers, /customers/{customer_id} і спеціальною формою редагування /customers/{customer_id}/edit.

Замовлення логічно виступає центральною, об’єднуючою операційною сутністю системи. Будь-яке замовлення має власну ідентифікаційну назву, обов’язковий зв’язок із конкретним профілем клієнта, опціональний зв’язок із певним товаром з каталогу, задану менеджером кількість одиниць продукції, розраховану підсумкову суму до сплати, поточний операційний статус, визначений канал продажу, адресу фактичної доставки й службову примітку для внутрішнього корпоративного використання. Набір статусів замовлення не є довільним текстовим полем, він суворо лімітований константами на рівні коду і включає значення new (нове замовлення), confirmed (підтверджене клієнтом), processing (знаходиться в процесі обробки/пакування у постачальника), shipped (передано в службу логістики), completed (успішно виконано, гроші отримані) та cancelled (скасовано з певних причин). Такий раціональний та обмежений підхід дозволяє дуже ефективно відстежувати хід опрацювання замовлення без необхідності ускладненого моделювання повного логістичного циклу.

Товарний каталог призначений для надійного, систематизованого збереження продуктових карток. Кожна картка включає назву, оригінальний вихідний опис або технічні характеристики товару, базову ціну продажу, складський артикул (SKU), статус актуальної наявності у постачальника (наприклад, available, out_of_stock), найменування самого постачальника та головне репрезентативне зображення. Унікальною особливістю системи є те, що безпосередньо під час створення нової одиниці товару застосунок ініціює запуск складної інтелектуальної обробки контенту. Спочатку завантажений графічний файл передається до внутрішнього сервісу видалення фону. Потім введені менеджером базові текстові дані використовуються для автоматизованого формування професійного комерційного опису за допомогою моделі штучного інтелекту. У таблиці products зберігаються як початкові поля, так і готові результати алгоритмічної обробки: комерційна назва (title),

короткий опис (short_summary), переваги (key_features), пошукові теги (seo_keywords) та безпосередньо очищене зображення, перетворене в рядок image_base64.

Окрема система внутрішніх задач використовується для зручних операційних нагадувань. Запис задачі містить короткий заголовок, розширений текстовий опис, класифікатор типу задачі (наприклад, follow_up), граничний строк виконання (дедлайн), поточний статус і, що дуже зручно для навігації, прямий лінк-зв'язок із клієнтом або специфічним замовленням. У поточній реалізації закладено лише два базові стани для кожної задачі: open (активна задача, що потребує уваги) і done (успішно закрита задача). Швидке перемикання стану однією кнопкою винесено в окремий маршрут /tasks/{task_id}/toggle, що дозволяє закривати нагадування не переходячи на інші сторінки.

Інтелектуальна обробка товарної картки концептуально складається з двох потужних та технічно незалежних підсистем. Видалення візуального фону на зображеннях реалізовано локально у файлі коду app/services/background_removal.py. Для цього застосовується відкрита бібліотека комп'ютерного зору rembg (яка використовує архітектуру U-Net для сегментації об'єктів) і графічна бібліотека PIL. Оброблений алгоритмом результат повертається у вигляді PNG-файлу з альфа-каналом (прозорістю), що кодується стандартом Base64 для прямого і надійного збереження в текстовому полі бази даних SQLite. Паралельно з цим процесом, генерація якісного тексту реалізована у файлі app/services/description_generator.py і базується на зовнішніх мережевих викликах до сучасного Gemini API від Google. Цей сервіс формує суворо структуровану текстову відповідь у форматі JSON із чітко розмежованими полями. Це значно спрощує реляційне збереження результату та дозволяє гнучко проводити ручне редагування кожного окремого фрагмента в інтерфейсі менеджера перед остаточною публікацією товару [26].

Для забезпечення можливостей технічного масштабування та інтеграції зі сторонніми платформами, розробниками було окремо виділено та реалізовано

API-ендпоінт POST /api/v1/process-product. Його стратегічне призначення – забезпечення віддаленого програмного сценарію взаємодії. У цьому випадку зовнішній програмний клієнт (наприклад, інший інтернет-магазин, скрипт імпорту товарів або мобільний застосунок) передає на сервер вихідні параметри: назву, деталі й файл зображення товару. Вже за лічені секунди він отримує готову JSON-відповідь із результатами алгоритмічної обробки. Безпечний доступ до цього ендпоінту гарантовано через механізм обов'язкового API-ключа (перевірка відбувається через залежність FastAPI verify_api_key). Варто наголосити, що зазначений публічний API цілеспрямовано не створює жодного CRM-запису в базі даних автоматично. Його головна архітектурна мета – бути універсальним, незалежним, безпечним і повторно використовуваним модулем штучного інтелекту для будь-якої зовнішньої системи.

При описі розробки важливо окремо зазначити архітектурні обмеження поточної версії прототипу. У дипломному репозиторії не реалізовано API-інтеграції з реальними постачальниками, банківські платіжні шлюзи, інтеграцію з API Нової Пошти чи Укрпошти для генерації товарно-транспортних накладних, автоматичне вивантаження на маркетплейси, розширену рольову модель і фінансову аналітику. Ці напрями належать до подальшого розвитку. Поточна версія зосереджена на робочому CRM/e-commerce ядрі, яке пов'язує користувача, клієнтів, товари, замовлення, задачі та модуль інтелектуальної обробки контенту [27].

Таблиця 3.1 докладно структурує розроблені функціональні блоки системи, перелічує конкретні реалізовані дії та зазначає наявні обмеження, щоб скласти об'єктивну картину готовності програмного продукту.

Таблиця 3.1 – Деталізовані функціональні можливості розробленого застосунку

Функціональний макро-блок	Реалізовані конкретні дії у вебінтерфейсі та API	Основні файли-артефакти програмної реалізації	Технічні обмеження поточної версії прототипу
1	**2**	**3**	**4**
Система облікових записів (Автентифікація)	Реєстрація нового користувача, безпечний вхід за паролем, вихід, збереження і розшифрування сесії	Код у `app/main.py`, `app/auth.py`, `app/auth_web.py`, шаблони `login.html`, `register.html`	Повна відсутність розгалуженої системи прав, немає супер-адміністратора, лише ізольовані менеджери
Головна оглядова панель (Статистичний Dashboard)	Системний підрахунок всіх клієнтів, товарів, активних та закритих замовлень, моніторинг прострочених задач	SQL-функція `get_dashboard_summary(user_id)`, шаблон `index.html`	Не реалізовані візуальні лінійні графіки продажів та звіти про доходи/витрати
Управління клієнтською базою (CRM модуль)	Створення профілю, детальне читання даних, зміна інформаційних полів, безповоротне каскадне видалення	Реляційна таблиця `customers`, група маршрутів `/customers`, часткові HTMX-шаблони	Відсутня автоматизована комунікаційна історія дзвінків/email та маркетингова сегментація лідів
Блок обробки замовлень	Оформлення продажу, перегляд зв’язків з клієнтом і товаром, зміна суми/кількості, швидка зміна статусу	Таблиця `orders`, константа `ORDER_STATUSES`, група вебмаршрутів `/orders`	Немає фінансового блоку для часткових оплат, відстеження посилок по ТТН і історії статусів

Продовження таблиці 3.1

1	2	3	4
Ведення товарного каталогу	Комплексне створення картки, візуальний перегляд, ручне коригування згенерованих ШІ-полів	Таблиця `products`, часткові шаблони товарів (`partials/product_list.html` та ін.)	Технічно не підтримується масовий пакетний імпорт та експорт каталогу у форматах CSV/XML/Excel
Організатор операційних задач	Планування нагадування, перегляд загального списку, зміна опису, швидке закриття однією кнопкою (`toggle`)	Таблиця бази даних `tasks`, група маршрутів `/tasks`	Не реалізована повноцінна інтеграція та двостороння синхронізація із зовнішніми електронними календарями
ШІ-обробка медіафайлів та генерація тексту	Прецизійне видалення фону на загружених фотографіях, розумна генерація заголовка, опису, переваг і SEO-тегів	Файли-сервіси `background_removal.py`, `description_generator.py`	Жорстка залежність результату від якості початкового фото та тимчасової доступності серверів Gemini API
API для сторонніх інтеграцій	Безконтактна віддалена програмна обробка інформації через передачу захищеного ключа доступу Bearer	Ендпоінт контролера `/api/v1/process-product`, логіка авторизації у `verify_api_key`	Даний API-маршрут повертає тільки структуру JSON і самостійно не здійснює запис всередині бази CRM

З точки зору щоденної, рутинної експлуатації платформи середньостатистичним менеджером, основний бізнес-процес виглядає як логічна послідовність кроків: вхід до системи через екран автентифікації, швидкий візуальний контроль оглядової панелі для оцінки завантаження та виявлення нагальних справ. Далі, за необхідності, відбувається внесення даних нового клієнта, якщо покупець звернувся до магазину вперше. Наступний етап

– швидка підготовка якісної товарної картки. Менеджеру більше не потрібно годинами писати описи і вирізати товари у Photoshop; завдяки ШІ-інструментарію він робить це за лічені секунди. Після цього відбувається фінальне оформлення сутності замовлення в базі даних та обов’язкове планування операційної задачі для наступного контрольного дзвінка. По завершенні усіх дій, спеціаліст просто оновлює статус замовлення й натискає кнопку закриття супровідної задачі. Інтерфейс миттєво, за допомогою HTMX, відображає ці зміни.

3.2 Проєктування архітектури застосунку

В основі архітектури розробленої програмної системи закладено модель монолітного вебзастосунку з логічним поділом на кілька рівнів. До них належать користувацький інтерфейс на боці браузера, маршрутизація FastAPI на стороні сервера, прикладні сервіси штучного інтелекту для обробки медіа та тексту, а також шар реляційного збереження даних на базі SQLite. Для задач дипломного проєкту такий підхід є доцільним, оскільки дозволяє запускати систему локально або в Docker-контейнері без надмірної інфраструктури. Інформаційний цикл від HTML-форми до збереження даних залишається прозорим для аналізу.

Глобальна концептуальна схема цієї архітектури дуже детально відображена та задокументована у графічному файлі thesis/report_assets/diagrams/01_system_architecture.png. Технічним ядром системи безперечно є серверний вебзастосунок на фреймворку FastAPI. Завдяки вбудованому серверу ASGI (Uvicorn), він постійно і асинхронно очікує та обробляє вхідні HTTP-запити. У відповідь цей сервер повертає або повні відрендерені HTML-сторінки, або оптимізовані часткові HTML-фрагменти (використовуючи технологію HTMX для часткового оновлення DOM-дерева браузера). Застосунок напряму комунікує із локальною базою даних SQLite для

виконання необхідних SQL-транзакцій. Коли ж виникає потреба специфічної обробки контенту, сервер ініціює виклики двох паралельних спеціалізованих сервісів: один запускає ресурсомісткі математичні операції для видалення графічного фону на процесорі сервера, а другий виконує HTTPS-запит до стороннього алгоритму Gemini API.

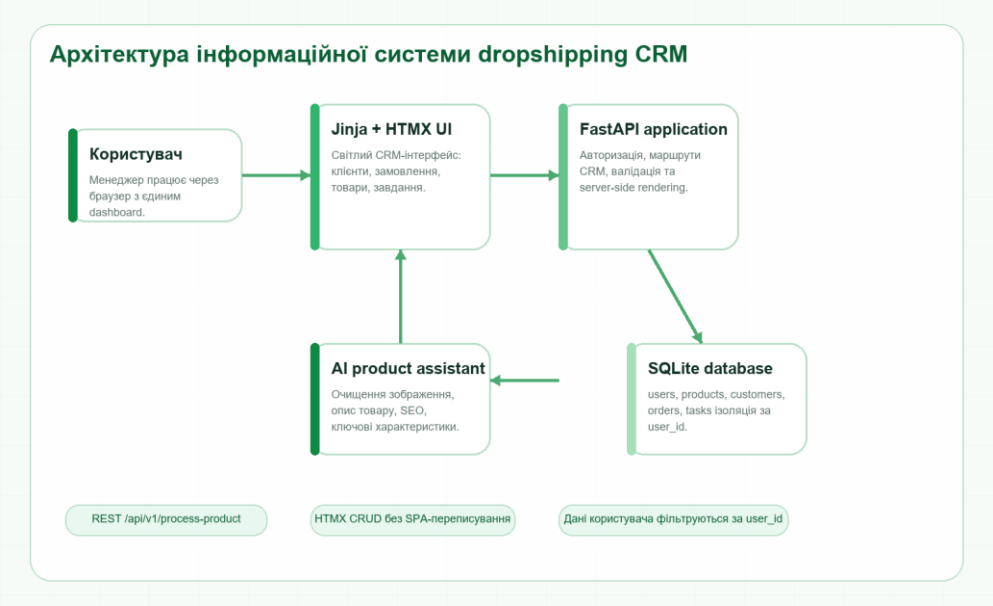

Рисунок 3.1 – Загальна архітектура інформаційної системи

Рисунок 3.2 – Доменний workflow обробки замовлення дропшипінг-платформи

Організаційно весь розроблений програмний код розподілено на чотири відповідальні функціональні шари:

– шар представлення (Frontend/UI): фізично складається із Jinja2-шаблонів, таблиць стилів CSS та декларативних атрибутів бібліотеки HTMX. Ці атрибути забезпечують динамічну AJAX-поведінку сторінки без необхідності писати окрему JavaScript-логіку для кожної дії;

– шар контролерів (Routing/API): охоплює всі маршрути FastAPI, прописані у main.py. Вони виконують функцію «регулювальника»: приймають мережеві запити, перевіряють автентичність криптографічної сесії, суворо валідують вхідні дані з вебформ, викликають сервісні функції з інших шарів та формують остаточну HTML- або JSON-відповідь для клієнта;

– шар прикладної бізнес-логіки (Services): містить вузькоспеціалізовані програмні модулі (зокрема background_removal.py і description_generator.py). Вони інкапсулюють у собі складні обчислення та взаємодію з нейромережевими алгоритмами поза межами звичайної рутинної CRUD-логіки додавання записів;

– шар доступу до даних (Data/Persistence): повністю базується на можливостях СКБД SQLite та включає в себе великий комплекс функцій у файлі app/database.py. Ці функції повністю приховують від контролерів специфіку написання сирих SQL-запитів, обробляють підключення і гарантують безпечне виконання транзакцій.

Важливою частиною архітектурного проєктування є контейнеризація, задокументована у файлі docker-compose.yml. Єдиний сервіс app збирається за інструкціями Dockerfile, відкриває вебпорт 8000, зчитує конфігурацію з .env і монтує постійний том product_data. Саме туди зберігається файл бази даних /data/products.db, що забезпечує відтворюваність середовища та збереження даних після перезапуску контейнера.

Користувацька частина та використання HTMX-технологій.

Інтерфейс користувача платформи побудований за перевіреною роками парадигмою класичного серверного рендерингу (Server-Side Rendering). Це

означає генерацію повністю готового до відображення у браузері HTML-коду безпосередньо на стороні Python-бекенду. Фундамент візуальної частини складають базові файли шаблонів: base.html, login.html, register.html та основний робочий простір менеджера index.html. Файл base.html виконує роль головного каркаса: він містить спільну незмінну структуру будь-якої сторінки, такі як підключення базових веб-шрифтів, скриптів HTMX та файлів каскадних таблиць стилів (CSS). Шаблони login.html і register.html спеціалізовані виключно на критичних процесах входу в систему, в той час як масивний файл index.html організовує складний розгалужений простір, розподіляючи контент між зручними вкладками.

Застосування шаблонізатора Jinja2 узгоджується зі специфікою розроблюваної CRM-системи. Основні дії користувача зводяться до заповнення форм, перегляду табличних списків і роботи з бічними панелями деталізації. Залучення важких frontend-фреймворків формату Single Page Application у цьому контексті було б надмірним ускладненням, оскільки потребувало б дублювання маршрутизації, системи збирання та управління станом. Натомість FastAPI формує HTML на сервері, а логіка відображення залишається пов’язаною з обробниками бекенду.

Незважаючи на використання серверного рендерингу, для плавної роботи інтерфейсу застосовано HTMX. Ця технологія дозволяє оновлювати окремі блоки сторінки без повного перезавантаження. Наприклад, коли менеджер додає нового клієнта або змінює статус замовлення, сервер повертає лише потрібний HTML-фрагмент: рядок таблиці, оновлений список або повідомлення. Такі часткові шаблони зібрані в директорії app/templates/partials/, а в HTML використовуються атрибути hx-post, hx-target та hx-swap.

З візуальної точки зору користувацький інтерфейс має робочий і мінімалістичний характер. Акцент зроблено на продуктивності менеджера: компактні інформаційні зведення, структуровані таблиці, зрозумілі форми введення даних і бічні панелі деталізації. Скриншоти 01_login.png – 08_detail_drawer.png фіксують основні стани системи: вхід, реєстрацію,

оглядову панель, клієнтів, замовлення, товари з ШІ-помічником, задачі та детальний перегляд запису.

Серверна частина та маршрутизація FastAPI.

Потужна серверна складова розробленого застосунку цілком спирається на асинхронні можливості надсучасного Python-фреймворку FastAPI. Головним програмним хабом усієї системи є файл app/main.py. Саме в ньому ініціалізується екземпляр застосунку app = FastAPI(...), підключаються глобальні проміжні шари (middlewares) для безпеки, монтуються директорії для віддачі статичних файлів і шаблонів, та, найголовніше, декларативно за допомогою декораторів реєструються всі HTTP-маршрути (ендпоінти). Також у цьому файлі окремо визначено публічний API-маршрут POST /api/v1/process-product, розроблений виключно для програмної безкористувацької інтеграції через обмін серіалізованими даними у форматі JSON.

Опрацювання будь-якого стандартного веб-запиту на сервері завжди слідує чітко визначеній захищеній послідовності. При надходженні запиту від браузера, фреймворк спершу ініціює процес обробки через SessionMiddleware, автоматично розшифровуючи вміст сесійної cookie клієнта на основі заданого складного секретного ключа (settings.secret_key). На наступному етапі, в тілі конкретного викликаного маршруту, обов’язково викликається допоміжна функція get_session_user. Її завдання – перевірити, чи дійсно присутній у сесії активний ідентифікатор користувача. Якщо він відсутній – сервер негайно перериває обробку запиту і безпечно перенаправляє (через HTTP статус 302) клієнта на сторінку автентифікації /login.

Архітектура вебмаршрутів логічно згрупована за основними бізнес-сутностями CRM:

– маршрути автентифікації: Включають /login, /register, /logout. Відповідають за ініціалізацію та криптографічне руйнування сесій, перевірку унікальності логіна користувача у базі під час його першої реєстрації;

– маршрути клієнтів (/customers): Забезпечують прийом POST-запитів зі стандартними HTML-формами для створення нового контакту в базі, PATCH-

запитів для редагування вже існуючого профілю клієнта та DELETE-запитів для його каскадного видалення;

– маршрути товарів (/products): Окрім класичного читання та редагування тексту, саме POST /products бере на себе найскладнішу задачу обробки форми. Ця форма містить не лише текст, а й завантажений бінарний файл UploadFile. Відбувається жорстка програмна валідація MIME-типу файлу (система пропускає лише image/jpeg та image/png) і строгий контроль його максимального розміру. Після успішної перевірки ініціюються алгоритми ШІ [28];

– маршрути замовлень (/orders): Управляють складним процесом збереження торгових угод. Особливо варто відзначити унікальний маршрут PATCH /orders/{order_id}/status, який дозволяє одним швидким натисканням перевести замовлення на наступний етап (статус) без відкриття важкої сторінки повного редагування всіх полів;

– маршрути задач (/tasks): Надають розширені можливості планування і, аналогічно до замовлень, мають свій оптимізований HTMX-ендпоінт PATCH /tasks/{task_id}/toggle для миттєвої зміни стану задачі на done або назад на open.

Математична та нейромережева обробка штучного інтелекту відбувається безпосередньо під час виконання маршруту створення нового товару. Якщо в процесі локального видалення фону алгоритмом виникне технічна аномалія (наприклад, файл сильно пошкоджений), або якщо мережеві сервери зовнішньої моделі Gemini тимчасово не відповідатимуть на запити генерації тексту через перевантаження, сервер на FastAPI зобов’язаний діяти максимально стійко. Спеціальний програмний блок try...except надійно перехоплює всі подібні винятки виконання. Замість аварійного падіння всієї програми з критичною помилкою 500 (Internal Server Error), бекенд повертає користувачу акуратне форматоване HTML-повідомлення, пояснюючи зрозумілою мовою причину, через яку AI-сервіс саме зараз недоступний. Таким чином, застосунок демонструє високий рівень відмовостійкості (resilience).

Детальний опис структури бази даних та SQL-функцій.

Ядром довгострокового зберігання інформації у створеній системі є локальна реляційна СКБД SQLite. Усі функції звернення до бази зібрані, систематизовані та локалізовані у єдиному файлі app/database.py. Застосування SQLite у цьому навчально-прикладному проєкті є архітектурно найбільш вигідним рішенням: база даних має вигляд єдиного компактного бінарного файлу за шляхом /data/products.db. Це повністю і назавжди усуває необхідність встановлення, важкого налаштування, тюнінгу та постійного адміністрування масивних серверних рішень бази даних. В умовах експлуатації через Docker, директорія із файлом бази автоматично монтується як незалежний системний том, що захищає накопичені дані від зникнення після перезбирання контейнера.

При кожному старті застосунку запускається процедура init_db(), що проводить ініціалізацію або верифікацію наявної схеми. Вона послідовно виконує SQL-команди створення таблиць CREATE TABLE IF NOT EXISTS, а також реалізує спрощений механізм легких системних міграцій. За допомогою службових запитів PRAGMA table_info система здатна самостійно визначити відсутні колонки і прозоро додати їх до існуючої бази (командою ALTER TABLE ADD COLUMN).

Структура моделі даних базується на п’яти основних взаємопов’язаних таблицях, які забезпечують зберігання інформації з дотриманням реляційної цілісності.

Таблиця users відповідає за збереження облікових записів (логін, хешований пароль). Таблиця products містить дані товарного каталогу, включно з вихідними характеристиками та згенерованим ШІ контентом (структурований опис у JSON, зображення у Base64). Таблиця customers зберігає профілі покупців із контактними та логістичними даними. Таблиця orders фіксує комерційні угоди, їхні статуси та зовнішні ключі на клієнта й товар. Таблиця tasks забезпечує роботу органайзера задач із дедлайнами та статусами виконання.

Усі об’єкти, крім записів users, містять обов’язкове поле user_id для забезпечення мультитенантності та ізоляції даних. Для підтримки цілісності

бази реалізовано механізми безпечного оновлення зовнішніх ключів під час видалення зв’язаних сутностей (наприклад, скидання customer_id у замовленнях при видаленні клієнта).

Архітектурні зв’язки між таблицями бази даних візуалізовані на рисунку 3.3 (ER-модель бази даних).

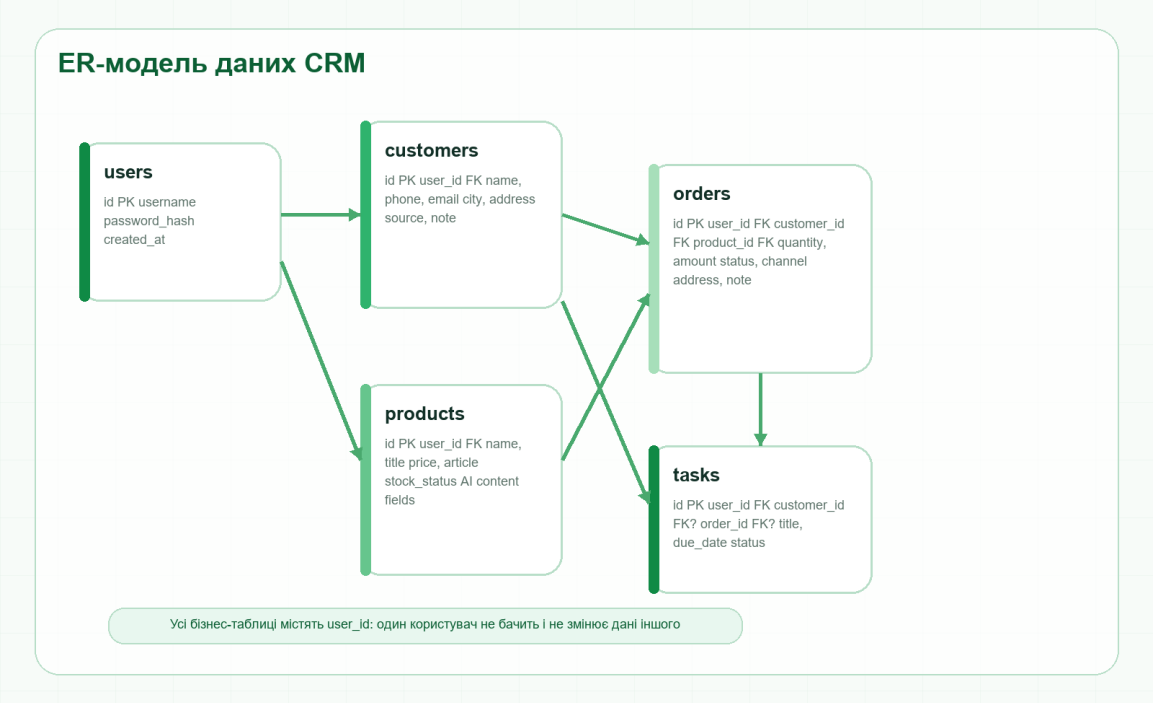

Рисунок 3.3 – ER-модель бази даних CRM-системи

Механізм ізоляції користувацьких даних за user_id.

Безпека збереження комерційної інформації у розробленій системі базується на ізоляції даних на рівні окремих записів бази. Оскільки платформа розроблялася як багатокористувацька, важливим завданням є обмеження доступу кожного менеджера лише до власних клієнтів, товарів, замовлень і задач.

Головним елементом цього механізму виступає поле user_id, яке додано до змістовних таблиць бази (products, customers, orders, tasks). Процес взаємодії з інформацією реалізовується у кілька послідовних етапів.

Під час успішного проходження процедури логіну, ідентифікатор користувача витягується з таблиці users та надійно записується у зашифровану сесію веббраузера за допомогою словника request.session. Ця сесія підписується та шифрується криптографічним ключем фреймворку і в принципі не може бути підроблена чи модифікована клієнтом. При кожному наступному HTTP-запиті спеціальна функція get_session_user розпаковує цей ідентифікатор. Якщо система виявляє його відсутність – сесія вважається невалідною.

Найважливіший етап безпеки відбувається під час генерації SQL-запитів. Отриманий із сесії user_id завжди передається у функціональні модулі всередині database.py. Кожен запит на читання інформації (інструкція SELECT) на рівні СКБД обмежується жорсткою умовою. Наприклад: SELECT * FROM customers WHERE user_id = ?. Як наслідок, запит технічно здатен повернути тільки ті рядки таблиці, що належать конкретному власнику, що робить витік чужих даних фізично неможливим.

Аналогічно, запити на оновлення (UPDATE) чи видалення (DELETE) виконують таку саму перевірку. Наприклад, редагування товару має умову UPDATE products SET ... WHERE id=? AND user_id=?. Це зменшує ризик Insecure Direct Object Reference (IDOR), оскільки підміна ідентифікатора запису не дає доступу до чужих даних без збігу user_id. Для автономного API-маршруту передбачено окрему логіку: залежність verify_api_key перевіряє переданий Bearer токен і дозволяє доступ лише за умови збігу системних секретів конфігурації.

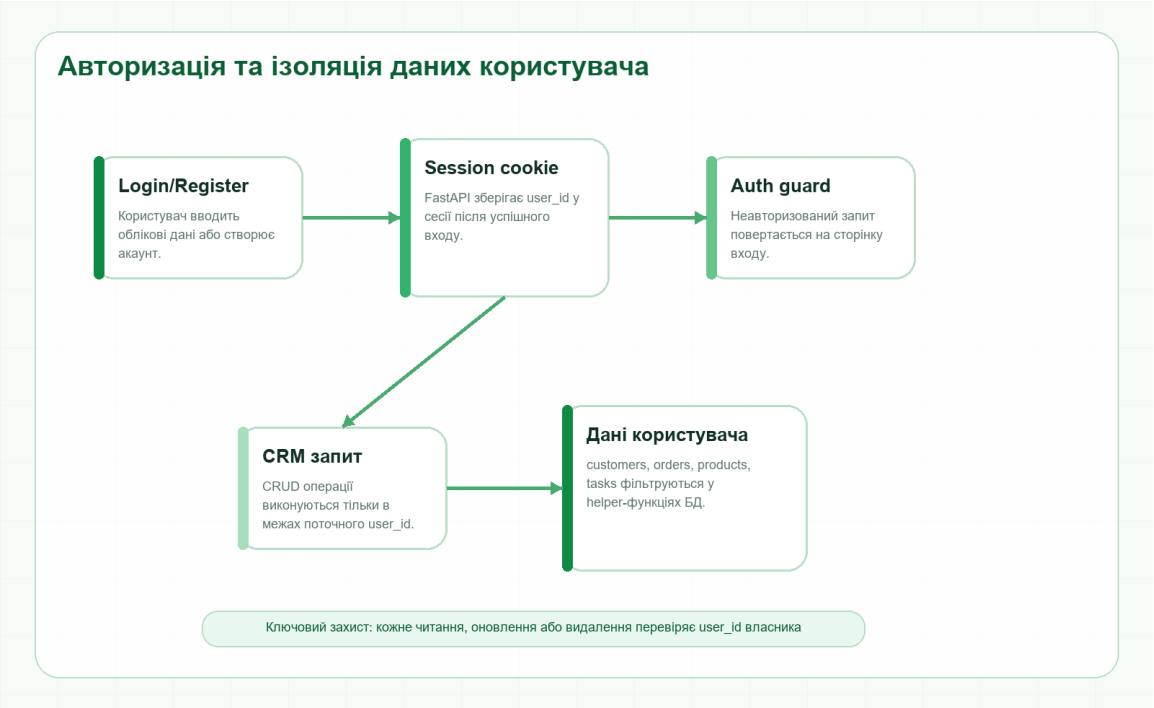

Рисунок 3.4 – Модель авторизації та ізоляції даних за user_id

Рисунок 3.5 – Pipeline модуля інтелектуальної обробки товарного контенту

3.3 Сценарії використання та перевірка працездатності застосунку

Практична ілюстрація щоденної роботи створеного застосунку відображає життєвий цикл управління продажами: безпечна автентифікація, огляд інформаційної панелі, робота з табличними списками сутностей, залучення ШІ для підготовки контенту та адміністрування замовлень і задач.

Сценарій взаємодії починається зі сторінки реєстрації та входу (Рисунок 3.6, Рисунок 3.7). Після безпечної перевірки облікових даних користувач переспрямовується на оглядову інформаційну панель (Рисунок 3.8). Dashboard агрегує статистичні показники: загальну кількість клієнтів і товарів, статус поточних замовлень та лічильник відкритих нагальних задач.

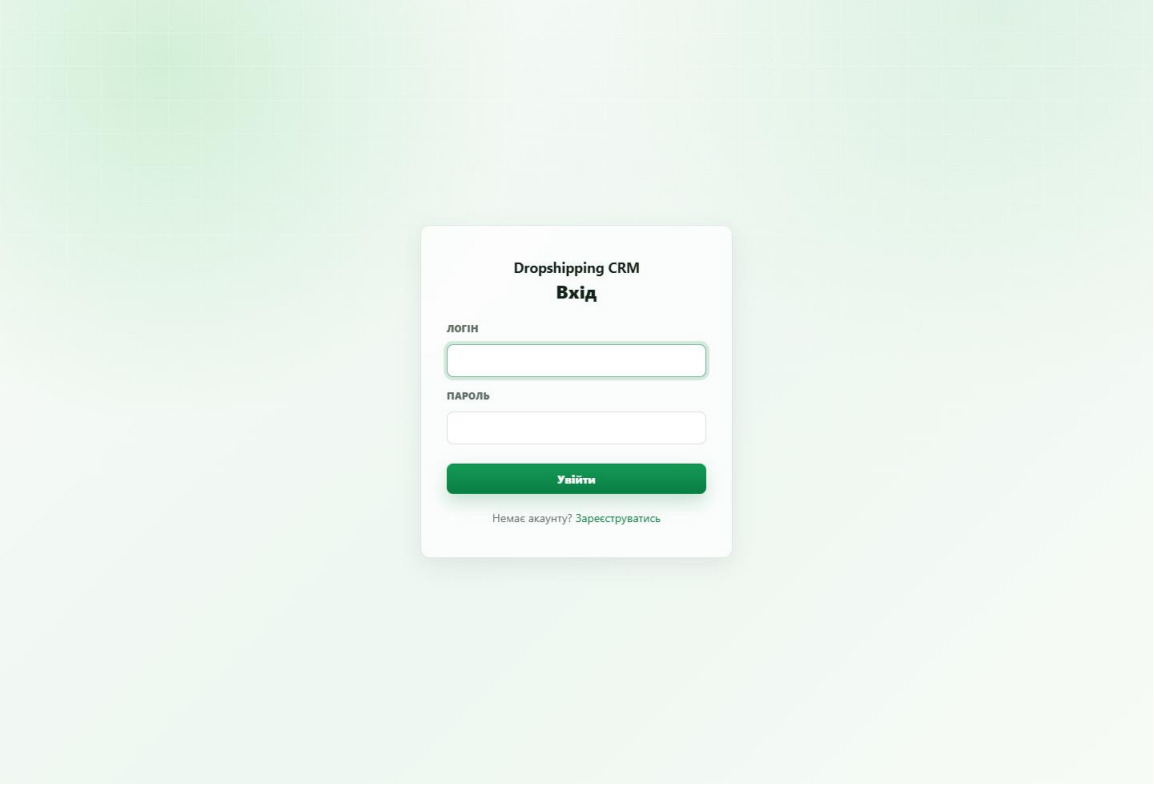

Рисунок 3.6 – Сторінка входу до CRM-системи

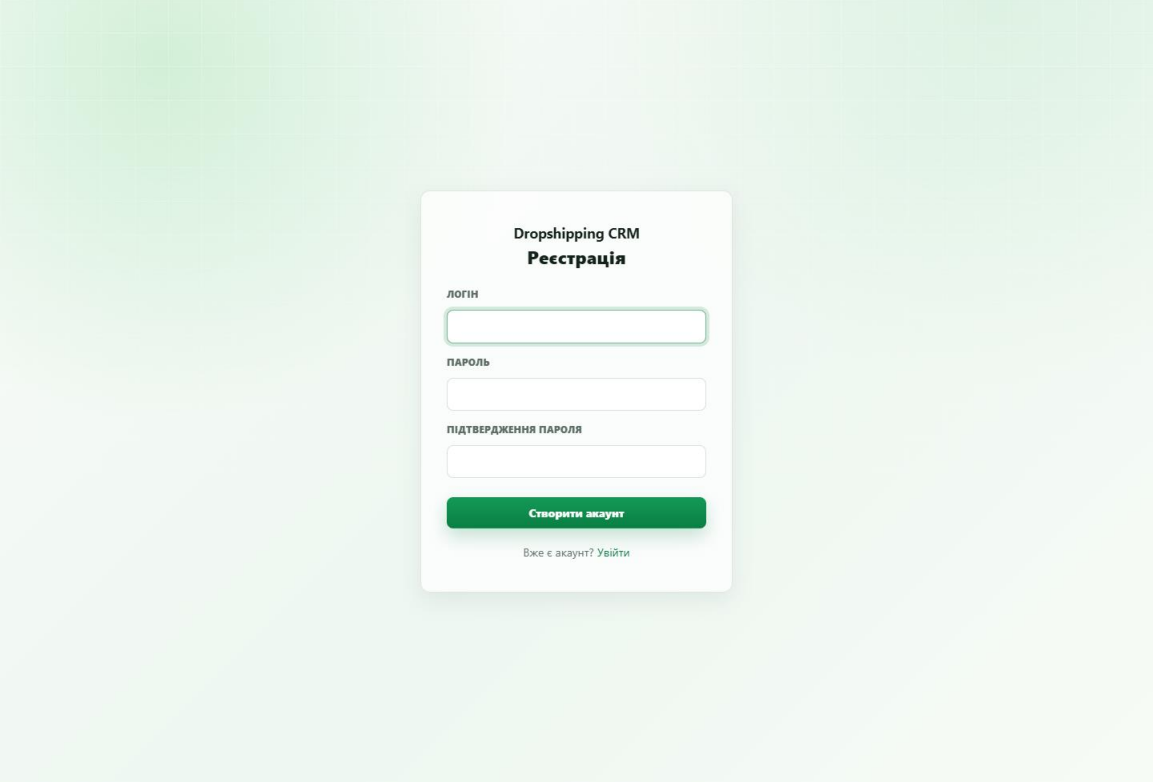

Рисунок 3.7 – Форма реєстрації користувача

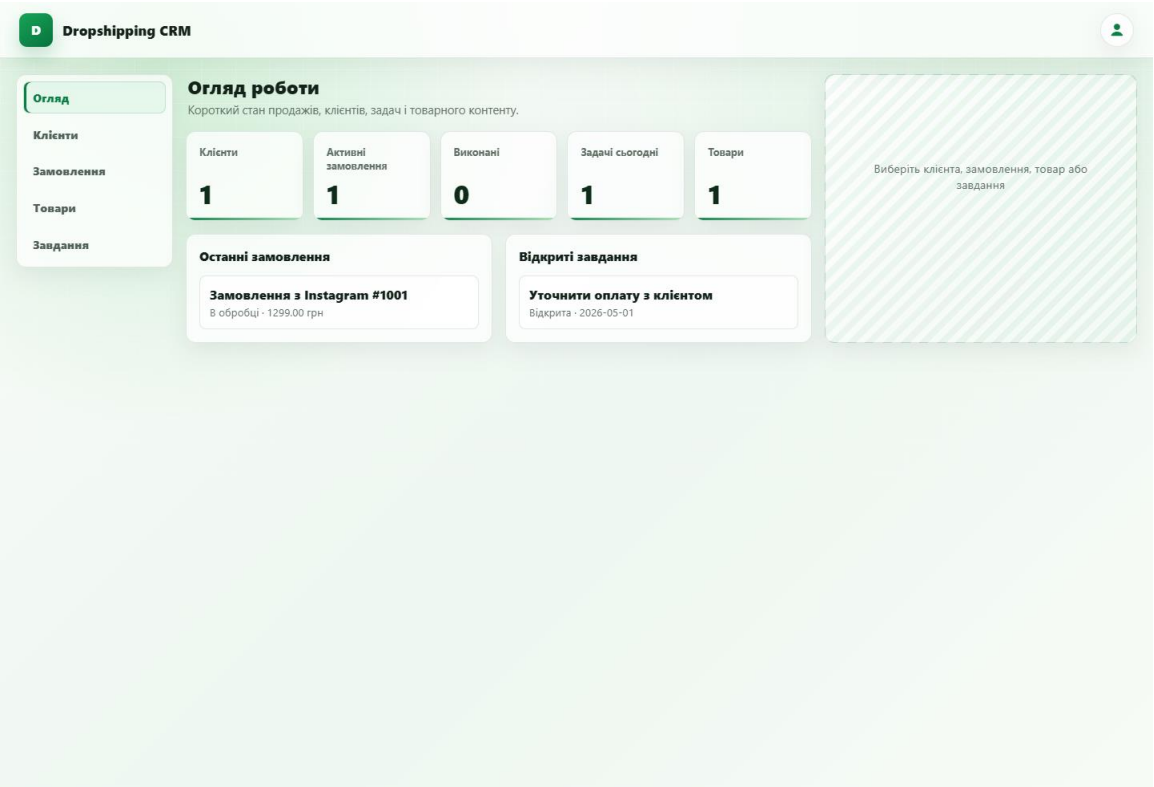

Рисунок 3.8 – Оглядова інформаційна панель CRM-системи

Під час роботи з клієнтською базою менеджер використовує розділ керування клієнтами (Рисунок 3.9), де створює нові профілі, що містять контактні та логістичні дані. Для ефективного огляду передбачено бічну висувну панель деталізації (Рисунок 3.13), яка зводить воєдино всю комунікаційну історію, пов’язані замовлення та задачі конкретного покупця.

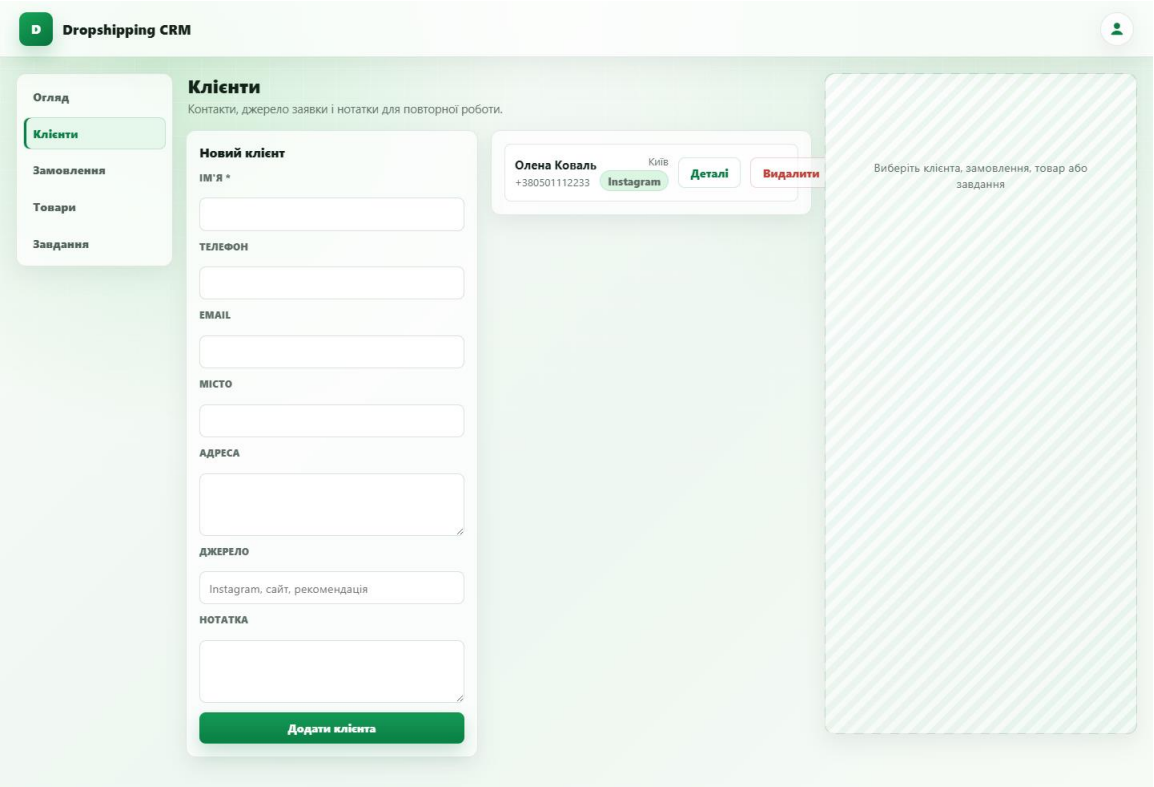

Рисунок 3.9 – Секція керування клієнтами

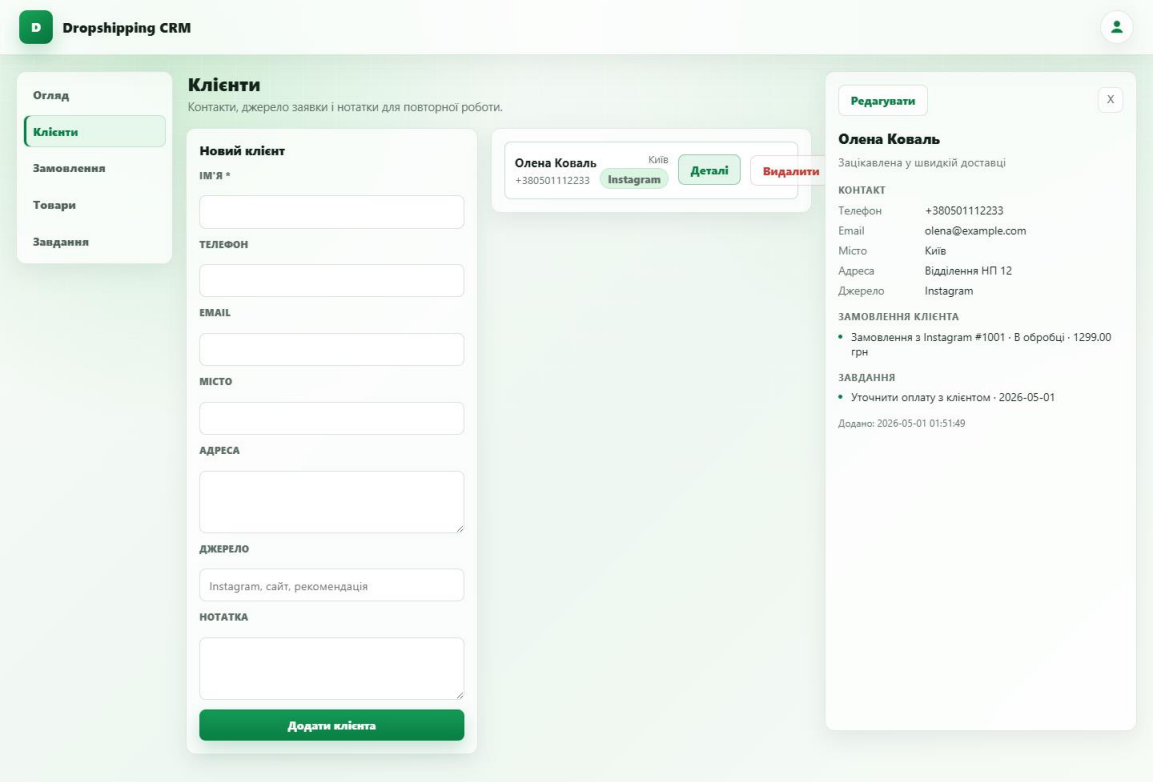

Рисунок 3.10 – Бічна панель деталізації запису

Ключовим елементом роботи є наповнення товарного каталогу (Рисунок 3.11). Менеджер вносить базові дані та завантажує фотографію, після чого ініціюється виклик ШІ-асистента. Система очищує зображення та звертається до Gemini API для генерації комерційного опису. Результат зберігається в базі та доступний для використання при оформленні продажу. Наступним етапом є фіксація угоди у розділі замовлень (Рисунок 3.10), де обирається клієнт, товар і встановлюється поточний операційний статус. Організацію подальших дій забезпечує розділ задач (Рисунок 3.12), що дозволяє планувати нагадування та швидко змінювати їх статус.

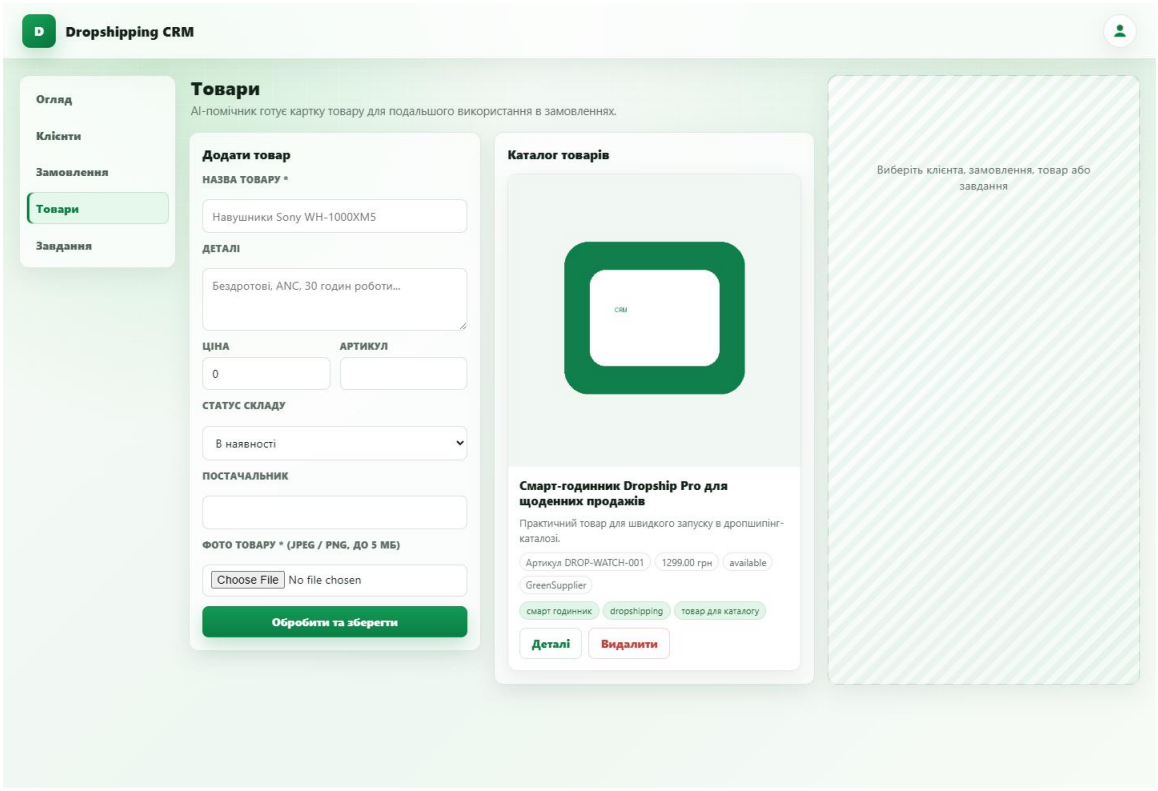

Рисунок 3.11 – Секція товарів і ШІ-помічника для товарного контенту

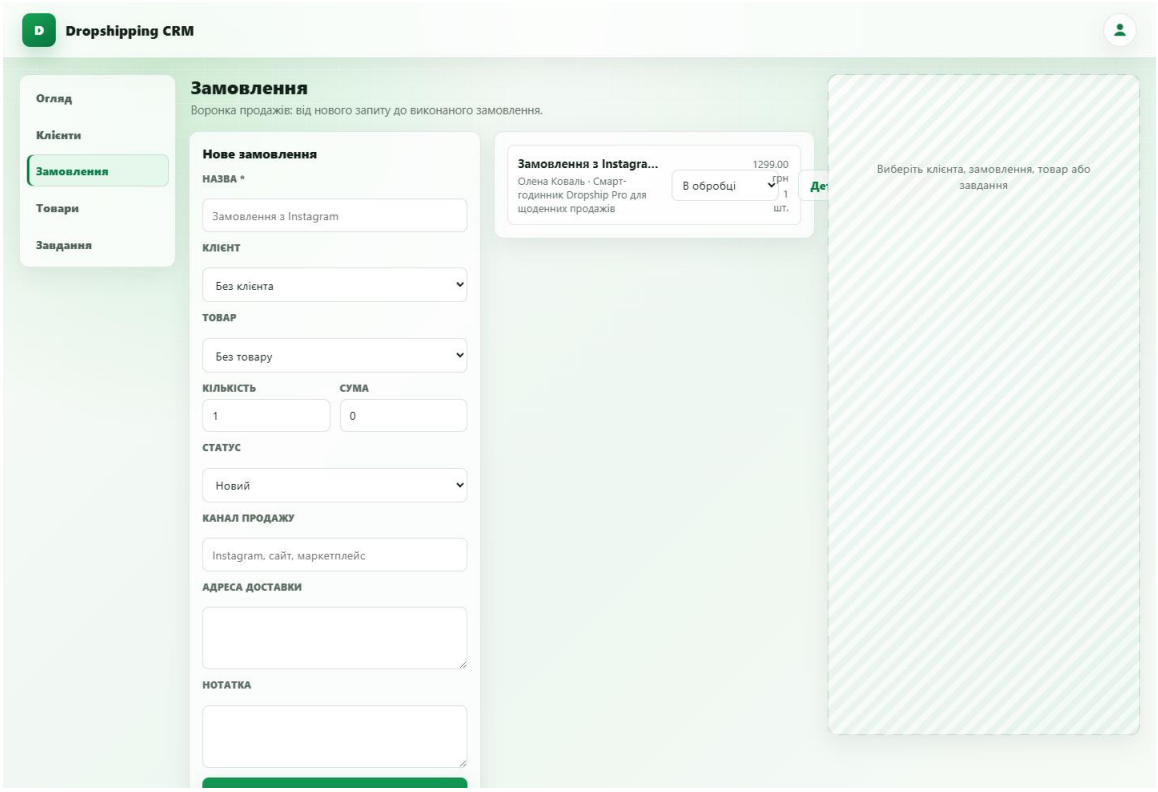

Рисунок 3.12 – Секція керування замовленнями

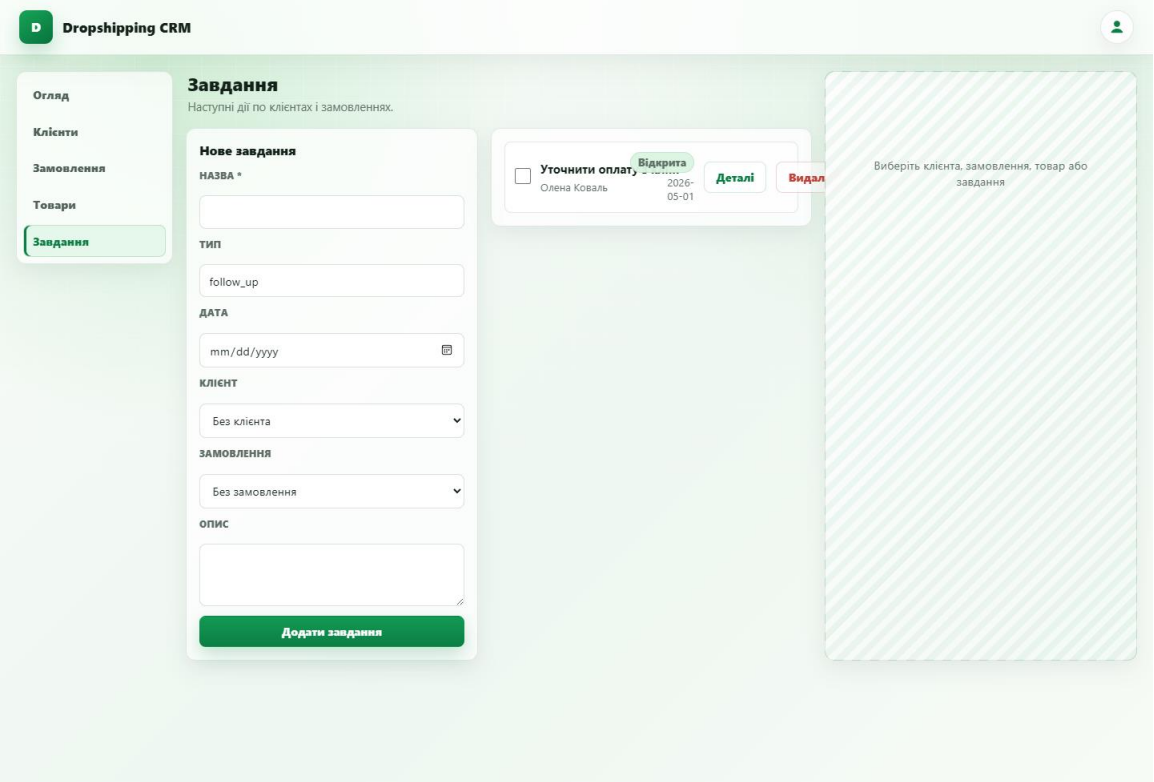

Рисунок 3.13 – Секція задач менеджера

Окремим напрямом роботи системи є можливість її використання через програмний API. Сторонні системи можуть надсилати запити із ключем Bearer, передаючи вихідні параметри товару, та отримувати у відповідь JSON із результатами ШІ-обробки без запису до внутрішньої БД.

Комплексна перевірка підтвердила коректне функціонування всіх базових сценаріїв, надійність ізоляції сесій між користувачами та стабільність інтеграції з ШІ-модулями. У таблиці 3.2 подано узагальнені результати перевірки.

Таблиця 3.2 – Результати перевірки працездатності застосунку

Напрям тестування	Очікуваний результат	Статус
1	**2**	**3**
Автентифікація та сесії	Успішний вхід, зберігання зашифрованої сесії, вихід	Виконано
Керування клієнтами	Створення, редагування, збереження даних	Виконано

Продовження таблиці 3.2

1	2	3
Оформлення замовлень	Прив’язка до клієнта, зміна статусів через HTMX	Виконано
Робота ШІ-модуля	Видалення фону, генерація JSON від Gemini API	Виконано
Ізоляція даних	Заборони доступу до записів інших `user_id`	Виконано
Запуск у Docker	Ініціалізація БД, стабільна доступність портів	Виконано

Отже, розділ демонструє працездатне ядро системи, що ефективно поєднує керування CRM-сутностями з можливостями штучного інтелекту, а перевірка підтверджує надійність механізмів безпеки та збереження даних.

ВИСНОВКИ

У кваліфікаційній роботі розглянуто проєктування та розробку інформаційної системи управління замовленнями дропшипінг-платформи з модулем інтелектуальної обробки контенту. Логіка роботи побудована від предметної області до реалізації: спочатку проаналізовано CRM/e-commerce контекст, далі окремо досліджено підготовку товарного контенту, після цього описано застосунок із його маршрутами, моделлю даних, інтерфейсом і перевірками.

У першому розділі проаналізовано розвиток CRM-систем, їх поширення в електронній комерції та специфіку використання в дропшипінгу. Для цієї моделі цінність має не окремий товарний запис, а зв’язка клієнта, замовлення, задачі, каналу продажу, адреси доставки й товарного контенту. Огляд існуючих рішень показав нішу для компактної CRM/e-commerce системи, орієнтованої на малу дропшипінг-команду й навчально-прикладний сценарій.

У другому розділі досліджено модуль інтелектуальної обробки товарного контенту. Розглянуто мету використання ШІ в товарній картці, напрями оцінювання, роль інструкції для генерації структурованого опису, критерії якості та обмеження складних вхідних даних. Обґрунтовано комбінований підхід, за якого rembg використовується для видалення фону товарного зображення, Gemini - для формування структурованого опису, а остаточна перевірка й редагування залишаються за користувачем.

У третьому розділі описано розробку CRM/e-commerce застосунку. Практичним результатом є вебзастосунок на основі FastAPI, Jinja, HTMX і SQLite, що запускається через Docker Compose. У реалізації наявні реєстрація та вхід користувача, сесійна автентифікація, ізоляція даних за user_id, персональний каталог товарів, клієнтські записи, замовлення, задачі, оглядова панель і API-ендпоінт для обробки товару.

Реалізована система демонструє працездатне CRM/e-commerce ядро для дропшипінг-сценарію, але не є завершеною промисловою платформою.

Інтеграції з постачальниками, платіжні модулі, служби доставки, розширені ролі, журнал аудиту, складський облік і поглиблена аналітика залишаються напрямами подальшого розвитку. Таке розмежування не прикрашає результат зайвими обіцянками й відокремлює фактичний функціонал від перспективних можливостей.

Поставлену мету досягнуто: спроєктовано та реалізовано інформаційну систему, що поєднує керування сутностями дропшипінг-платформи з модулем інтелектуальної обробки товарного контенту. Практична цінність результату полягає у тому, що клієнти, замовлення, товари, задачі, обробка зображення та генерація опису працюють у межах одного прикладного середовища, а не як непов’язані демонстраційні фрагменти. Оформлення пояснювальної записки узгоджено з вимогами до структури науково-технічних звітів [29]. Бібліографічні описи подано з урахуванням правил складання джерел посилання [30].

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Chaffey, D., Hemphill, T., & Edmundson-Bird, D. (2019). *Digital business and e-commerce management* (7th ed.). Pearson.

2. Laudon, K. C., & Traver, C. G. (2021). *E-commerce 2020–2021: Business, technology and society* (16th ed.). Pearson.

3. Davenport, T., Guha, A., Grewal, D., & Bressgott, T. (2020). How artificial intelligence will change the future of marketing. *Journal of the Academy of Marketing Science, 48*, 24–42.

4. Buttle, F., & Maklan, S. (2019). Customer relationship management: Concepts and technologies (4th ed.). Routledge.

5. Buttle, F., & Maklan, S. (2019). Customer relationship management: Concepts and technologies (4th ed.). Routledge.

6. Meena, P., & Sahu, P. (2021). Customer relationship management research from 2000 to 2020: An academic literature review and classification. *Vision, 25*(2), 136–158.

7. Romano, N. C., Jr., & Fjermestad, J. (2001). Electronic commerce customer relationship management: An assessment of research. *International Journal of Electronic Commerce, 6*(2), 61–113.

8. Chiang, W. K., & Feng, Y. (2010). Retailer or e-tailer? Strategic pricing and economic-lot-size decisions in a competitive supply chain with drop-shipping. *Journal of the Operational Research Society, 61*(11), 1645–1653.

9. Peinkofer, S. T., Esper, T. L., Smith, R. J., & Williams, B. D. (2019). Assessing the impact of drop-shipping fulfilment operations on the upstream supply chain. *International Journal of Production Research, 57*(11), 3598–3621.

10. Hübner, A., Kuhn, H., & Wollenburg, J. (2016). Last mile fulfilment and distribution in omni-channel grocery retailing: A strategic planning framework. *International Journal of Retail & Distribution Management, 44*(3), 228–247.

11. Hermann, E., & Puntoni, S. (2025). Generative AI in marketing and principles for ethical design and deployment. *Journal of Public Policy & Marketing, 44*(3).

12. Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., ... Amodei, D. (2020). Language models are few-shot learners. *Advances in Neural Information Processing Systems, 33*, 1877–1901.

13. Liu, P., Yuan, W., Fu, J., Jiang, Z., Hayashi, H., & Neubig, G. (2023). Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. ACM Computing Surveys, 55(9), 1–35.

14. Rother, C., Kolmogorov, V., & Blake, A. (2004). GrabCut: Interactive foreground extraction using iterated graph cuts. *ACM Transactions on Graphics, 23*(3), 309–314.

15. Ronneberger, O., Fischer, P., & Brox, T. (2015). U-Net: Convolutional networks for biomedical image segmentation. In *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015* (pp. 234–241). Springer.

16. Кобилін, О. А., & Творошенко, І. С. (2021). *Методи цифрової обробки зображень: навчальний посібник*. ХНУРЕ.

17. Творошенко, І. С. (2021). *Технології прийняття рішень в інформаційних системах: навчальний посібник*. ХНУРЕ.

18. Гороховатський, В. О., & Творошенко, І. С. (2021). *Методи інтелектуального аналізу та оброблення даних: навчальний посібник*. ХНУРЕ.

19. Daradkeh, Y. I., Gorokhovatskyi, V., Tvoroshenko, I., Gadetska, S., & Al-Dhaifallah, M. (2021). Methods of classification of images on the basis of the values of statistical distributions for the composition of structural description components. *IEEE Access, 9*, 92964–92973.

20. Daradkeh, Y. I., Gorokhovatskyi, V., Tvoroshenko, I., & Zeghid, M. (2022). Cluster representation of the structural description of images for effective classification. *Computers, Materials & Continua, 73*(3), 6069–6084.

21. Гороховатський, В. О., Творошенко, І. С., & Чмутов, Ю. В. (2022). Застосування систем ортогональних функцій для формування простору ознак у методах класифікації зображень. *Сучасні інформаційні системи, 6*(3), 5–12.

22. Гороховатський, В., Передрій, О., Творошенко, І., & Марков, Т. (2023). Матриця відстаней для множини компонентів структурного опису як інструмент для створення класифікатора зображень. *Сучасні інформаційні системи, 7*(1), 5–13.

23. Gorokhovatskyi, V., Tvoroshenko, I., Kobylin, O., & Vlasenko, N. (2023). Search for visual objects by request in the form of a cluster representation for the structural image description. *Advances in Electrical and Electronic Engineering, 21*(1), 19–27.

24. Tvoroshenko, I., Gorokhovatskyi, V., Kobylin, O., & Tvoroshenko, A. (2023). Application of deep learning methods for recognizing and classifying culinary dishes in images. *International Journal of Academic and Applied Research, 7*(9), 57–70.

25. Karakonstantyn, D., & Tvoroshenko, I. (2024). About the issue of optimization the performance of the server part of the information system. In *Abstracts of XII International Scientific and Practical Conference "Prospective directions of modern science and education in the world"* (pp. 364–366).

26. Bohdan, N., Tvoroshenko, I., Gorokhovatskyi, V., & Kobylin, O. (2025). Development of a hybrid method to enhance context memory for a chatbot application based on large language models. *International Journal of Academic Information Systems Research, 9*(10), 7–18.

27. Suprun, A., Tvoroshenko, I., Gorokhovatskyi, V., & Yakovleva, O. (2025). Development and research of a method for the combined use of large language models for text generation. *International Journal of Academic and Applied Research, 9*(10), 249–263.

28. Гороховатський, В., & Творошенко, І. (2026). *Продуктивні моделі аналізу даних у методах розпізнавання зображень: монографія*. ХНУРЕ.

29. Tvoroshenko, I., & Tkachenko, D. (2020). Mechanisms of image classification based on descriptors of local features. In *Abstracts of IV International Scientific and Practical Conference "Integration of scientific bases into practice"* (pp. 443–448).

30. Daradkeh, Y. I., Tvoroshenko, I., Gorokhovatskyi, V., Latiff, L. A., & Ahmad, N. (2021). Development of effective methods for structural image recognition using the principles of data granulation and apparatus of fuzzy logic. *IEEE Access, 9*, 13417–13428.