ДЕРЖАВНИЙ УНІВЕРСИТЕТ
ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНИХ ТЕХНОЛОГІЙ
НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ
КАФЕДРА ІНФОРМАЦІЙНИХ СИСТЕМ ТА ТЕХНОЛОГІЙ

КВАЛІФІКАЦІЙНА РОБОТА

на тему: «**Система керування онлайн-замовленнями та логістикою мовою Python**»

на здобуття освітнього ступеня бакалавра
зі спеціальності 126 Інформаційні системи та технології
(код,найменування спеціальності)
освітньо-професійної програми Інформаційні системи та технології
(назва)

Кваліфікаційна робота містить результати власних досліджень. Використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело

_____ Карина ГРИШКО
(підпис) *(ім'я, ПРІЗВИЩЕ здобувача)*

Виконав: здобувач вищої освіти група ІСД-42
Карина ГРИШКО
(ім'я, ПРІЗВИЩЕ)
Керівник Оксана ТКАЛЕНКО
к.т.н.,доцент *(ім'я, ПРІЗВИЩЕ)*
Рецензент: _____
к.т.н.,доцент *(ім'я, ПРІЗВИЩЕ)*

Київ 2026

ДЕРЖАВНИЙ УНІВЕРСИТЕТ
ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНИХ ТЕХНОЛОГІЙ

Навчально-науковий інститут інформаційних технологій

Кафедра Інформаційних систем та технологій
Ступінь вищої освіти бакалавр
Спеціальність 126 Інформаційні системи та технології
Освітньо-професійна програма Інформаційні системи та технології

ЗАТВЕРДЖУЮ
Завідувач кафедрою ІСТ
Каміла СТОРЧАК

“____” _____2026 року

З А В Д А Н Н Я
НА КВАЛІФІКАЦІЙНУ РОБОТУ

Гришко Карині Віталіївні
(прізвище, ім’я, по батькові здобувача)

1.Тема кваліфікаційної роботи: Система керування онлайн-замовленнями та логістикою мовою Python
керівник кваліфікаційної роботи: Оксана ТКАЛЕНКО к.т.н., доцент
(ім’я, ПРІЗВИЩЕ, науковий ступінь, вчене звання)
затверджені наказом Державного університету інформаційно-комунікаційних технологій від “20” лютого 2026 р. № 51

2. Строк подання кваліфікаційної роботи «1» червня 2026 р.

3. Вихідні дані кваліфікаційної роботи:
1. Принципи побудови веб-систем електронної комерції.
2. Технології розробки вебзастосунків мовою Python із використанням фреймворку Django.
3. Методи організації баз даних та логістики доставки у системах онлайн-замовлень.
4. Науково-технічна література та офіційна документація Python, Django, SQLite.

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити):
1. Аналіз сучасних вебсистем електронної комерції та онлайн-замовлень.

2. Дослідження технологій розробки вебзастосунків мовою Python та фреймворку Django.
3. Проєктування архітектури системи керування онлайн-замовленнями та логістикою доставки.
4. Розробка бази даних вебсистеми.
5. Реалізація функціоналу вебзастосунку та інтерфейсу користувача.
6. Реалізація модуля логістики доставки.
7. Тестування та аналіз роботи програмної системи.

5.Перелік ілюстраційного матеріалу: *презентація*

6. Дата видачі завдання «20» лютого 2026 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів кваліфікаційної роботи	Строк виконання етапів роботи	Примітка
1.	Підбір та аналіз технічної літератури з теми онлайн-замовлень, логістики та Python	20.02-05.03.26	Виконано
2.	Аналіз існуючих систем керування онлайн-замовленнями та логістичними процесами	04.03-19.03.25	Виконано
3.	Дослідження архітектури системи та вибір технологій (Python, Django, SQL, API)	20.03-1.04.25	Виконано
4.	Проєктування системи керування онлайн-замовленнями та логістикою	01.04-29.04.25	Виконано
5.	Розробка програмної реалізації системи (backend, бази даних, логіка замовлень)	30.03-02.05.25	Виконано
6.	Тестування та налагодження функціоналу системи	03.05-05.05.25	Виконано
7.	Оформлення бакалаврської роботи та підготовка демонстраційних матеріалів	05.05-30.05.25	Виконано

Здобувач вищої освіти _____ Карина ГРИШКО
(підпис) *(ім'я, ПРІЗВИЩЕ)*

Керівник кваліфікаційної роботи _____ Оксана ТКАЛЕНКО
(підпис) *(ім'я, ПРІЗВИЩЕ)*

РЕФЕРАТ

Текстова частина кваліфікаційної роботи на здобуття ступня бакалавр: 66 с., 57 рис., 9 табл., 20 джерел.

Метою дипломної роботи є удосконалення системи керування онлайн замовлення шляхом розробки вебсистеми, яка керує онлайн-замовленнями та логістикою доставки, вона забезпечує автоматизацію процесів електронної комерції, обробку замовлень, керування товарами та контроль доставки продукції.

Об'єктом дослідження є процес автоматизації онлайн-замовлень та логістики доставки у вебсередовищі.

Предметом дослідження є методи та засоби розробки вебсистем керування онлайн-замовленнями та логістикою доставки з використанням мови програмування Python і фреймворку Django.

У процесі виконання дипломної роботи було проведено аналіз сучасних вебсистем електронної комерції, досліджено особливості організації онлайн-замовлень та логістики доставки, визначено основні вимоги до програмної системи та обґрунтовано вибір технологій розробки.

У роботі спроєктовано архітектуру програмної системи та структуру бази даних, реалізовано модулі авторизації користувачів, каталогу товарів, кошика, оформлення замовлень та логістики доставки. Для реалізації серверної частини вебзастосунку використано мову програмування Python та фреймворк Django. Для збереження інформації використано систему керування базами даних SQLite.

У процесі розробки створено вебінтерфейс користувача із використанням HTML, CSS та шаблонів Django. Реалізований інтерфейс забезпечує зручну взаємодію користувача із системою, перегляд товарів, оформлення замовлень та контроль доставки.

КЛЮЧОВІ СЛОВА: PYTHON, DJANGO, ВЕБСИСТЕМА, ОНЛАЙН-ЗАМОВЛЕННЯ, ЕЛЕКТРОННА КОМЕРЦІЯ, ЛОГІСТИКА ДОСТАВКИ, БАЗА ДАНИХ, SQLITE, ВЕБЗАСТОСУНОК, ІНТЕРФЕЙС КОРИСТУВАЧА.

ЗМІСТ

ВСТУП

У сучасних умовах стрімкого розвитку інформаційних технологій та цифровізації бізнес-процесів особливого значення набувають системи електронної комерції, зокрема системи керування онлайн-замовленнями та логістикою доставки. Зростання популярності інтернет-магазинів, маркетплейсів і сервісів доставки обумовлює необхідність створення ефективних програмних рішень, які забезпечують автоматизацію обробки замовлень, контроль їх виконання та оптимізацію логістичних процесів.

Актуальність теми зумовлена тим, що сучасні підприємства стикаються з великою кількістю замовлень, які потребують швидкої обробки, точного обліку та своєчасної доставки. Традиційні підходи до організації цих процесів не забезпечують необхідного рівня швидкості, гнучкості та масштабованості. Водночас веборієнтовані інформаційні системи дозволяють централізовано керувати всіма етапами обслуговування замовлення – від його створення до доставки кінцевому споживачу. Це підвищує ефективність роботи підприємства, знижує кількість помилок і покращує якість обслуговування клієнтів.

Стан дослідження даної проблематики свідчить про значну увагу науковців і практиків до розробки систем електронної комерції та логістики. У працях дослідників розглядаються питання автоматизації бізнес-процесів, побудови вебсистем, використання сучасних фреймворків, а також оптимізації логістичних маршрутів. Проте існує потреба у створенні інтегрованих систем, які поєднують функціонал управління замовленнями, взаємодії з користувачем та контролю доставки в єдиному вебсередовищі з використанням сучасних інструментів програмування.

Метою дипломної роботи є удосконалення системи керування онлайн замовлення шляхом розробки вебсистеми, яка керує онлайн-замовленнями та логістикою доставки, вона забезпечує автоматизацію процесів електронної комерції, обробку замовлень, керування товарами та контроль доставки продукції.

Для досягнення поставленої мети у роботі було визначено такі основні завдання:

1. Дослідити теоретичні основи функціонування систем онлайн-замовлень та логістики доставки.
2. Провести аналіз сучасних технологій веброзробки для створення систем електронної комерції.
3. Спроєктувати структуру інформаційної системи керування онлайн-замовленнями та логістикою.
4. Виконати моделювання основних бізнес-процесів системи.
5. Реалізувати програмне рішення засобами мови Python та фреймворку Django.
6. Провести тестування системи та оцінити ефективність її роботи.

Об'єктом дослідження є процеси обробки онлайн-замовлень та організації доставки у вебсередовищі. Предметом дослідження виступають методи, моделі та програмні засоби побудови систем керування замовленнями та логістикою доставки.

У процесі виконання роботи використано методи теоретичного аналізу, моделювання інформаційних систем, об'єктно-орієнтованого програмування, а також методи проєктування баз даних і вебзастосунків. Для реалізації програмної частини застосовано мову програмування Python та фреймворк Django, що дозволяє створювати масштабовані та функціональні вебсистеми.

Наукова новизна отриманих результатів полягає у розробці інтегрованого підходу до створення вебсистеми керування онлайн-замовленнями та логістикою доставки, який передбачає об'єднання в межах єдиного вебзастосунку функцій керування товарами, обробки замовлень, обліку користувачів та контролю процесів доставки. На відміну від окремого використання незалежних модулів, запропонований підхід забезпечує централізовану взаємодію між компонентами системи через єдину базу даних та спільну серверну логіку фреймворку Django. У роботі запропоновано структуру вебсистеми, що базується на архітектурі MVT та забезпечує гнучкість, масштабованість і можливість подальшого розширення функціоналу. Реалізований підхід дозволяє автоматизувати основні процеси

електронної комерції, зменшити дублювання даних та підвищити ефективність керування онлайн-замовленнями і логістикою доставки.

Практична значущість роботи полягає у можливості використання розробленої системи для автоматизації діяльності інтернет-магазинів або сервісів доставки. Реалізоване програмне рішення може бути адаптоване під різні бізнес-потреби та інтегроване з іншими інформаційними системами.

Структура дипломної роботи включає вступ, три розділи, висновки, список використаних джерел та додатки. У першому розділі розглянуто теоретичні основи функціонування систем онлайн-замовлень і логістики. Другий розділ присвячено аналізу предметної області та проєктуванню системи. У третьому розділі описано процес розробки та реалізації вебсистеми, а також наведено результати її тестування.

Апробація результатів та публікації. Основні положення і результати бакалаврської роботи доповідались на науково практичних конференціях, що проходили на базі Державного університету інформаційно-комунікаційних технологій. VII міжнародна науково-технічна конференція «Сучасний стан та перспективи розвитку iot» з тезами на тему «Система керування онлайн-замовленнями та логістикою мовою python».

1 ТЕОРЕТИЧНІ ОСНОВИ СИСТЕМ ОНЛАЙН-ЗАМОВЛЕНЬ ТА ЛОГІСТИКИ

1.1 Загальна характеристика онлайн-замовлень та логістики

Сучасний етап розвитку інформаційного суспільства ознаменується активним упровадженням цифрових технологій у всі сфери діяльності, зокрема у сферу торгівлі та логістики. Одним із ключових напрямків трансформації бізнесу є використання систем інтернет-замовлень, які гарантують можливість віддаленої взаємодії між продавцем і покупцем. Такі системи дають змогу автоматизувати процес оформлення, обробки та виконання замовлень, що суттєво підвищує результативність діяльності підприємств. Інтернет-замовлення є процесом придбання товарів або послуг через вебінтерфейс чи мобільні застосунки, який включає вибір товару, формування кошика, введення даних користувача, підтвердження купівлі та здійснення сплати. Особливістю цього процесу є його інтеграція з інформаційними системами підприємства, такими як системи управління складом, бухгалтерський облік та логістичні модулі. Це забезпечує безперервність інформаційних потоків і дозволяє швидко реагувати на зміни попиту. [1]

Логістика постачання є невід’ємною складовою систем інтернет-замовлень та охоплює сукупність процесів, пов’язаних із транспортуванням товарів від постачальника до кінцевого споживача. Вона включає планування маршрутів, управління запасами, обробку замовлень, контроль виконання доставки та взаємодію з кур’єрськими службами. Дієва організація логістики є критично важливою для гарантування вчасного виконання замовлень і задоволення потреб клієнтів. Системи інтернет-замовлень та логістики функціонують як єдина інтегрована інформаційна система, в якій кожен етап обробки замовлення супроводжується відповідними інформаційними потоками. Від моменту створення замовлення до його доставки відбувається обмін даними між різними

компонентами системи, що дозволяє забезпечити прозорість процесів і нагляд на кожному етапі. Загальний опис процесу інтернет-замовлення та логістики постачання наведено в таблиці 1.1. [2]

Таблиця 1.1

Основні етапи процесу онлайн-замовлення та доставки

Етап процесу	**Характеристика**
Формування замовлення	Вибір товарів, додавання до кошика, введення даних користувача
Обробка замовлення	Перевірка наявності товару, підтвердження замовлення
Оплата	Виконання транзакції через платіжні системи
Комплектація	Підготовка товару до відправлення зі складу
Доставка	Транспортування товару до клієнта
Завершення замовлення	Отримання товару та підтвердження виконання

Однією з головних ознак сучасних систем онлайн-замовлень є їх спрямованість на клієнта. Це виявляється у наданні зручного інтерфейсу, можливості моніторингу статусу замовлення у реальному часі, а також у застосуванні індивідуальних пропозицій. Водночас для компанії суттєвими є аспекти гнучкості системи, її стабільності та здатності опрацьовувати великі обсяги відомостей. З технічного погляду системи онлайн-замовлень втілюються у вигляді вебдодатків, що оперують за клієнт-серверною будовою. Клієнтська частина відповідає за комунікацію з користувачем, тоді як серверна частина надає обробку основної логіки, збереження даних та з’єднання з іншими сервісами. Важливе значення у таких системах мають бази даних, які гарантують збереження відомостей про користувачів, продукцію, замовлення та їхні стани. Логістичний складник системи також зазнав суттєвих перетворень під впливом цифрових технологій. Застосування автоматизованих систем керування дає змогу

вдосконалити шляхи доставки, знизити видатки та наростити швидкість обслуговування. Окрім того, сучасні системи підтримують з’єднання із зовнішніми службами доставки, що розширює потенціал підприємств і дозволяє забезпечити прибуття товару у різні області. На рис. 1.1 відображена схема загальної структури системи онлайн-замовлень та логістики доставки для більшого розуміння системи. [3]

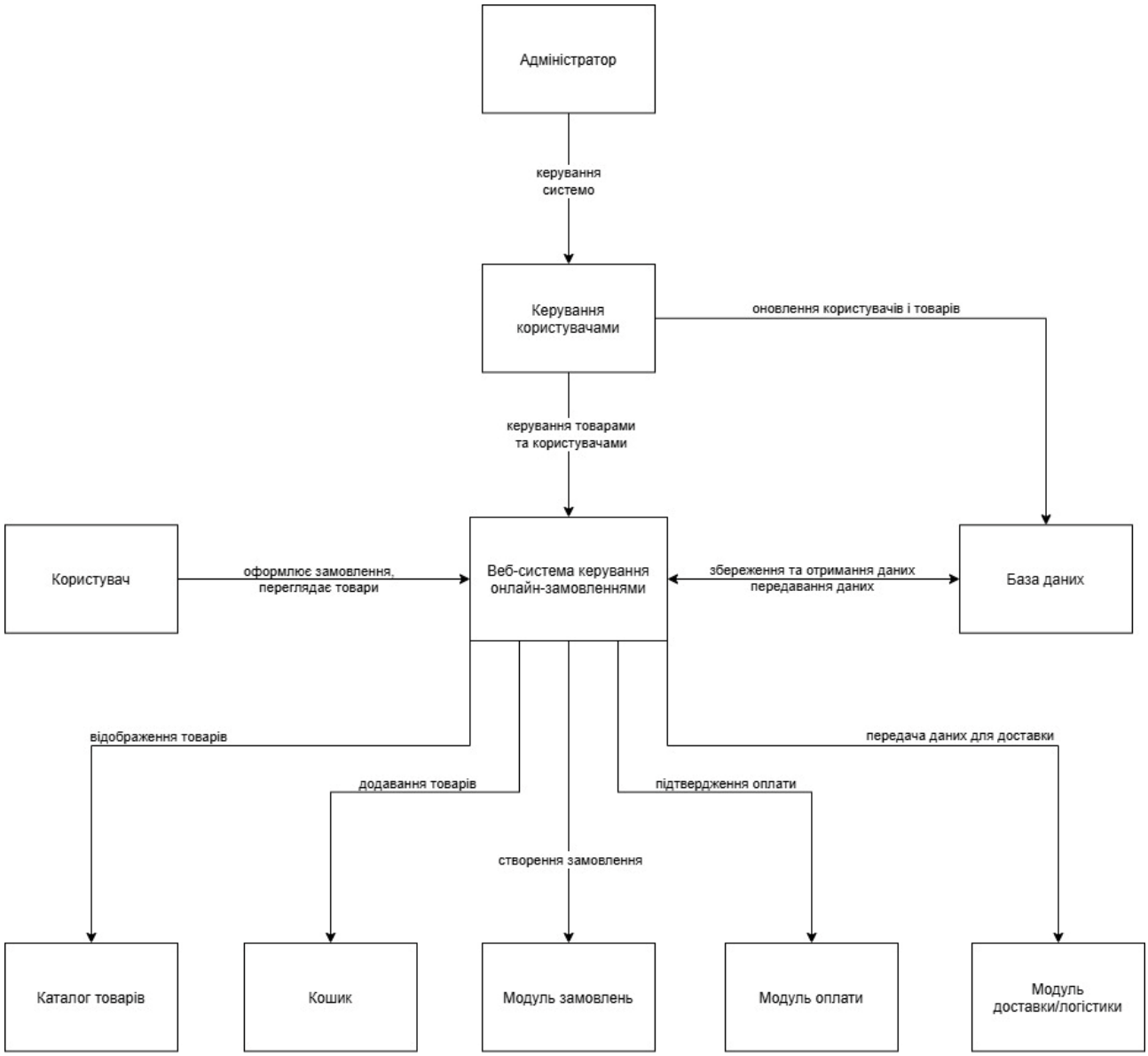


Рис. 1.1 Загальна структура системи онлайн-замовлень та логістики доставки

Системи онлайн-замовлень та логістики доставки є комплексними інформаційними системами, які об'єднують у собі функції керування замовленнями, опрацювання відомостей та організації доставки. Їхнє

впровадження є обов'язковою умовою для якісного функціонування сучасного бізнесу в умовах цифрової економіки.

1.2 Аналіз сучасних веб-систем електронної комерції

Електронні комерційні вебплатформи сьогодення вважаються ключовим складником цифровізованої економіки, адже вони уможливлюють автоматизацію реалізації товарів та послуг через інтернет-мережу. Стрімкий розвиток інформаційних технологій, зростання кількості мережевих користувачів та поширення покупок онлайн посприяли інтенсивному впровадженню електронної торгівлі у діяльність бізнесів будь-якого розміру. На цей час такі вебсистеми застосовуються як гігантами світового масштабу, так і невеликими місцевими крамницями. [1]

Першочерговим завданням цих систем є забезпечення комфортної взаємодії між сторонами продажу та купівлі, а також автоматизація обробки запитів на придбання, керування асортиментом, здійснення транзакцій та організації доставки. Завдяки експлуатації вебтехнологій бізнеси набувають здатності продавати свою продукцію безперервно, не маючи географічних обмежень, що суттєво розширює їхній ринок збуту. Типова сучасна система електронної комерції здебільшого містить низку головних структурних елементів: довідник продукції, механізм реєстрації клієнтів, віртуальний кошик, форму для оформлення замовлення, вбудований платіжний модуль, адміністративний інтерфейс та компонент для управління логістикою транспортування. Усі ці складові взаємодіють у межах спільної бази даних та серверного апарату системи.

Характерною рисою новітніх вебплатформ є впровадження багатошарової архітектури, яка дає змогу логічно розділити клієнтський інтерфейс, ядро бізнес-процесів та рівень зберігання інформації. Подібний підхід забезпечує потенціал для розширення, спрощує подальше обслуговування системи та підвищує її швидкодію. Окрім того, ці вебрішення активно інтегруються з зовнішніми сервісами, зокрема системами розрахунків, кур'єрськими службами, CRM-програмами та

платформами для аналітики. На актуальному ринку доступна велика кількість комерційних платформ, які різняться функціоналом, архітектурними рішеннями та технологіями розробки. До найвідоміших зразків відносять такі ресурси як Amazon, Shopify, Prom.ua та eBay. Ці системи втілюють широкий спектр можливостей, спрямованих на прискорення бізнес-операцій та покращення зручності для кінцевого споживача. Аби провести порівняння ключових ознак поточних інтернет-систем для торгівлі, зроблена таблиці 1.2. [2]

Таблиця 1.2

Порівняльна характеристика сучасних вебсистем електронної комерції

Назва системи	**Основне призначення**	**Особливості**	**Підтримка логістики**
Amazon	Глобальний маркетплейс	Велика кількість інтегрованих сервісів	Так
Shopify	Платформа для створення інтернет-магазинів	Гнучке налаштування магазину	Так
Prom.ua	Маркетплейс для малого та середнього бізнесу	Просте створення онлайн-магазину	Так
eBay	Система онлайн-аукціонів та продажів	Підтримка міжнародної торгівлі	Частково

Розгляд актуальних платформ електронної торгівлі свідчить, що більшість із них втілюють подібний набір базових можливостей, проте відчутні розбіжності полягають у ступені автоматизації, здатності до масштабування та застосованих технічних рішеннях. Наприклад, великі світові сервіси впроваджують складні архітектури з розподіленою обробкою даних та хмарні обчислення для підтримання стійкості під значними навантаженнями. На противагу цьому, менші рішення

сфокусовані на максимальній простоті впровадження та зручності користування. Ключовим компонентом будь-якої сучасної інтернет-системи є механізм керування обліковими записами користувачів. Цей блок відповідає за перевірку ідентичності (автентифікацію), надання дозволів (авторизацію), безпечне зберігання персональної інформації та визначення прав доступу залежно від ролі. Завдяки цьому стає можливим чітко розмежувати повноваження між особами, які керують системою (адміністраторами), тими, хто управляє асортиментом (менеджерами), та кінцевими покупцями.

Не менш істотною частиною є модуль, що займається обробкою транзакцій. Після того, як клієнт завершує оформлення покупки, система автоматично фіксує цей факт у базі даних, оновлює наявність товару, готує необхідну документацію для логістичних служб та інформує клієнта про поточний статус виконання його замовлення. У новітніх розробках також інтегровано функціонал для моніторингу руху товару в режимі реального часу. Особлива увага приділяється візуальній складовій та взаємодії з відвідувачем. Платформи електронної комерції мали б мати адаптивний дизайн, що гарантує коректне відображення як на стаціонарних комп'ютерах, так і на портативних гаджетах. Зрозуміла структура меню, швидкість пошуку потрібних позицій та інтуїтивно зрозумілий процес оформлення покупки – усе це безпосередньо впливає на результативність роботи платформи та рівень задоволеності тих, хто нею користується. [20]

Одним із головних векторів розвитку індустрії електронної торгівлі вважається інтеграція інструментів аналітики та механізмів персоналізації. Системи аналізують дії користувачів у мережі, їхню купівельну історію та індивідуальні вподобання, а на основі цього формують індивідуальні пропозиції щодо товарів. Це, своєю чергою, сприяє зростанню обсягів реалізації та покращенню якості комунікації з клієнтською базою. Окрім функціонального наповнення, вирішального значення набуде питання захищеності платформ електронної комерції. Оскільки ці системи оперують конфіденційними користувацькими даними та фінансовими реквізитами, критично важливою є імплементація передових методів захисту, включно із шифруванням даних,

використанням багатоетапної верифікації користувача та проактивним захистом від кібератак. [15]

Підсумовуючи, сучасні інтернет-платформи для комерційної діяльності являють собою складні, взаємопов'язані системи, які об'єднують процеси продажу продукції, управління виконанням замовлень, логістичні операції та комунікацію з аудиторією. Їхній розвиток націлений на підвищення рівня автоматизації бізнес-процесів, забезпечення гнучкості при зростанні навантажень та вдосконалення обслуговування споживачів.

1.3 Технології розробки веб-систем (Python, Django)

Створення сучасних веброзробок постає однією з ключових ланок розвитку IT-сектору. Вебзастосунки знайшли своє місце у сфері електронної торгівлі, фінансових установ, навчальних систем, систем управління підприємствами та у чисельних інших царинах. Для побудови подібних рішень залучають різноманітні мови програмування, каркаси й технології, що гарантують втілення функціоналу, обробку інформації та взаємодію кінцевих користувачів із системою через інтернет-мережу. Серед найбільш затребуваних мов для побудови вебрішень виділяється Python. Ця мова відзначається лаконічним синтаксисом, високою динамікою розробки та розширеними можливостями для злиття з різними програмними середовищами. Python активно задіються як для малих веброєктів, так і в масштабних корпоративних структурах. Завдяки наявності великого арсеналу бібліотек та фреймворків, Python дає змогу реалізовувати комплексні програмні рішення, мінімізуючи часові витрати. Важливою перевагою Python є його багатофункціональність. Мова підтримує об'єктно-орієнтований, процедурний та функціональний стилі програмування, що робить її придатною для вирішення різнопланових програмних завдань. Окрім веброзробки, Python знаходить застосування у галузі аналітики даних, штучного інтелекту, у автоматизації процесів та розробці мережевих утиліт. [10], [13]

Для спорудження вебсистем на базі Python повсюдно використовуються спеціалізовані програмні каркаси. Одним із найпопулярніших є Django – високорівневий, відкритий вебфреймворк, який забезпечує швидке та захищене створення вебзастосунків. Django суттєво спрощує розробку складних систем завдяки інтеграції готових модулів та вбудованих механізмів для роботи з базами даних, перевірки автентичності користувачів та адміністрування. Каркас Django ґрунтується на архітектурній концепції MVT (Model-View-Template), що є варіацією класичної архітектури MVC. Цей підхід забезпечує поділ програмної логіки на окремі частини, що полегшує технічну підтримку та масштабування системи. Модель відповідає за маніпуляції з даними та БД, Представлення (View) обробляє логіку отриманих запитів, а Шаблони формують візуальну частину для користувача. Основні складові архітектури Django представлені у Таблиці 1.3. [5], [11], [18]

Таблиця 1.3

Основні компоненти фреймворку Django

Компонент	**Призначення**
Model	Робота з базою даних та структура даних
View	Обробка логіки вебзапитів
Template	Формування інтерфейсу користувача
URL Dispatcher	Маршрутизація запитів
Admin Panel	Адміністрування системи
ORM	Взаємодія з базою даних без SQL-запитів

Однією з ключових особливостей Django є наявність ORM (Object Relational Mapping) – механізму, який дозволяє працювати з базою даних через об’єкти Python без необхідності написання складних SQL-запитів. Це значно спрощує процес розробки та зменшує ймовірність виникнення помилок при роботі з даними. Фреймворк також містить вбудовану адміністративну панель, яка автоматично створюється на основі моделей бази даних. Адміністративна частина дозволяє керувати користувачами, товарами, замовленнями та іншими елементами системи

через графічний інтерфейс без необхідності додаткового програмування. Важливою перевагою Django є високий рівень безпеки. Фреймворк має вбудовані механізми захисту від найбільш поширених типів вебатак, зокрема SQL-ін'єкцій, CSRF-атак, XSS-атак та підробки сесій користувачів. Це дозволяє створювати безпечні вебсистеми для роботи з персональними та фінансовими даними. [11]

Для реалізації клієнтської частини вебсистем зазвичай використовуються HTML, CSS та JavaScript. HTML забезпечує структуру сторінок, CSS відповідає за оформлення інтерфейсу, а JavaScript використовується для реалізації динамічної взаємодії з користувачем. У сучасних вебзастосунках ці технології працюють спільно із серверною частиною, реалізованою за допомогою Python та Django. У процесі роботи вебсистеми браузер користувача надсилає HTTP-запити до сервера. Серверна частина, реалізована на Django, обробляє ці запити, виконує необхідні операції з базою даних та формує відповідь у вигляді HTML-сторінки або JSON-даних. Такий підхід забезпечує взаємодію між користувачем та системою у режимі реального часу. Сучасні вебсистеми також активно використовують реляційні бази даних. У Django підтримується робота з PostgreSQL, MySQL, SQLite та іншими системами керування базами даних. Для невеликих проєктів часто використовується SQLite, тоді як для масштабних систем застосовується PostgreSQL завдяки її продуктивності та надійності. Впровадження Python разом із фреймворком Django є цілком виправданим рішенням при проєктуванні платформ для управління онлайн-замовленнями та системою доставки товарів. Це зумовлено тим, що дані інструменти сприяють прискореному циклу розробки, забезпечують високу масштабованість та полегшують подальше супроводження розробленого софту. Ба більше, вони відкривають можливості для втілення комплексної бізнес-логіки, а також безшовної інтеграції з різноманітними платіжними шлюзами, логістичними операторами та іншими сторонніми програмними комплексами. [3], [6-10], [16,], [17]

Отже, пара технологій Python і Django постає як потужний ресурс для створення актуальних веб-рішень у сфері електронної комерції. Їхнє застосування дає змогу формувати програмні продукти, які вирізняються високою стабільністю,

надійним рівнем захисту та розширеним функціоналом, що життєво необхідно для успішної автоматизації операцій із керування замовленнями та оптимізації логістичних потоків.

1.4 Архітектура систем керування замовленнями

Конструкція архітектури для систем, що оперують замовленнями, є фундаментальною для зведення актуальних мережевих платформ електронної комерції. Вона диктує програмну компоновку, правила взаємодії між окремими складниками системи, послідовність опрацювання відомостей та способи інформаційного обміну між кінцевим користувачем і серверною інфраструктурою. Грамотно спроектована структура гарантує стійкість функціонування платформи, її здатність до розширення, захищеність та потенціал для майбутньої еволюції. Призначення систем управління замовленнями полягає в механізації процедур оформлення, опрацювання та моніторингу виконання замовлень. Вони консолідують у собі функціональні блоки для взаємодії з клієнтами, каталогом товарів, віртуальним кошиком, платіжними операціями, логістикою та звітністю. Щоб забезпечити належну працездатність, всі складові повинні бути зведені в уніфіковану інформаційну матрицю.

У нинішніх веб-рішеннях перевага надається багатошаровій архітектурі, що передбачає розподіл системи на логічно відокремлені шари. Такий метод дозволяє полегшити обслуговування програмного коду, збільшити швидкодію та забезпечити автономність окремих модулів. Типова архітектура систем управління замовленнями охоплює три ключові рівні: рівень клієнта, рівень сервера та рівень сховища даних. Рівень клієнта відповідає за комунікацію з користувачем через його веб-інтерфейс. Серверний рівень здійснює виконання основної бізнес-логіки системи, тоді як база даних відповідає за персистентне зберігання та маніпуляції з даними. Ключові рівні архітектурного плану відображено у Таблиці 1.4. [3]

Таблиця 1.4

Основні рівні архітектури системи керування замовленнями

Рівень системи	**Призначення**
Клієнтський рівень	Взаємодія користувача з вебсистемою
Серверний рівень	Обробка логіки роботи системи
Рівень бази даних	Зберігання інформації про товари, користувачів та замовлення

Фронтенд системи розробляється із застосуванням HTML, CSS та JavaScript та відповідає за презентацію даних для кінцевого споживача. Через веб-інтерфейс замовник отримує можливість переглядати асортимент продуктів, здійснювати пошук, оформлювати купівлю та контролювати перебіг доставки. Інтерфейс зобов'язаний бути адаптивним, інтуїтивно зрозумілим і забезпечувати швидке переміщення між різними сторінками. Бекенд слугує ядром системи управління замовленнями. Саме він займається обробкою HTTP-запитів, валідацією введених даних, взаємодією з інформаційним сховищем та формуванням відповіді для користувача. У веб-додатках, створених на Python та Django, серверна логіка структурується згідно з архітектурною парадигмою MVT (Model-View-Template). Модель у цій системній структурі відповідає за визначення схеми даних та роботу з базою даних. Саме через моделі реалізується маніпуляція сутностями, такими як товари, клієнти, категорії, замовлення та інші елементи системи. Рендеринг (Представлення) забезпечує виконання бізнес-правил та обробку звернень від користувачів, тоді як шаблони займаються візуальним конструюванням веб-сторінок. [4], [6], [8], [9], [20]

Ключовим елементом архітектури є база даних, що слугує для централізованого зберігання всієї інформації. У системах управління замовленнями ця база містить відомості про продукти, зареєстрованих користувачів, адреси для доставки, актуальні статуси замовлень, фінансові транзакції та історію покупок.

Для забезпечення високої продуктивності системи база даних повинна гарантувати швидкий доступ до даних та підтримувати їхню цілісність. У сучасних реалізаціях активно застосовується API – програмний інтерфейс для взаємодії між різними модулями системи або зовнішніми сервісами. API уможливлює інтеграцію веб-системи з платіжними шлюзами, логістичними операторами, мобільними додатками та платформами третіх сторін. Використання API суттєво розширює функціональний потенціал системи та сприяє автоматизації багатьох рутинних процесів. Для ілюстрації взаємодії головних з'єднаних частин системи розглянемо сценарій оформлення замовлення. Після того, як користувач додає обраний товар у віртуальний кошик та підтверджує намір придбати його, клієнтський модуль надсилає відповідний запит на сервер. Сервер перевіряє коректність наданої інформації, створює новий запис у базі даних та оновлює статус замовлення. Далі система інформує логістичний модуль або службу доставки. Після успішного виконання доставки статус замовлення фіналізується та відображається клієнту. Сучасні архітектурні підходи в системах управління замовленнями також підтримують механізми масштабування. Це дає змогу підвищувати пропускну здатність системи зі зростанням кількості користувачів чи обсягу інформації. Для досягнення цієї мети можуть бути задіяні хмарні обчислення, технології балансування запитового навантаження та розподілені сховища даних. При проектуванні архітектури особлива увага приділяється аспектам безпеки. Оскільки система оперує персональною та фінансовою інформацією клієнтів, критично важливим є застосування надійних методів автентифікації, шифрування даних та захисту від несанкціонованого втручання. Архітектура повинна гарантувати безпечний обмін даними між клієнтом і сервером, а також надійний захист бази даних від потенційних загроз. Для порівняння основних компонентів архітектури системи використано таблицю 1.5. [15], [19]

Таблиця 1.5

Характеристика компонентів системи керування замовленнями

Компонент	**Основні функції**
Вебінтерфейс	Відображення інформації та взаємодія з користувачем
Серверна логіка	Обробка замовлень та бізнес-процесів
База даних	Збереження та обробка інформації
API	Інтеграція із зовнішніми сервісами
Система автентифікації	Захист облікових записів користувачів
Логістичний модуль	Контроль доставки та статусів замовлень

Таким чином, архітектура систем керування замовленнями являє собою складну багаторівневу структуру, яка забезпечує взаємодію між користувачами, серверною частиною та базою даних. Використання сучасних архітектурних підходів дозволяє створювати надійні, масштабовані та безпечні вебсистеми, здатні ефективно обробляти онлайн-замовлення та керувати логістикою доставки.

2 АНАЛІЗ ТА ПРОЄКТУВАННЯ СИСТЕМИ

2.1 Аналіз предметної області

Сфера застосування систем для керування онлайн-замовленнями та організації транспортування включає етапи електронної торгівлі, що стосуються реалізації продукції через Інтернет, обробки запитів на купівлю, проведення фінансових операцій та налагодження логістики постачання продукції кінцевому покупцеві. В умовах сучасного переходу бізнесу у цифровий простір, подібні системи виступають ключовим засобом автоматизації підприємницької діяльності, оскільки вони дають змогу гарантувати оперативне обслуговування клієнтів, раціональне керування інформаційними потоками та моніторинг кожного кроку реалізації замовлення. Ядром цієї предметної сфери є послідовність взаємодій між клієнтом, програмним комплексом та підрозділом, відповідальним за доставку. Клієнт, використовуючи веб-інтерфейс, має можливість переглянути асортимент товарів, скласти бажаний перелік покупок, узгодити метод транспортування та здійснити розрахунок. Інформаційна система відповідає за опрацювання введених даних, підтвердження наявності позицій на складі, збереження відомостей у сховищі даних та перепровадження потрібної інформації до логістичного блоку. Транспортна служба займається переміщенням товару та інформуванням про актуальний стан його доставки. Робота системи електронного замовлення охоплює низку взаємозалежних стадій. На початковому етапі відвідувач знаходить потрібний товар та вміщує його у віртуальний кошик. Після затвердження замовлення, система створює відповідний запис у базі даних та ініціює його обробку. Опісля, замовлення направляється до блоку, що відповідає за фінансові транзакції та логістику постачання. Щойно всі необхідні кроки будуть завершені, клієнт отримує свою продукцію та сповіщення про остаточне виконання замовлення. Для кращого розуміння основних процесів предметної області зображено таблицю 2.1 з основними учасники системи онлайн-замовлень. [1],

Таблиця 2.1

Основні учасники системи онлайн-замовлень

Учасник системи	**Основні функції**
Користувач	Перегляд товарів, оформлення замовлення, оплата
Адміністратор	Керування товарами, замовленнями та користувачами
Система	Обробка інформації та автоматизація процесів
Служба доставки	Доставка товарів та оновлення статусу замовлення
Платіжна система	Проведення електронних платежів

Огляд сфери діяльності засвідчує, що одне з наріжних завдань системи полягає у впровадженні автоматизації для опрацювання запитів на придбання. При застосуванні традиційних методів значна частка операцій виконується вручну, що неминуче тягне за собою затримки, виникнення помилок та зростання часових витрат. Впровадження інтернет-систем дає змогу суттєво спростити ці процедури завдяки єдиному центру керування даними та автоматизованому виконанню рутинних кроків. Невід’ємною складовою цієї сфери є організація доставки та перевезення. Це охоплює етапи планування, структурування та моніторингу руху товарів від продавця до споживача. У нинішніх реаліях клієнти очікують експрес-доставки та спроможності відслідковувати прогрес замовлення у режимі реального часу. Отже, системи електронної комерції зобов'язані підтримувати інтеграційні зв’язки зі службами перевезення і забезпечувати миттєве поновлення відомостей щодо статусу виконання кожного замовлення. Сфера також включає процеси, пов'язані з керуванням асортиментом та обліком складських ресурсів. Системі необхідно гарантувати актуальність відомостей про наявність товарів, їхню вартість, технічні характеристики та належність до певної групи. Автоматичне оновлення інформації усуває ймовірність оформлення замовлення на продукт, який фактично відсутній на складі. [2]

Однією з вирішальних характеристик сучасних рішень є наявність механізмів для ідентифікації та підтвердження особи користувачів. Це дає змогу забезпечити

захист персональних даних, збереження історії купівель та індивідуалізацію комунікації з клієнтом. Крім того, система повинна підтримувати розмежування прав доступу, зокрема для покупців та осіб, відповідальних за адміністрування. Для забезпечення належної працездатності системи критично важливим є налагодження взаємодії між абсолютно всіма її складовими. Інформація стосовно товарів, користувачів, замовлень та фінансових транзакцій має осязати єдине сховище даних, що гарантує повноту та узгодженість відомостей. Серверна частина системи відповідає за опрацювання основної логіки функціонування та забезпечує комунікацію між користувацьким інтерфейсом та базою даних. Під час детального вивчення предметної області було визначено ключові функціональні спроможності, які має втілити система управління електронними замовленнями та логістикою перевезень, вони зображені в таблиці 2.2. [19]

Таблиця 2.2

Основні функції системи

Функція	**Характеристика**
Реєстрація користувачів	Створення та керування обліковими записами
Каталог товарів	Перегляд та пошук товарів
Кошик	Формування списку товарів для покупки
Оформлення замовлення	Створення замовлення та вибір способу доставки
Оплата	Проведення електронних платежів
Відстеження доставки	Контроль статусу виконання замовлення
Адміністрування	Керування товарами та замовленнями

Розгляд існуючих вебсистем електронної комерції засвідчує, що найбільш дієвими є системи, які комбінують зрозумілий інтерфейс користувача, автоматизацію процесів та здатність інтегруватися із зовнішніми сервісами. Надзвичайно важливою є підтримка розширюваності системи, оскільки чисельність користувачів та обсяг відомостей можуть постійно збільшуватися. Під час творення системи керування онлайн-замовленнями треба брати до уваги

потреби щодо продуктивності, захисту та сталості. Система мусить гарантувати стрімку обробку запитів, охорону особистих даних користувачів та стабільну діяльність навіть при суттєвому навантаженні. У підсумку аналізу сфери встановлено, що розробка вебсистеми керування онлайн-замовленнями та логістикою пересилання є на часі завданням, спрямованим на автоматизацію процесів електронної комерції та зростання ефективності опрацювання замовлень. Застосування новітніх вебтехнологій та інтеграція логістичних механізмів дають змогу забезпечити швидке та належне обслуговування користувачів.

2.2 Формування вимог до системи

Створення вимог до системи є одним із найважливіших етапів розробки програмного забезпечення, адже саме на цьому етапі окреслюються ключові функціональні можливості, властивості та приписи майбутньої системи. Від коректності окреслення вимог залежить дієвість роботи програмного добутку, його відповідність потребам споживачів та потенціал подальшого розвитку системи. Система урядування інтернет-замовленнями та логістикою перевезення мусить гарантувати автоматизацію головних бізнес-процедур електронної комерції. До таких процедур входять керування продукцією, опрацювання замовлень, здійснення розрахунків, нагляд за доставкою та комунікація з клієнтами. Водночас система має забезпечувати стійку роботу у вебінфраструктурі, комфортний інтерфейс та охорону даних. Формування вимог проводиться на підставі аналізу сфери діяльності, вивчення наявних систем електронної комерції та з'ясування потреб користувачів. У ході проєктування були взяті до уваги особливості роботи онлайн-крамниць, потреба у злитті логістичних процесів та застосуванні актуальних вебразробок. Усі вимоги до системи можна умовно розподілити на функціональні та нефункціональні. Функціональні вимоги окреслюють перелік дій, які має здійснювати система, тоді як нефункціональні вимоги описують ознаки ґатунку, віддачі, захищеності та стабільності програмного забезпечення. [1], [2]

Серед головних завдань цієї системи виділяється робота з обліковими записами клієнтів. Система зобов'язана мати функціонал для реєстрації нових осіб, входу в систему та відновлення втраченого доступу. Для гарантування конфіденційності особистих відомостей слід впровадити методи шифрування паролів та процедури підтвердження особи користувача. Значущою складовою функціонування системи є блок управління асортиментом продукції. Він має забезпечувати надійне зберігання відомостей про товари, включно з їхніми категоріями, детальними описами, ціною та доступністю на складі. Клієнт повинен мати змогу оперативно знаходити потрібні позиції та ознайомлюватися з їхніми повними характеристиками. Система також повинна підтримувати функціонал, пов'язаний з віртуальним кошиком клієнта. Користувач має право додавати продукцію до кошика, коригувати обсяги або усувати позиції, що стали непотрібними. Щойно клієнт підтвердить намір придбати товар, система генерує замовлення та фіксує всі дані у сховищі даних. Невід'ємною частиною є модуль, що відповідає за організацію доставки товарів. Він має передбачати передачу даних про сформовані замовлення логістичній службі, актуалізацію стану виконання замовлення та інформування про це клієнта. Передбачена можливість вибору методу доставки та введення адреси для отримання придбаного товару. [19]

Окрім вимог, що стосуються безпосередньої функціональності, суттєве значення мають якісні (нефункціональні) параметри системи. Вони визначають рівень якості програмного продукту та його спроможність ефективно працювати під реальними експлуатаційними навантаженнями. Безпека системи становить собою ключову умову. З огляду на те, що робота системи охоплює обробку персональних відомостей користувачів та фінансової інформації щодо платежів, критично важливо гарантувати захист від будь-якого несанкціонованого доступу, впровадити захищені протоколи для комунікації даних та надійні методи підтвердження особи (автентифікації). [15]

Не менш вагомим чинником виступає швидкодія самої системи. Вебдодаток зобов'язаний оперативно реагувати на запити користувачів, навіть у ситуаціях високого рівня завантаження. Це вимагає впровадження оптимізованих методів

взаємодії з базою даних та побудови високоефективної серверної інфраструктури. Має бути закладена здатність системи до масштабування. Це дасть змогу в перспективі розширювати спектр функціональних можливостей та нарощувати абонентську базу без істотної деградації робочих характеристик. Паралельно слід забезпечити коректне функціонування системи у всіх актуальних вебпереглядах та на портативних (мобільних) ґаджетах. Під час фіксації вимог також було визначено категорії користувачів системи. Основними є такі суб'єкти, як клієнт (покупець) та уповноважений керуючий (адміністратор). Клієнт залучає систему для ознайомлення з асортиментом, оформлення купівельних актів та контролю логістичних процесів. Адміністратор же відповідає за адміністрування номенклатури, контролювання виконаних замовлень та управління обліковими записами користувачів. [10]

Отже, після етапу збору вимог було встановлено ключові функціональні та нефункціональні аспекти системи, призначеної для управління замовленнями в інтернеті та логістикою доставки. Ці специфіковані вимоги слугуватимуть фундаментом для подальших кроків: розробки архітектури системи, детального моделювання бізнес-процесів, а також безпосередньо програмування.

2.3 Моделювання системи

Створення моделі системи – це ключовий етап у розробці програмного забезпечення, адже він дає змогу формалізувати архітектуру майбутньої системи, встановити, як взаємодіятимуть її складові частини, а також задокументувати головні бізнес-операції. Завдяки моделюванню формується абстрактний образ системи, який слугує основою для подальшого кодування та аналізу роботи створеного програмного забезпечення. Коли ми моделюємо систему для управління мережевими замовленнями та маршрутизацією доставки, це допомагає окреслити ключові функціональні блоки, типи користувачів та послідовності обробки запитів. Головний вектор моделювання полягає у створенні чіткої архітектури, налагодженні взаємозв'язків між елементами системи та полегшенні майбутнього

втілення інтернет-додатку. Під час цієї роботи з моделювання були ідентифіковані головні суб’єкти, залучені до системи: клієнт, управитель (адміністратор), а також зовнішні системи, що відповідають за логістику та фінансові транзакції. Клієнт взаємодіє із системою через веб-інтерфейс, здійснюючи перегляд асортименту, оформлення замовлень та моніторинг статусу доставки. Управитель відповідає за адміністрування товарного каталогу, облікових записів та самих замовлень. Система ж забезпечує обробку даних, їх надійне зберігання та автоматизацію основних операцій електронної комерції. Визначення функціонального складу системи є однією з центральних ліній моделювання. З цією метою було виділено базові модулі, кожен з яких відповідає за реалізацію певної функції вебзастосунку.

Основу системи складає декілька взаємопов’язаних модулів, кожен із яких відповідає за окрему частину роботи вебзастосунку. Модуль користувачів забезпечує реєстрацію, авторизацію та керування обліковими записами користувачів. Завдяки цьому користувачі можуть створювати власні акаунти, входити до системи та змінювати персональні дані. Модуль каталогу призначений для зберігання та відображення інформації про товари. Саме через нього користувач отримує доступ до переліку продукції, описів, характеристик і зображень товарів. Для формування покупок використовується модуль кошика, який дозволяє додавати, видаляти та змінювати кількість обраних товарів перед оформленням замовлення. Модуль замовлень відповідає за створення, обробку та збереження інформації про замовлення користувачів. Після оформлення покупки дані передаються до модуля оплати, який забезпечує виконання електронних платежів та підтвердження успішної транзакції. Для організації доставки товарів використовується модуль доставки. Він дозволяє контролювати логістичні процеси, відстежувати статус замовлення та оновлювати інформацію про доставку. Окрему роль виконує адміністративний модуль, за допомогою якого адміністратор може керувати товарами, замовленнями, користувачами та загальним функціонуванням системи.

Опис ключових шляхів проходження інформації між частинами системи є невід'ємною складовою її моделювання. Коли система отримує запит через HTTP

від користувача, вона проводить обробку цих відомостей, здійснює звернення до сховища даних (бази даних) і готує кінцевий результат у формі вебсторінки. Для втілення такого механізму взаємодії застосовується архітектурна парадигма MVT, яку підтримує програмний фреймворк Django. Під час функціонування система здійснює обмін даними з базою для фіксації відомостей стосовно осіб, товарних позицій, оформлених замовлень та проведення платежів. Передача відомостей відбувається між незалежними складовими (модулями), що гарантує збереження повноти та відповідності наявних даних. Як приклад, після завершення процесу формування замовлення, відповідний модуль направляє дані до компонентів, що відповідають за фінансові операції та організацію доставки. У процесі розробки моделі були також ідентифіковані ключові складові бази даних цього програмного комплексу. Серед них фігурують користувачі, продукція, групи товарів, транзакції замовлень, вміст кошика та деталі доставки. Кожна така складова володіє своїм комплектом характеристик та взаємозв'язками з іншими компонентами системи. Перелік головних складових системи відображено у таблиці 2.3. [5], [18], [11]

Таблиця 2.3

Основні сутності системи

Сутність	**Основні атрибути**
Користувач	Ім’я, електронна пошта, пароль
Товар	Назва, опис, ціна, кількість
Категорія	Назва категорії
Кошик	Список товарів, кількість
Замовлення	Номер замовлення, дата, статус
Оплата	Спосіб оплати, сума, статус
Доставка	Адреса, спосіб доставки, статус

Ключовим етапом проєктування є встановлення кореляцій між елементами системи. Індивідуальний клієнт здатний ініціювати різноманітні замовлення, при цьому кожне таке замовлення може містити сукупність позицій. Окрім того,

замовлення інтегрується з процесами оплати та логістики. Подібна система зв'язків формує осмислену структуру сховища даних і уможливлює впровадження продуктивного механізму зберігання відомостей. Процес також сприяє визначенню хронології виконання головних бізнес-операцій. Після селекції позиції клієнт поміщає її у віртуальний кошик, після чого система генерує замовлення та фіксує його дані у сховищі. Слідом відбувається транзакція оплати, а дані про замовлення маршрутизуються до відповідального за логістику блоку. Після успішного виконання доставки клієнт отримує сповіщення про завершення операції.

Отже, було ідентифіковано архітектуру вебдодатку, його головні функціональні блоки, шляхи переміщення даних та кореляції між усіма елементами. Отримані матеріали слугують фундаментом для подальшого проєктування програмної архітектури та втілення системи для управління інтернет-замовленнями та процесами доставки.

2.4 Проєктування бази даних

Розробка бази даних є однією з центральних стадій створення вебзастосунку для управління онлайн-замовленнями та логістикою перевезення. Від коректності побудови бази даних залежить працездатність усього проєкту, швидкість опрацювання відомостей, надійність зберігання даних, а також можливість інтеграції нових функціональних можливостей у програмне забезпечення у майбутньому. Головна мета проєктування бази даних – формування логічної схеми для збереження відомостей, яка гарантує цілісність даних, усуває надлишковість інформації та підтримує взаємодію між різними частинами системи. На етапі розробки системи були визначені головні категорії об'єктів предметної сфери, які вимагають постійного зберігання у базі даних. До них належать зареєстровані користувачі, асортимент товарів, групи товарів, тимчасовий покупочний кошик, оформлені замовлення, платіжні операції та логістичні аспекти. Кожна з цих категорій має свій набір характеристик та взаємозв'язків, що забезпечують належне функціонування програми. База даних мусить забезпечити надійне збереження

відомостей про тих, хто зареєструвався у системі. Для кожного користувача зберігається особиста інформація, облікове ім'я, адреса електронної пошти, зашифрований пароль та його рольовий статус у системі. Розподіл ролей дозволяє встановити різний рівень доступу для звичайних покупців та адміністраторів системи. [16], [11]

Неодмінною складовою бази даних є таблиця, що містить інформацію про товари. Вона включає назву товару, його опис, вартість, кількість доступних одиниць та належність до певної категорії. Для зручності роботи з каталогом товарів передбачено окремий розділ категорій, що дає змогу групувати товари за певними критеріями та впроваджувати механізми пошуку та фільтрації. Після вибору необхідних позицій користувач додає їх до свого кошика. Для реалізації цієї функції виділено спеціальну сутність кошика, яка тимчасово зберігає відомості про обрані товари до моменту, коли покупець оформлює замовлення. Після підтвердження купівлі система створює запис про замовлення, прив'язуючи його до відповідного користувача. Таблиця замовлень є однією з ключових у структурі бази даних. Вона містить дані про унікальний номер замовлення, дату його створення, поточний статус виконання, загальну вартість та інформацію про замовника. Оскільки одне замовлення може містити багато різних товарів, між таблицями замовлень і товарів встановлюється зв'язок типу «множина до множини». Окремо в системі ведеться облік платіжних операцій та логістики. Таблиця оплати фіксує спосіб, яким було здійснено платіж, суму транзакції та її поточний статус. Таблиця доставки зберігає дані про адресу одержувача, обраний метод транспортування товару та актуальний стан перевезення. Під час розробки схеми бази даних важливим кроком є визначення взаємозв'язків між усіма сутностями. Користувач може розмістити будь-яку кількість замовлень, проте кожне замовлення асоціюється лише з одним користувачем. Товари можуть фігурувати в багатьох замовленнях, і так само, у кожному замовленні може бути велика кількість товарів. Подібні взаємозв'язки забезпечують логічну узгодженість даних та підтримують їхню цілісність. Для підвищення швидкодії системи необхідно, щоб база даних була належним чином нормалізована. Нормалізація допомагає скоротити дублювання

відомостей та сприяє створенню оптимальної структури зберігання інформації. У цій системі структура бази даних відповідає основним постулатам нормалізації, що гарантує ефективну роботу навіть з великими масивами даних. При проєктуванні бази даних також бралася до уваги необхідність забезпечення високої продуктивності системи. Для прискорення пошуку товарів, користувачів та історій замовлень є доцільним застосування індексації на тих полях, які найчастіше використовуються у пошукових запитах. Це скорочує час, необхідний для опрацювання інформації, та підвищує загальну швидкість роботи вебзастосунку.

Для технічної реалізації бази даних у цьому проєкті оптимальним є використання реляційної системи керування базами даних. Наразі застосовується SQLite, яка повністю підтримується фреймворком Django та забезпечує легку інтеграцію з вебсервісом. Використання механизму ORM у Django дозволяє оперувати з базою даних через моделі на Python, мінімізуючи потребу ручного написання SQL-запитів. Для кращої візуалізації структури бази даних побудована ER-діаграма, яка чітко відобразить усі ключові сутності системи та зв'язки, що існують між ними. На рис. 2.1 зображено ER-діаграма бази даних системи. [12], [17]

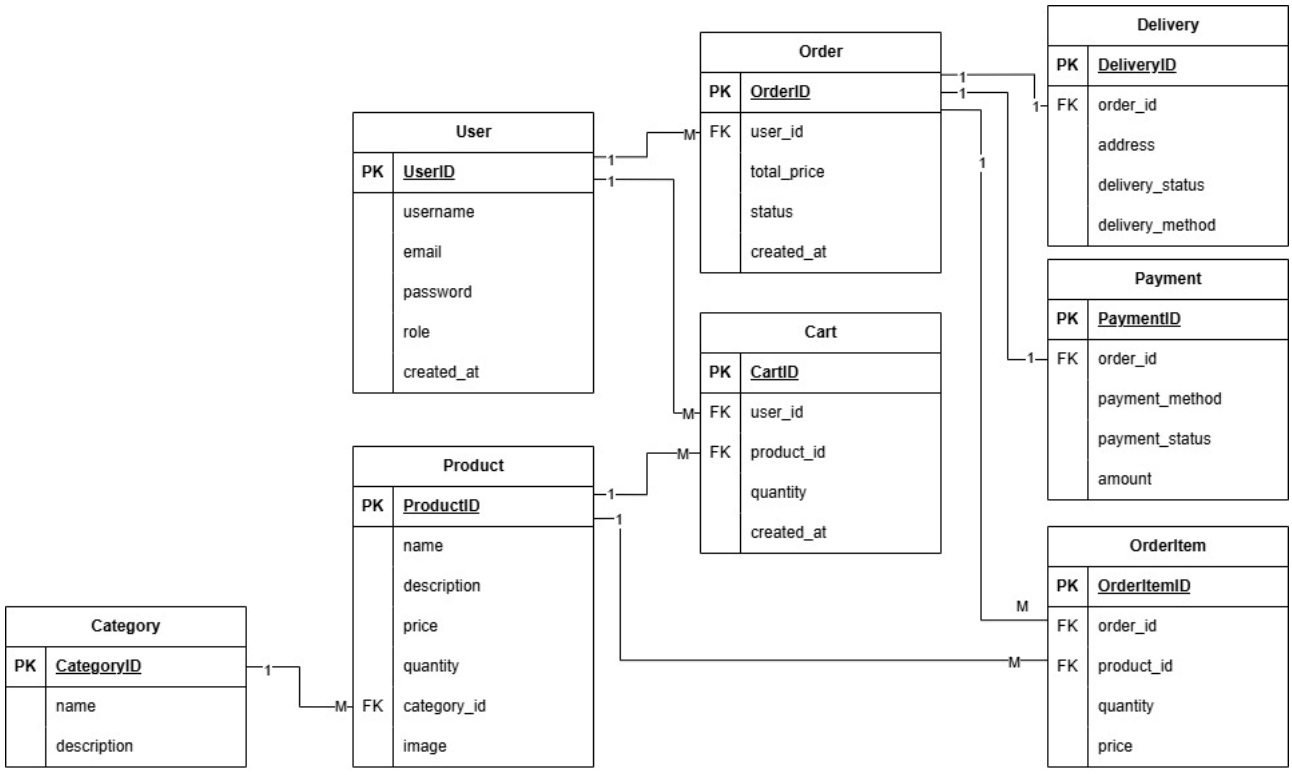


Рис. 2.1 ER-діаграма бази даних системи

Основною сутністю бази даних є таблиця ‘User’, яка призначена для збереження інформації про користувачів системи. У таблиці зберігаються логін користувача, адреса електронної пошти, пароль, роль у системі та дата створення облікового запису. Кожен користувач має унікальний ідентифікатор ‘id’, який використовується як первинний ключ таблиці. Для збереження інформації про категорії товарів використовується таблиця ‘Category’. Вона містить назву категорії та її короткий опис. Одна категорія може включати декілька товарів, тому між таблицями ‘Category’ і ‘Product’ реалізований зв’язок типу «один до багатьох». Таблиця ‘Product’ призначена для збереження інформації про товари, доступні у системі. У ній містяться назва товару, опис, ціна, кількість доступних одиниць, зображення товару та зовнішній ключ ‘category_id’, який забезпечує зв’язок із таблицею категорій. Кожен товар має власний первинний ключ ‘id’. Для реалізації функціональності кошика використовується таблиця ‘Cart’. Вона забезпечує тимчасове збереження товарів, які користувач додає перед оформленням замовлення. Таблиця містить зовнішні ключі ‘user_id’ та ‘product_id’, що забезпечують зв’язок із користувачем і товаром відповідно. Однією з центральних таблиць системи є таблиця ‘Order’, яка містить інформацію про оформлені замовлення. У таблиці зберігаються дата створення замовлення, загальна сума покупки, статус виконання та зовнішній ключ ‘user_id’, який визначає користувача, що створив замовлення. Один користувач може створити декілька замовлень, тому між таблицями ‘User’ та ‘Order’ реалізовано зв’язок типу «один до багатьох». Оскільки одне замовлення може містити декілька товарів, а один товар може входити до складу багатьох замовлень, у системі використовується проміжна таблиця ‘OrderItem’. Вона реалізує зв’язок типу «багато до багатьох» між таблицями ‘Order’ і ‘Product’. Таблиця містить зовнішні ключі ‘order_id’ та ‘product_id’, а також інформацію про кількість товарів і їх вартість у межах конкретного замовлення. Для збереження інформації про оплату використовується таблиця ‘Payment’. Вона містить спосіб оплати, суму платежу, статус транзакції та зовнішній ключ ‘order_id’, який забезпечує зв’язок із таблицею замовлень.

Кожному замовленню відповідає один запис про оплату. Таблиця ‘Delivery’ призначена для збереження інформації про доставку товару. У ній містяться адреса доставки, спосіб транспортування, статус доставки та зовнішній ключ ‘order_id’. Таблиця забезпечує контроль логістичних процесів та відстеження стану виконання доставки.

В результаті проєктування бази даних було сформовано структуру збереження інформації, яка забезпечує ефективну взаємодію між компонентами системи керування онлайн-замовленнями та логістикою доставки. Спроєктована база даних підтримує основні бізнес-процеси електронної комерції, забезпечує цілісність інформації та створює основу для подальшої реалізації вебзастосунку засобами Python та Django.

3 РОЗРОБКА ВЕБСИСТЕМИ

3.1 Архітектура програмної системи

Розроблена система керування онлайн-замовленнями та логістикою доставки реалізована у вигляді багатомодульного вебзастосунку з використанням мови програмування Python та фреймворку Django. Під час створення програмної системи було використано архітектурний шаблон MVT (Model-View-Template), який є основою роботи фреймворку Django та забезпечує розділення логіки роботи системи, інтерфейсу користувача та механізму взаємодії з базою даних. Архітектура системи побудована таким чином, щоб забезпечити:

- модульність;
- масштабованість;
- простоту підтримки;
- зручність розширення функціоналу;
- стабільну взаємодію між компонентами.

У структурі проєкту кожен функціональний модуль виконує окреме завдання та взаємодіє з іншими компонентами через Django ORM та систему маршрутів. До складу системи входить декілька основних модулів, кожен із яких виконує окрему функцію. Модуль users відповідає за роботу користувачів, реєстрацію, авторизацію та керування профілем. Модуль products реалізує каталог товарів, систему категорій та взаємодію з продукцією. Для роботи кошика використовується модуль cart, який забезпечує тимчасове збереження товарів перед оформленням замовлення. Модуль orders відповідає за створення та обробку замовлень користувачів, а delivery реалізує контроль логістики доставки. Також у системі присутні модулі reviews та favorites, які забезпечують роботу відгуків і списку обраних товарів. Адміністрування вебсистеми здійснюється через Django Admin Panel. Клієнтська частина вебзастосунку реалізована за допомогою HTML, CSS, Bootstrap та шаблонів Django. Вона забезпечує взаємодію користувача із системою через

веббраузер. Серверна частина реалізована мовою Python та виконує обробку HTTP-запитів, реалізацію бізнес-логіки, взаємодію з базою даних і перевірку введених користувачем даних. Після обробки інформації система формує HTML-сторінки та повертає їх користувачу через вебінтерфейс.

Для збереження інформації використовується база даних SQLite. Робота з базою даних реалізована через Django ORM, що дозволяє працювати з таблицями як з Python-об’єктами. Після надсилання запиту користувачем серверна частина обробляє отримані дані, звертається до відповідних моделей та формує HTML-сторінку або відповідь системи. Одним із головних модулів системи є модуль користувачів. У ньому реалізовано:

- реєстрацію;
- авторизацію;
- редагування профілю;
- збереження адрес доставки.

Для зберігання адрес використовується окрема модель UserAddress. У ній зберігається інформація про місто, вулицю, номер будинку, квартиру та контактний номер телефону користувача. Це дозволяє використовувати декілька адрес доставки при оформленні замовлень. Завдяки цьому користувач може використовувати декілька адрес доставки при оформленні замовлень. Модуль товарів реалізує основний функціонал електронної комерції. У системі створено структуру категорій та підкатегорій товарів. Для кожного товару у базі даних зберігається назва, опис, ціна, акційна ціна, відсоток знижки, кількість продукції на складі, характеристики товару, інформація про гарантію та доставку, а також фотографії продукції. У модулі реалізовано автоматичне обчислення рейтингу товару на основі відгуків користувачів. Для підвищення зручності користування створено систему обраних товарів. Користувач може додавати продукцію до списку улюблених для швидкого доступу. Модуль кошика забезпечує тимчасове збереження вибраних товарів перед оформленням замовлення. Користувач може додавати товари, змінювати їхню кількість, видаляти непотрібні позиції та переглядати автоматично розраховану вартість покупки. Після підтвердження покупки система створює

замовлення та переносить інформацію з кошика у модуль orders. Модуль замовлень відповідає за створення замовлення, збереження інформації про товари та обробку даних користувача. Після підтвердження покупки система автоматично створює запис у базі даних та формує інформацію про доставку. У системі реалізовано декілька статусів замовлень:

- нове;
- підтверджене;
- оплачене;
- відправлене;
- доставлене;
- скасоване.

Окрему роль виконує модуль доставки. Він відповідає за логістику та контроль виконання доставки товарів. Для кожної доставки система зберігає адресу користувача, спосіб доставки, поточний статус та дату створення замовлення. Це дозволяє контролювати процес логістики та відображати актуальну інформацію у системі. Модуль доставки інтегрований із системою замовлень та дозволяє відстежувати процес виконання покупки. Для адміністрування системи використовується Django Admin Panel. Через адміністративний інтерфейс адміністратор може додавати та редагувати товари, керувати категоріями, змінювати статуси замовлень і доставок, переглядати користувачів та контролювати загальну роботу системи. У проєкті також реалізовано систему статистики, яка дозволяє переглядати кількість користувачів, замовлень, доставок та загальний дохід вебсистеми. Архітектура вебсистеми дозволяє легко розширювати функціонал проєкту. За потреби можуть бути додані нові модулі, наприклад:

- онлайн-оплата;
- інтеграція з поштовими сервісами;
- система повідомлень;
- API для мобільного застосунку.

Для кращого розуміння зроблені. На рис. 3.1 зображений скріншот структури самого проєкту Django.

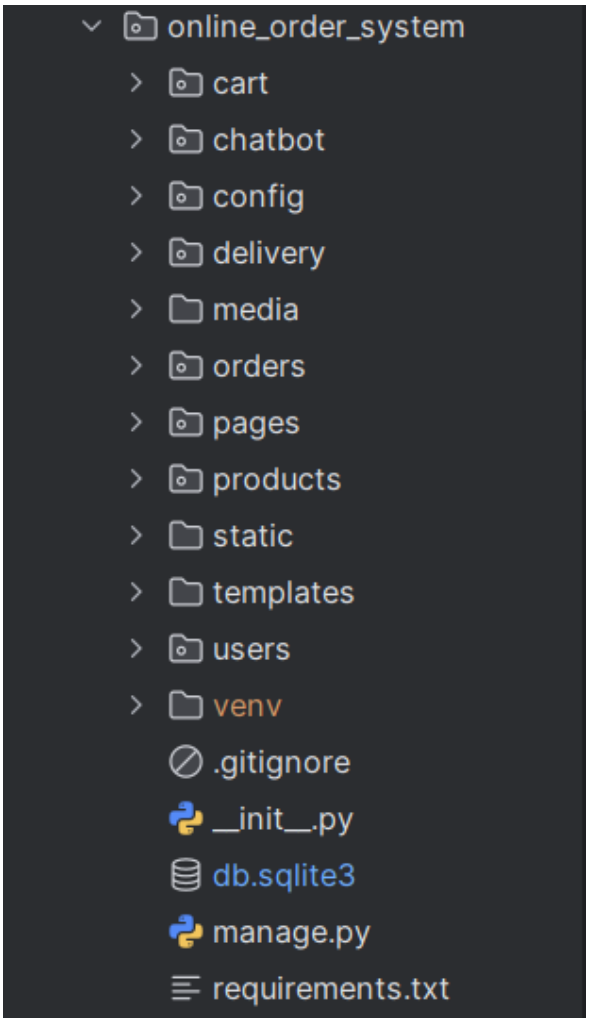


Рис. 3.1 Скріншот структура проєкту Django у середовищі розробки

На рис. 3.2., рис. 3.3, зображено скріншоти структури модулів users і products в середовищі проєкту.

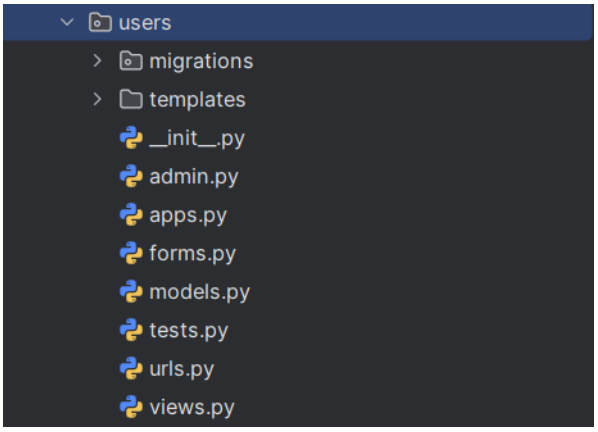


Рис. 3.2 Скріншот структури модулів users

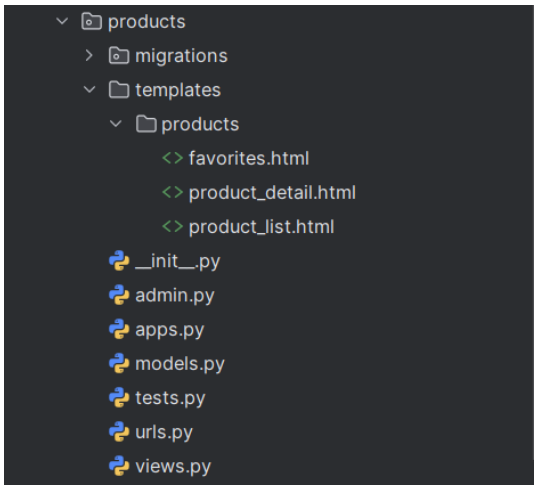


Рис. 3.3 Скріншот структури модулів products

На рис. 3.4, рис. 3.5 зображені скріншоти структури модулів orders і delivery у середовищі даного проєкту.

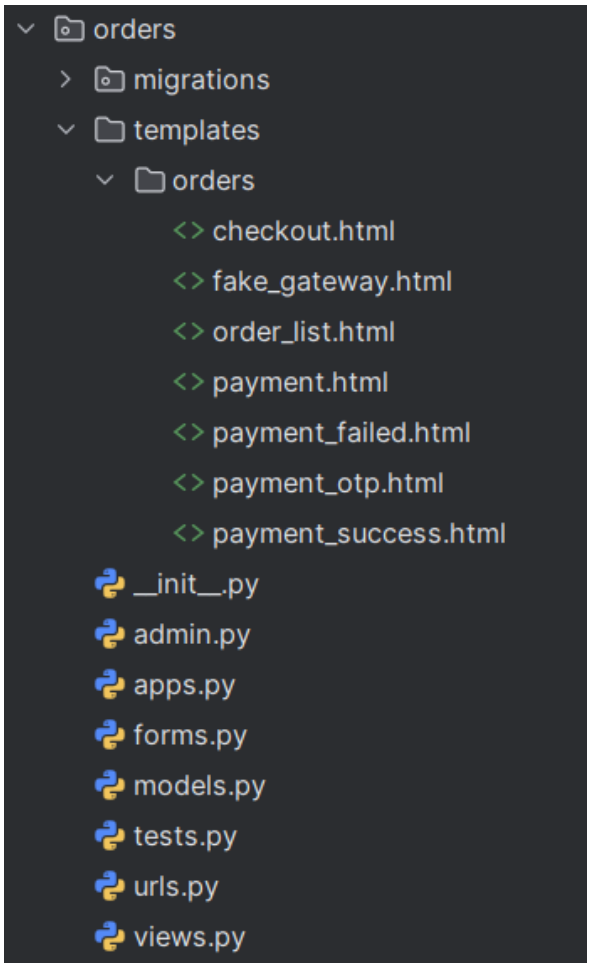


Рис. 3.4 Скріншот структури модулів orders

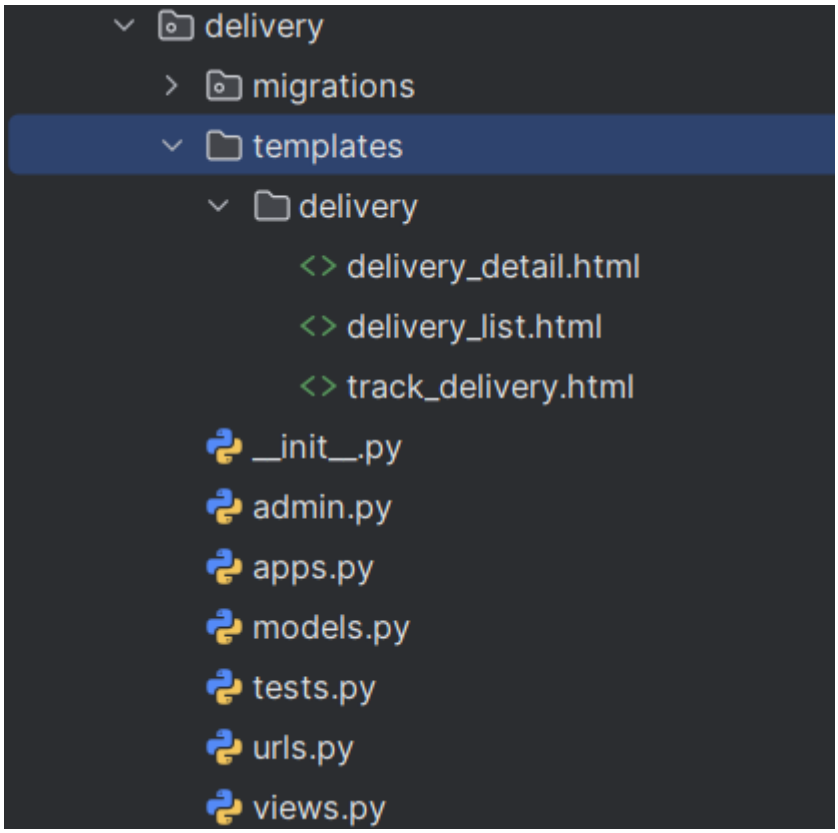


Рис. 3.5 Скріншот структури модулів delivery

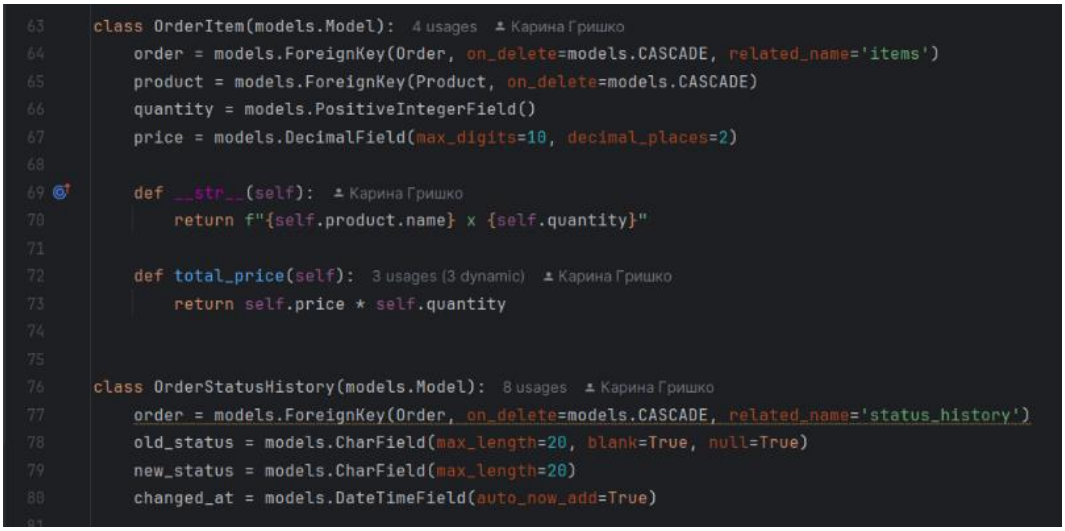
На рис. 3.6 зображено скріншот фрагмента коду моделі Product, а саме модель каталогу, яка зберігає всю інформацію про конкретну позицію товару.

```
39  class Product(models.Model):  10 usages  Карина Гришко
40      category = models.ForeignKey(Category, on_delete=models.CASCADE)
41      name = models.CharField(max_length=200)
42      description = models.TextField()
43      specifications = models.TextField(blank=True, null=True)
44      warranty = models.CharField(max_length=255, blank=True, null=True)
45      return_policy = models.TextField(blank=True, null=True)
46      delivery_info = models.TextField(blank=True, null=True)
47      price = models.DecimalField(max_digits=10, decimal_places=2)
48      old_price = models.DecimalField(max_digits=10, decimal_places=2, blank=True, null=True)
49      discount_percent = models.IntegerField(default=0)
50      is_promo = models.BooleanField(default=False, verbose_name="promo")
51      is_new = models.BooleanField(default=False)
52      stock = models.PositiveIntegerField(default=0)
53      image = models.ImageField(upload_to='products/', blank=True, null=True)
54
```

Рис. 3.6 Скріншот фрагмента коду моделі Product

На рис. 3.7, зображено скріншот фрагмента коду моделі Order. А саме OrderItem, вона пов'язує замовлення з конкретним товаром і зберігає його кількість

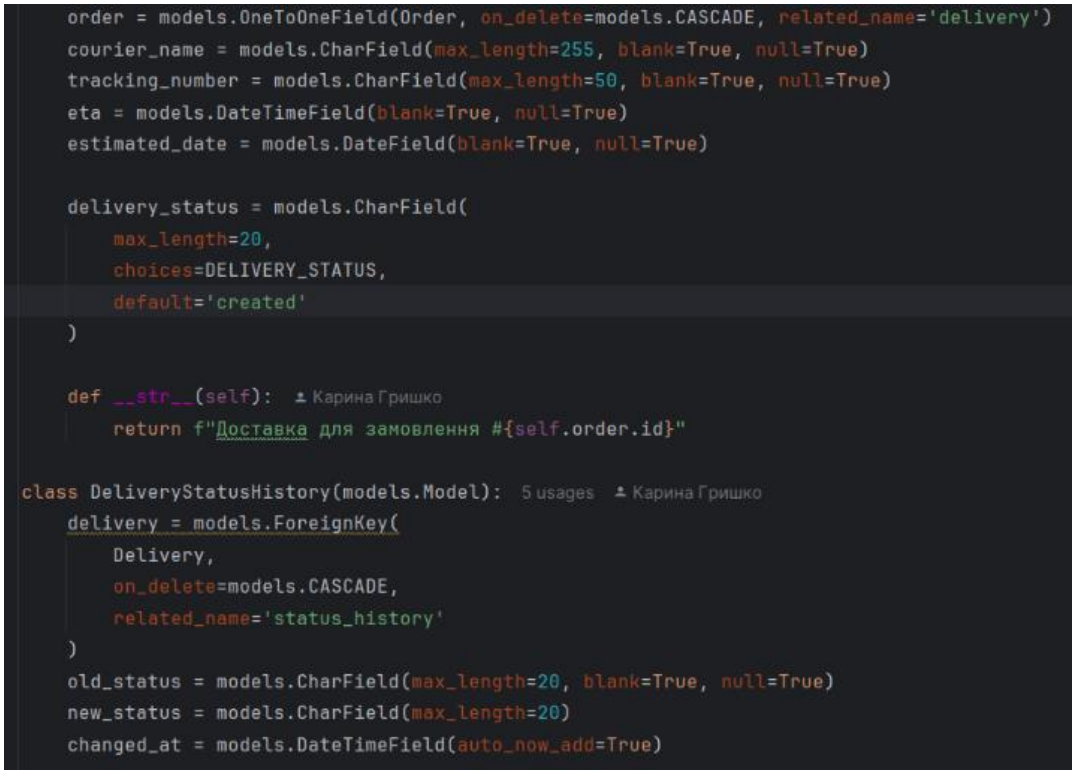
та фіксовану ціну на момент покупки та OrderStatusHistory, вона дозволяє відстежувати весь шлях замовлення від створення до видачі.



```
class OrderItem(models.Model):  4 usages  Карина Гришко
    order = models.ForeignKey(Order, on_delete=models.CASCADE, related_name='items')
    product = models.ForeignKey(Product, on_delete=models.CASCADE)
    quantity = models.PositiveIntegerField()
    price = models.DecimalField(max_digits=10, decimal_places=2)

    def __str__(self):  Карина Гришко
        return f"{self.product.name} x {self.quantity}"

    def total_price(self):  3 usages (3 dynamic)  Карина Гришко
        return self.price * self.quantity


class OrderStatusHistory(models.Model):  8 usages  Карина Гришко
    order = models.ForeignKey(Order, on_delete=models.CASCADE, related_name='status_history')
    old_status = models.CharField(max_length=20, blank=True, null=True)
    new_status = models.CharField(max_length=20)
    changed_at = models.DateTimeField(auto_now_add=True)
```

Рис. 3.7 Скріншот фрагмента коду моделі Order

На рис. 3.8 зображено скріншот фрагмента коду моделі Delivery, а саме клас Delivery і DeliveryStatusHistory. Delivery потрібна для зберігає всю інформацію про процес доставки конкретного замовлення., а DeliveryStatusHistory потрібна для логування та аудиту змін у процесі доставки.



```
    order = models.OneToOneField(Order, on_delete=models.CASCADE, related_name='delivery')
    courier_name = models.CharField(max_length=255, blank=True, null=True)
    tracking_number = models.CharField(max_length=50, blank=True, null=True)
    eta = models.DateTimeField(blank=True, null=True)
    estimated_date = models.DateField(blank=True, null=True)

    delivery_status = models.CharField(
        max_length=20,
        choices=DELIVERY_STATUS,
        default='created'
    )

    def __str__(self):  Карина Гришко
        return f"Доставка для замовлення #{self.order.id}"

class DeliveryStatusHistory(models.Model):  5 usages  Карина Гришко
    delivery = models.ForeignKey(
        Delivery,
        on_delete=models.CASCADE,
        related_name='status_history'
    )
    old_status = models.CharField(max_length=20, blank=True, null=True)
    new_status = models.CharField(max_length=20)
    changed_at = models.DateTimeField(auto_now_add=True)
```

Рис. 3.8 Скріншот фрагмента коду моделі Delivery

3.2 Блок-схема роботи вебсистеми

Для кращого розуміння принципу функціонування розробленої вебсистеми було створено блок-схему роботи системи, яка демонструє послідовність взаємодії між користувачем, серверною частиною, базою даних та адміністративним модулем На 3.9 зображено початок роботи з керуванням вебсистеми онлайн-замовленнями та логістикою доставлення. Клієнт реєструється, або заходить в акаунт система перевіряє чи успішна реєстрація, або вхід. Якщо ні, то виводить повідомлення про помилку відправляє на блок реєстрації чи входу. Якщо успішно, то клієнту переходить на каталог товару і він переїдає товари, потім він вибирає товар і якщо товар відсутній, то його не можна додавати в кошик, а якщо є, то товар додається в кошик. Після цього, якщо користувач хоче додати ще, то іде цикл, він повертається до каталогу і додає товари, які йому треба. Коли він додав товари клієнт переглядає кошик. Після перегляду кошика, клієнт і оформляє замовлення. Він водить дані доставки та оплати, а система перевіряє чи коректні дані, якщо ні, то виводяться помилки, а якщо коректно то дані зберігаються в БД. Після збереження в БД, замовлення передається адміністратору, він переглядає і вирішує підтверджувати замовлення чи ні. Якщо ні, то замовлення відхиляється, повідомляється клієнту і робота завершується. Якщо підтверджує, то формується доставка, призначається статус залежно від етапу. Товар забирає клієнт самостійно з магазину чи доставляється. Оновлюється статус доставки доставлено. Можливо після цього залишається відгук й завершується робота.

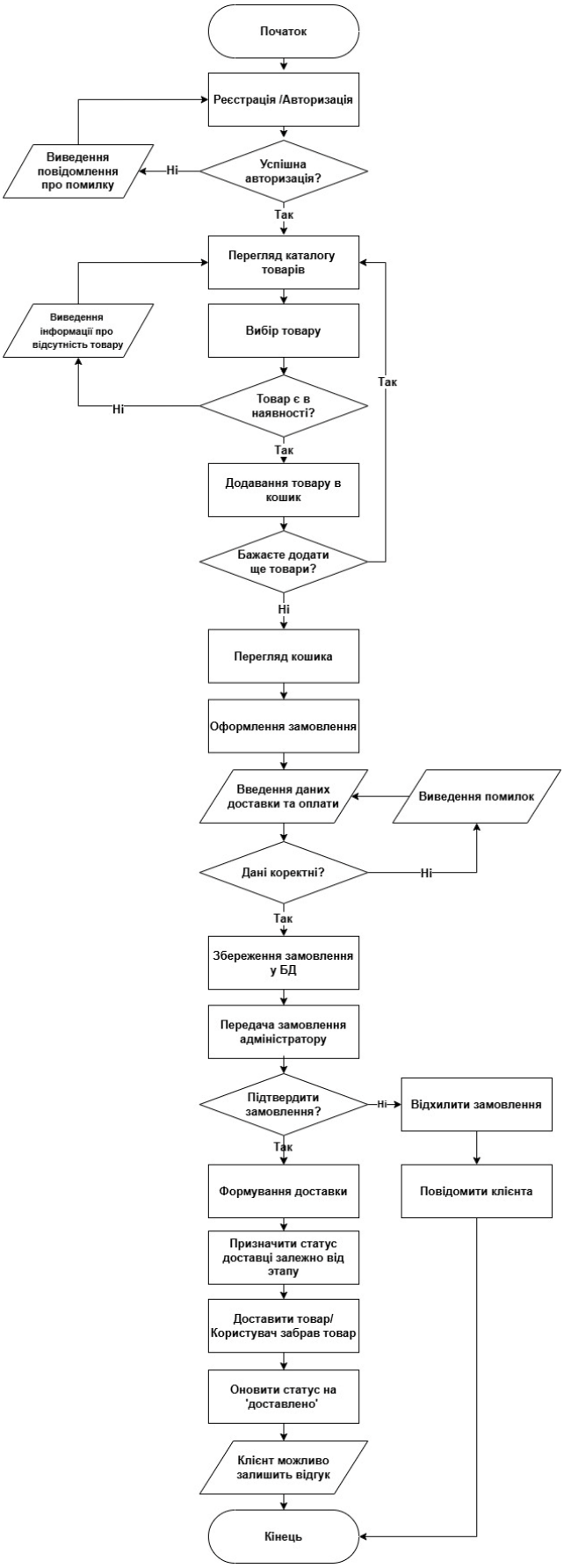


Рис. 3.9 блок-схема роботи вебсистеми керування онлайн-замовленнями та логістикою доставки

Блок-схема демонструє логіку взаємодії основних компонентів вебсистеми керування онлайн-замовленнями та логістикою доставки. У схемі відображено взаємозв’язок між користувацьким модулем, системою обробки замовлень, базою даних та адміністративною частиною вебзастосунку. Такий підхід дозволяє наочно представити послідовність виконання операцій, перевірку коректності введених даних, процес формування замовлення та подальше керування доставкою товару. Особливістю розробленої системи є інтеграція функцій керування товарами, оформлення замовлень та логістики доставки в межах єдиного вебзастосунку. Усі модулі взаємодіють через спільну базу даних і серверну логіку, реалізовану засобами фреймворку Django. Це забезпечує централізоване зберігання даних, автоматизацію процесів обробки замовлень та спрощує адміністрування системи. Розроблена структура дозволяє підвищити ефективність роботи вебзастосунку, забезпечити контроль етапів доставки та покращити взаємодію між клієнтом і адміністратором системи.

Розроблена структура системи забезпечує гнучкість та можливість подальшого масштабування. За необхідності функціонал вебсистеми може бути розширений шляхом додавання нових модулів, зокрема інтеграції платіжних сервісів, служб доставки або систем аналітики. Крім цього, використання архітектури MVT дозволяє розділити логіку обробки даних, інтерфейс користувача та механізми взаємодії з базою даних, що спрощує підтримку та модернізацію системи.

Таким чином, блок-схема роботи вебсистеми демонструє повний цикл функціонування програмного рішення та підтверджує реалізацію інтегрованого підходу до керування онлайн-замовленнями та логістикою доставки в межах єдиної інформаційної системи.

3.3 Опис процесу розробки вебсистеми

Процес розробки вебсистеми керування онлайн-замовленнями та логістикою доставки виконувався поетапно та включав аналіз предметної області, проєктування структури системи, створення бази даних, реалізацію функціональних модулів Основною метою розробки було створення єдиної інформаційної системи, яка забезпечує автоматизацію процесів оформлення замовлень, керування товарами та контролю доставки. На початковому етапі розробки було проведено аналіз сучасних вебсистем електронної комерції та визначено основні функціональні вимоги до програмного забезпечення. У результаті аналізу встановлено, що система повинна забезпечувати можливість реєстрації та авторизації користувачів, перегляду каталогу товарів, пошуку продукції, додавання товарів у кошик, оформлення замовлень та контролю доставки. Крім цього, окрему увагу було приділено створенню адміністративного модуля для керування товарами, замовленнями та користувачами системи. Для реалізації серверної частини вебзастосунку було обрано мову програмування Python та фреймворк Django. Використання Django дозволило реалізувати серверну логіку системи, забезпечити взаємодію між окремими модулями та спростити роботу з базою даних. Однією з переваг фреймворку є підтримка архітектури MVT, що дозволяє розділити логіку обробки даних, інтерфейс користувача та механізми взаємодії із базою даних. Це значно спрощує подальшу підтримку та модернізацію системи.

На наступному етапі було виконано проєктування структури бази даних. Для зберігання інформації у системі використовувалась база даних SQLite. Було створено основні моделі даних, які відповідають за роботу з користувачами, товарами, категоріями товарів, замовленнями та інформацією про доставку. Між окремими сутностями були реалізовані логічні зв’язки, що дозволило забезпечити коректну взаємодію між компонентами вебзастосунку та централізоване зберігання інформації.

Після створення структури бази даних було реалізовано функціональні модулі системи. У процесі розробки створено систему реєстрації та авторизації користувачів, що забезпечує захист персональних даних та розмежування доступу

до функціоналу вебзастосунку. Для користувачів реалізовано можливість перегляду каталогу товарів, пошуку продукції за категоріями, роботи з кошиком та оформлення онлайн-замовлень. Після підтвердження замовлення інформація автоматично передається до адміністративної частини системи для подальшої обробки. Окрему увагу було приділено реалізації модуля керування доставкою. Адміністратор системи має можливість переглядати список замовлень, змінювати їх статус, підтверджувати обробку та контролювати процес доставки товару. Завдяки інтеграції всіх функціональних модулів у межах єдиного вебзастосунку забезпечується централізована обробка інформації та автоматизація взаємодії між користувачем, адміністратором і базою даних. Для створення інтерфейсу користувача використовувалися HTML, CSS, JavaScript та Bootstrap. Застосування Bootstrap дозволило реалізувати адаптивний дизайн вебзастосунку та забезпечити коректне відображення сторінок на різних пристроях. Інтерфейс системи було розроблено з урахуванням принципів зручності використання та простоти навігації між основними функціональними модулями.

3.4 Реалізація функціоналу

У процесі розробки вебсистеми було реалізовано основний функціонал, необхідний для повноцінної роботи системи онлайн-замовлень та логістики доставки. Усі функціональні можливості створювалися безпосередньо на основі структури реального проєкту, реалізованого за допомогою Python та Django. Основною задачею під час реалізації функціоналу було створення зручної та стабільної системи, яка дозволяє користувачам переглядати товари, оформлювати замовлення та контролювати процес доставки через єдиний вебінтерфейс. Після запуску вебзастосунку користувач потрапляє на головну сторінку системи, де відображаються товари, категорії та акційні пропозиції. Для формування каталогу використовується модуль products, який взаємодіє з базою даних через Django ORM. У каталозі реалізовано можливість сортування товарів за ціною, назвою та популярністю. Кожен товар у системі містить детальну інформацію, зокрема назву,

опис, характеристики, ціну, розмір знижки, фото та рейтинг. Окремо реалізовано блок схожих товарів, що дозволяє користувачу швидко знаходити продукцію однієї категорії. Для зручності навігації було створено систему категорій та підкатегорій. Така структура дозволяє логічно групувати товари та спрощує пошук необхідної продукції у каталозі. У системі також реалізовано механізм пошуку товарів за назвою. Однією з головних функцій вебзастосунку є робота кошика покупця. Користувач може додавати товари до кошика без переходу на окрему сторінку оформлення. Після додавання система автоматично оновлює кількість товарів та загальну вартість замовлення. У модулі cart реалізовано зміну кількості товарів, видалення позицій та автоматичний перерахунок ціни. Дані кошика тимчасово зберігаються у системі до моменту підтвердження покупки. Після підтвердження покупки користувач переходить до оформлення замовлення. У процесі оформлення система зберігає інформацію про:

- товари;
- кількість;
- адресу доставки;
- користувача;
- спосіб доставки.

Після успішного оформлення автоматично створюється новий запис у таблиці замовлень.

У проєкті реалізовано окремий модуль orders, який відповідає за керування замовленнями. Для кожного замовлення система автоматично генерує статус. Це дозволяє контролювати процес виконання покупки та відображати актуальну інформацію користувачу. Після створення замовлення інформація передається до модуля доставки. У модулі delivery реалізовано механізм контролю логістики. Адміністратор може змінювати статус доставки через адміністративну панель, а користувач – переглядати поточний стан замовлення у власному кабінеті. Важливою частиною функціоналу є особистий кабінет користувача. Через нього користувач може переглядати історію замовлень, редагувати власний профіль та

керувати адресами доставки. Для цього використовується окрема модель UserAddress.

У системі також реалізовано функціонал обраних товарів. Користувач може додавати продукцію до списку улюблених для швидкого доступу у майбутньому. Даний функціонал реалізований через модель Favorite. Окрему увагу було приділено системі відгуків. Після перегляду товару користувач може залишити коментар та оцінку. На основі отриманих оцінок система автоматично формує рейтинг товару. Для адміністрування вебсистеми використовується Django Admin Panel. Через адміністративну панель можна додавати нові товари, змінювати категорії, редагувати інформацію про користувачів та керувати замовленнями. У проєкті реалізовано систему статистики, яка дозволяє переглядати кількість замовлень, користувачів та доставок. Це дозволяє контролювати загальний стан роботи системи. Під час реалізації функціоналу використовувалися:

- Django ORM;
- HTML;
- CSS;
- Bootstrap;
- шаблони Django;
- SQLite.

Використання Django ORM значно спростило роботу з базою даних та дозволило швидко реалізувати взаємозв’язки між моделями. Для підвищення зручності користувачів у системі реалізовано повідомлення про успішні дії. Наприклад, після додавання товару до кошика або оформлення замовлення користувач отримує відповідне повідомлення. Також реалізовано захист сторінок користувача через систему авторизації Django. Незареєстрований користувач не може оформлювати замовлення або переглядати особистий кабінет. На рис. 3.10, відображається скріншот головної сторінки, та видніється частина катологу товарів.

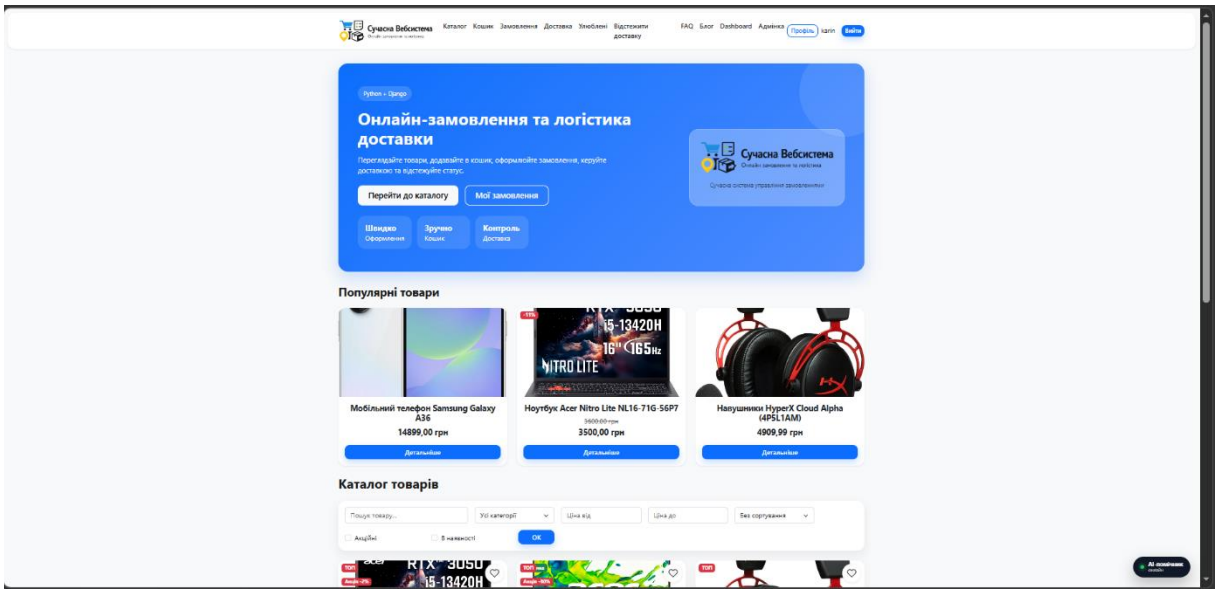


Рис. 3.10 Скріншот головної сторінки вебсистеми

На рис. 3.11 показано скріншот каталогу товарів вже детальніше.

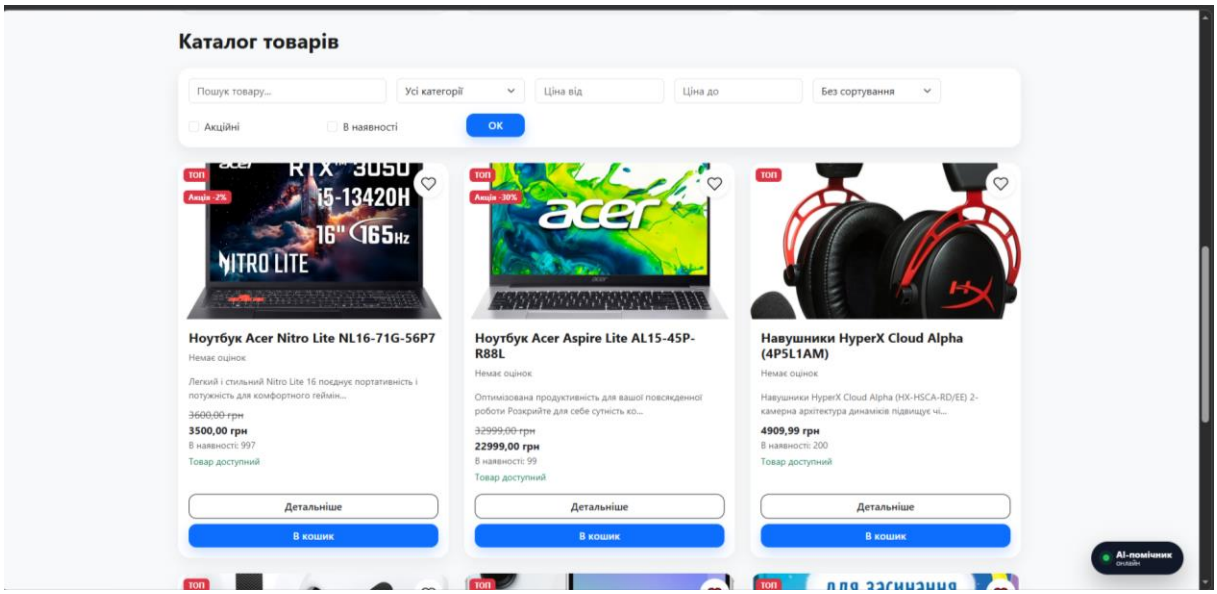


Рис. 3.11 Скріншот каталогу товарів

На рис. 3.12, зображено вже каталог товарів з системою сортування по категоріям.

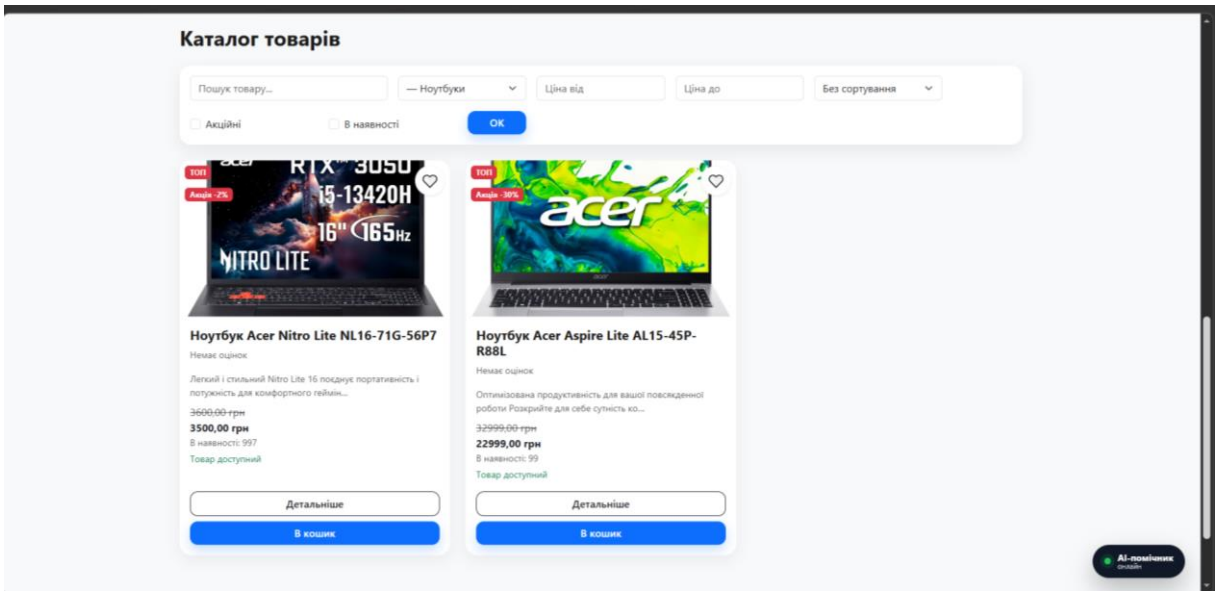


Рис. 3.12 Скріншот систему категорій

На рис. 3.13 зображено сторінка товару з детальною інформацією та можливостю додати в кошик, чи в улюблені. Також можна повернутися до каталогу товарів.

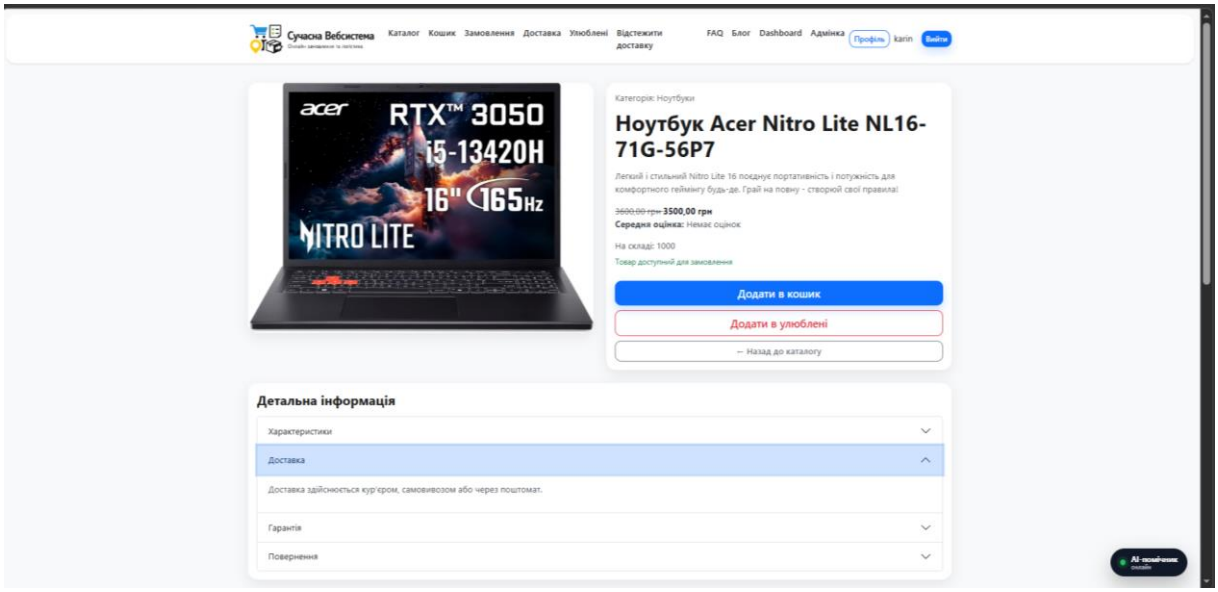


Рис. 3.13 Скріншот сторінку товару

На та рис. 3.14 зображені скріншот улюблених товарів користувача.

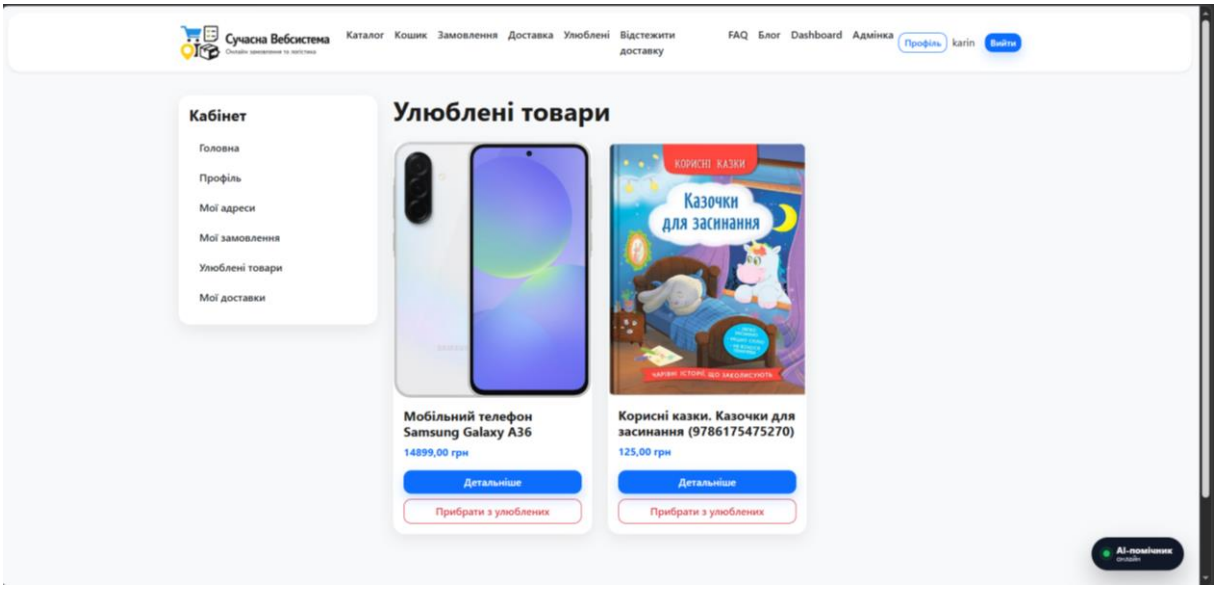


Рис. 3.14 Скріншот улюблених товарів користувача

Таким чином, реалізований функціонал забезпечує повноцінну роботу системи онлайн-замовлень та логістики доставки, а також дозволяє автоматизувати основні процеси електронної комерції.

3.5 Реалізація бази даних

Для збереження інформації у вебсистемі використовується система керування базами даних SQLite. Вибір даної СКБД обумовлений простотою інтеграції з Django, швидким налаштуванням та достатньою продуктивністю для реалізації вебсистеми онлайн-замовлень і логістики доставки. База даних є одним із ключових компонентів створеного проєкту, оскільки саме вона забезпечує збереження інформації про користувачів, товари, замовлення, доставку та взаємодію між усіма модулями системи. У проєкті робота з базою даних реалізована за допомогою Django ORM. Це дозволило працювати з таблицями бази даних через Python-моделі без необхідності створення складних SQL-запитів вручну. Кожна модель у системі відповідає окремій таблиці бази даних. Структура бази даних була спроєктована таким чином, щоб забезпечити правильний зв’язок між усіма компонентами вебсистеми. Основними сутностями проєкту є користувачі, товари, категорії,

кошик, замовлення, доставка, відгуки та список обраних товарів. Однією з головних моделей системи є модель Product, яка використовується для збереження інформації про товари. У ній зберігаються назва товару, опис, характеристики, ціна, акційна ціна, відсоток знижки, кількість товару на складі та фотографії продукції. Саме ця модель забезпечує роботу каталогу товарів у вебсистемі. Для організації структури каталогу реалізовано модель Category. Вона підтримує вкладені категорії через self ForeignKey, що дозволяє створювати підкатегорії та формувати ієрархічну структуру продукції. Завдяки цьому товари у вебсистемі розподіляються за категоріями, що значно спрощує навігацію та пошук продукції.

Важливу роль у роботі системи відіграє модель Order. Вона відповідає за збереження інформації про оформлені замовлення користувачів. Після підтвердження покупки система автоматично створює новий запис у таблиці замовлень та пов’язує його з користувачем. Для кожного замовлення у системі також створюються записи у моделі OrderItem. Саме ця модель зберігає перелік товарів, які входять до конкретного замовлення, їхню кількість та вартість. Такий підхід дозволяє організувати правильний зв’язок між замовленнями та товарами. Окремо у проєкті реалізовано модель Delivery, яка відповідає за логістику доставки. У ній зберігається інформація про адресу доставки, спосіб доставки, статус та дату створення доставки. Завдяки цьому система дозволяє контролювати процес виконання замовлення. Для збереження адрес користувачів створено модель UserAddress. Вона дозволяє користувачу використовувати декілька адрес доставки та спрощує процес оформлення замовлень. У вебсистемі також реалізовано систему відгуків через модель Review. Вона використовується для збереження коментарів користувачів та оцінок товарів. На основі оцінок система автоматично розраховує середній рейтинг продукції. Для реалізації списку обраних товарів використовується модель Favorite, яка пов’язує користувачів із товарами, доданими до улюблених. Під час розробки структури бази даних основна увага приділялася правильній організації зв’язків між таблицями. У системі активно використовуються зовнішні ключі ForeignKey, які забезпечують цілісність даних та дозволяють уникнути дублювання інформації. Завдяки використанню Django ORM

створення таблиць та оновлення структури бази даних виконувалося через систему міграцій. Це значно спростило процес розробки та дозволило швидко вносити зміни до структури моделей. У процесі роботи вебсистеми база даних забезпечує:

- збереження інформації про користувачів;
- роботу каталогу товарів;
- обробку замовлень;
- контроль доставки;
- збереження відгуків;
- роботу списку обраних товарів.

На рис. 3.15 зображено скріншот фрагмента коду для збереження інформації про користувача, а також на рис. 3.16 таблиці, де зберігається інформація користувача.

```python
def register_view(request):  2 usages  Карина Гришко
    if request.user.is_authenticated:
        return redirect('product_list')

    if request.method == 'POST':
        form = RegisterForm(request.POST)
        if form.is_valid():
            user = form.save()
            login(request, user)
            messages.success(request, message: 'Реєстрація успішна. Ви увійшли в систему.')
            return redirect('product_list')
    else:
        form = RegisterForm()

    return render(request, template_name: 'users/register.html', context: {'form': form})


def login_view(request):  2 usages  Карина Гришко
    if request.user.is_authenticated:
        return redirect('product_list')

    if request.method == 'POST':
        form = LoginForm(request, data=request.POST)
        if form.is_valid():
            user = form.get_user()
            login(request, user)
            messages.success(request, message: 'Ви успішно увійшли в систему.')
            return redirect('product_list')
        else:
            messages.error(request, message: 'Неправильне ім'я користувача або пароль.')
```

Рис. 3.15 Скріншот фрагмента коду для збереження інформації про користувача

Таблиця: auth_user — Filter in any column

	id	password	last_login	is_superuser	username	last_name	email	is_staff	is_active	date_joined	first_name
	Фі...	Фільтр	Фільтр	Фільтр	Фільтр	Фільтр	Фільтр	Фільтр	Фільтр	Фільтр	Фільтр
1	1	pbkdf2_sha2...	2026-05...	1	karin		karina...	1	1	2026-04-19 09:27:09.429486	
2	2	pbkdf2_sha2...	2026-04...	0	karina		karina...	0	1	2026-04-19 15:01:59.460681	
3	3	pbkdf2_sha2...	2026-05...	0	Георгій		Georgm...	0	1	2026-05-16 11:45:48.000767	

Рис. 3.16 Скріншот SQL таблиці користувачів

На рис. 3.17 скріншот частинної роботи каталогу товарів. А на рис. 3.18 збереження продуктів в таблиці.

```python
def product_list(request):  2 usages  Карина Гришко
    products = Product.objects.select_related('category').annotate(avg_rating=Avg('reviews__rating'))
    categories = Category.objects.select_related('parent').order_by( *field_names: 'parent__id', 'name')

    query = request.GET.get('q')
    category_id = request.GET.get('category')
    min_price = request.GET.get('min_price')
    max_price = request.GET.get('max_price')
    sort = request.GET.get('sort')

    only_promo = request.GET.get('only_promo')
    only_available = request.GET.get('only_available')

    if query:
        products = products.filter(name__icontains=query)

    if category_id:
        selected_category_obj = Category.objects.filter(id=category_id).first()
```

Рис. 3.17 Скріншот фрагмента коду роботи каталогу товарів

	id	name	description	price	stock	image	category_id	discount_percent	is_promo	old_price	is_new	delivery_info
1	1	Ноутбук Acer Nitro Lite NL16-71G-56P7	Легкий і стильний Nitro Lite 16 …	3500	997	products/001.png	5	11	1	3600	0	
2	2	Мобільний телефон Samsung Galaxy A36	Galaxy A36 вирізняється елегантним …	14899	55454	products/005.png	9	0	0	NULL	0	
3	3	Навушники HyperX Cloud Alpha …	Навушники HyperX Cloud Alpha (HX-…	4909.99	200	products/003.png	7	0	0	NULL	0	Доставка кур'єром, поштом
4	4	Корисні казки. Казочки для засинання…	«Казочки для засинання» написані …	125	99	products/006.png	10	0	0	NULL	0	
5	5	Навушники Apple AirPods (4-те …	Навушники Apple AirPods 4…	7399	98	products/004.png	8	0	0	NULL	0	
6	6	Ноутбук Acer Aspire Lite AL15-45P-…	Оптимізована продуктивність для вашо…	22999	99	products/002.png	5	0	1	32999	0	

Рис. 3.18 Скріншот SQL таблиці продуктів

На рис. 3.19 скріншот фрагмента коду де обробляється замовлення, а рис. 3.20 таблиці де зберігається інформація про доставку доставки.

```python
25      def checkout(request):
77          if request.method == 'POST':
78              form = OrderForm(request.POST)
79
80              if form.is_valid():
81                  order = form.save(commit=False)
82                  order.user = request.user
83                  order.status = 'awaiting_payment'
84                  order.coupon_code = coupon_code if coupon_code == 'SALE10' else ''
85                  order.discount_percent = discount_percent
86                  order.delivery_method = delivery_method
87                  order.delivery_price = delivery_price
88                  order.payment_method = form.cleaned_data.get('payment_method')
89                  order.save()
```

Рис. 3.19 Скріншот фрагмента коду обробки замовлень

Таблиця: delivery_delivery

Filter in any column

	id	courier_name	estimated_date	order_id	eta	tracking_number	delivery_status
	Фі...	Фільтр	Фільтр	Фільтр	Фільтр	Фільтр	Фільтр
1	10	NULL	NULL	10	2026-05-16 19:04:04.992796	TRK-2026-010	in_transit
2	11	NULL	NULL	11	2026-05-16 19:31:00.073137	TRK-2026-011	delivered
3	12	Андрій	2026-05-17	12	2026-05-19 07:48:28	TRK-2026-012	courier_assigned
4	13	NULL	NULL	13	2026-05-21 11:54:32.805877	TRK-2026-013	created
5	14	Андрій	2026-05-19	14	2026-05-21 11:56:11	TRK-2026-014	courier_assigned

Рис. 3.20 Скріншот SQL таблиці доставки

На рис. 3.21, скріншот фрагмента коду контролю доставки товари, а на рис. 3.22, Скріншот SQL таблиці історії статусу доставки

```python
@login_required  2 usages  Карина Гришко
def update_delivery_status(request, delivery_id):
    delivery = get_object_or_404(Delivery, id=delivery_id)

    status_flow = ['created', 'packed', 'courier_assigned', 'in_transit', 'delivered']

    try:
        current_index = status_flow.index(delivery.delivery_status)
    except ValueError:
        messages.error(request, message: 'Невідомий статус доставки')
        return redirect( to: 'delivery_detail', delivery_id=delivery.id)

    if current_index < len(status_flow) - 1:
        new_status = status_flow[current_index + 1]

        DeliveryStatusHistory.objects.create(
            delivery=delivery,
            old_status=delivery.delivery_status,
            new_status=new_status
        )

        delivery.delivery_status = new_status
        delivery.save()

        messages.success(request, message: f'Статус доставки оновлено: {delivery.get_delivery_status_display()}')
    else:
        messages.info(request, message: 'Доставка вже завершена (Доставлено)')
```

Рис. 3.21 Скріншот фрагмента коду контролю доставки

Таблиця: delivery_deliverystatushistory

Filter in any colu

	id	old_status	new_status	changed_at	delivery_id
	Фі...	Фільтр	Фільтр	Фільтр	Фільтр
1	10	created	packed	2026-05-13 19:40:28.609509	11
2	11	packed	courier_assigned	2026-05-13 19:40:30.356695	11
3	12	courier_assigned	in_transit	2026-05-13 19:54:26.985226	11
4	13	in_transit	delivered	2026-05-13 19:54:43.232213	11
5	14	created	packed	2026-05-13 20:07:40.736198	10
6	15	packed	courier_assigned	2026-05-13 20:07:41.565605	10

Рис. 3.22 Скріншот SQL таблиці історії статусу доставки

На рис. 3.23 скріншот коду збереження відгуків, рис. 3.24, таблиці де зберігаються відгуки користувачів.

```
    favorites = Favorite.objects.filter(user=request.user).select_related('product')
    return render(request, template_name: 'products/favorites.html', context: {'favorites': favorites})


@login_required  2 usages  Карина Гришко
def add_review(request, product_id):
    product = get_object_or_404(Product, id=product_id)

    text = request.POST.get('text', '').strip()
    rating = request.POST.get('rating')

    if text and rating:
        Review.objects.create(
            user=request.user,
            product=product,
            text=text,
            rating=int(rating)
        )
        messages.success(request, message: 'Відгук успішно додано.')
    else:
        messages.error(request, message: 'Заповніть текст відгуку та виберіть оцінку.')

    return redirect( to: 'product_detail', product_id=product.id)
```

Рис. 3.23 Скріншот фрагмента коду збереження відгуків

Таблиця: products_review

Filter in any column

	id	text	created_at	product_id	user_id	rating
	Фі...	Фільтр	Фільтр	Фільтр	Фільтр	Фільтр
1	1	А де фото?	2026-04-19 16:49:56.533149	4	1	5
2	2	А?	2026-04-19 17:57:05.089961	4	1	4
3	3	Не прийшов товар	2026-04-19 17:57:15.135147	4	1	1
4	4	Вау	2026-04-19 17:57:21.826666	4	1	5
5	5	Потужний, для дитини, для навчання…	2026-05-15 21:24:43.132542	1	1	5

Рис. 3.24 Скріншот SQL таблиці відгуків

На рис. 3.25зображено скріншот фрагмента коду який відповідає за список обраних товарів, на рис. 3.26. таблиця яка зберігає улюблені товари користувачів.

```python
def toggle_favorite(request, product_id):
    product = get_object_or_404(Product, id=product_id)
    favorite, created = Favorite.objects.get_or_create(user=request.user, product=product)

    if not created:
        favorite.delete()
        messages.info(request, message: f'Товар "{product.name}" видалено з улюблених.')
    else:
        messages.success(request, message: f'Товар "{product.name}" додано в улюблені.')

    return redirect( to: 'product_detail', product_id=product.id)


@login_required  2 usages  Карина Гришко
def favorite_list(request):
    favorites = Favorite.objects.filter(user=request.user).select_related('product')
    return render(request, template_name: 'products/favorites.html', context: {'favorites': favorites})
```

Рис. 3.25 Скріншот фрагмента для роботи списку обраних товарів

Таблиця: products_favorite

	id	product_id	user_id
	Фі...	Фільтр	Фільтр
1	2	2	1
2	3	4	1

Рис. 3.26 Скріншот SQL таблиці обраних товарів користувачів

4 ІНТЕРФЕЙС, ЛОГІСТИКА ТА ТЕСТУВАННЯ СИСТЕМИ

4.1 Інтерфейс користувача

Інтерфейс користувача у розробленій вебсистемі реалізовано за допомогою HTML, CSS, Bootstrap та шаблонів Django. Основною задачею під час створення інтерфейсу було забезпечення зручної взаємодії користувача із системою онлайн-замовлень та логістики доставки. Під час розробки особлива увага приділялася простоті навігації, швидкому доступу до основних функцій системи та адаптивності інтерфейсу для різних типів пристроїв. Завдяки використанню Bootstrap вебзастосунок коректно працює як на персональних комп'ютерах, так і на планшетах та мобільних пристроях. Головна сторінка вебсистеми є центральним елементом взаємодії користувача із застосунком. На ній відображається каталог товарів, категорії продукції, акційні товари та основна інформація про систему. Для покращення навігації у верхній частині сторінки реалізовано меню переходу між основними розділами вебсистеми. На головній сторінці також реалізовано систему пошуку товарів, що дозволяє швидко знаходити продукцію за назвою. Для більшої зручності користувач може використовувати фільтрацію та сортування товарів. На рис. 4.1, рис. 4.2. показано роботу пошуку товарів та сортування товарів.

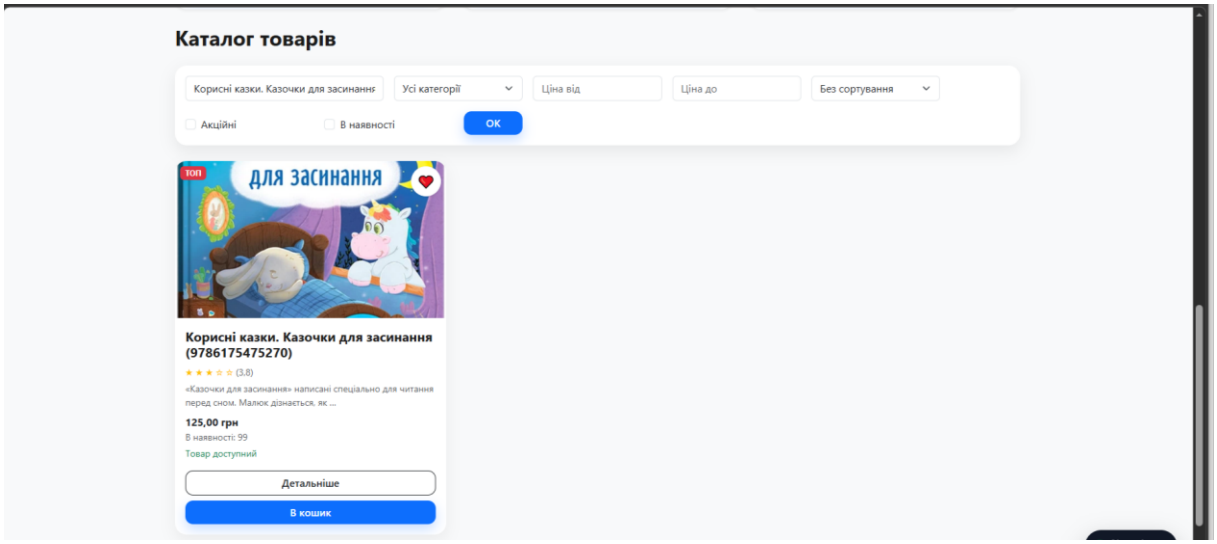


Рис. 4.1 Скріншот роботи пошуку товарів

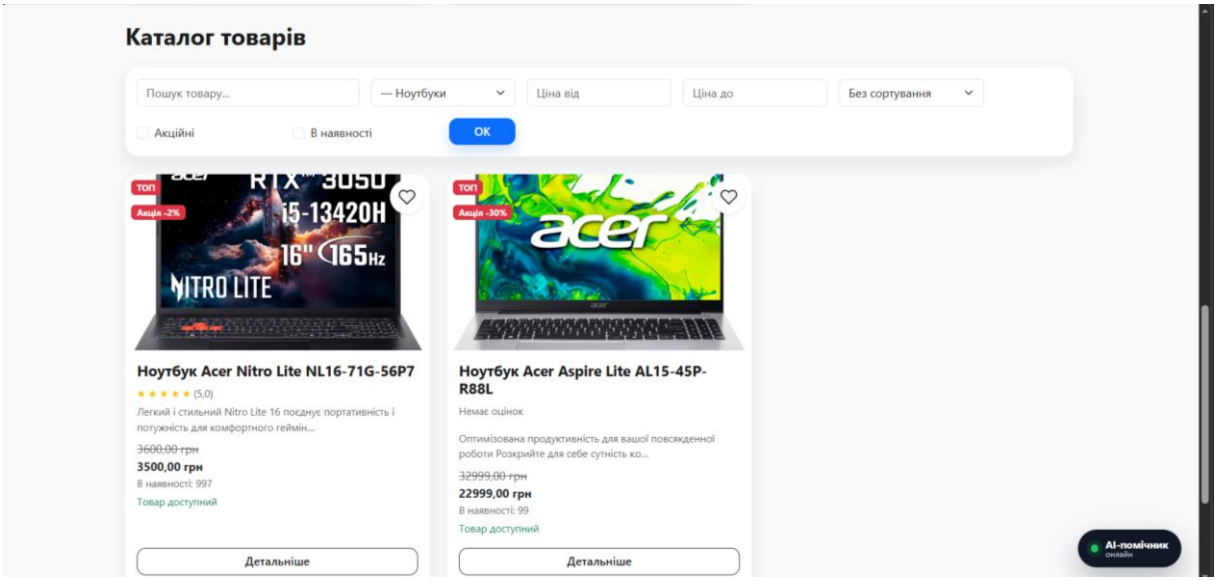


Рис. 4.2 Скріншот роботи сортування категорій товару

Каталог товарів реалізовано у вигляді карток продукції. Для кожного товару відображається фотографія, назва, ціна, рейтинг та коротка інформація про товар. Такий підхід дозволяє користувачу швидко переглядати великий обсяг продукції та знаходити необхідні товари. Після переходу на сторінку окремого товару користувач отримує повну інформацію про продукцію. На сторінці відображається опис товару, характеристики, рейтинг, система відгуків та рекомендовані товари. Також реалізовано кнопку додавання товару до кошика та можливість додати товар до списку обраних. Для покращення взаємодії користувача із системою реалізовано повідомлення про успішні дії. Наприклад, після додавання товару до кошика система автоматично повідомляє користувача про успішне виконання операції. На рис. 4.3 зображено скріншот кошику з товарами і підсумок сплати, а на рис. 4.4 зображено кошик з товарами після додачі товару.

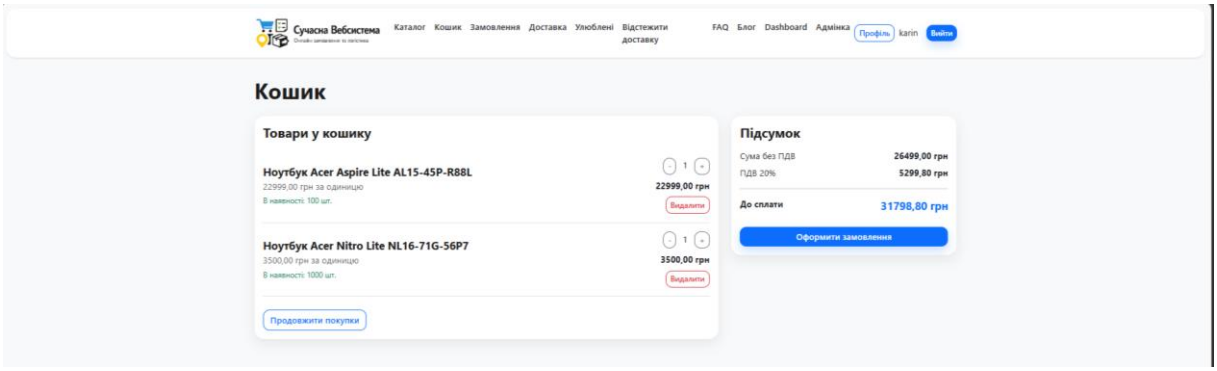


Рис. 4.3 Скріншот кошику

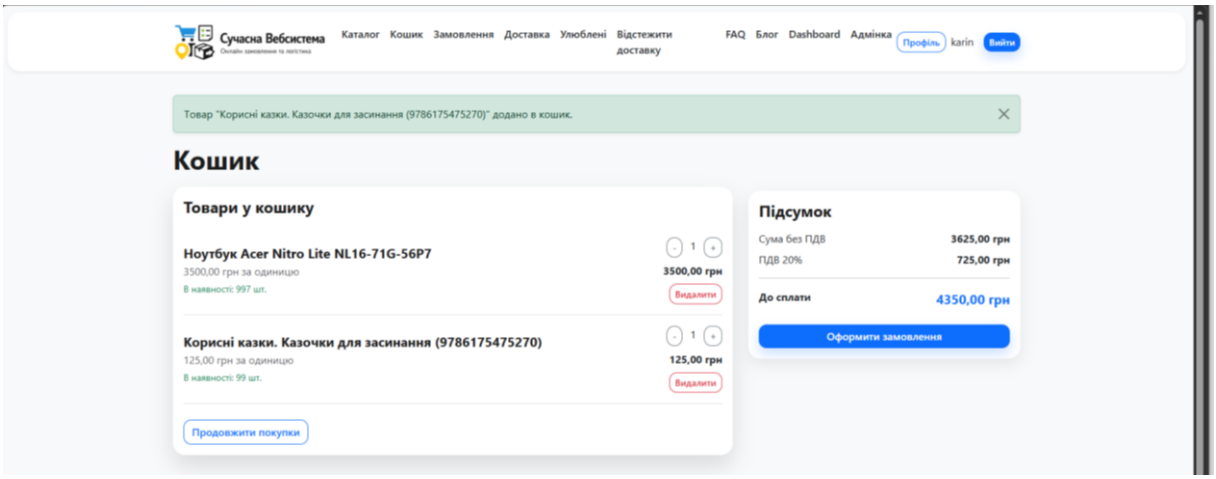


Рис. 4.4 Скріншот повідомлення про успішне додавання товару до кошика

На, рис. 4.5 зображено скріншот з відгуками користувачів на товар.

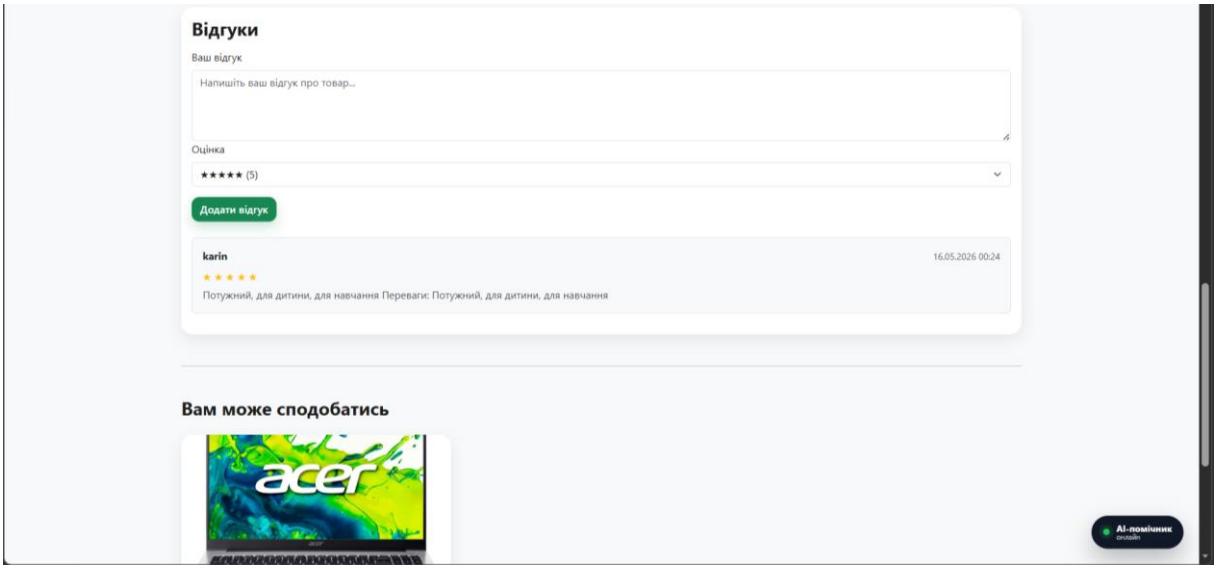


Рис. 4.5 Скріншот відгуків користувачів

Одним із важливих елементів інтерфейсу є кошик покупця. У ньому користувач може переглядати обрані товари, змінювати їхню кількість, видаляти непотрібні позиції та переглядати загальну вартість замовлення. Після підтвердження покупки користувач переходить до сторінки оформлення замовлення. У даному розділі реалізовано введення адреси доставки, контактних даних та підтвердження покупки. Особистий кабінет користувача дозволяє переглядати історію замовлень, редагувати профіль та керувати адресами доставки. Інтерфейс особистого кабінету реалізований максимально просто та зрозуміло, що забезпечує швидкий доступ до необхідної інформації. На рис. 4.6 скріншот прикладу оформлення форми замовлення. На рис. 4.7. скріншот зображено особистого кабінету користувача.

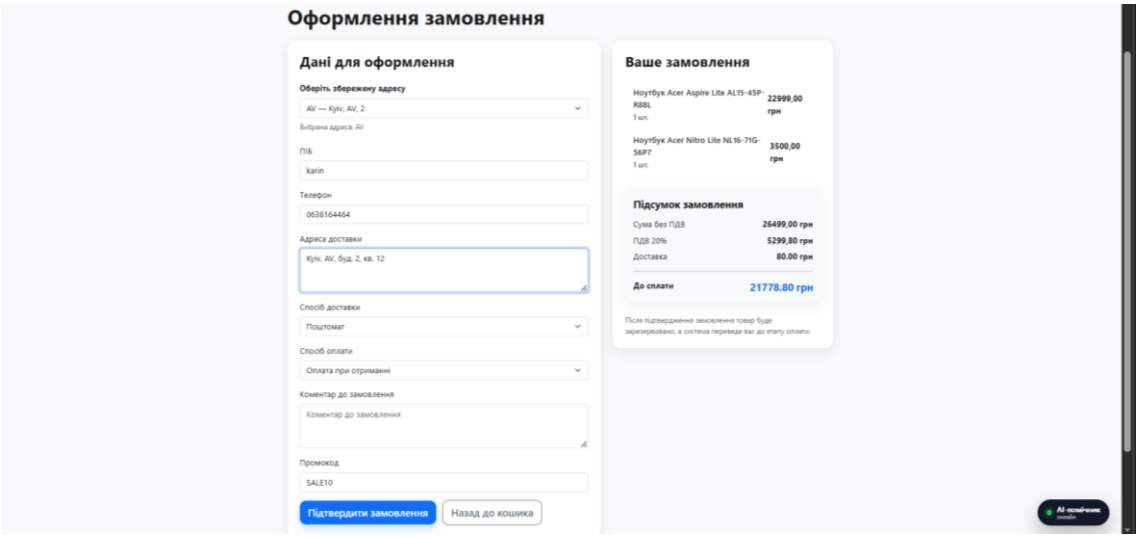


Рис. 4.6 Скріншот форми оформлення замовлення

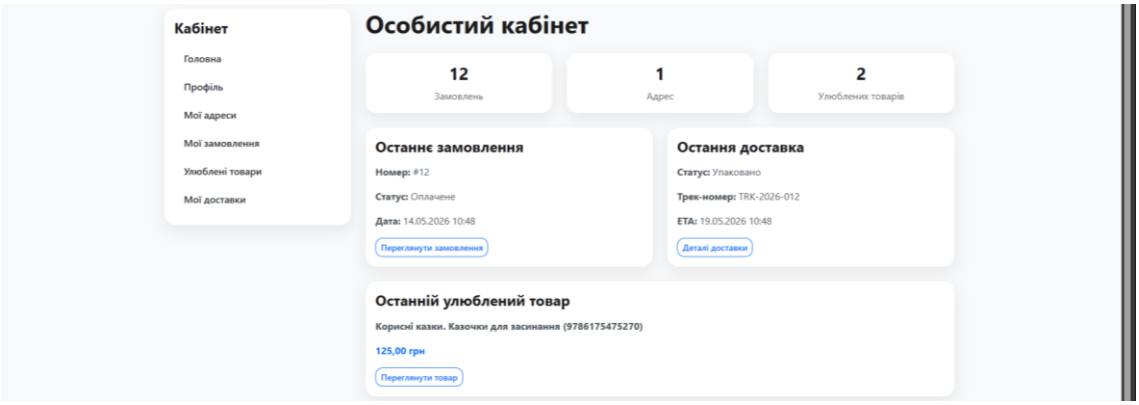


Рис. 4.7 Скріншот особистого кабінету користувача

У вебсистемі також реалізовано адміністративний інтерфейс на основі Django Admin Panel. Через адміністративну панель адміністратор може керувати товарами, категоріями, користувачами, доставками та замовленнями. Адміністративна панель значно спрощує процес керування системою, оскільки дозволяє швидко змінювати інформацію без необхідності редагування бази даних вручну. Для зручності адміністрування у панелі реалізовано таблиці замовлень, доставки та користувачів. Також адміністратор може змінювати статуси замовлень та контролювати процес доставки товарів. Рис. 4.8 показує аналітику системи онлайн-замовлень , а рис. 4.9 всіх замовлень користувача.

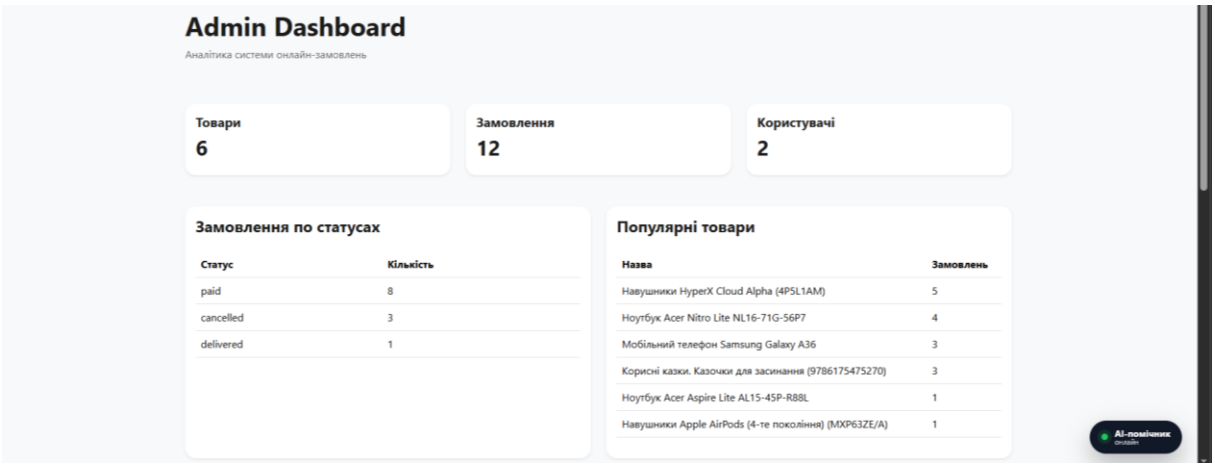


Рис. 4.8 Скріншот аналітики системи онлайн-замовлень

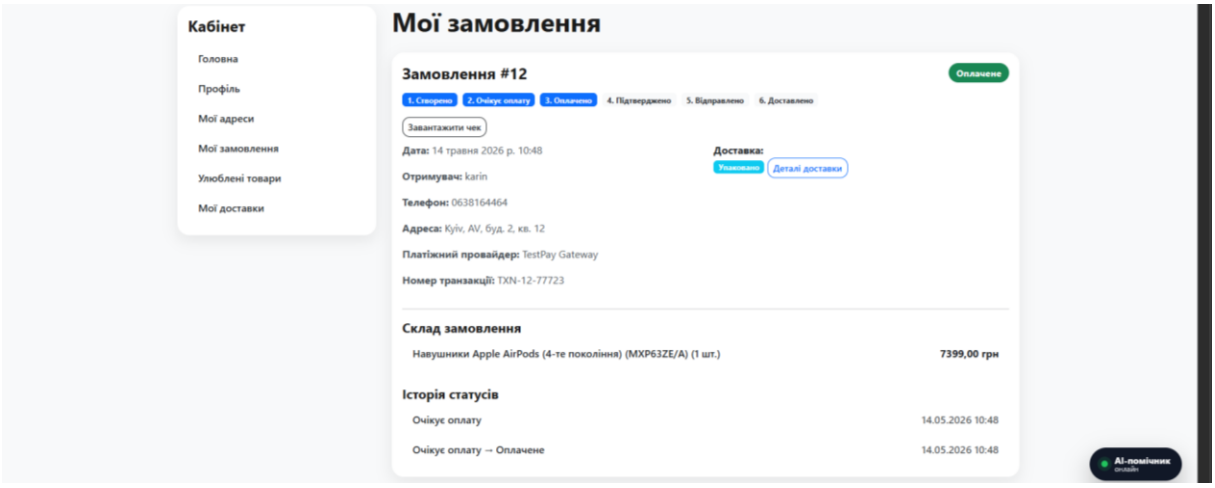


Рис. 4.9 Скріншот замовлень користувача

На рис. 4.10 показано деталі доставки до зміни статусу у замовлення, а рис. 4.11 вже після зміни.

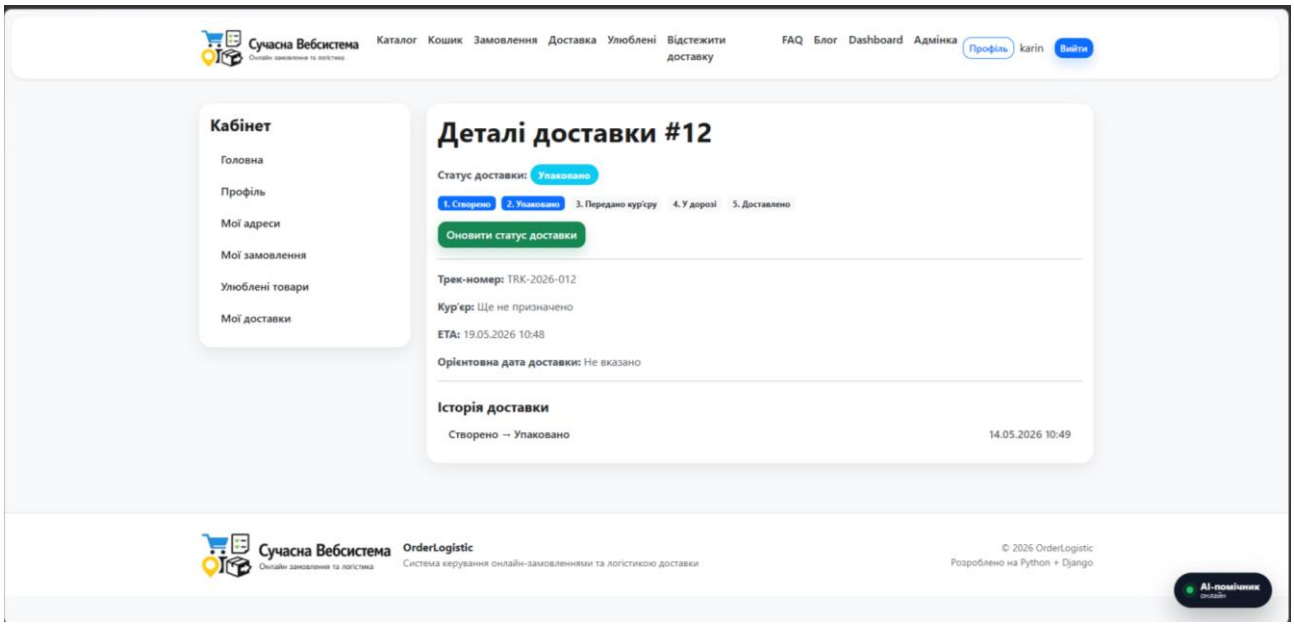


Рис. 4.10 Скріншот деталі доставки до зміни статусу

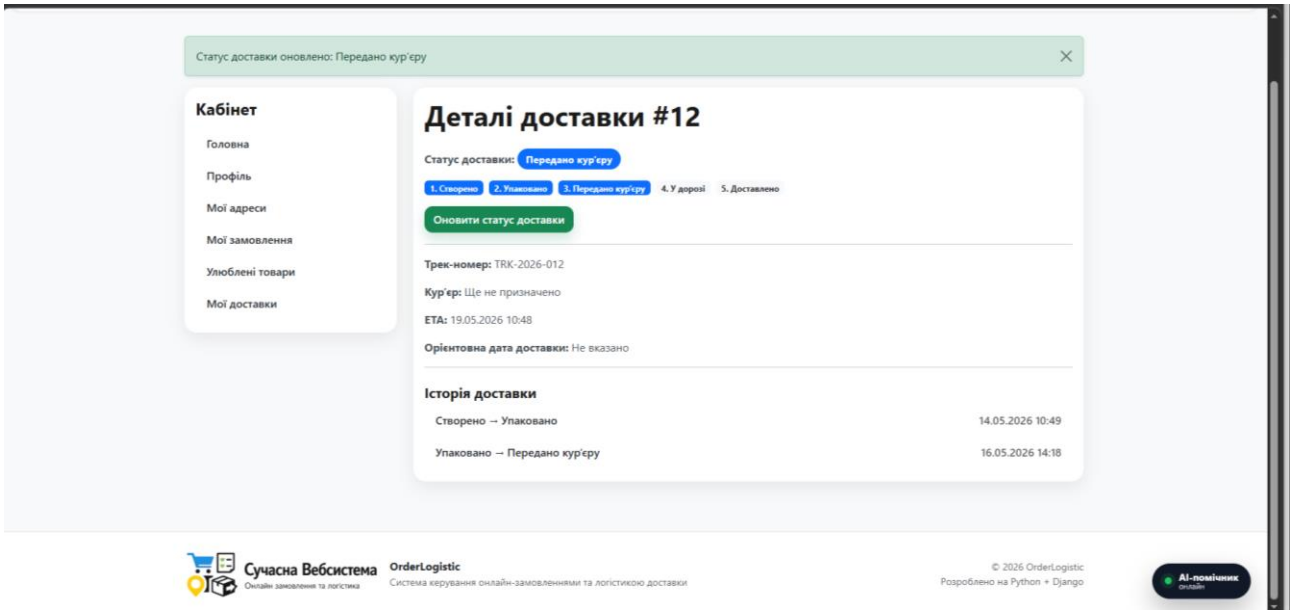


Рис. 4.11 Скріншот деталі доставки після зміни статусу

Під час створення інтерфейсу головна увага приділялася логічному розташуванню елементів та простоті використання. Усі основні функції системи знаходяться у швидкому доступі, що значно покращує взаємодію користувача із

вебзастосунком. Таким чином, реалізований інтерфейс забезпечує зручну роботу із системою онлайн-замовлень та логістики доставки, а також дозволяє ефективно взаємодіяти з усіма функціональними можливостями створеного вебзастосунку.

4.2 Реалізація логістики доставки

Одним із найважливіших компонентів розробленої вебсистеми є модуль логістики доставки, який забезпечує обробку та контроль виконання замовлень після їх оформлення користувачем. Реалізація даного функціоналу дозволяє автоматизувати процес доставки товарів та забезпечує взаємодію між користувачем, системою замовлень і адміністративною частиною вебзастосунку. Модуль доставки у створеному проєкті безпосередньо інтегрований із системою замовлень. Після підтвердження покупки система автоматично створює новий запис доставки та пов’язує його із відповідним замовленням користувача. Завдяки цьому вся інформація про оформлену покупку та процес її виконання зберігається у єдиній системі. Для реалізації логістики використовується окрема модель Delivery, яка містить основну інформацію про доставку. У ній зберігається адреса користувача, контактні дані, спосіб доставки, дата створення замовлення та поточний статус виконання доставки. Така структура дозволяє контролювати всі етапи логістики та забезпечує правильну взаємодію між різними компонентами вебсистеми. Під час оформлення замовлення користувач вводить власні контактні дані та адресу доставки. Після підтвердження покупки система автоматично передає інформацію до модуля доставки та створює новий запис у базі даних, як показано на рис. 4.12.

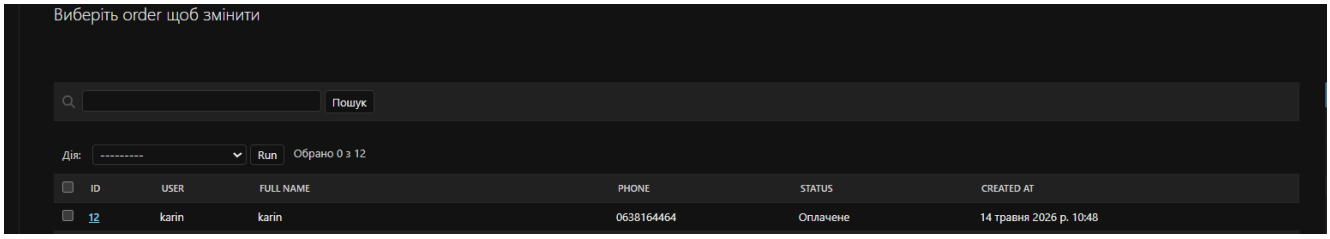


Рис. 4.12 Скріншот створеного запису доставки у Django Admin Panel

Однією з головних функцій модуля доставки є система статусів замовлень. Вона дозволяє контролювати поточний стан виконання доставки та забезпечує користувача актуальною інформацією про його покупку. Після створення замовлення система автоматично встановлює початковий статус доставки. Надалі адміністратор може змінювати статус залежно від етапу виконання замовлення. Наприклад, доставка може перебувати у стані обробки, бути переданою кур'єру або вже доставленою користувачу. Завдяки реалізованій системі статусів користувач може відстежувати процес доставки через особистий кабінет, а адміністратор – контролювати виконання всіх замовлень через адміністративну панель. У проєкті логістика доставки тісно пов'язана із системою замовлень. Після зміни статусу інформація автоматично оновлюється у базі даних та відображається у вебінтерфейсі. Це дозволяє підтримувати актуальний стан замовлень та забезпечує коректну роботу всієї системи. Для забезпечення правильної взаємодії між компонентами у системі реалізовано зв'язок між моделями Order та Delivery через ForeignKey. Завдяки цьому кожна доставка прив'язується до конкретного користувача та оформленого замовлення.

Важливу роль у реалізації логістики відіграє адміністративна частина системи. Через Django Admin Panel адміністратор може переглядати список усіх доставок, змінювати статуси, переглядати адреси користувачів та контролювати виконання замовлень. Усі дані про доставки відображаються у вигляді таблиць, що значно спрощує процес адміністрування вебсистеми. Адміністратор може швидко знайти потрібне замовлення, переглянути інформацію про користувача та змінити поточний статус доставки. На рис. 4.13 показано скріншот списку доставок у Django Admin Panel, рис. 4.14 зображено список змін статусу у замовлення в Django Admin Panel.

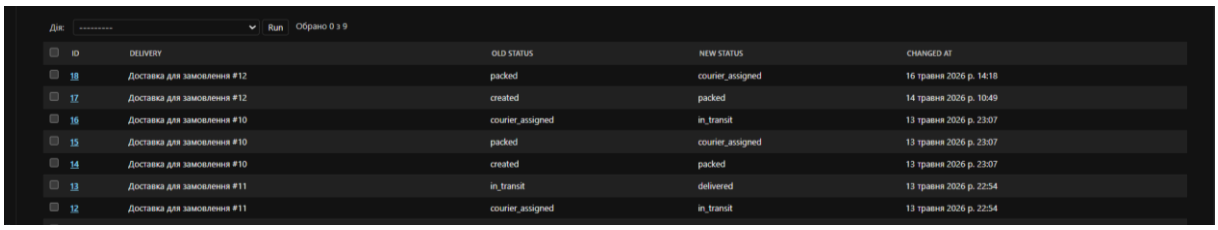


Рис. 4.13 Скріншот списку доставок у Django Admin Panel

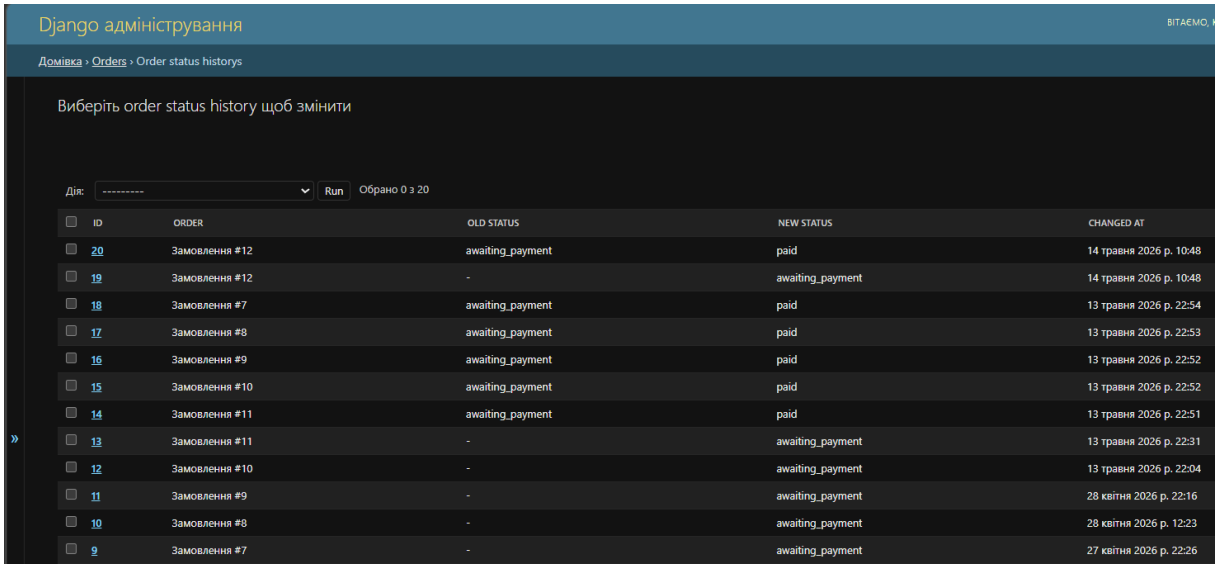


Рис. 4.14 Скріншот зміни статусу замовлення у Django Admin Panel

Окрему увагу під час реалізації було приділено взаємодії користувача із системою доставки. У особистому кабінеті користувач може переглядати історію своїх замовлень, інформацію про доставку та поточний статус виконання покупки. Такий підхід значно покращує зручність використання вебсистеми, оскільки користувач отримує можливість контролювати процес доставки без необхідності звернення до адміністратора. Для реалізації модуля доставки використовувалися можливості Django ORM та механізм моделей Django. Це дозволило забезпечити правильне збереження даних у базі та реалізувати взаємозв’язок між користувачами, замовленнями та доставками. Під час створення системи також було реалізовано перевірку правильності введених даних під час оформлення замовлення. Це дозволяє уникнути помилок у контактній інформації та адресах доставки. Крім цього, у системі реалізовано автоматичне оновлення інформації про доставку після зміни статусу замовлення. Завдяки цьому користувач завжди бачить актуальний стан виконання покупки.

На рис. 4.15 показано скріншот моделі Delivery у файлі models.py, у рис. 4.16 таблиця доставки у базі даних. А на рис. 4.17 скріншот відображення статусу доставки в особистому кабінеті.

```python
class Delivery(models.Model):  9 usages  Карина Гришко
    DELIVERY_STATUS = [
        ('created', 'Створено'),
        ('packed', 'Упаковано'),
        ('courier_assigned', 'Передано кур'єру'),
        ('in_transit', 'У дорозі'),
        ('delivered', 'Доставлено'),
    ]

    order = models.OneToOneField(Order, on_delete=models.CASCADE, related_name='delivery')
    courier_name = models.CharField(max_length=255, blank=True, null=True)
    tracking_number = models.CharField(max_length=50, blank=True, null=True)
    eta = models.DateTimeField(blank=True, null=True)
    estimated_date = models.DateField(blank=True, null=True)

    delivery_status = models.CharField(
        max_length=20,
        choices=DELIVERY_STATUS,
        default='created'
    )
```

Рис. 4.15 Скріншот моделі Delivery у файлі models.py

Таблиця: delivery_delivery

Filter in any column

	id	courier_name	estimated_date	order_id	eta	tracking_number	delivery_status
	Фі...	Фільтр	Фільтр	Фільтр	Фільтр	Фільтр	Фільтр
1	10	NULL	NULL	10	2026-05-16 19:04:04.992796	TRK-2026-010	in_transit
2	11	NULL	NULL	11	2026-05-16 19:31:00.073137	TRK-2026-011	delivered
3	12	NULL	NULL	12	2026-05-19 07:48:28.335675	TRK-2026-012	courier_assigned

Рис. 4.16 Скріншот таблиці доставки у базі даних

Деталі доставки #12

Статус доставки: Передано кур'єру

1. Створено | 2. Упаковано | 3. Передано кур'єру | 4. У дорозі | 5. Доставлено

Оновити статус доставки

Трек-номер: TRK-2026-012

Кур'єр: Андрій

ETA: 19.05.2026 10:48

Орієнтовна дата доставки: 17 травня 2026 р.

Історія доставки

Створено → Упаковано 14.05.2026 10:49

Упаковано → Передано кур'єру 16.05.2026 14:18

Рис. 4.17 Скріншот відображення статусу доставки в особистому кабінеті користувача

Під час розробки логістики доставки основна увага приділялася, стабільності роботи системи та простоті використання як для користувача, так і для адміністратора. Таким чином, реалізований модуль логістики доставки забезпечує повноцінний контроль процесу виконання замовлень, автоматизує обробку доставки товарів та забезпечує стабільну взаємодію між усіма компонентами створеної вебсистеми онлайн-замовлень.

4.3 Тестування системи

Після завершення розробки вебсистеми було проведено тестування основного функціоналу для перевірки стабільності роботи, правильності обробки даних та взаємодії між модулями проєкту. Основною метою тестування було переконатися, що система онлайн-замовлень та логістики доставки працює коректно та забезпечує правильне виконання всіх основних операцій. Тестування проводилося безпосередньо на створеному вебзастосунку, реалізованому за допомогою Python та Django. Під час перевірки особлива увага приділялася роботі користувацької частини системи, адміністративної панелі та взаємодії з базою даних. Одним із перших етапів тестування була перевірка системи авторизації та реєстрації користувачів. Було перевірено правильність створення нових облікових записів, роботу входу до системи та збереження інформації у базі даних.

Після успішної авторизації користувача було протестовано роботу каталогу товарів. У процесі тестування перевірялася коректність відображення товарів, робота категорій, пошуку та сортування продукції.

На рис. 4.18 показаний скріншот результату працездатності головної сторінки без авторизації.

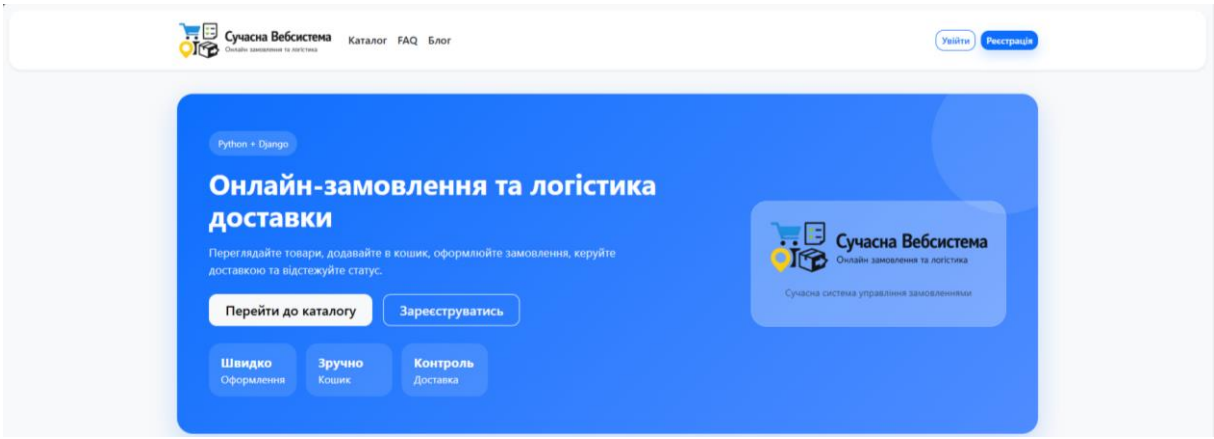


Рис. 4.18 Скріншот перевірки головної сторінки без авторизації

На рис. 4.19 скріншот результату перевірки при не надійного пароля.

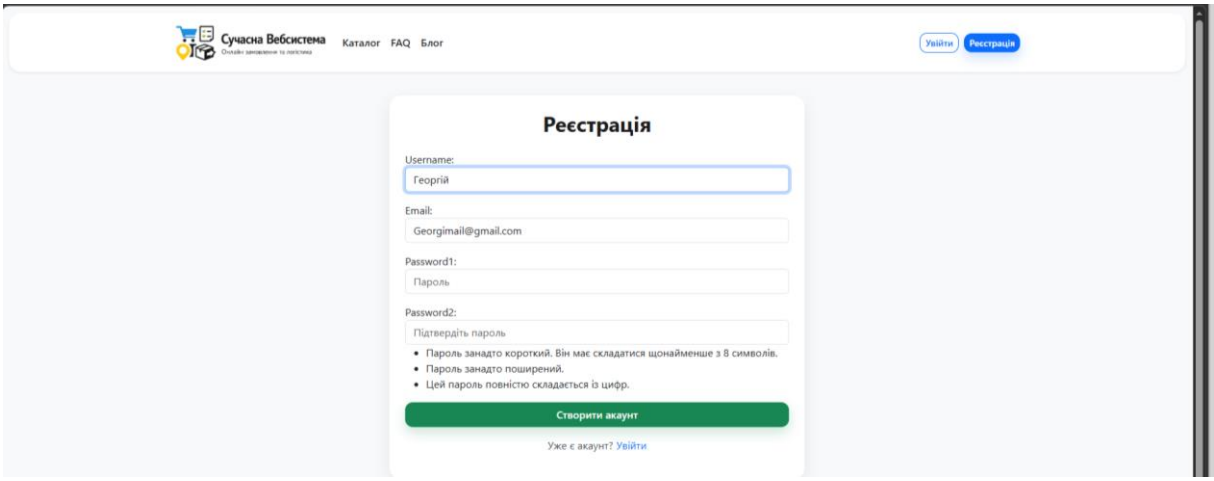


Рис. 4.19 Скріншот перевірки при неуспішній реєстрації

На рис. 4.20 зображено результат успішної реєстрації, автоматичного перехода на головну сторінку та показу повідомлення про успішну реєстрацію.

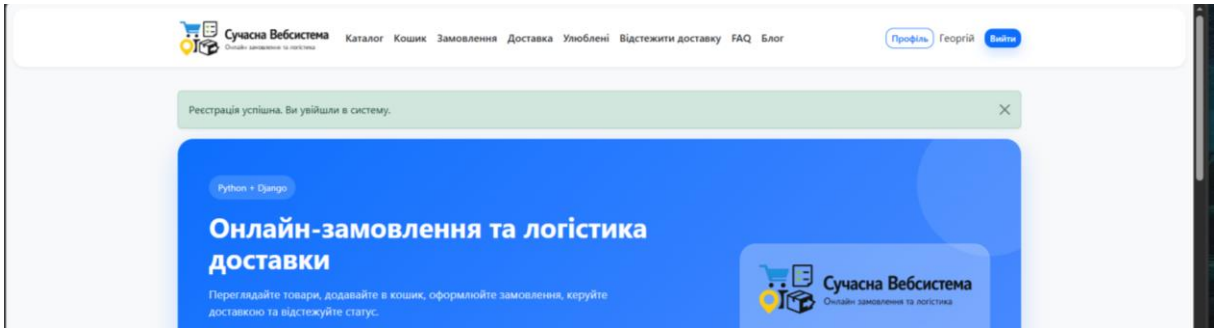


Рис. 4.20 Скріншот перевірки при успішній реєстрації

Окрему увагу було приділено тестуванню сторінки товару. Перевірялася правильність відображення інформації про продукцію, фотографій, рейтингу та системи відгуків. Також було протестовано роботу кнопок додавання товару до кошика та списку обраних. Під час тестування кошика перевірялася можливість додавання товарів, зміни кількості продукції, видалення позицій та автоматичного перерахунку загальної вартості замовлення. Після цього було протестовано процес оформлення замовлення. Перевірялася правильність введення контактних даних, збереження адрес доставки та створення нового запису у таблиці замовлень. Рис. 4.21 та рис. 4.22 показують скріншот перевірки працездатності порожнього кошика й кошика коли додали користувач товар.

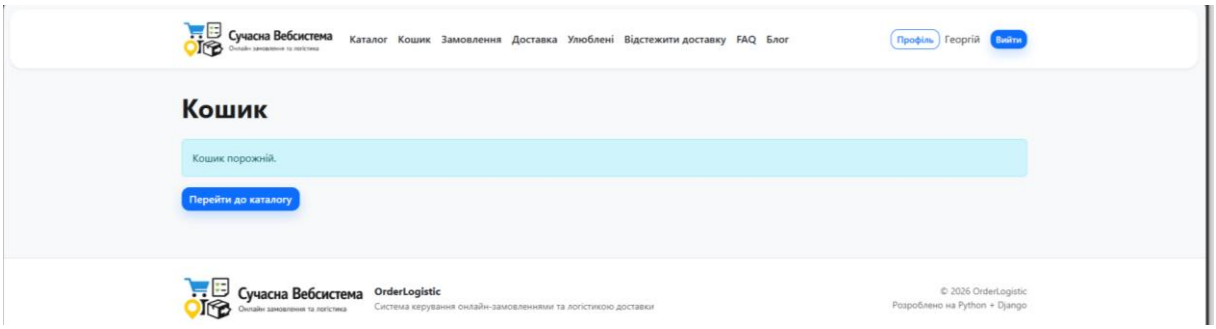


Рис. 4.21 Скріншот перевірки при відсутності товару в кошику

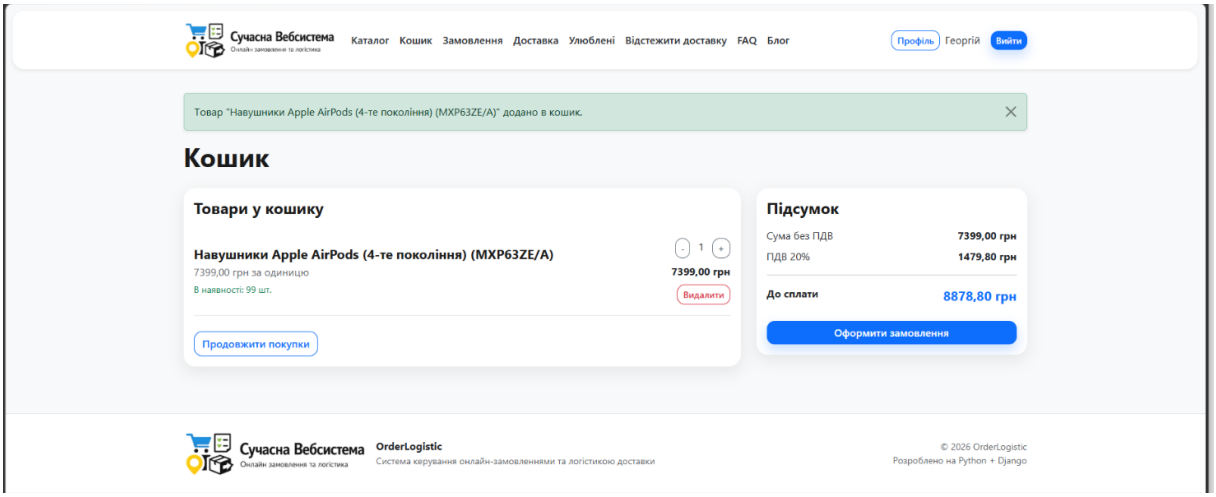


Рис. 4.22 Скріншот перевірки при додавання товару в кошик

На рис. 4.23 показано оформлення замовлення, потім на рис. 4.24 показано успішне оформлення і переходження до сторінки оплати.

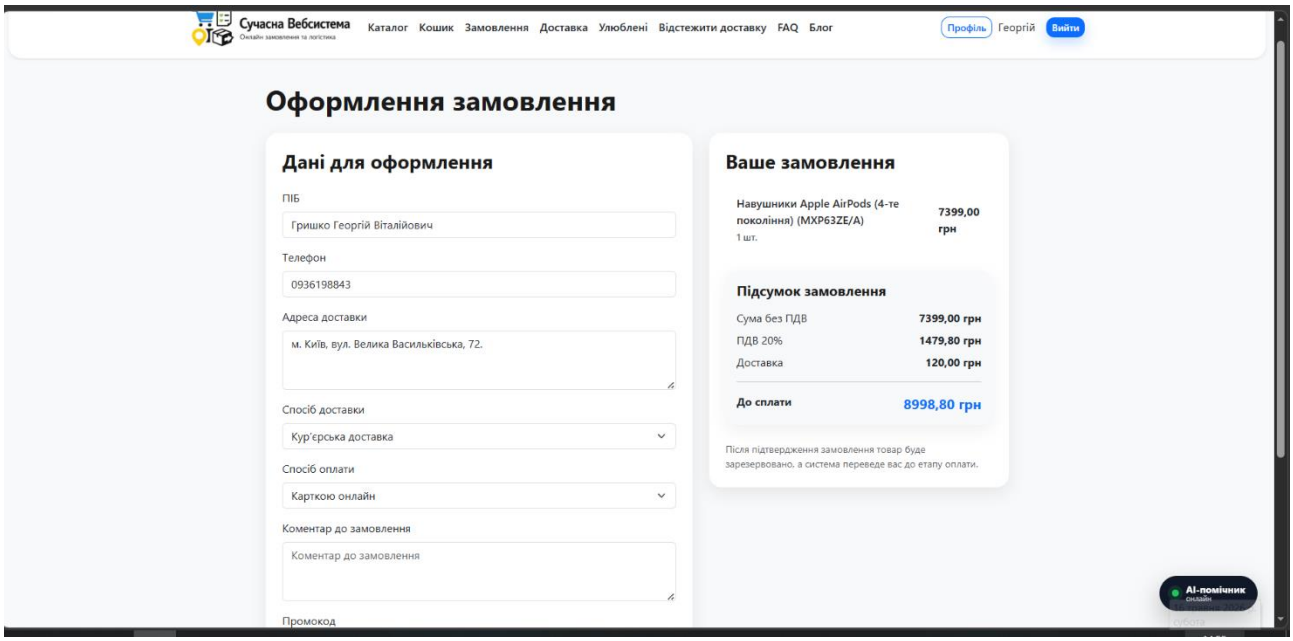


Рис. 4.23 Скріншот перевірки сторінки оформлення замовлення

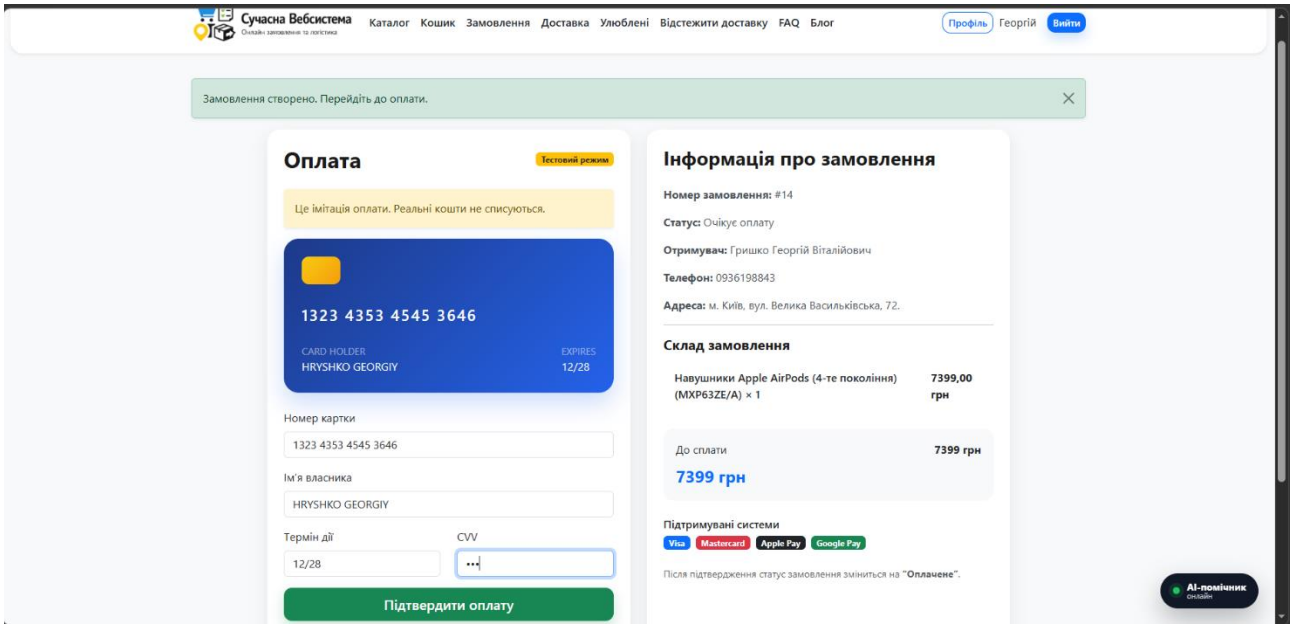


Рис. 4.24 Скріншот перевірки тестової оплати онлайн

На рис. 4.25 скріншот при успішній оплаті , рис. 4.26 показує що замовлення успішно прийнялося і показується на сторінці.

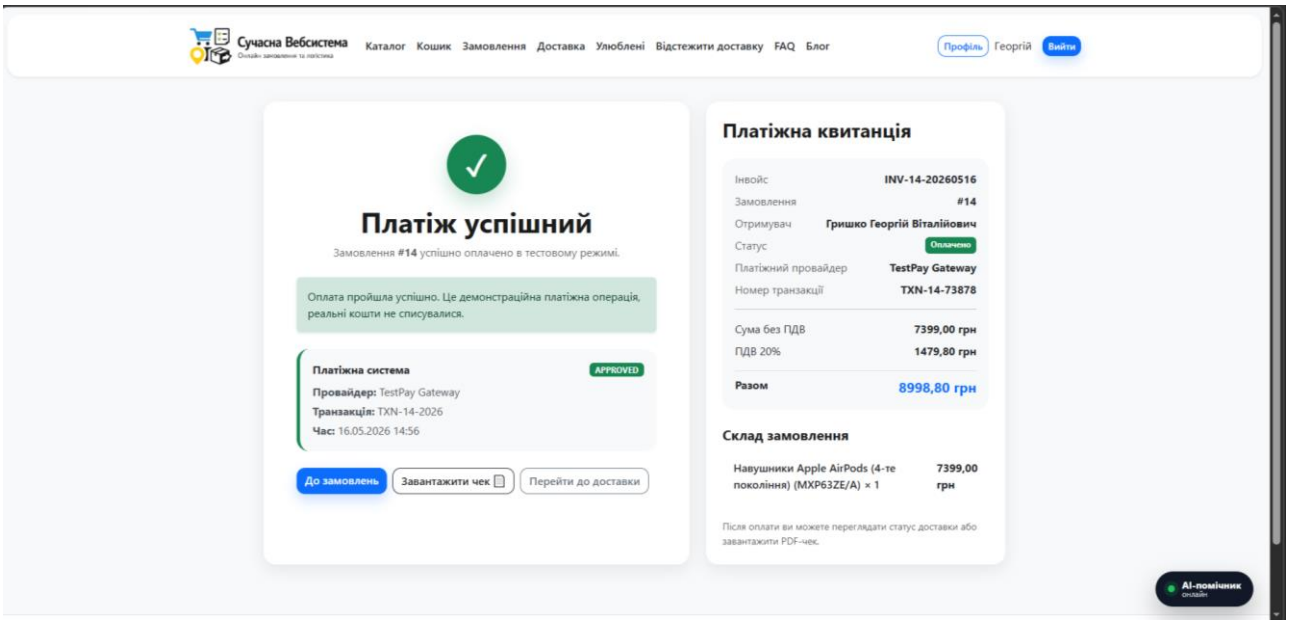


Рис. 4.25 Скріншот перевірки тестового успішного платіжу

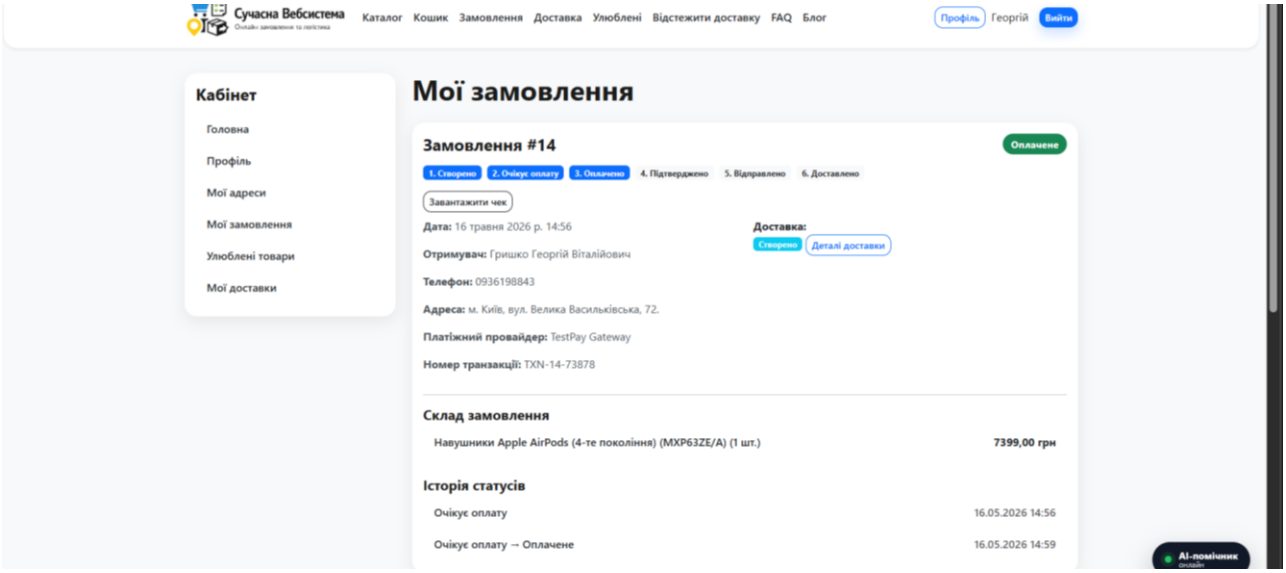


Рис. 4.26 Скріншот перевірки знаходження замовлення і його статусу

Також у процесі тестування системи замовлень перевірялася правильність створення записів у моделях Order та OrderItem. Було протестовано на зміну статусів замовлень та оновлення інформації у базі даних. Окремо було проведено перевірку модуля доставки. Під час тестування контролювалася правильність створення доставки після оформлення покупки, зміна статусів доставки та відображення інформації у особистому кабінеті користувача. Було встановлено, що після зміни статусу доставки через Django Admin Panel інформація автоматично

оновлюється у вебінтерфейсі та коректно відображається користувач. На рис. 3.27, рис. 4.28, рис. 4.29 показаний успішний етап перевірки деталей замовлення при зміни статусу.

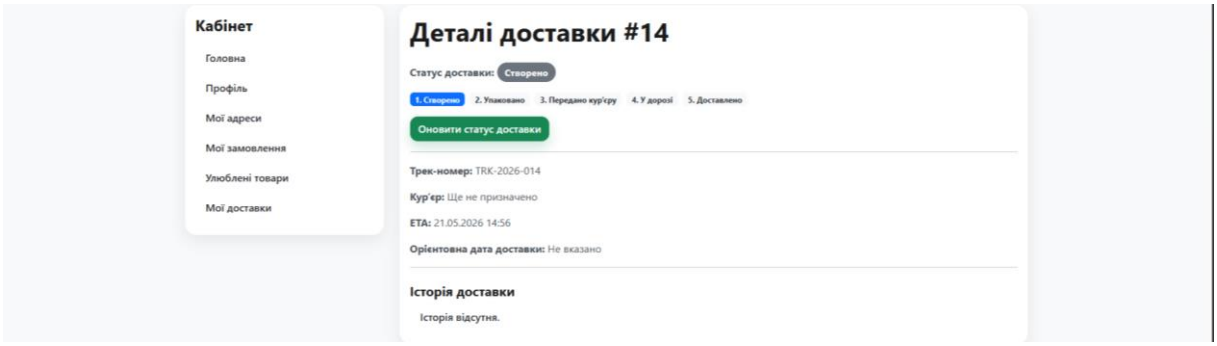


Рис. 4.27 Скріншот перевірки деталі доставки до зміним статусу

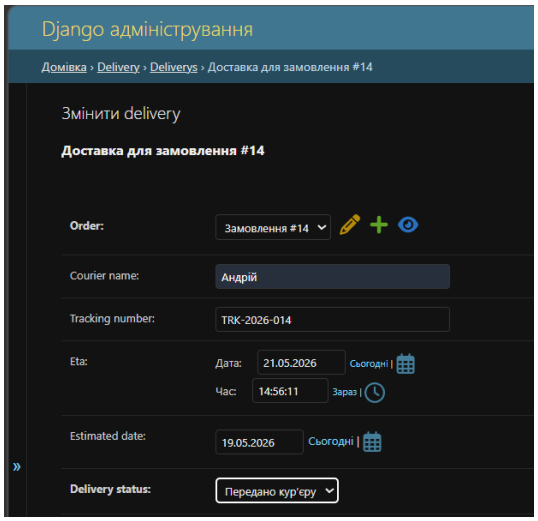


Рис. 4.28 Скріншот перевірки призначення кур’єра і дати доставки та зміни статусу в Django Admin Panel

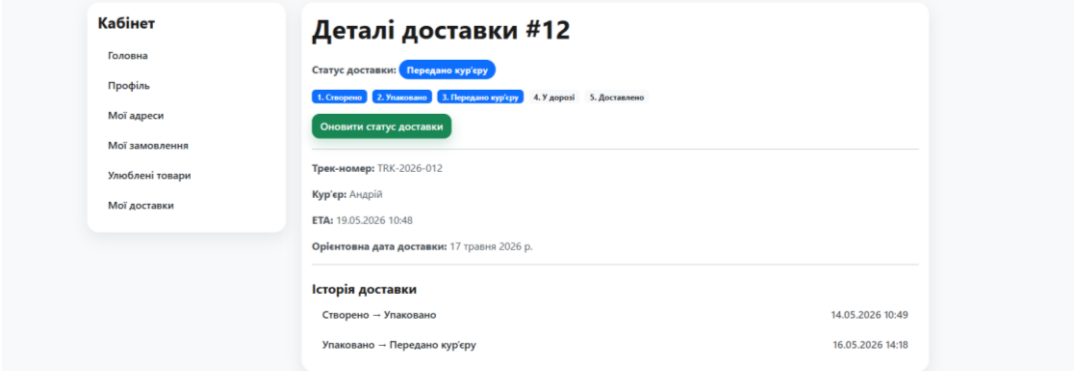


Рис. 4.29 Скріншот перевірки деталі доставки після зміним статусу

Для перевірки коректності роботи розробленої вебсистеми було проведено тестування основних функціональних модулів. У процесі тестування перевірялась стабільність роботи системи, правильність обробки даних, взаємодія серверної частини з базою даних, а також швидкість виконання основних операцій користувача. Особлива увага приділялась перевірці роботи авторизації, оформлення замовлень, функціонування адміністративної панелі та модуля керування доставкою. Результати тестування вебсистеми знаходяться в таблиці 4.1.

Таблиця 4.1

Результати тестування вебсистеми

Параметр	**Результат**
Час завантаження головної сторінки	1,4 с
Час авторизації користувача	0,8 с
Час оформлення замовлення	1,2 с
Середній час відповіді сервера	0,6 с

Результати тестування підтвердили коректність роботи функціональних модулів вебсистеми, стабільність взаємодії між серверною частиною та базою даних, а також правильність обробки користувацьких запитів. У процесі тестування критичних помилок виявлено не було. Отримані результати свідчать про можливість практичного використання системи для автоматизації процесів онлайн-замовлень та логістики доставки. Під час тестування адміністративної панелі було перевірено роботу керування товарами, категоріями, користувачами та доставками. Адміністратор може швидко редагувати інформацію та контролювати роботу всієї системи через єдиний інтерфейс. Також було протестовано роботу бази даних та взаємодію між моделями. Під час перевірки не було виявлено помилок у зв’язках між таблицями, а всі дані коректно зберігалися через Django ORM. У процесі тестування особлива увага приділялася стабільності роботи вебсистеми при виконанні основних операцій. Було перевірено:

- створення нових замовлень;

- роботу доставки;
- взаємодію між користувачем та системою;
- коректність збереження інформації;
- оновлення статусів;
- роботу адміністративної панелі.

Таким чином, проведене тестування підтвердило правильність реалізації функціоналу вебсистеми, стабільність її роботи та готовність до використання у якості системи онлайн-замовлень і логістики доставки.

ВИСНОВКИ

У результаті виконання кваліфікаційної роботи було досягнуто поставленої мети – розроблено вебсистему керування онлайн-замовленнями та логістикою доставки засобами мови програмування Python та фреймворку Django. Створена система забезпечує автоматизацію процесів електронної комерції, обробку замовлень, керування товарами та контроль доставки. Для досягнення поставленої мети було виконано такі завдання:

1. Досліджено теоретичні основи функціонування систем онлайн-замовлень та логістики доставки. Встановлено, що використання вебсистем дозволяє автоматизувати процеси обробки замовлень, підвищити швидкість роботи та покращити якість обслуговування користувачів.
2. Проведено аналіз сучасних технологій веброзробки для створення систем електронної комерції. Встановлено, що мова програмування Python та фреймворк Django є ефективними засобами для створення безпечних, масштабованих і функціональних вебзастосунків.
3. Спроєктовано структуру інформаційної системи керування онлайн-замовленнями та логістикою доставки. У результаті визначено архітектуру системи, основні модулі та структуру бази даних, що забезпечує ефективну організацію та обробку даних.
4. Виконано моделювання основних бізнес-процесів системи. Встановлено взаємозв’язки між основними сутностями та визначено інформаційні потоки між модулями вебзастосунку, що дозволило логічно організувати роботу системи.
5. Реалізовано програмне рішення засобами Python та Django. У результаті створено вебзастосунок із функціями реєстрації користувачів, перегляду товарів, роботи з кошиком, оформлення замовлень та контролю доставки.
6. Проведено тестування системи та оцінено ефективність її роботи. Результати тестування підтвердили коректність функціонування основних модулів,

стабільність роботи системи та можливість її практичного використання у сфері електронної комерції.

Отже, у ході виконання кваліфікаційної роботи було створено функціональну вебсистему керування онлайн-замовленнями та логістикою доставки, яка може бути використана як основа для автоматизації діяльності інтернет-магазинів та сервісів доставки.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Горобченко О. А. Особливості функціонування та створення інтернет-магазину. Інвестиції: практика та досвід. 2023. № 16. С. 71–76. URL: https://doi.org/10.32702/2306-6814.2023.16.71
2. Шалева О. І., Середа І. С. Можливості диверсифікації діяльності роздрібного торговельного підприємства шляхом створення інтернет-магазину. Науковий вісник ПУЕТ: Економічні науки. 2023. № 3 (109). URL: https://doi.org/10.37734/2409-6873-2023-3-9
3. A comprehensive approach to home service booking system development in a django web application / S. K. Pandey et al. International Journal of Research - GRANTHAALAYAH. 2025. Vol. 13, no. 4. URL: https://doi.org/10.29121/granthaalayah.v13.i4.2025.6192
4. Bootstrap Documentation / Bootstrap Team. URL: https://getbootstrap.com/docs/5.3/getting-started/introduction/ (дата звернення: 14.05.2026).
5. Chandiramani A. Management of django web development in python. Journal of Management and Service Science (JMSS). 2021. Vol. 1, no. 2. P. 1–17. URL: https://doi.org/10.54060/jmss/001.02.005
6. CSS: Cascading Style Sheets / Mozilla Developer Network. URL: https://developer.mozilla.org/en-US/docs/Web/CSS (дата звернення: 14.05.2026).
7. Django Forms / Django Software Foundation. URL: https://docs.djangoproject.com/en/stable/topics/forms/ (дата звернення: 14.05.2026).
8. HTML: HyperText Markup Language / Mozilla Developer Network. URL: https://developer.mozilla.org/en-US/docs/Web/HTML (дата звернення: 14.05.2026).
9. JavaScript Guide / Mozilla Developer Network. URL: https://developer.mozilla.org/en-US/docs/Web/JavaScript/Guide (дата звернення: 14.05.2026).

10. Manoj Kumar, Rainu Nandal. Python's role in accelerating web application development with django. International Research Journal on Advanced Engineering and Management (IRJAEM). 2024. Vol. 2, no. 06. P. 1902–1915. URL: https://doi.org/10.47392/irjaem.2024.0307
11. Models and Databases / Django Software Foundation. URL: https://docs.djangoproject.com/en/stable/topics/db/models/ (дата звернення: 14.05.2026).
12. Official Website / DB Browser for SQLite. URL: https://sqlitebrowser.org/ (дата звернення: 14.05.2026).
13. Python Documentation / Python Software Foundation. URL: https://docs.python.org/3/ (дата звернення: 14.05.2026).
14. Python Tutorial / W3Schools. URL: https://www.w3schools.com/python/ (дата звернення: 14.05.2026).
15. Rother K. Documentation. Pro Python Best Practices. Berkeley, CA, 2017. P. 245–259. URL: https://doi.org/10.1007/978-1-4842-2241-6_17
16. SQLite / K. P. Gaffney et al. Proceedings of the VLDB Endowment. 2022. Vol. 15, no. 12. P. 3535–3547. URL: https://doi.org/10.14778/3554821.3554842
17. SQLite Home Page / SQLite Documentation. URL: https://www.sqlite.org/docs.html (дата звернення: 14.05.2026).
18. Srivastava A. K. Django, the python web framework. International Journal of Scientific Research in Engineering and Management. 2022. Vol. 06, no. 05. URL: https://doi.org/10.55041/ijsrem13183
19. The user authentication system / Django Software Foundation. URL: https://docs.djangoproject.com/en/stable/topics/auth/ (дата звернення: 14.05.2026).
20. Uzayr S. B. The bootstrap framework. Bootstrap. Boca Raton, 2022. P. 19–70. URL: https://doi.org/10.1201/9781003309383-2

ДЕМОНСТРАЦІЙНІ МАТЕРІАЛИ (Презентація)

ДЕРЖАВНИЙ УНІВЕРСИТЕТ ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНИХ ТЕХНОЛОГІЙ

НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

КАФЕДРА ІНФОРМАЦІЙНИХ СИСТЕМ ТА ТЕХНОЛОГІЙ

«Система керування онлайн-замовленнями та логістикою мовою Python»

Виконав студент 4 курсу
групи ІСД-42
Гришко Карина Віталіївна
Керівник роботи: Викладач кафедри ІСТ
Ткаленко Оксана Миколаївна

Київ - 2025

МЕТА, ОБ’ЄКТ ТА ПРЕДМЕТ ДОСЛІДЖЕННЯ

Мета дослідження: розробка вебсистеми керування онлайн-замовленнями та логістикою доставки шляхом використання мови програмування Python і фреймворку Django для автоматизації процесів електронної комерції.

Об’єкт дослідження: процес автоматизації обробки онлайн-замовлень, керування товарами та організації логістики доставки продукції.

Предмет дослідження: вебсистема керування онлайн-замовленнями та логістикою доставки, реалізована засобами Python та Django.

ТЕХНІЧНІ ЗАВДАННЯ

1. Проаналізувати особливості функціонування систем онлайн-замовлень та логістики доставки.
2. Провести аналіз сучасних технологій веброзробки для створення систем електронної комерції.
3. Обрати програмні засоби та технології для реалізації вебсистеми.
4. Розробити вебзастосунок для керування товарами, оформлення замовлень та контролю доставки.
5. Спроєктувати структуру бази даних системи та реалізувати взаємодію з SQLite через Django ORM.
6. Реалізувати модулі користувачів, замовлень, кошика та логістики доставки.
7. Провести тестування вебзастосунку на коректність роботи та стабільність функціонування.

ФУНКЦІОНАЛЬНІ ВИМОГИ

Користувач повинен мати можливість:

1. пройти реєстрацію та авторизацію у системі;
2. переглядати каталог товарів та інформацію про продукцію;
3. додавати товари до кошика та змінювати їх кількість;
4. оформлювати онлайн-замовлення через вебінтерфейс;
5. переглядати статус оформленого замовлення;
6. отримувати інформацію про доставку товару;
7. виконувати пошук та фільтрацію товарів у каталозі.

Система повинна забезпечувати:

1. автоматичну обробку замовлень користувачів;
2. взаємодію з базою даних SQLite через Django ORM;
3. збереження інформації про користувачів, товари та замовлення;
4. оновлення статусів замовлення та доставки;
5. стабільну роботу вебзастосунку та коректне відображення даних.

НЕФУНКЦІОНАЛЬНІ ВИМОГИ

1. Продуктивність: вебсистема повинна забезпечувати швидку обробку запитів користувачів, оформлення замовлень та взаємодію з базою даних без значних затримок при середньому навантаженні.
2. Масштабованість: архітектура системи повинна підтримувати можливість подальшого розширення функціоналу, зокрема додавання нових модулів, способів оплати, логістичних сервісів та адміністрування системи.
3. Зручність користування: інтерфейс вебзастосунку повинен бути простим, зрозумілим та зручним для користувачів під час перегляду товарів, оформлення замовлень та керування доставкою.
4. Безпека: система повинна забезпечувати захист персональних даних користувачів, безпечну авторизацію та контроль доступу до функціональних модулів вебзастосунку.
5. Надійність: вебзастосунок повинен стабільно працювати та забезпечувати коректне збереження інформації про товари, користувачів і замовлення у базі даних SQLite.

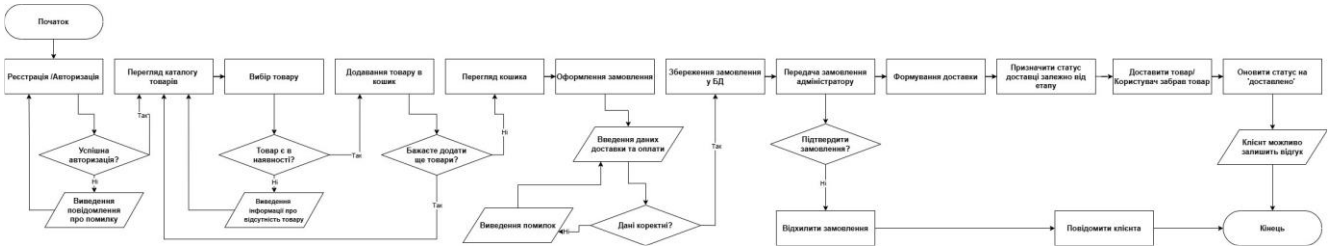
АНАЛОГИ

Назва системи	Основне призначення	Особливості	Підтримка логістики
Amazon	Глобальний маркетплейс	Велика кількість інтегрованих сервісів	Так
Shopify	Платформа для створення інтернет-магазинів	Гнучке налаштування магазину	Так
Prom.ua	Маркетплейс для малого та середнього бізнесу	Просте створення онлайн-магазину	Так
eBay	Система онлайн-аукціонів та продажів	Підтримка міжнародної торгівлі	Частково

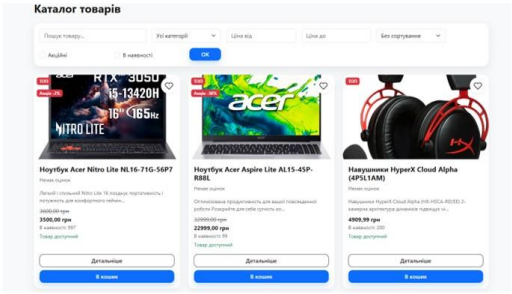
ПРОГРАМНІ ЗАСОБИ РЕАЛІЗАЦІЇ

Загальна структура системи онлайн-замовлень та логістики доставки

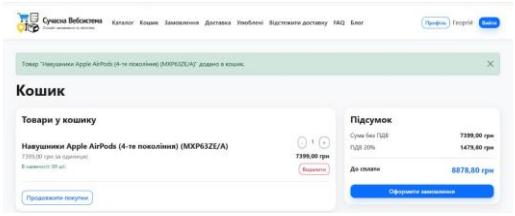
Блок-схема роботи вебсистеми керування онлайн-замовленнями та логістикою доставки



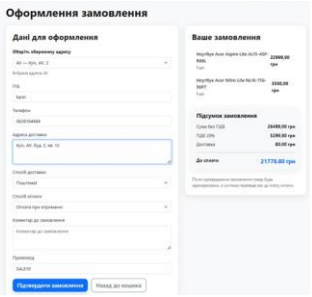
Інтерфейс системи



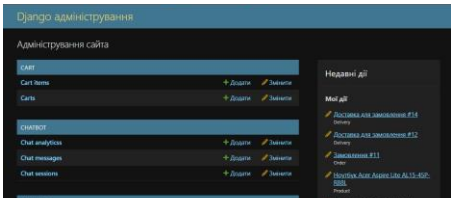
Вигляд сторінки головної в браузері



Вигляд кошика в браузері та повідомлення про додавання товару в кошик



Вигляд заповненого замовлення



Вигляд адміністрування сайта

ВИГЛЯД ДАНИХ В SQLite

Таблиця: delivery_deliverystatushistory

	id	old_status	new_status	changed_at	delivery_id
	Фі...	Фільтр	Фільтр	Фільтр	Фільтр
1	10	created	packed	2026-05-13 19:40:28.609509	11
2	11	packed	courier_assigned	2026-05-13 19:40:30.356695	11
3	12	courier_assigned	in_transit	2026-05-13 19:54:26.985226	11
4	13	in_transit	delivered	2026-05-13 19:54:43.232213	11
5	14	created	packed	2026-05-13 20:07:40.736198	10
6	15	packed	courier_assigned	2026-05-13 20:07:41.565605	10

Історія статусу доставки

Таблиця: products_favorite

	id	product_id	user_id
	Фі...	Фільтр	Фільтр
1	2	2	1
2	3	4	1

Улюблені товари користувачів

Таблиця: products_review

	id	text	created_at	product_id	user_id	rating
	Фі...	Фільтр	Фільтр	Фільтр	Фільтр	Фільтр
1	1	А де фото?	2026-04-19 16:49:56.533149	4	1	5
2	2	А?	2026-04-19 17:57:05.089961	4	1	4
3	3	Не прийшов товар	2026-04-19 17:57:15.135147	4	1	1
4	4	Вау	2026-04-19 17:57:21.826666	4	1	5
5	5	Потужний, для дитини, для навчання…	2026-05-15 21:24:43.132542	1	1	5

Таблиця відгуків

ВИСНОВКИ

1. Було проведено аналіз предметної області систем онлайн-замовлень та логістики доставки, що дозволило визначити основні вимоги до функціоналу вебзастосунку.
2. У ході роботи було досліджено сучасні технології веброзробки та обрано оптимальні засоби реалізації: Python, Django, HTML, CSS та SQLite.
3. Спроєктовано структуру бази даних і реалізовано вебсистему для керування товарами, оформлення замовлень та контролю доставки продукції.
4. Розроблено та протестовано вебзастосунок, який забезпечує автоматизацію процесів електронної комерції та відповідає функціональним і нефункціональним вимогам системи.

АПРОБАЦІЯ

Основні результати дипломної роботи були представлені в VII міжнародна науково-технічна конференція «Сучасний стан та перспективи розвитку iot» з тезою на тему «Система керування онлайн-замовленнями та логістикою мовою python»

ДЯКУЮ ЗА УВАГУ!