МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ХАРЧОВИХ ТЕХНОЛОГІЙ
Інститут (факультет) Автоматизації і комп'ютерних систем
Кафедра Інформаційних технологій, штучного інтелекту і кібербезпеки

«До захисту в ЕК»	**«До захисту допущено»**
Директор інституту(декан факультету)	Завідувач кафедри
_____ Андрій ФОРСЮК	_____ Сергій ГРИБКОВ
(підпис) (ім'я та прізвище)	(підпис) (ім'я та прізвище)
«2» червня 2025р.	«2» червня 2025р.

КВАЛІФІКАЦІЙНА РОБОТА
НА ЗДОБУТТЯ ОСВІТНЬОГО СТУПЕНЯ БАКАЛАВРА

зі спеціальності 122 «Комп'ютерні науки»
(код та назва спеціальності)

освітньо-професійної програми Інформаційні системи та штучний інтелект

на тему: «Розроблення мобільного застосунку для інтернет-магазину ItBox»

Виконав: здобувач 4 курсу, групи КН-4-4

Щербаков Артем Юрійович _____
(прізвище, ім'я по батькові повністю) (підпис)

Керівник Харкянен Олена Валеріївна _____
(прізвище , ім'я та по батькові повністю) (підпис)

Консультанти _____ _____
(ім'я та прізвище) (підпис)

_____ _____
(ім'я та прізвище) (підпис)

_____ _____
(ім'я та прізвище) (підпис)

Рецензент _____ _____
(ім'я та прізвище) (підпис)

Я як здобувач(ка) Національного університету харчових технологій розумію і підтримую політику університету з академічної доброчесності. Я не надавав(-ла) і не одержував(-ла) недозволеної допомоги під час підготовки цієї роботи. Використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело

Здобувач_____
(підпис)

Київ – 2025 р.

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ХАРЧОВИХ ТЕХНОЛОГІЙ

Інститут (факультет) Автоматизації і комп'ютерних систем
Кафедра Інформаційних технологій, штучного інтелекту і кібербезпеки
Освітній ступінь бакалавр
Спеціальність 122 «Комп'ютерні науки»
Освітньо-професійна програма Інформаційні системи та штучний інтелект

ЗАТВЕРДЖУЮ
Завідувач кафедри
Інформаційних технологій, штучного інтелекту і кібербезпеки
Сергій ГРИБКОВ

" 28 " квітня **2025 року**

З А В Д А Н Н Я

НА КВАЛІФІКАЦІЙНУ РОБОТУ ЗДОБУВАЧА
Щербаков Артем Юрійович
(прізвище, ім'я, по батькові)

1. Тема роботи **«Розроблення мобільного застосунку для інтернет-магазину ItBox»**
керівник роботи Харкянен Олена Валеріївна, к.т.н., доцент
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)
затверджені наказом закладу вищої освіти від 28 квітня 2025 року № 254-кс
2. Строк подання здобувачем роботи 30.05.2025 р.
3. Вихідні дані до роботи
4. Зміст пояснювальної записки (перелік питань, які потрібно розробити)
1. Дослідження предметної області та постановка задачі
2. Технічне завдання на проєктування
3. Розроблення мобільного додатку
5. Перелік графічного матеріалу
схема організаційної структури магазину ITBOX, модель бізнес-процесу оформлення замовлень в нотації BPMN, логічна модель бази даних, структура бази даних в СУБД MongoDB, фрагменти функціоналу у вигляді коду.

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
1	Харкянен Олена Валеріївна		
2	Харкянен Олена Валеріївна		
3	Харкянен Олена Валеріївна		

7. Дата видачі завдання 28.04.2025

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів виконання кваліфікаційної роботи	Строк виконання етапів роботи	Примітка
1	Постановка задачі. Визначення цілей, завдань та вимог до системи	28.04.2025 – 29.04.2025	Виконано
2	Аналіз предметної області. Огляд систем-аналогів.	29.04.2025– 10.05.2025	Виконано
4	Реалізація основних модулів системи та проєктування бази даних	11.05.2025– 20.05.2025	Виконано
5	Підготовка пояснювальної записки, презентації	21.05.2025– 25.05.2025	Виконано
6	Оформлення відгуку, рецензії, подання роботи	25.05.2025– 26.05.2025	Виконано

Здобувач _____ Щербаков А. Ю.
(підпис) (прізвище та ініціали)

Керівник роботи _____ Харкянен О. В.
(підпис) (прізвище та ініціали)

АНОТАЦІЯ

Кваліфікаційна робота на тему «Розробка мобільного застосунку для інтернет-магазину ItBox» складається з 3 розділів загальним обсягом 70 сторінок, містить 25 рисунків, 4 таблиць та 30 посилань на використані джерела.

Робота присвячена дослідженню предметної області електронної комерції в Україні, зокрема функціонування інтернет-магазину ItBox, та розробці мобільного застосунку для автоматизації процесів взаємодії з клієнтами. У роботі здійснено аналіз сучасного стану автоматизації в компанії, визначено недоліки поточної мобільної версії сайту, проаналізовано приклади застосунків основних конкурентів (Rozetka, Allo, Prom.ua тощо) та обґрунтовано доцільність створення власного рішення.

Розроблено повноцінний кросплатформенний мобільний застосунок із використанням React Native, PayloadCMS і MongoDB, що реалізує функції реєстрації, перегляду каталогу товарів, керування замовленнями та вподобаними товарами. Проведено техніко-економічне обґрунтування, яке підтвердило ефективність впровадження системи — очікуваний термін окупності складає 1,26 роки.

Ключові слова: МОБІЛЬНИЙ ЗАСТОСУНОК, ІНТЕРНЕТ-МАГАЗИН, ЕЛЕКТРОННА КОМЕРЦІЯ, ІНФОРМАЦІЙНА СИСТЕМА.

SUMARRY

The qualification thesis entitled *"Development of a Mobile Application for the ItBox Online Store"* consists of 3 chapters, with a total length of 70 pages. It includes 25 figures, 4 tables, and 30 references.

The work is dedicated to researching the e-commerce domain in Ukraine, specifically the operations of the ItBox online store, and developing a mobile application to automate customer interaction processes. The thesis analyzes the current state of automation within the company, identifies the limitations of the existing mobile web version, reviews competing solutions (Rozetka, Allo, Prom.ua), and justifies the development of a custom solution.

A full-featured cross-platform mobile application was developed using React Native, PayloadCMS, and MongoDB, implementing key functions such as user registration, product catalog browsing, order management, and wishlists. A techno-economic assessment was carried out, confirming the project’s viability with a projected payback period of 1.26 years.

Keywords: MOBILE APPLICATION, ONLINE STORE, E-COMMERCE, INFORMATION SYSTEM.

ЗМІСТ

ВСТУП

В умовах стрімкого розвитку електронної комерції особливого значення набуває створення зручних і функціональних мобільних застосунків для інтернет-магазинів. Сучасні користувачі очікують на швидкий доступ до сервісу з будь-якого мобільного пристрою, стабільну роботу застосунку та зрозумілий інтерфейс. Зважаючи на це розробка мобільного застосунку для онлайн-магазину ItBox є вчасною та доцільною, адже сприяє покращенню взаємодії між клієнтом і торговою платформою, а також підвищує ефективність електронних продажів.

Метою даної кваліфікаційної роботи є розроблення повноцінного мобільного застосунку для ItBox із використанням технологій React Native для клієнтської частини та Node.js — для серверної. Основний акцент зроблено на автоматизації таких процесів, як авторизація користувачів, перегляд товарного каталогу, формування замовлень та робота з базою даних.

У межах роботи здійснено аналіз предметної області, сформульовано функціональні вимоги до системи, підготовлено технічне завдання, спроєктовано архітектуру мобільного застосунку, реалізовано ключовий функціонал і проведено тестування.

Результатом проєктування і розробки є мобільний застосунок, готовий до впровадження в реальне середовище електронної торгівлі, що відповідає актуальним вимогам щодо продуктивності, зручності використання та технічної надійності.

РОЗДІЛ 1. ДОСЛІДЖЕННЯ ПРЕДМЕТНОЇ ОБЛАСТІ ТА ПОСТАНОВКА ЗАДАЧІ

1.1 Загальна характеристика компанії ItBox

ItBox — це інноваційний інтернет-магазин, який функціонує як підрозділ компанії Brain Computers. За понад 29 років присутності на українському ринку компанія заслужила репутацію надійного постачальника електроніки, побутової техніки, товарів для дому, спорту, туризму та дитячих потреб.

Однією з ключових переваг ItBox є розгалужена інфраструктура: компанія має регіональні склади, сервісні центри та пункти видачі у 23 областях України, що дозволяє швидко обробляти замовлення й забезпечувати високий рівень сервісу та гарантійного обслуговування.

Магазин активно підтримує програми лояльності для клієнтів, впроваджуючи бонусну систему ITbox-балів, нарахування відсотків на залишок балансу, а також забезпечує привабливі ціни на товари.

ItBox виступає офіційним дистриб’ютором і імпортером ряду міжнародних брендів, зокрема Exceleram, Apacer, MSI, ADATA, Doogee, а також реалізує продукцію під власною торговою маркою Vinga.

1.2 Організаційна структура компанії ItBox, роль і взаємодія підрозділів

Компанія ItBox має чітко сформовану організаційну структуру, до складу якої входять ключові функціональні підрозділи.

- логістичний відділ;
- відділ продажу;
- служба клієнтської підтримки;
- маркетинговий відділ;
- інтернет-магазин ItBox як підрозділ роздрібної торгівлі;
- ІТ-відділ;
- сервісний центр.

У межах цієї кваліфікаційної роботи автоматизації підлягає саме відділ роздрібної торгівлі. Його основні завдання включають.

– регулярне оновлення товарного каталогу;
– обробку замовлень, отриманих із вебсайту та мобільного застосунку;
– підтримку взаємодії з клієнтами через особистий кабінет;
– аналіз обсягів продажів та залишків товарів на складах.

На рисунку 1.1 наведено структуру компанії ItBox з виділенням підрозділу автоматизації, як об’єкта автоматизації.

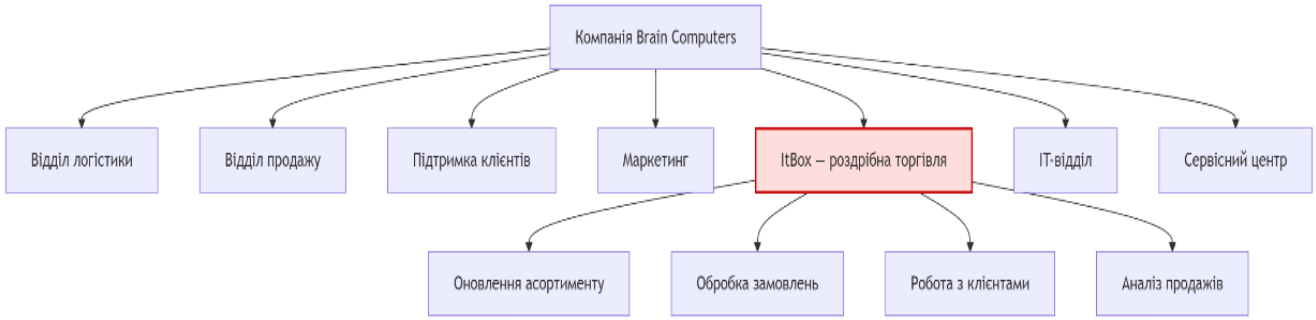


Рисунок 1.1 - Структура підприємства ItBox

1.3 Аналіз поточного стану автоматизації

На момент дослідження компанія ItBox уже має повноцінний веб-сайт із можливістю оформлення замовлень. Проте відсутній зручний мобільний застосунок, що негативно впливає на лояльність мобільних користувачів. Та в цілому обмежує конкурентоспроможність магазину на ринку продажів продукції.

Наявне рішення (мобільна версія сайту) не забезпечує.

- зручної навігації та швидкодії;
- персоналізації;
- офлайн-доступу до обраних товарів;
- push-сповіщення;
- повний доступ до програми лояльності (баланс балів тощо).

Для усунення наведених недоліків доцільно розроблення розроблення окремого мобільного застосунку, який дозволить підвищити якість

обслуговування, зменшити навантаження на відділ підтримки та стимулювати повторні покупки.

1.4 Моделювання та аналіз бізнес-процесів

1.4.1 Моделювання бізнес-процесів в нотації BPMN

На поточному етапі діяльність інтернет-магазину ItBox реалізується переважно через веб-інтерфейс. Основні взаємодії користувачів із системою — це перегляд товарів, оформлення замовлень, оплата та доступ до особистого кабінету — відбуваються виключно через вебсайт. Запити клієнтів обробляються за допомогою зовнішніх CRM-систем, а частина процесів — зокрема модерація замовлень і консультації — виконується вручну менеджерами.

Модель бізнес-процесе оформлення замовлення у нотації BPMN наведена на рисунку 1.2.

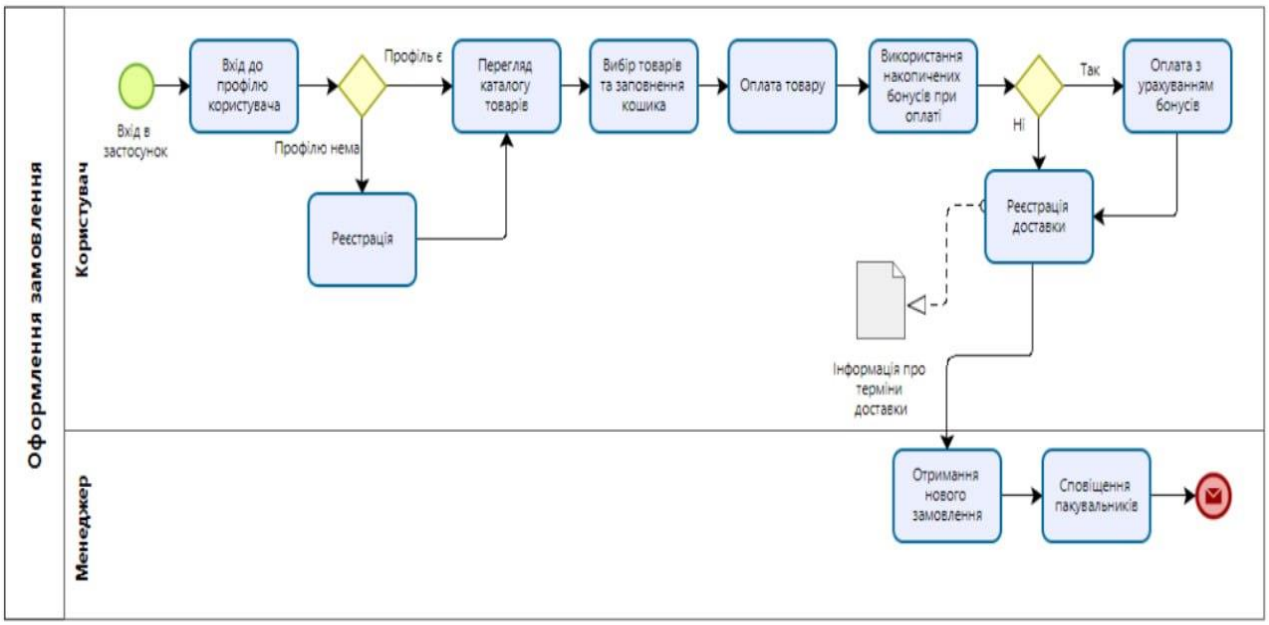


Рисунок 1.2 – Модель бізнес-процесу оформлення замовлення – статус AS - IS

Недоліки чинної моделі:

- низький рівень мобільної адаптації;
- відсутність персональних push-сповіщень;
- неможливість швидкої авторизації через мобільні сервіси;
- ускладнена навігація на мобільних пристроях.

1.4.2 Пропозиції щодо вдосконалення

Впровадження мобільного застосунку дозволить суттєво покращити зручність та ефективність взаємодії користувача з платформою ItBox.

Запропонована на рисунку 1.3 модель передбачає реалізацію таких функцій:

- оперативний доступ до товарного каталогу без потреби в браузері;
- можливість швидкої авторизації за допомогою біометричних даних або облікових записів Google/Facebook;
- push-сповіщення про акції, зміну статусу замовлень, накопичення балів тощо;
- перегляд балансу ITbox-балів у режимі реального часу;
- спрощений процес оформлення замовлень — усього за кілька кліків;
- повноцінна інтеграція з CRM-системою для автоматизації обробки замовлень та клієнтської підтримки.

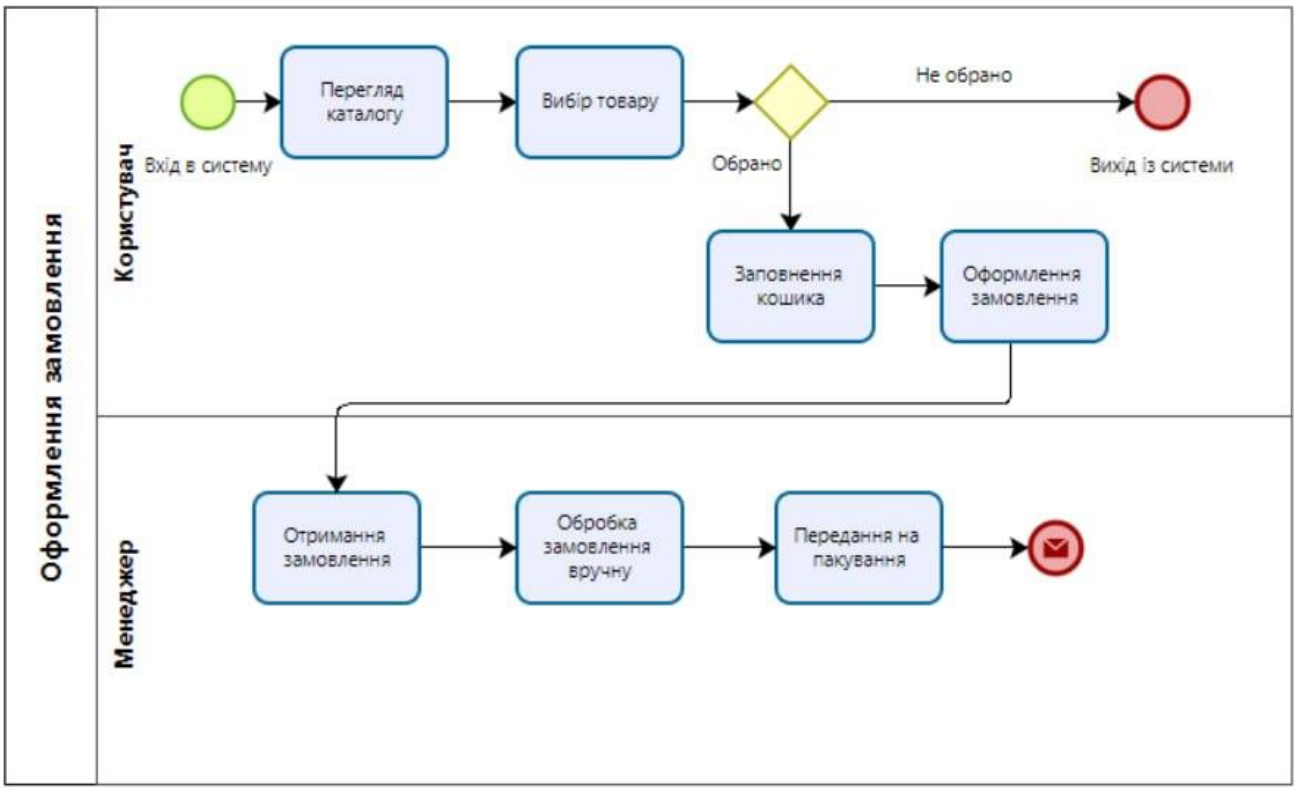


Рисунок 1.3 —Модель бізнес-процесу оформлення замовлення – статусу TO BE

1.5 Аналіз нинішнього стану комп’ютеризації компанії ItBox

Для вирішення виявлених проблем, пов’язаних із відсутністю мобільного каналу взаємодії з клієнтами, було проведено аналіз вітчизняного ринку програмних рішень у сфері електронної комерції. Особливу увагу приділено мобільним застосункам інтернет-магазинів, які дозволяють автоматизувати основні бізнес-процеси роздрібної торгівлі в онлайн-середовищі. У цьому підрозділі розглянуто найбільш поширені програмні аналоги, які частково або повністю задовольняють вимоги до подібних систем.

1.5.1. Мобільний застосунок Rozetka

Rozetka — один із лідерів українського ринку e-commerce, що пропонує повнофункціональний мобільний застосунок для Android та iOS. Він розроблений власними силами компанії з урахуванням високих вимог до навантаження, продуктивності та зручності користувача.

Функціональні можливості:

- персоналізований каталог товарів;
- власна система бонусів;
- оформлення замовлення в кілька кліків;
- push-сповіщення;
- онлайн оплата.

Переваги:

- висока швидкодія;
- надійна архітектура;
- стильний бренд та висока довіра.

Недоліки:

- висока вартість реалізації (орієнтовно $100 000+);
- замкнена система, не призначена для використання третіми сторонами.

1.5.2. Мобільний застосунок Allo

Allo — ще один великий гравець ринку, який має власний мобільний застосунок. У ньому реалізовано функції перегляду каталогу, оформлення замовлення, збереження обраних товарів та програми лояльності.

Функціональні можливості:

- каталог і фільтри;
- знижки, бонуси, акції;
- кабінет користувача;
- онлайн оплата.

Переваги:

- простий інтерфейс;
- швидке оформлення замовлення.

Недоліки:

- відсутність офлайн-функцій;
- менше персоналізації порівняно з Rozetka.

1.5.3. Мобільний застосунок Prom.ua

Prom.ua (від EVO) — маркетплейс із власним мобільним застосунком, який працює як агрегатор. Він дає змогу переглядати товари з різних магазинів, але не забезпечує повного брендування під один магазин.

Функціональні можливості:

- велика кількість магазинів в одному застосунку;
- стандартна корзина та замовлення;
- власна платіжна система;
- чат з продавцем.

Переваги:

- широкий вибір товарів;
- власна логістика.

Недоліки:

- не підходить для брендування;
- неможливість повного контролю над UX.

Проаналізувавши ринок мобільних застосунків інтернет-магазинів, можна зробити висновок, що конкуренти тривалий час використовують мобільні застосунки що вже є ринковим стандартом впровадження інформаційних

технологій для підтримки електронної комерції. Тому ItBox потребує аналогічного або більш досконалого функціоналу при впровадженні мобільного додатку.

1.5.4. Порівняння систем-аналогів

У таблиці 1.1 наведено порівняння основних інтернет-магазинів конкурентів за ключовими характеристиками, що враховувалися при розробленні власного застосунку ItBox.

Таблиця 1.1. *Порівняння основних інтернет-магазинів конкурентів*

Критерій	**Rozetka**	**Allo**	**Prom.ua**	**Власна розробка ItBox**
Вартість розробки	4 146 103₴+	2 902 272₴+	недоступна	165 844₴–207 305₴
Кастомізація інтерфейсу	+	+	–	+
Мобільна оптимізація	+	+	+	+
Підтримка push-сповіщень	+	+	–	+
Робота оф-лайн	–	–	–	частково
Власна бонусна система	+	+	–	+
Інтеграція з CRM/базами даних	+	+	–	+
Підтримка Apple Pay / GPay	+	+	+	+
Мова інтерфейсу (укр)	+	+	+	+
Контроль над даними користувачів	+	+	–	+

1.6 Техніко-економічне обґрунтування впровадження програмного забезпечення

Для підтвердження економічної ефективності розробки та впровадження мобільного застосунку ItBox було виконано розрахунок ключових показників доцільності реалізації проєкту.

Передпроєктне дослідження та аналіз вимог.

Трудомісткість (ТМ): 15 люд.-днів

Чисельність працюючих (ЧП): 3 особи

Тривалість (ТР): 5 днів

Витрати: 49 753₴ (зарплата персоналу) + 33 168₴ (маркетингові дослідження, консультації) = 82 922₴

Розробка технічного завдання.

Трудомісткість (ТМ): 12 люд.-днів

Чисельність працюючих (ЧП): 3 особи

Тривалість (ТР): 4 дні

Витрати: 74 629₴ (зарплата персоналу) + 29 022₴ (UI/UX платформа, інструменти для проектування) = 103 652₴

Проєктування архітектури та інтефейсу системи.

Трудомісткість (ТМ): 10 люд.-днів

Чисельність працюючих (ЧП): 2 осіб

Тривалість (ТР): 5 днів

Витрати: 456 071₴ (зарплата розробників) + 165844 (програмне забезпечення) = 621 915₴

Розробка та налаштування системи.

Трудомісткість (ТМ): 16 люд.-днів

Чисельність працюючих (ЧП): 2 особи

Тривалість (ТР): 8 днів

Витрати: 186 574₴ (зарплата тестувальників) + 62 191₴ (системи автоматизованого тестування)= 248 766₴

Інтеграція.

Трудомісткість (ТМ): 9 люд.-днів

Чисельність працюючих (ЧП): 3 особи

Тривалість (ТР): 3 дні

Витрати: 103 652₴ (зарплата) + 82 922₴ (інтеграція платіжних систем, налаштування серверів) = 186 574₴

Тестування та виправлення помилок.

Трудомісткість (ТМ): 20 люд.-днів

Чисельність працюючих (ЧП): 4 особи

Тривалість (ТР): 5 днів

Витрати: 58 045₴ (зарплата персоналу) + 37 314₴ (навчальні матеріали, відеоінструкції) = 95 360₴

Підготовка документації.

Трудомісткість (ТМ): 12 люд.-днів

Чисельність працюючих (ЧП): 4 особи

Тривалість (ТР): 3 дні

Витрати: 331 688₴ (зарплата команди підтримки) + 124 383₴ (резервне копіювання даних, моніторинг, регулярні оновлення) = 456 071₴

Загальні витрати = 82 922₴ + 103 652₴ + 621 915₴ + 248 766₴ + 186 574₴ + 95 360₴ + 456 071₴= 1 782 824₴

Очікуваний економічний ефект.

Зниження операційних витрат: Завдяки автоматизації процесів управління замовленнями та підтримки клієнтів очікується скорочення витрат на персонал та адміністративні послуги на 20%, що призведе до економії 331 668₴ на рік.

Збільшення обсягів продажу: Використання зручної та функціональної платформи дозволить обробляти до 25% більше замовлень завдяки покращенню взаємодії з клієнтами (швидший пошук товарів, ефективні рекомендації, нові маркетингові інструменти) і прогнозоване збільшення доходів на рік становить **621 915₴**.

Покращення обслуговування клієнтів.

Завдяки автоматизації клієнтської підтримки (використання чат-ботів, автоматичні сповіщення) очікується зменшення часу відповіді на клієнтські запити на 40% та річна економія ресурсів, пов'язана з підвищенням ефективності роботи персоналу, складе **207 305₴**.

Скорочення часу на обробку замовлень: Оптимізація логістики та процесів обробки замовлень дозволить знизити середній час обробки замовлення на 35% та річна економія за рахунок оптимізації становитиме **269 496₴**.

Загальний економічний ефект:

331 688₴ (зниження витрат) + 621 915₴ (збільшення доходів) + 331 668₴ (покращення обслуговування) + 269 496₴ (економія на часі) = 1 430 405₴ на рік.

Окупність проекту

Термін окупності проекту розраховується за формулою:

$$\text{Термін окупності} = \frac{\text{Витрати на проєкт}}{\text{Річний економічний ефект}} = \frac{1\ 795\ 262}{1\ 430\ 405} \approx 1{,}26 \text{ роки}$$

Отже, термін окупності складає приблизно 1,26 роки.

Термін окупності складає приблизно 1,26 роки, що означає, що за цей час витрати на проєкт будуть повністю покриті завдяки отриманому економічному ефекту. Після цього часу проєкт почне приносити чистий прибуток.

1.7. Обґрунтування доцільності проєктування й розроблення мобільного застосунку для інтернет-магазину ItBox

Проведений аналіз показав, що в компанії ItBox на сьогодні відсутнє зручне мобільне рішення, яке б дозволяло користувачам швидко оформлювати замовлення, переглядати каталог, слідкувати за балансом бонусів або отримувати актуальні сповіщення. Наявна мобільна версія сайту є лише частково адаптованою під смартфони і не забезпечує того рівня зручності, до якого звикли сучасні користувачі.

У процесі аналізу поточного стану автоматизації, наведеному у пункті 1.3, стало очевидно, що частина дій виконується вручну або через інтерфейс, який не призначений для мобільних пристроїв. Це створює зайві труднощі як для клієнтів, так і для працівників відділу роздрібної торгівлі.

У пункті 1.4.2 були розглянуті можливі зміни бізнес-процесів після впровадження застосунку. Зокрема, передбачається спрощення процесу оформлення замовлення, автоматизація комунікації з клієнтом за допомогою push-сповіщень, а також зменшення навантаження на менеджерів.

Також у пункті 1.5 було проаналізовано декілька прикладів мобільних застосунків конкурентів — Rozetka, Allo, Prom.ua. Вони вже давно пропонують своїм користувачам зручні застосунки, які допомагають економити час і стимулюють повторні покупки. У той же час, ці рішення не є відкритими або придатними для адаптації під потреби ItBox.

Власний застосунок дозволяє реалізувати саме той функціонал, який потрібен компанії та її клієнтам: простий інтерфейс, можливість швидкого замовлення, бонусна система, перегляд історії покупок, повідомлення про акції. До того ж, реалізація мобільного застосунку пропонується на сучасному технологічному стеку (React Native + Node.js), що забезпечить в майбутньому можливість легкого масштабування або зміни функціоналу без вкладення великих коштів.

Таким чином, розробка мобільного застосунку для інтернет-магазину ItBox є цілком обґрунтованим кроком. Це не просто про нову програму — це про кращу взаємодію з клієнтами, підвищення ефективності і можливість залишатися конкурентоспроможними на ринку.

РОЗДІЛ 2. ТЕХНІЧНЕ ЗАВДАННЯ НА ПРОЄКТУВАННЯ

2.1. Загальні положення

2.1.1. Найменування системи

«Мобільний додаток для інтернет-магазину ItBox»

2.1.2. Результати робіт

Результати робіт зі створення системи оформлюються згідно з вимогами ДСТУ на відповідні етапи розроблення. Порядок оформлення і передачі результатів визначається змістом і календарним планом виконання розробки.

2.1.3. Зміни та уточнення

У випадку необхідності на наступних стадіях робіт зі створення системи окремі положення можуть уточнюватися і розвиватися за погодженням із замовником.

2.2. Призначення і цілі створення системи

2.2.1. Призначення системи

Система призначена для автоматизації процесу онлайн-покупок в інтернет-магазині ItBox через мобільний додаток. Система надає користувачам зручний інтерфейс для перегляду каталогу товарів, здійснення замовлень, оплати, відстеження статусу доставки та інших супутніх операцій.

2.2.2. Цілі створення системи

Основною метою створення системи є забезпечення мобільного доступу до послуг інтернет-магазину ItBox у будь-який час та з будь-якого місця, що сприятиме розширенню клієнтської бази та збільшенню продажів.

Основні цілі створення системи:

- розширення каналів продажу товарів компанії ItBox;
- підвищення лояльності клієнтів за рахунок надання зручного мобільного інтерфейсу;
- збільшення конверсії продажів через персоналізацію пропозицій;
- оптимізація процесу здійснення покупок для користувачів;
- забезпечення безпечного та швидкого процесу оплати товарів;

- отримання аналітичної інформації про поведінку користувачів для подальшої оптимізації бізнес-процесів.

2.3. Характеристика об'єкта автоматизації

2.3.1. Короткі відомості про об'єкт автоматизації

Об'єктом автоматизації є процес взаємодії клієнтів з інтернет-магазином ItBox через мобільний додаток. Базовий об'єкт впровадження — компанія ItBox, що спеціалізується на продажу електроніки та аксесуарів.

Основні бізнес-процеси, що підлягають автоматизації:

- перегляд каталогу товарів та пошук за різними критеріями;
- оформлення замовлення та вибір способу доставки;
- оплата замовлення через інтегровані платіжні системи;
- відстеження статусу замовлення;
- управління профілем користувача;
- отримання повідомлень про акції, знижки та спеціальні пропозиції.

2.4. Вимоги до системи

2.4.1. Вимоги до системи в цілому

2.4.1.1. Вимоги до структури і функціонування системи

Архітектура системи

Система повинна мати клієнт-серверну архітектуру з використанням сучасних технологій:

- мобільний додаток (клієнт) — React Native;
- серверна частина —PayloadCMS;
- база даних — MongoDB.

Діагностування функціонування системи

Система повинна забезпечувати діагностику помилок та збоїв, надаючи користувачам відповідні сповіщення. При цьому повинна фіксуватися детальна технічна інформація про помилки для подальшого аналізу розробниками.

Можливості розвитку і модернізації

Структура і технологія програмного забезпечення системи повинні забезпечувати можливість масштабування, додавання нових функцій та адаптації до змін вимог бізнесу без необхідності повного переписування коду.

Режими функціонування

Функціонування системи має забезпечувати:

- онлайн-режим з повним функціоналом при наявності підключення до Інтернету;
- можливість синхронізації даних при відновленні підключення до Інтернету.

2.4.1.2. Вимоги до користувачів системи

Категорії користувачів

Система передбачає наступні категорії користувачів:

- незареєстровані користувачі — мають доступ до обмеженого функціоналу (перегляд каталогу товарів, пошук);
- зареєстровані користувачі — мають доступ до повного функціоналу (оформлення замовлень, відстеження доставки, історія покупок);
- адміністратори — мають доступ до адміністративної панелі (управління товарами, обробка замовлень, налаштування системи).

Вимоги до кваліфікації користувачів

Для роботи з мобільним додатком користувачі повинні:

- мати базові навички користування смартфоном;
- вміти встановлювати додатки з App Store або Google Play;
- мати базові навички роботи з інтернет-магазинами;
- розуміти принципи здійснення онлайн-платежів.

Показники призначення

Характеристики якості системи

Система повинна забезпечувати:

- швидкість завантаження головної сторінки додатку — не більше 3 секунд;

- швидкість переходу між сторінками додатку — не більше 1 секунди;
- швидкість пошуку товарів — не більше 2 секунд;
- кількість одночасних підключень до сервера — не менше 10 000;
- час відгуку сервера на запит — не більше 500 мс.

Показники надійності

Основними показниками надійності є:

- коефіцієнт доступності системи — не менше 99,5%;
- середній час відновлення працездатності системи — не більше 30 хвилин;
- ймовірність безвідмовної роботи — не менше 0,95;
- максимальний час простою системи — не більше 4 годин на місяць.

Забезпечення надійності

Для забезпечення надійності необхідно передбачити:

- резервне копіювання бази даних не рідше одного разу на добу;
- кешування даних на стороні клієнта для забезпечення роботи в офлайн-режимі;
- автоматичне відновлення сесії користувача після збоїв;
- моніторинг працездатності системи в режимі реального часу;
- механізми перевірки цілісності даних при їх передачі.

2.4.1.5. Вимоги до безпеки

Для забезпечення безпеки при експлуатації системи потрібно дотримуватись вимог ДСТУ: ДСТУ 2293-99, ДСТУ ISO 6309:2007, ДСТУ 12.0.230:2008, ДСТУ 7237:2011, ДСТУ 7238:2011, ДСТУ 7239:2011.

2.4.1.6. Вимоги з ергономіки та технічної естетики

Загальні ергономічні і естетичні вимоги до системи повинні відповідати держстандартам ДСТУ 8604:2015, ДСТУ 7298:2013. Інтерфейс мобільного додатку повинен відповідати сучасним тенденціям UI/UX дизайну та керівництвам з дизайну для iOS та Android.

2.4.1.7. Вимоги до експлуатації

Види обслуговування

Види обслуговування системи визначаються у відповідності з ДСТУ EN 13306:2019. Загальні вимоги з експлуатації, технічного обслуговування і ремонту повинні відповідати ДСТУ 3576-97.

Вимоги до технічних засобів

Для функціонування мобільного додатку необхідні.

- смартфон з операційною системою iOS не нижче версії 13.0 або Android не нижче версії 8.0;
- підключення до Інтернету (Wi-Fi або мобільний інтернет);
- не менше 100 МБ вільного місця на пристрої.

Для функціонування серверної частини необхідні.

- сервер з можливістю розгортання Node.js додатків;
- сервер бази даних MongoDB;
- доступ до мережі Інтернет зі статичною IP-адресою.

2.4.1.8. Вимоги до захисту інформації від несанкціонованого доступу

Для надійності збереження і доступу до інформації необхідно використовувати засоби захисту:

Автентифікація користувачів за допомогою:

логіна та пароля;

Шифрування даних:

- використання HTTPS для обміну даними між клієнтом та сервером;
- шифрування персональних даних у базі даних;
- шифрування платіжної інформації відповідно до стандарту PCI DSS.

Управління доступом:

- розмежування прав доступу до функцій системи;
- ведення журналу дій користувачів;
- автоматичний вихід з системи після певного періоду неактивності.

2.4.1.9. Вимоги щодо збереження інформації при аваріях

Засоби резервного збереження

Необхідно передбачити:

- щоденне інкрементальне резервне копіювання даних;
- щотижневе повне резервне копіювання даних;
- зберігання резервних копій протягом не менше 3 місяців;
- автоматичне відновлення даних після збоїв системи.

Розміщення резервних копій

Резервні копії повинні зберігатися на окремих фізичних серверах, розташованих у різних географічних локаціях для забезпечення максимальної надійності.

2.4.1.10. Вимоги щодо захисту від впливу зовнішніх діянь

Захист від електромагнітних полів

Електрична складова електромагнітного поля завад у приміщеннях не повинна перевищувати 0,3 В/м² в діапазоні частот від 0,15 до 300 МГц. Для захисту від впливу електромагнітних полів та індустріальних завад слід передбачити різноманітні екрани та фільтри.

Захист від шкідливих факторів

Засоби захисту від шкідливих факторів повинні відповідати вимогам ДБН В.2.2-9-2009.

2.4.1.11. Вимоги до патентної чистоти

Під час створення системи патентні дослідження не проводяться. Усі використані бібліотеки та компоненти повинні мати відкриту ліцензію, що дозволяє їх комерційне використання.

2.4.1.12. Вимоги щодо стандартизації і уніфікації

У системі кодування інформації необхідно використовувати UTF-8. Дати, час та інші метрики повинні відповідати міжнародним стандартам.

2.4.2. Вимоги до функцій системи

2.4.2.1. Перелік функцій із зазначенням вхідної та вихідної інформації

В таблиці 2.1 наведено перелік функцій із зазначенням вхідної та вихідної інформації.

Таблиця 2.1. *Перелік функцій, вхідної та вихідної інформації*

№ п/п	**Найменування функції**	**Вхідна інформація**	**Вихідна інформація**
1	Реєстрація та автентифікація користувачів	Дані користувача (ім'я, email, пароль)	Створений профіль користувача, токен автентифікації
2	Перегляд каталогу товарів	Параметри фільтрації та сортування	Список товарів, що відповідають заданим критеріям
3	Пошук товарів	Пошуковий запит, фільтри	Результати пошуку товарів
4	Управління кошиком покупок	Дії користувача (додавання, видалення, зміна кількості товарів)	Оновлений стан кошика
5	Оформлення замовлення	Дані про доставку, спосіб оплати	Створене замовлення, інформація для оплати
6	Оплата замовлення	Платіжні дані користувача	Статус оплати, чек
7	Відстеження статусу замовлення	Ідентифікатор замовлення	Інформація про поточний статус і місцезнаходження замовлення
8	Управління профілем користувача	Особисті дані, налаштування	Оновлений профіль
9	Перегляд історії замовлень	Запит на отримання історії	Список попередніх замовлень з деталями
10	Отримання push-повідомлень	Налаштування сповіщень	Повідомлення про акції, статус замовлення, тощо

2.4.3. Вимоги до видів забезпечення

2.4.3.1. Вимоги до математичного забезпечення

Система не вимагає спеціального математичного забезпечення для реалізації покладених на неї функцій. Достатньо можливостей обраних технологій (React Native, PayloadCMS, MongoDB).

2.4.3.2. Вимоги до інформаційного забезпечення

Організація інформаційного забезпечення

Інформаційне забезпечення системи повинно містити дані, достатні для виконання всіх покладених на систему функцій.

Основні сутності:

- користувачі;
- товари;
- категорії товарів;
- замовлення;
- статуси замовлень;
- способи доставки;
- способи оплати;
- відгуки та рейтинги.

Захист даних

Слід передбачити захист даних від руйнування при аваріях і порушеннях у енергоживленні системи — використання резервних копій БД та тимчасове зберігання даних на стороні клієнта.

2.4.3.3. Вимоги до лінгвістичного забезпечення

Мови розроблення

Для розроблення програмних засобів повинні використовуватися:

- JavaScript/TypeScript для клієнтської та серверної частин;
- SQL/NoSQL для роботи з базою даних;
- HTML/CSS для веб-інтерфейсу адміністративної панелі.

Організація діалогу

Організація діалогу користувача з системою має будуватися на інтуїтивно зрозумілому інтерфейсі з використанням стандартних компонентів мобільних додатків (кнопки, меню, форми введення тощо).

2.4.3.4. Вимоги до програмного забезпечення

Загальносистемне програмне забезпечення

До загальносистемного ПЗ належить:

- операційні системи клієнтських пристроїв — iOS, Android;
- серверна операційна система — Linux Ubuntu 20.04 LTS або новіша;
- система управління БД — MongoDB;
- серверна платформа — Node.js.

Загальні вимоги до системного ПЗ

- мінімальні вимоги до ресурсів технічних засобів;
- максимальна швидкодія;
- надійність та безпека;
- повне задоволення потреб функціональних завдань системи.

Вимоги до технологій розробки

- клієнтська частина: React Native, Redux/MobX для управління станом додатку;
- серверна частина: PayloadCMS на базі Node.js;
- база даних: MongoDB;
- API: REST або GraphQL;
- автоматизація процесів: CI/CD з використанням GitHub Actions або Jenkins.

Вимоги до СУБД

- надійність зберігання даних;
- ефективне управління необхідним обсягом і структурою даних;
- швидкість виконання запитів;
- підтримка транзакцій;

- масштабованість.

Програмні засоби взаємодії з користувачем

- інтуїтивно зрозумілий інтерфейс;
- контроль введення даних з відображенням повідомлень про помилки;
- адаптивний дизайн для різних розмірів екранів;
- підтримка жестів (свайп, пінч, тощо);
- багатомовний інтерфейс.

Вимоги до розробки спеціального ПЗ

- модульна структура програм;
- використання сучасних патернів проєктування;
- відповідність UI стандартам iOS та Android;
- можливість розширення функціоналу;
- оптимізація для роботи на пристроях з різною продуктивністю.

2.4.3.5. Вимоги до технічного забезпечення

Вимоги до обладнання наведені у таблиці 2.2.

Таблиця 2.2. *Вимоги до технічного забезпечення системи*

№ п/п	**Основні характеристики**
Технічне забезпечення для сервера	
1	CPU: не менше 4 ядер, частота не менше 2.5 GHz
2	RAM: не менше 8 GB
3	HDD/SSD: не менше 100 GB
4	Підключення до мережі: 1 Gbps
Технічне забезпечення для розробки	
1	CPU: не менше 4 ядер, частота не менше 2.3 GHz
2	RAM: не менше 16 GB

№ п/п	Основні характеристики
3	SSD: не менше 256 GB
4	Дисплей: не менше 15"
Вимоги до клієнтських пристроїв	
1	iOS: iPhone 6s або новіше, iOS 13.0+
2	Android: API level 26 (Android 8.0) або вище
3	RAM: не менше 2 GB
4	Вільне місце: не менше 100 MB

Обмін інформацією

Засоби обчислювальної техніки повинні забезпечувати обмін інформації в обсягах, зазначених у вимогах до інформаційного забезпечення.

2.4.3.6. Вимоги до метрологічного забезпечення

Система не має вимірювальних каналів, вимірювального обладнання і приладів, тому вимоги до цього виду забезпечення не висуваються.

2.4.3.7. Вимоги до організаційного забезпечення

Стандарти організаційного забезпечення

Організаційне забезпечення системи розробляється відповідно до вимог державного стандарту по АСУП.

Кадрове забезпечення

При впровадженні системи не передбачається збільшення штатної чисельності підприємства. Для роботи з системою має бути призначений відповідальний співробітник.

Вимоги до функціонування системи

- наказом керівника визначається список співробітників, які мають доступ до системи;
- контроль і прийняття рішень при аварійних ситуаціях здійснює відповідальний за систему;
- регулярне навчання співробітників роботі з системою;
- дотримання правил безпеки та конфіденційності інформації.

2.5. Склад і зміст робіт по створенню системи

2.5.1. Стадії створення системи і терміни виконання робіт

Стадії створення системи і терміни виконання робіт наведені у таблиці 2.3.

Таблиця 2.3. *Найменування робіт при створенні системи*

№ п/п	Найменування робіт	Строки виконання робіт
1	Передпроєктне дослідження та аналіз вимог	28.04.2025 - 02.05.2025
2	Розробка технічного завдання	03.05.2025 - 06.05.2025
3	Проєктування архітектури ті інтерфейсу системи	07.05.2025 - 11.05.2025
4	Розробка та налаштування системи	12.05.2025 - 19.05.2025
5	Інтеграція	20.05.2025 - 22.05.2025
6	Тестування та виправлення помилок	23.05.2025 - 27.05.2025
7	Підготовка документації	28.05.2025 - 30.05.2025

2.5.2. Діаграма Ганта

Далі на буде наведено діаграму Ганта, яка була створена на основі термінів зазначених у таблиці 2.3.

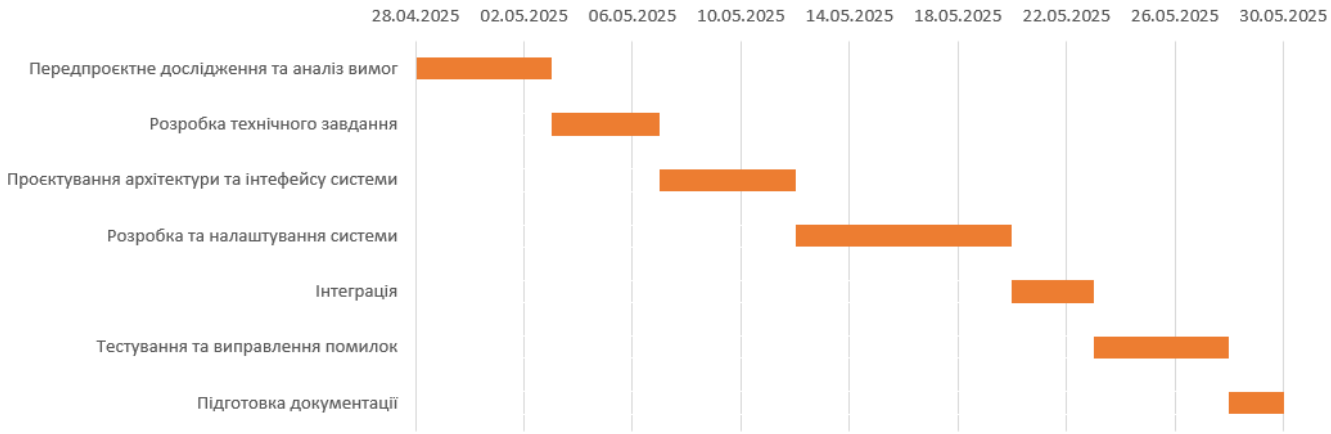


Рисунок 2.4 – Діаграма Ганта

2.6. Порядок контролю і приймання системи

2.6.1. Загальні умови приймання

Система вводиться на діючому підприємстві ItBox. При введенні в дію система повинна пройти приймальні випробування згідно з ДСТУ 3974-2000.

2.6.2. Випробування системи

Випробування для визначення працездатності і рішення про можливість приймання системи в дослідну експлуатацію проводять розробники разом із замовником. Програму випробувань складає розробник і затверджує замовник.

2.6.3. Дослідна експлуатація

Здача в дослідну експлуатацію здійснюється на основі технічного завдання та інструкції користувача. За результатами дослідної експлуатації формується перелік доробок і рекомендовані строки їх виконання.

2.6.4. Оформлення результатів

Введення в дію системи оформлюється актом здачі-прийому, який підписується представниками замовника та розробника.

2.7. Вимоги до складу і змісту робіт із підготовки до введення системи в дію

Для успішного введення мобільного додатку для інтернет-магазину ItBox в дію замовник виконує ряд підготовчих робіт:

- проводить укомплектування технічних засобів згідно з вимогами до технічного забезпечення;
- забезпечує доступ до мережі Інтернет з достатньою пропускною здатністю;
- забезпечує доступ до серверних потужностей для розгортання системи;
- організовує навчання персоналу роботі з адміністративною панеллю та основними функціями системи;
- призначає відповідальних осіб за підтримку та експлуатацію системи;
- забезпечує інтеграцію із внутрішніми системами обліку та управління товарами;

- проводить підготовку маркетингових матеріалів для просування мобільного додатку серед клієнтів;
- Організовує тестову групу користувачів для бета-тестування додатку;
- Розробляє план комунікацій для інформування клієнтів про новий канал продажів;
- Проводить дослідну експлуатацію і вводить систему в дію на основі результатів тестування.

2.8. Вимоги до документації

2.8.1. Перелік документації

На систему розробляється комплекс документації у складі:

1. Технічне завдання на розробку мобільного додатку;
2. Технічний проект, що включає:
    - пояснювальну записку;
    - опис архітектури системи;
    - модель даних;
    - керівництво користувача;
    - керівництво адміністратора.
1. Програма та методика випробувань;
2. Акт приймально-здавальних випробувань;
3. Експлуатаційна документація:
    - інструкція з установки мобільного додатку;
    - інструкція з налаштування серверної частини;
    - інструкція з адміністрування системи.

2.8.2. Вимоги до оформлення документації

Документація на систему розробляється у відповідності з вимогами Державних стандартів:

- ДСТУ 3008-2015. Документація. Звіти у сфері науки і техніки. Структура та правила оформлювання;

- ДСТУ 3974-2000 Система розроблення та поставлення продукції на виробництво. Правила виконання дослідно-конструкторських робіт;
- ДСТУ ISO/IEC 25051:2016 Інженерія програмного забезпечення. Вимоги до якості готового до використання програмного продукту (RUSP) і інструкції щодо тестування;
- ДСТУ ISO/IEC/IEEE 26511:2016 Системна та програмна інженерія. Вимоги до менеджерів інформації користувача;
- ДСТУ ISO/IEC/IEEE 26512:2016 Системна та програмна інженерія. Вимоги до укладачів інформації користувача.

2.8.3. Порядок внесення змін до документації

Внесення змін до документації здійснюється у відповідності з ДСТУ ГОСТ 2.503:2013 Єдина система конструкторської документації. Правила внесення змін.

2.9. Джерела розробки

2.9.1. Нормативні документи

При розробленні технічного завдання на систему використано наступні документи:

- ДСТУ 3008-2015. Документація. Звіти у сфері науки і техніки. Структура та правила оформлювання;
- ДСТУ 3973–2000 Система розроблення та поставлення продукції на виробництво;
- ДСТУ Б В.2.5–82:2016 Електробезпека в будівлях і спорудах. Вимоги до захисних заходів від ураження електричним струмом;
- ДСТУ ISO/IEC 27001:2015 Інформаційні технології. Методи захисту. Системи управління інформаційною безпекою. Вимоги;
- ДСТУ ISO/IEC TR 29110-5-1-2:2015 Інженерія програмного забезпечення. Профілі життєвого циклу для малих організацій;
- ДСТУ ISO/IEC/IEEE 29119-1:2015 Тестування програмного забезпечення. Частина 1. Поняття і визначення;
- ДСТУ ISO 9241-210:2019 Ергономіка взаємодії людина-система. Частина 210. Людиноцентричне проєктування інтерактивних систем.

РОЗДІЛ 3. РОЗРОБЛЕННЯ МОБІЛЬНОГО ДОДАТКУ

3.1. Опис та обґрунтування вибору програмно-технічних засобів розроблення програмного продукту

Для розробки мобільного додатку інтернет-магазину ItBox було обрано сучасний технологічний стек, який забезпечує ефективну розробку, хорошу продуктивність та підтримку кросплатформенності.

Основа розробки:

- React Native та Expo - основний фреймворк для розробки мобільного додатку, що дозволяє створювати нативні мобільні додатки з використанням JavaScript і React. Expo забезпечує набір готових інструментів і сервісів для швидшої розробки;
- TypeScript - статично типізована надбудова над JavaScript, яка забезпечує безпеку типів, що є критичним для підтримки якості коду в масштабних проєктах;
- Expo Router - система навігації, яка використовується для маршрутизації в додатку, що дозволяє створювати інтуїтивну навігацію між екранами.

Інтерфейс та UI-компоненти:

- React Native компоненти - використовуються базові компоненти React Native для створення інтерфейсу користувача;
- компоненти Expo - використовуються для роботи з системними можливостями, такими як шрифти, сховища та зображення;
- власні UI-компоненти - для забезпечення єдиного стилю та UX у додатку розроблені власні компоненти, такі як Button, Card та інші.

Управління станом та взаємодія з сервером:

- React Context API - використовується для глобального управління станом додатку та передачі даних між компонентами без необхідності передавання пропсів через проміжні компоненти;

- Axios - HTTP-клієнт для взаємодії з бекенд-сервером, що дозволяє здійснювати різні типи запитів (GET, POST, PUT, DELETE);
- AsyncStorage і SecureStore - використовуються для зберігання локальних даних і токену авторизації відповідно.

Обґрунтування вибору технологій:

1. React Native + Expo було обрано через:
    - можливість розробки під iOS та Android з єдиною кодовою базою;
    - швидкий цикл розробки завдяки гарячому перезавантаженню;
    - велика спільнота і значна кількість бібліотек;
    - експо спрощує процес розробки, тестування і публікації додатку.
2. TypeScript забезпечує:
    - раннє виявлення помилок під час розробки;
    - кращу документацію коду та автодоповнення в IDE;
    - безпечніше рефакторування коду;
    - кращу масштабованість проєкту.
3. React Context API обрано для управління станом через:
    - простоту використання порівняно з Redux;
    - достатню потужність для потреб додатку;
    - вбудованість у React без зовнішніх залежностей.

3.2. Проєктування та створення бази даних

Логічну модель бази даних наведено на рисунку 3.5.

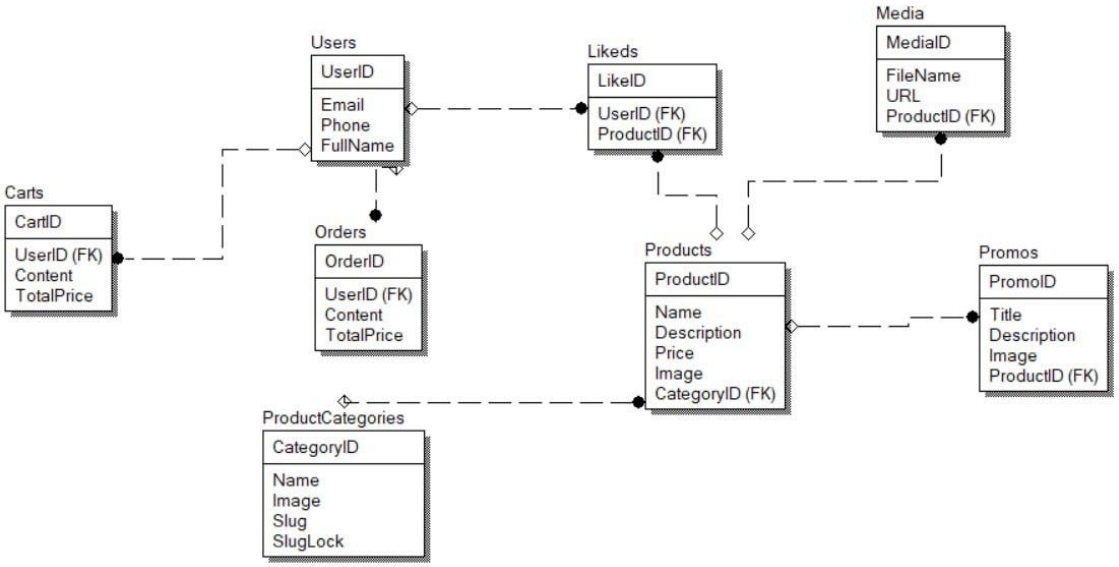


Рисунок 3.5 – Логічна модель бази даних

MongoDB — це документоорієнтована NoSQL СУБД, яка зберігає дані у форматі BSON (розширення JSON).

Основні відомості:

- тип: NoSQL, документоорієнтована;
- мова запитів: власна, подібна до JSON;
- структура даних: колекції та документи замість таблиць і рядків;
- гнучкість: не потребує жорсткої схеми;
- масштабованість: легко масштабувати горизонтально (sharding);
- Швидкодія: висока продуктивність при великих обсягах неструктурованих даних;
- Застосування: мобільні додатки, аналітика, IoT, швидкі вебсервіси.

Переваги:

- гнучка структура даних;
- легко оновлювати модель без міграцій;
- добре інтегрується з JavaScript/Node.js;
- вбудована реплікація і відновлення після збоїв.

Спроектуємо структуру бази даних мобільного додатку, що складається з наведених нижче сутностей.

1. Users (Користувачі) - зберігає інформацію про зареєстрованих користувачів:
    - id: унікальний ідентифікатор;
    - email: електронна пошта користувача;
    - password: пароль (зберігається захищено);
    - fullName: повне ім'я;
    - phone: номер телефону;
    - інші поля для автентифікації та безпеки.
2. Products (Товари) - каталог товарів магазину:
    - id: унікальний ідентифікатор;
    - name: назва товару;
    - description: опис товару;
    - price: ціна;
    - image: основне зображення;
    - gallery: галерея зображень;
    - discount: знижка (якщо є);
    - oldPrice: стара ціна (до знижки);
    - rating: рейтинг товару;
    - category: категорія (зв'язок з таблицею категорій);
    - features: характеристики товару;
    - slug: URL-сумісна назва для посилань.
3. ProductCategories (Категорії товарів):
    - id: унікальний ідентифікатор;
    - name: назва категорії;
    - image: зображення категорії;
    - slug: URL-сумісна назва для посилань.
4. Cart (Кошик) - зберігає товари у кошику користувача:

- id: унікальний ідентифікатор;
- user: зв'язок з користувачем;
- content: масив елементів кошика (товар і кількість);
- totalPrice: загальна вартість.

5. Order (Замовлення) - оформлені замовлення:
    - id: унікальний ідентифікатор;
    - user: зв'язок з користувачем;
    - content: товари в замовленні;
    - totalPrice: загальна вартість;
    - status: статус замовлення (очікування, завершено, в дорозі, очікує отримання);
    - delivery: спосіб доставки (Нова пошта, Укрпошта, самовивіз);
    - payment: спосіб оплати (картка, готівка);
    - fullName, address, phone: контактні дані для доставки.
6. Liked (Вподобані товари) - товари, додані користувачем до списку вподобаних:
    - id: унікальний ідентифікатор;
    - user: зв'язок з користувачем;
    - products: список вподобаних товарів.
7. Media (Медіафайли) - зберігає інформацію про зображення:
    - id: унікальний ідентифікатор;
    - url: посилання на файл;
    - thumbnailURL: посилання на мініатюру;
    - filename: ім'я файлу;
    - mimeType: тип файлу;
    - filesize: розмір файлу;
    - width, height: розміри зображення.
8. Promo (Акції) - інформація про акційні пропозиції:
    - id: унікальний ідентифікатор;

- title: заголовок акції;
- description: опис акції;
- image: зображення акції.

Зв'язки між таблицями:

- User → Cart: один до одного (у користувача один кошик);
- User → Orders: один до багатьох (користувач має багато замовлень);
- User → Liked: один до одного (один список вподобаних);
- Product → Category: багато до одного (товари належать категоріям);
- Product → Cart/Order: багато до багатьох через проміжні таблиці.

Структуру бази даних у СУБД MongoDB наведено у Додатку Г.

Взаємодія з базою даних відбувається через RESTful API, розташований за адресою **https://itbox.veiag.dev/api.**

3.3. Реалізація функцій системи

3.3.1. Авторизація та реєстрація користувачів

Для забезпечення персоналізованої роботи з додатком реалізовано функції авторизації та реєстрації. Цей функціонал реалізовано в хуку **useAuth**, який надає наступні можливості:

1. реєстрація нових користувачів - функція приймає email, пароль, ім'я та телефон і відправляє їх на сервер:
   register: (email: string, password: string, fullName: string, phone: string) => Promise<void>
2. автентифікація - функція логіну перевіряє облікові дані та отримує токен:
   login: (email: string, password: string) => Promise<void>
3. вихід із системи - виконує вихід і видаляє збережені дані авторизації:
   logout: () => Promise<void>
4. збереження стану авторизації - використовується SecureStore для збереження токену між сесіями.

3.3.2. Управління кошиком покупок

Функціонал кошика реалізовано в компоненті **CartContext** і надає наступні можливості:

- додавання товарів у кошик;
- синхронізація даних кошика;
- робота з каталогом товарів;
- список вподобаних товарів;

Програмний код функцій взаємодії з мобільним застосунком наведено у Додатку Д

3.3.3. Пошук товарів

Для швидкого пошуку необхідних товарів реалізовано функціонал пошуку:

Функція пошуку:

```
const fetchSearchResults = async (page = currentPage) => {
  setLoading(true);
  try {
    const queryParams = {
      limit: limitPerPage,
      page: page,
      sort: getSortParam(),
      where: buildWhereClause()
    };
    const queryString = stringify(queryParams);
    const response = await fetch(`${PAYLOAD_API_URL}/products?${queryString}`,
{
      method: 'GET',
      headers: {
        'Content-Type': 'application/json',
      },
    });
    if (response.ok) {
      const data = await response.json();
```

```
        setSearchResults(data.docs || []);
        setPaginationInfo({
          hasNextPage: data.hasNextPage,
          hasPrevPage: data.hasPrevPage,
          limit: data.limit,
          nextPage: data.nextPage,
          page: data.page,
          pagingCounter: data.pagingCounter,
          prevPage: data.prevPage,
          totalDocs: data.totalDocs,
          totalPages: data.totalPages
        });
        setCurrentPage(data.page);
      } else {
        console.error('Error fetching search results');
      }
    } catch (error) {
      console.error('Search fetch error:', error);
    } finally {
      setLoading(false);
    }
  };
```

Функція генерування параметрів сортування пошуку:

```
const getSortParam = () => {
    switch (sortBy) {
      case 'price_asc':
        return 'price';
      case 'price_desc':
        return '-price';
```

```
      case 'newest':
        return '-createdAt';
      case 'relevance':
        return '-createdAt';
      default:
        return '-createdAt';
    }
  };
```

Функція генерування параметрів запиту

```
const buildWhereClause = () => {
    const whereClause: Record<string, any> = {};
    if (query.trim()) {
      whereClause.name = {
        contains: query.trim(),
      };
    }
    if (filters.category) {
      whereClause['category.slug'] = {
        equals: filters.category,
      };
    }
    if (filters.priceMin && !isNaN(Number(filters.priceMin))) {
      whereClause.price = {
        ...(whereClause.price || {}),
        greater_than_equal: Number(filters.priceMin),
      };
    }
    if (filters.priceMax && !isNaN(Number(filters.priceMax))) {
      whereClause.price = {
        ...(whereClause.price || {}),
```

```
      less_than_equal: Number(filters.priceMax),
    };
  }
  if (filters.hasDiscount) {
    whereClause.discount = {
      not_equals: 0,
    };
  }
  return whereClause;
};
```

3.3.4. Інтеграція з API

Взаємодія з бекендом відбувається через RESTful API. Основна URL API:

```
export const PAYLOAD_API_URL = 'https://itbox.veiag.dev/api';
export const AUTH_TOKEN_KEY = 'payloadAuthToken';

const URL = "https://itbox.veiag.dev";
export const getImageURL = (url:string)=>{
  return URL+url;
}
```

Уся взаємодія з API побудована на стандартній функції Javascript – Fetch , тобто звичайними запитами (GET для отримання данних , POST для створення даних , PATCH – для оновлення даних , DELETE – для видалення даних) .

Особливістю PayloadCMS з підключеною базою даних MongoDB , є те , що він має дуже потужну систему RESTful API , за допомогою якої можна виконувати будь-які операції з базою даних . Наприклад , програмний код для отримання останніх 2 замовлень авторизованого користувача :

```
const fetchOrders = async () => {
    try {
      const query = stringify({
```

```
      depth:2,
      limit: 2,
      where: {
       'user': {
        'equals': user?.id
       }
      },
      sort: '-createdAt',
     })
     const response = await fetch(`${PAYLOAD_API_URL}/order?${query}`, {
      method: 'GET',
     });
     if (!response.ok) {
      throw new Error('Network response was not ok');
     }
     const data = await response.json();
     setLastOrders(data.docs);
     console.log('Fetched orders:', data.docs);
    } catch (error) {
     console.error('Error fetching orders:', error);
    }
```

Код отримання схожих товарів для сторінки товару (товари з тієї ж категорії):

```
const fetchSimilarProducts = async () => {
    try {
     const query = stringify({
      limit: 5,
      page: 1,
      where: {
       category: {
```

```
            equals: typeof product?.category === 'string' ? product?.category :
product?.category.id
          },
          id: {
            not_equals: product?.id
          }
        }
      })
      const res = await fetch(`${PAYLOAD_API_URL}/products?${query}`, {
        method: 'GET',
        headers: {
          'Content-Type': 'application/json',
        },
      });
      if (res.ok) {
        const data = await res.json();
        setSimilarProducts(data.docs);
      }
    } catch (error) {
      console.error("Error fetching similar products:", error)
    }
  }
  if(product) {
    fetchSimilarProducts()
  }
```

Код отримання 4 товарів з кожної категорії для головної сторінки:

```
  const fetchProducts = async () => {
        const productsByCategory: { [key: string]: Product[] } = {};
        for (const category of categoriesToShow) {
          const query = stringify({
```

```
            limit: 4,
            page: 1,
            where: {
              "category.slug": {
                equals: category,
              },
            },
          });
          const res = await fetch(
            `${PAYLOAD_API_URL}/products?${query}`,
            {
              method: "GET",
              headers: {
                "Content-Type": "application/json",
              },
            }
          );
          if (res.ok) {
            const data = await res.json();
            productsByCategory[category] = data?.docs;
          }
        }
        setProducts(productsByCategory);
      };
      fetchProducts();
    }, []);
```

3.4. Інструкція користувача

Реєстрація та вхід

1. Запуск додатку (див. рис. 3.6): Після встановлення додатку на свій пристрій, запустіть його, натиснувши на іконку ItBox.

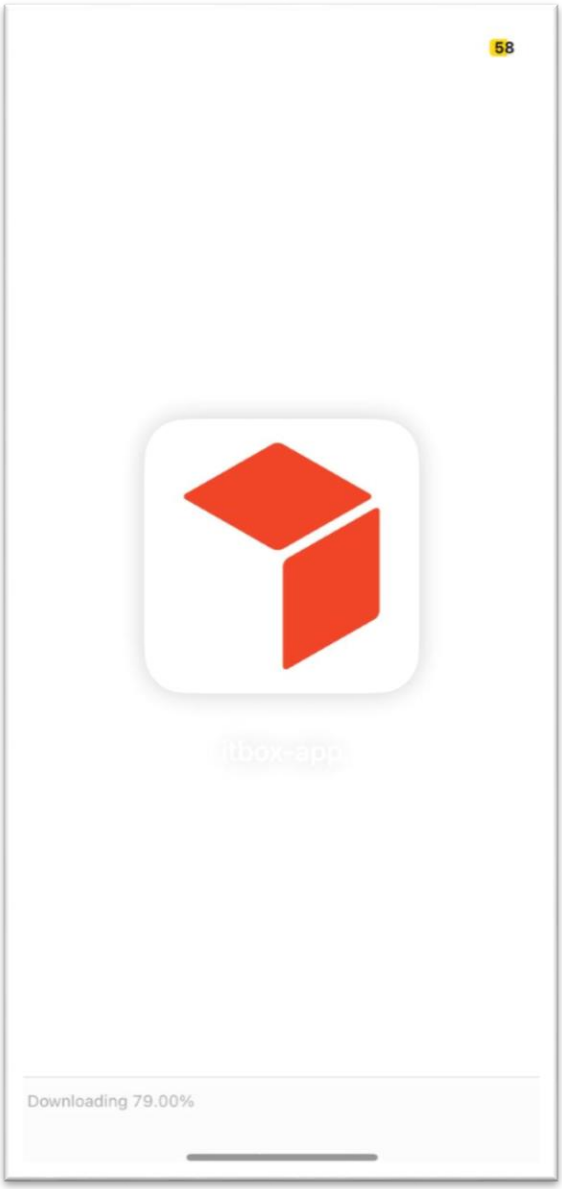


Рисунок 3.6 – Запуск додатку

2. Реєстрація нового облікового запису (див. рис. 3.7):
    - на екрані входу натисніть кнопку "Зареєструватися";
    - заповніть форму реєстрації: електронна пошта, пароль, повне ім'я та номер телефону;
    - натисніть кнопку "Зареєструватися";

- після успішної реєстрації виконайте вхід, використовуючи свої облікові дані.

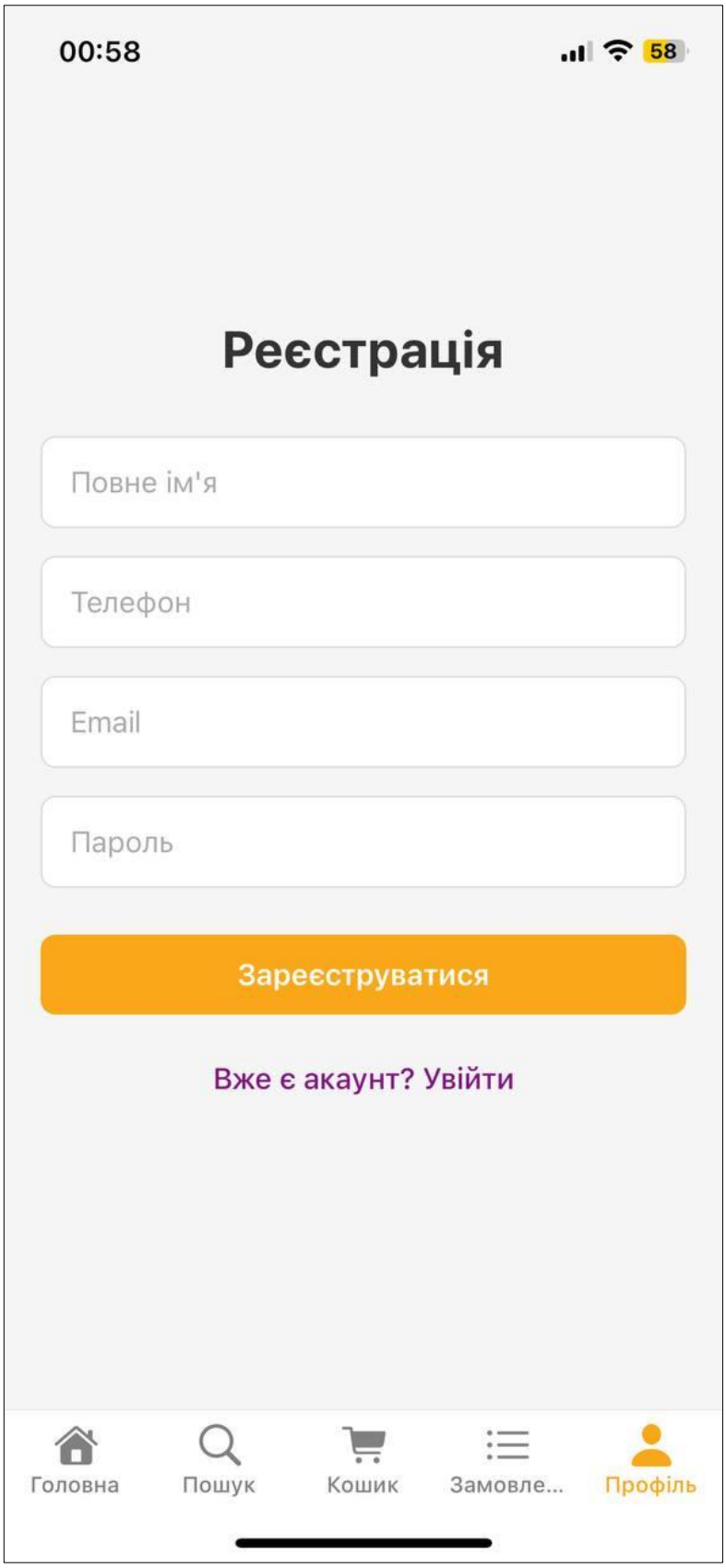


Рисунок 3.7 – Сторінка реєстрація

3. Вхід в існуючий обліковий запис (див. рис. 3.8):
    - на екрані входу введіть електронну пошту та пароль;
    - натисніть кнопку "Увійти".

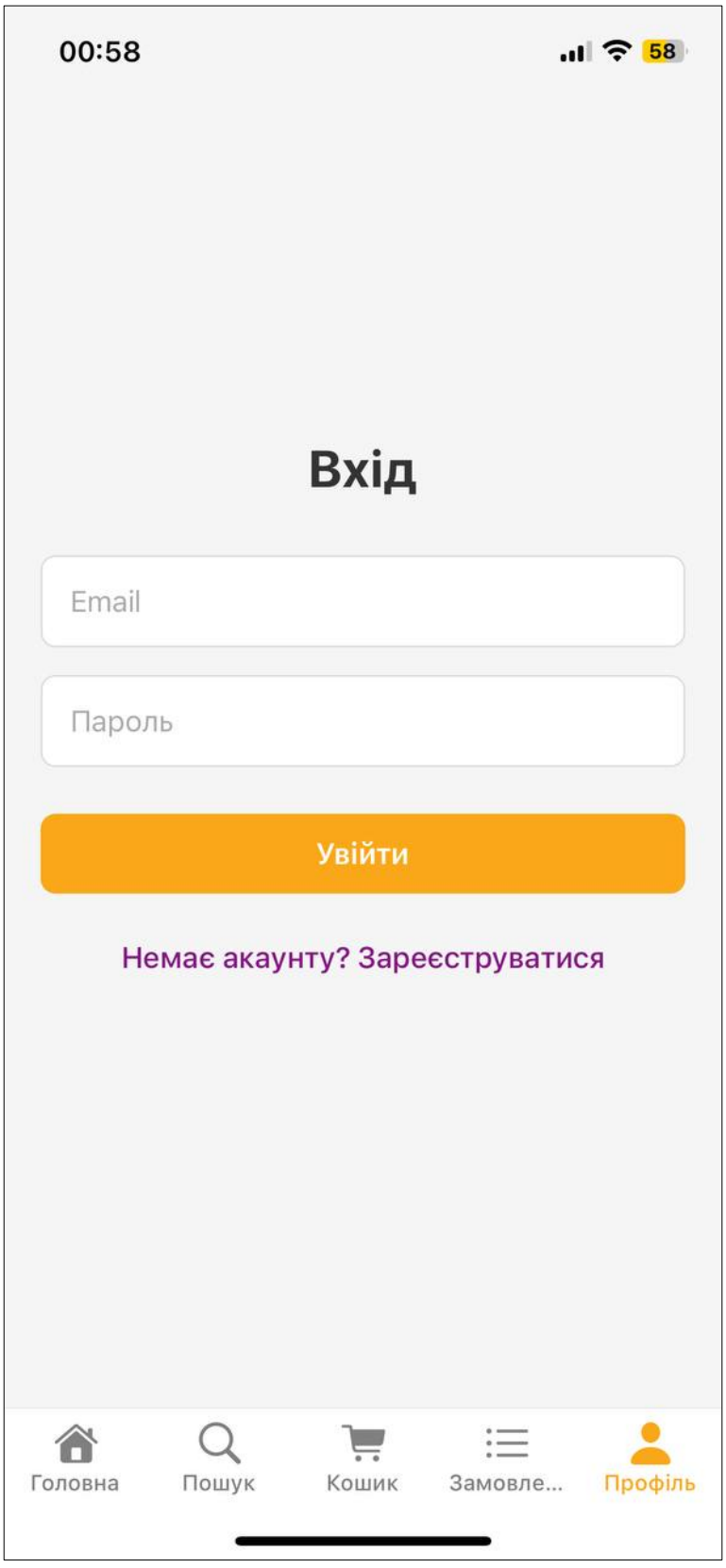


Рисунок 3.8 – Сторінка авторизації

Перегляд каталогу та пошук товарів

1. Головний екран (див. рис. 3.9):
    - на головному екрані відображаються категорії товарів та популярні товари;
    - прокрутіть екран вниз, щоб переглянути більше товарів;
    - натисніть на категорію, щоб переглянути всі товари у цій категорії.

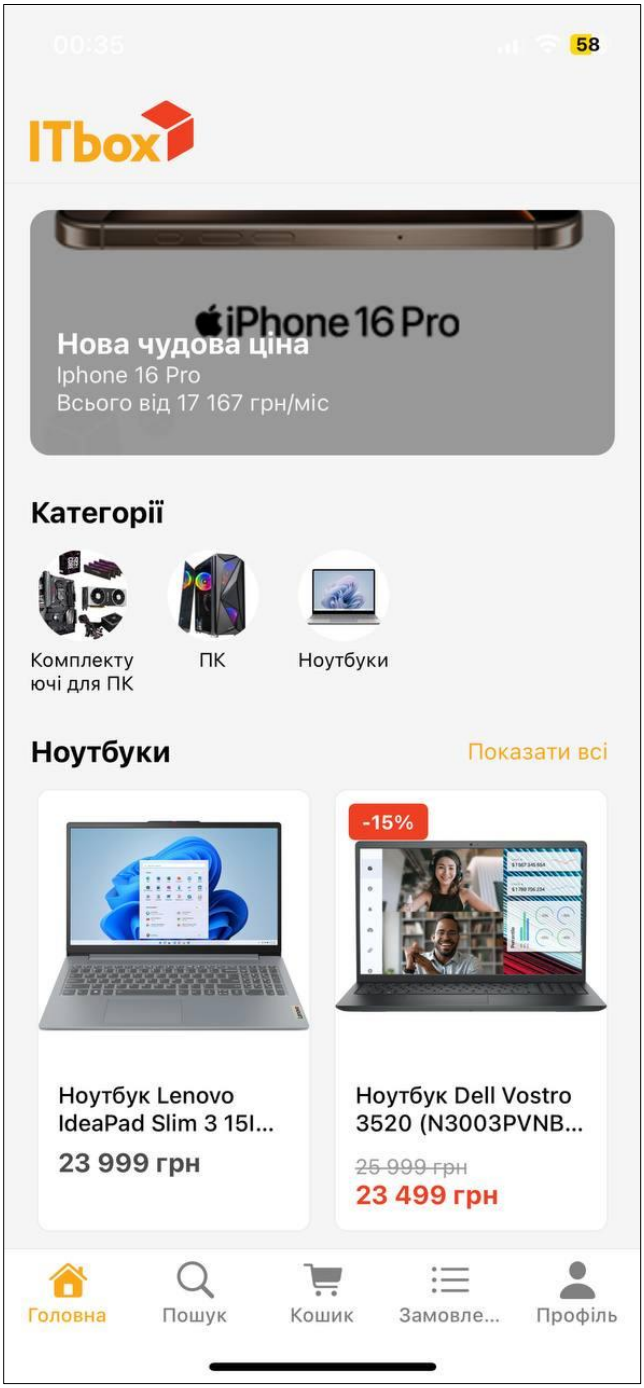


Рисунок 3.9 - Головний екран

2. Пошук товарів (див. рис. 3.10):
    - натисніть на значок пошуку в нижній панелі навігації;
    - введіть назву товару у пошукове поле;
    - результати пошуку відображатимуться автоматично;
    - ви можете фільтрувати результати за категорією або сортувати за ціною, рейтингом тощо.

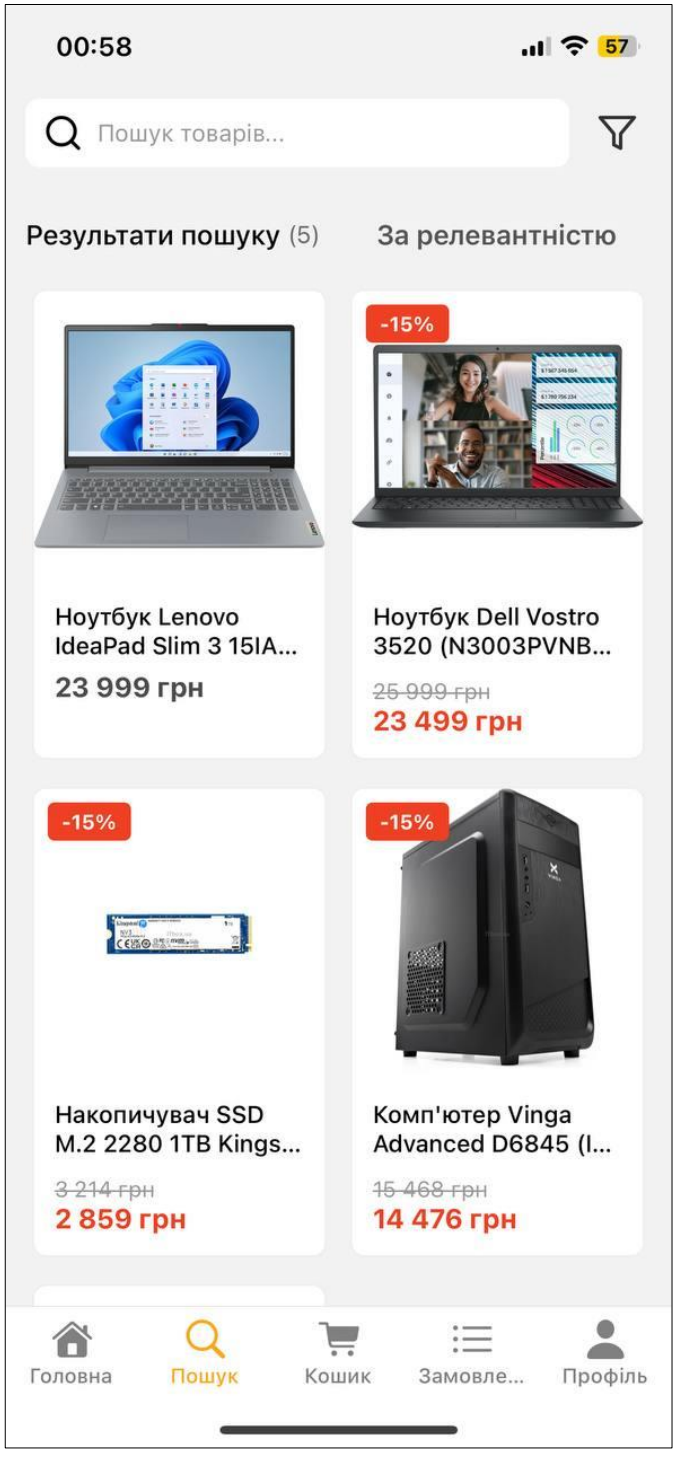


Рисунок 3.10 – Сторінка пошук товарів

3. Детальний перегляд товару (див. рис. 3.11):
    - натисніть на карточку товару, щоб переглянути детальну інформацію;
    - на сторінці товару ви побачите опис, характеристики, ціну та галерею зображень;
    - прокрутіть вниз, щоб побачити подібні товари.

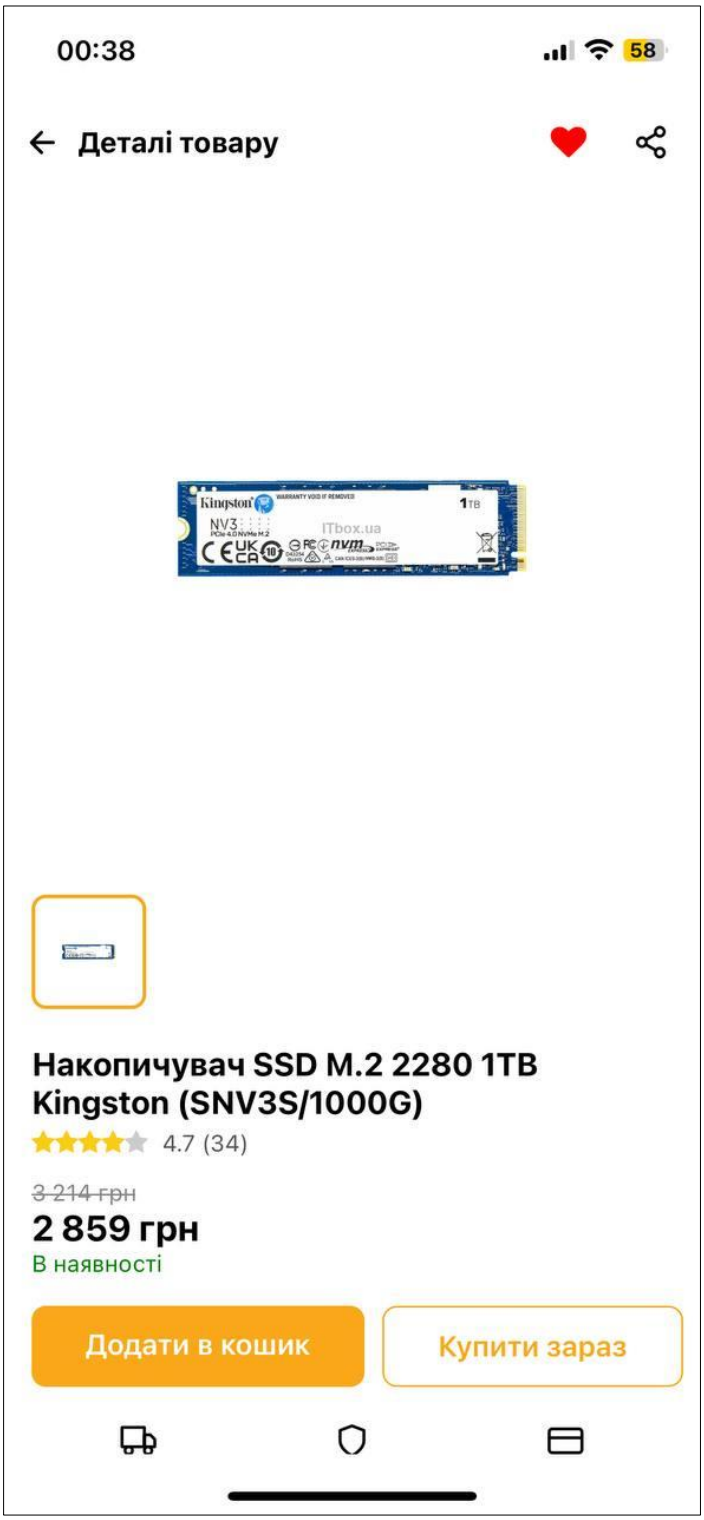


Рисунок 3.11 – Детальний перегляд товару

Робота з кошиком

1. Додавання товару до кошика (див. рис. 3.12):
    - на сторінці товару натисніть кнопку "Додати до кошика";
    - ви можете вибрати кількість товару перед додаванням;
    - після додавання ви побачите повідомлення про успішне додавання.

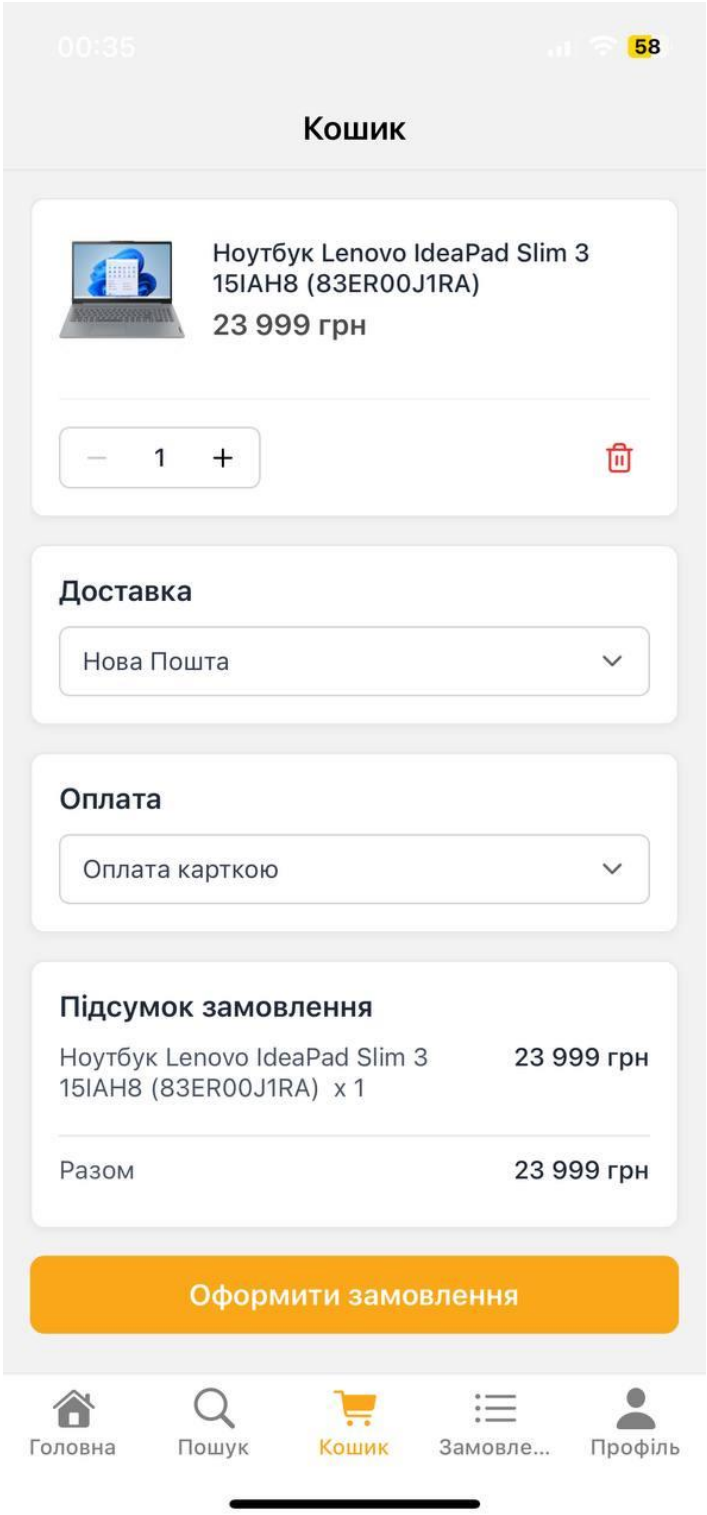


Рисунок 3.12 – Функція додавання товару в кошик

2. Перегляд кошика (див. рис. 3.13):
    - натисніть на значок кошика в нижній панелі навігації;
    - у кошику відображатимуться всі додані товари з кількістю та загальною вартістю.

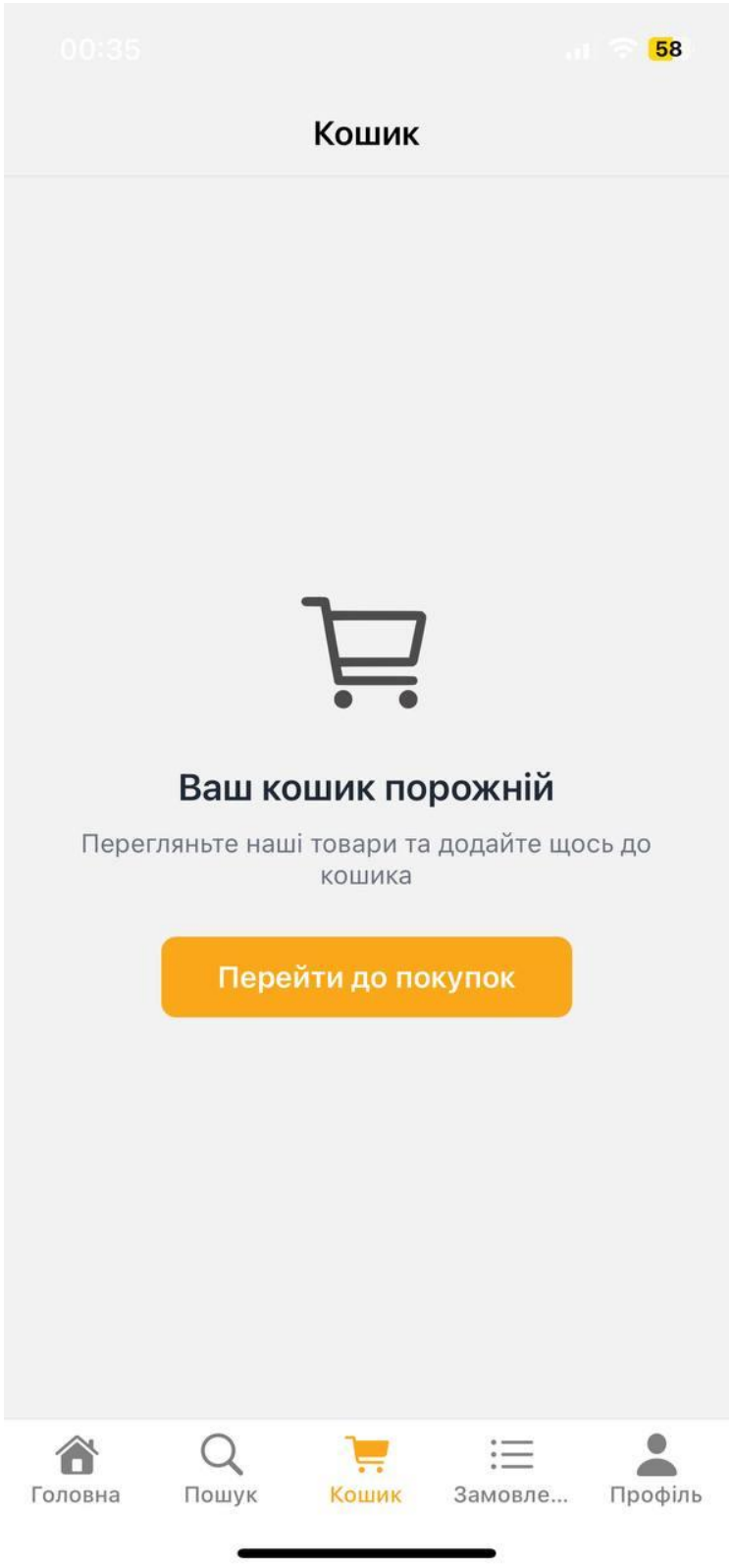


Рисунок 3.13 – Сторінка перегляду кошика

3. Зміна кількості товару в кошику (див. рис. 3.14):
    - натисніть на "+" або "-" біля товару, щоб змінити кількість;
    - натисніть на значок видалення, щоб повністю видалити товар з кошика.

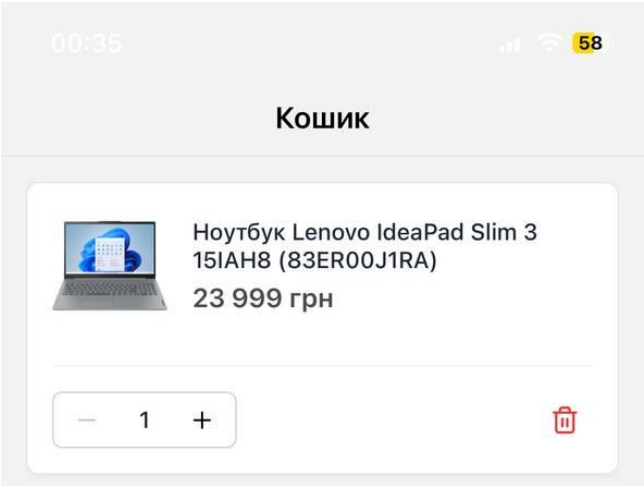


Рисунок 3.14 - Зміна кількості товару в кошику

4. Оформлення замовлення (див. рис. 3.15):
    - у кошику натисніть кнопку "Оформити замовлення";
    - заповніть форму доставки: ім'я, адресу, телефон;
    - виберіть спосіб доставки (Нова пошта, Укрпошта, самовивіз);
    - виберіть спосіб оплати (картка, готівка);
    - перевірте дані замовлення та натисніть "Підтвердити замовлення".

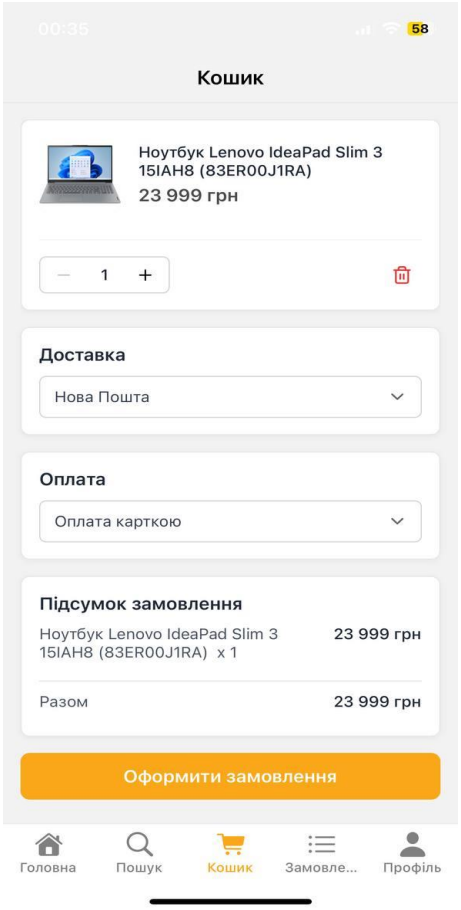


Рисунок 3.15 – Сторінка оформлення замовлення

Робота зі списком вподобаних товарів

1. Додавання товару до вподобаних (див. рис. 3.16):
    - на сторінці товару або в каталозі натисніть на значок сердечка;
    - товар буде додано до списку вподобаних.

Рисунок 3.16 – Сторінка вподобання товару

2. Перегляд вподобаних товарів (див. рис. 3.17):
    - натисніть на пункт "Вподобані" в меню профілю;
    - тут відображатимуться всі товари, які ви додали до списку вподобаних.

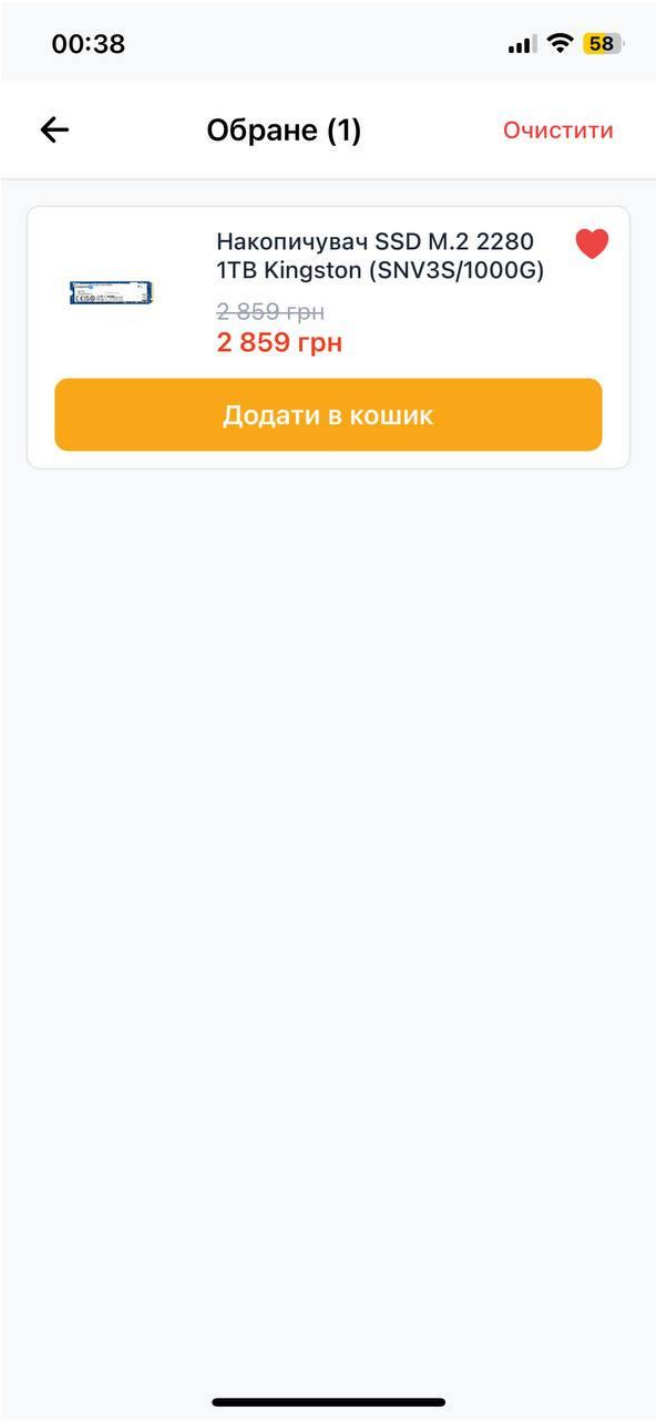


Рисунок 3.17 – Сторінка вподобаних товарів

3. Видалення товару зі списку вподобаних (див. рис. 3.18):
    - на сторінці вподобаних товарів натисніть на значок сердечка біля товару;
    - товар буде видалено зі списку вподобаних.

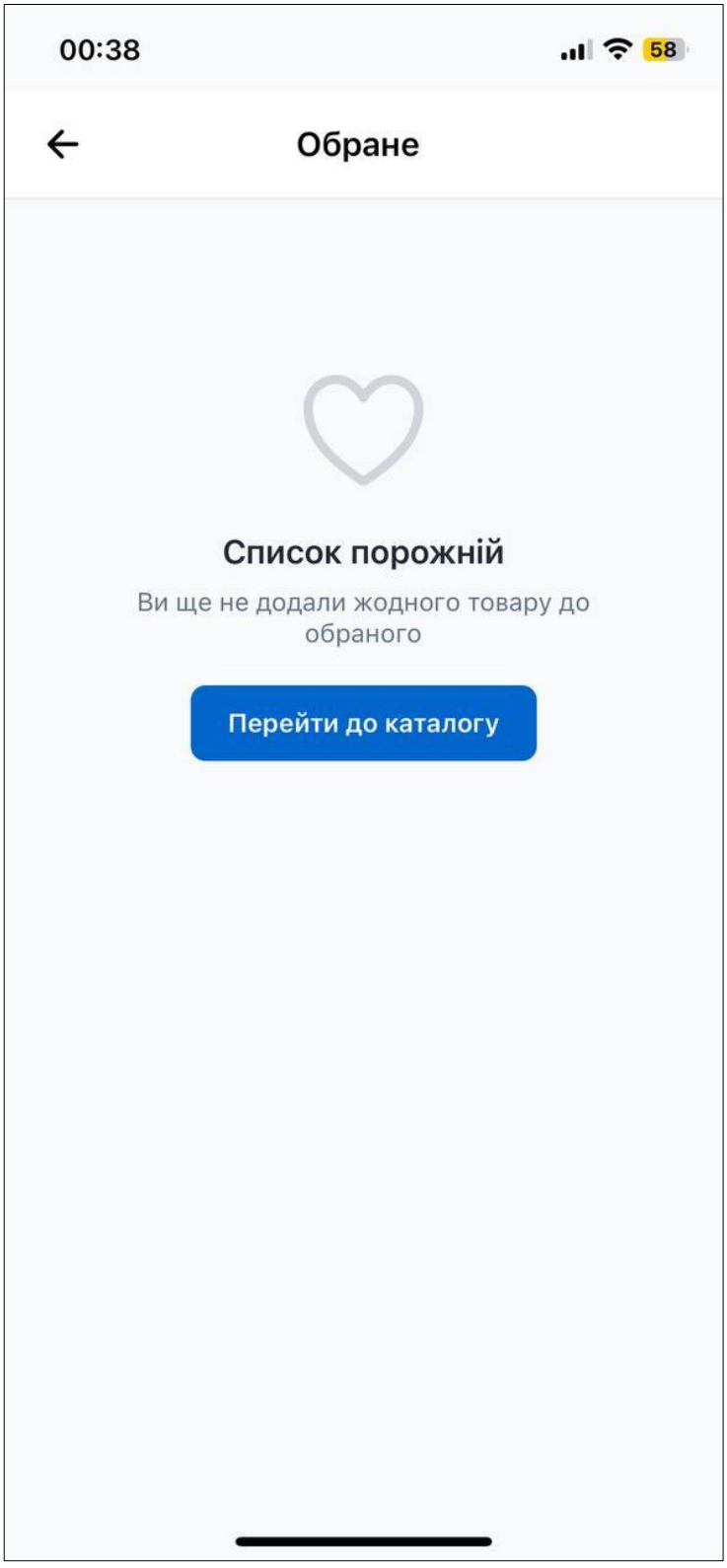


Рисунок 3.18 – Сторінка порожнього списку уподобань

Управління особистим профілем

1. Перегляд профілю (див. рис. 3.19):
    - натисніть на значок профілю в нижній панелі навігації;
    - тут ви побачите свої особисті дані та меню налаштувань.

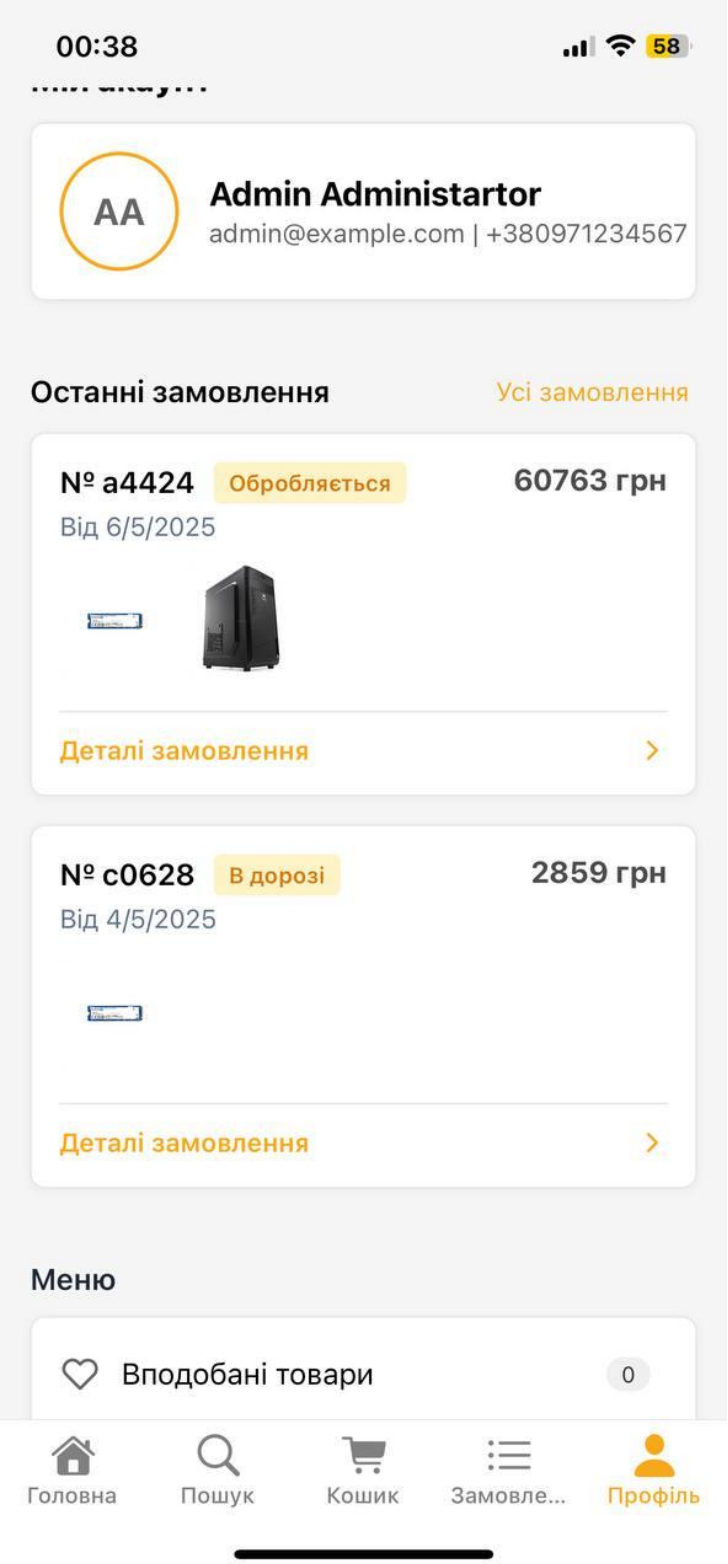


Рисунок 3.19 – Сторінка перегляду профіля

2. Перегляд історії замовлень (див. рис. 3.20):
    - у профілі натисніть на пункт "Мої замовлення";
    - тут відображатимуться всі ваші попередні замовлення;
    - натисніть на замовлення, щоб переглянути його деталі та статус.

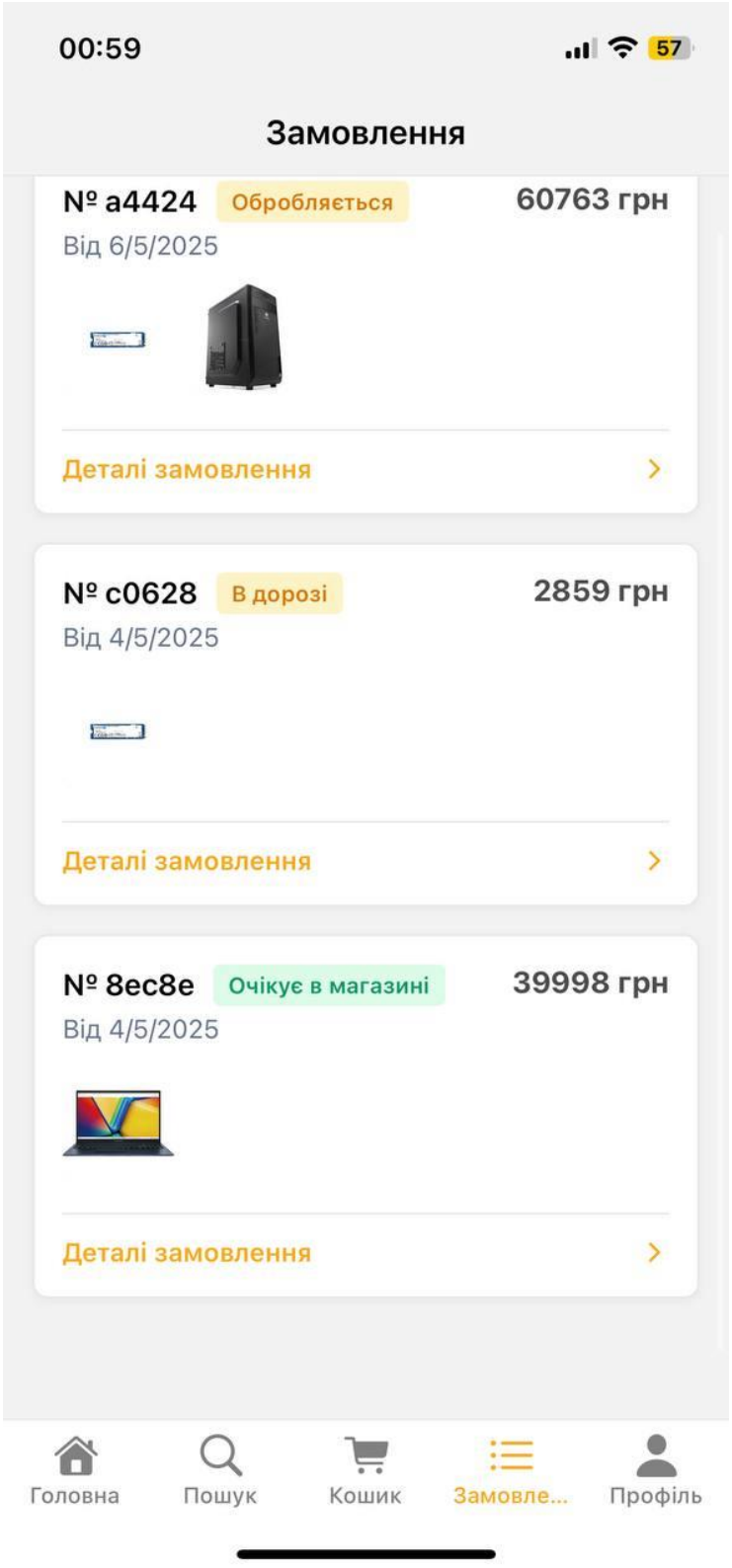


Рисунок 3.20 – Сторінка історії замовлень

3. Вихід з облікового запису (див. рис. 3.21):
    - у профілі натисніть на кнопку "Вийти";
    - ви будете перенаправлені на екран входу.

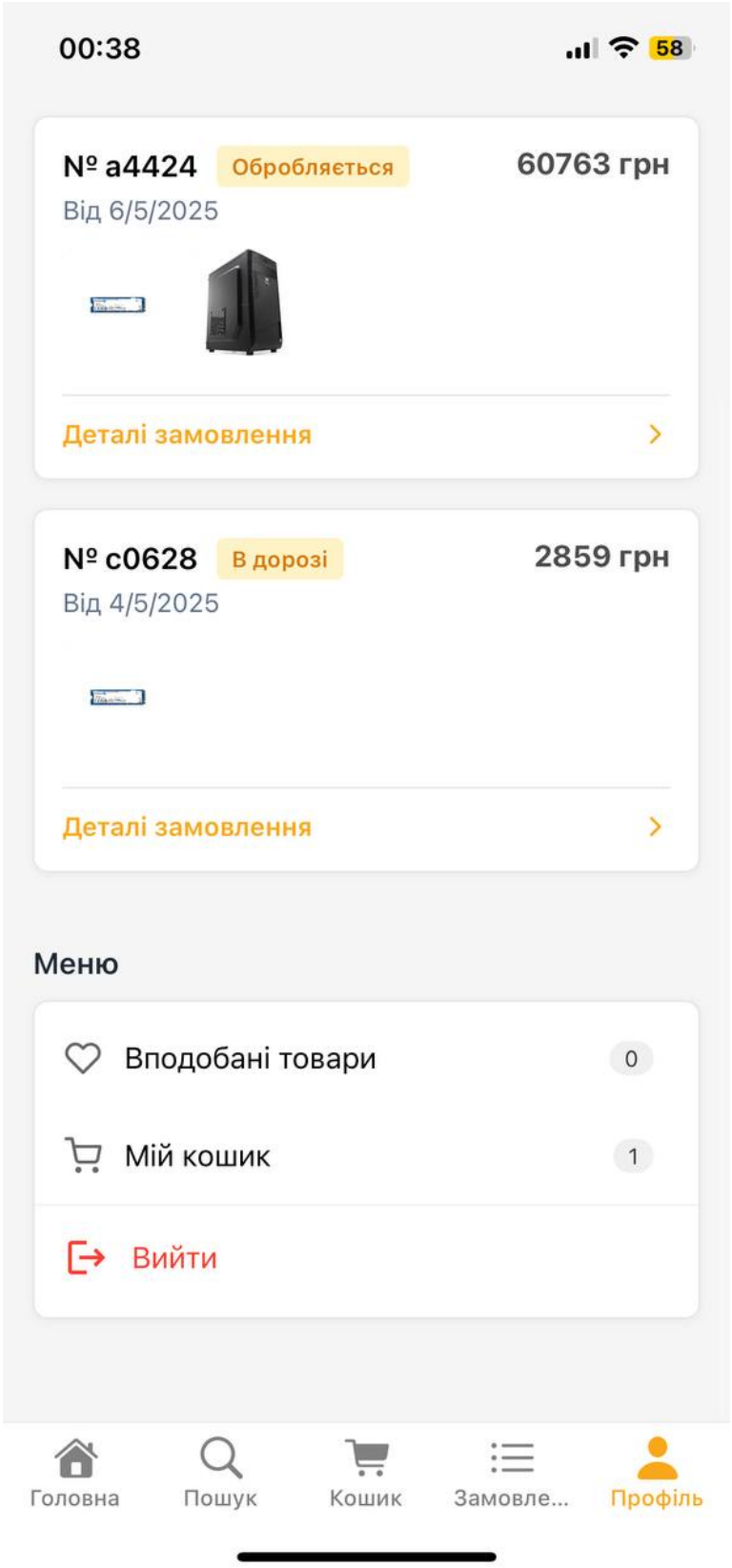


Рисунок 3.21 – Функція вийти

3.4 Тестування програмного продукту

3.4.1 Таблиця результатів тестування

В таблиці 3.4 наведено приклади конкретних тестів, що було проведено для перевірки ключових функцій системи.

Таблиця 3.4 Результати тестування окремих функцій застосунку

№	Пріоритет	Перевірка	Очікуваний результат	Iphone 14	Iphone 15	Iphone 16
1	високий	Авторизація користувача	Успішний вхід до системи	Passed	Passed	Passed
2	високий	Створення нового облікового запису	Створюється новий користувач	Passed	Passed	Passed
3	низький	Додавання товару в кошик	Товар з’являється на сторінці кошика	Passed	Passed	Passed
4	високий	Оформлення замовлення	Збереження даних та формування замовлення	Passed	Passed	Passed
5	середній	Редагування профілю користувача	Зміни зберігаються в системі	Passed	Passed	Passed
6	низький	Відображення даних з порожніми полями	Система коректно обробляє пусті значення	Passed	Passed	Passed
7	середній	Пошук в каталозі	Система коректно відобразить ціль пошуку	Passed	Passed	Passed

3.4.2 Висновки за результатами тестування

На підставі проведеного тестування можна зробити такі висновки:

1. Програмний продукт стабільно функціонує в усіх нових версіях пристроїв під які орієнтований додаток.
2. Основні функції (авторизація, робота з кошиком, оформлення замовлення) відповідають заданим вимогам.
3. Система забезпечує задовільний рівень безпеки для захисту збережених даних та обмеження несанкціонованого доступу.

Під час тестування було виявлено кілька незначних помилок, які було усунуто в процесі доопрацювання. В цілому, розроблена система відповідає початковим вимогам і може бути впровадженою для використання.

ВИСНОВОК

Аналіз діяльності компанії ItBox, яка функціонує як підрозділ Brain Computers, виявив суттєвий недолік у поточній стратегії роздрібної торгівлі — відсутність повноцінного мобільного застосунку. В умовах сучасного ринку e-commerce, коли мобільний трафік суттєво переважає десктопний, це зменшує конкурентоспроможність компанії та обмежує доступ до значної частини потенційних клієнтів.

Основні гравці ринку (Rozetka, Allo, Prom.ua) активно використовують мобільні застосунки як ключовий канал взаємодії з клієнтами. При цьому виявлено, що рівень мобільної оптимізації та персоналізації клієнтського досвіду став де-факто ринковим стандартом, а не конкурентною перевагою. Отже мобільний застосунок для ItBox доцільний через зростання мобільного трафіку, потребу в зручному інтерфейсі, швидкому доступі до товарів і персоналізованих функцій, що підвищує продажі та лояльність клієнтів. Розробка мобільного додатку здійснена на основі сучасного технологічного стеку на основі React Native та Node.js. Переваги React Native в мобільній розробці:

- Кросплатформеність (iOS і Android з одним кодом)
- Висока швидкодія для UI
- Швидкий розвиток і оновлення завдяки Hot Reload
- Велика спільнота та багато готових бібліотек

Переваги Node.js в мобільній розробці:

- Висока продуктивність при обробці запитів
- Один JavaScript-стек клієнт–сервер
- Легке масштабування
- Багато пакетів через npm

Розроблений мобільний застосунок ItBox вирішує ключові проблеми бізнесу:

– підвищує лояльність користувачів через забезпечення зручного мобільного досвіду;

– автоматизує процеси комунікації з клієнтами через систему сповіщень;

- забезпечує додатковий канал продажів, що потенційно збільшує ринкову частку компанії;
- підвищує ефективність відділу роздрібної торгівлі завдяки автоматизації рутинних операцій;
- дозволяє залишатися конкурентоспроможними в умовах мобільноорієнтованого споживчого ринку.

Детальні розрахунки ефективності впровадження мобільного застосунку ItBox продемонстрували економічну доцільність проекту з терміном окупності 1,26 роки. Загальна вартість розробки та впровадження склала 1 795 262, а очікуваний річний економічний ефект — $34500.

В цілому, розробка та впровадження мобільного застосунку для інтернет-магазину ItBox є стратегічно важливим кроком у розвитку компанії, що не лише відповідає сучасним вимогам ринку, але й створює технологічний фундамент для подальшого зростання і розширення бізнесу.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. ДСТУ 3973-2000. СИСТЕМА РОЗРОБЛЕННЯ ТА ПОСТАВЛЕННЯ ПРОДУКЦІЇ НА ВИРОБНИЦТВО. Чинний від 27.11.2000. Київ : ДЕРЖСТАНДАРТ УКРАЇНИ, 2001. 43 с. URL: https://www.pdau.edu.ua/sites/default/files/node/2805/dstu39732000.pdf (дата звернення: 15.05.2025).
2. ДСТУ 3008:2015. ЗВІТИ У СФЕРІ НАУКИ І ТЕХНІКИ. На заміну ДСТУ 3008-95 ; чинний від 22.06.2015. Київ : ДП «УкрНДНЦ», 2016. 26 с. URL: https://science.kname.edu.ua/images/dok/derzhstandart_3008_2015.pdf (дата звернення: 15.05.2025).
3. ДСТУ 2293:2014. ОХОРОНА ПРАЦІ. Терміни та визначення основних понять. На заміну ДСТУ 2293-99 ; чинний від 01.05.2015. Київ : МІНЕКОНОМРОЗВИТКУ УКРАЇНИ, 2015. 18 с. (дата звернення: 15.05.2025).
4. ДСТУ ISO 6309:2007. ПРОТИПОЖЕЖНИЙ ЗАХИСТ. Знаки безпеки Форма та колір. На заміну ГОСТ 12.4.026–76 ; чинний від 30.03.2007. Київ : ДЕРЖСПОЖИВСТАНДАРТ УКРАЇНИ, 2008. 12 с. URL: https://www.ksv.biz.ua/GOST/DSTY_ALL/DSTY1/dsty_iso_6309-2007.pdf (дата звернення: 15.05.2025).
5. ДСТУ EN 13306:2019. ТЕХНІЧНЕ ОБСЛУГОВУВАННЯ. Терміни та визначення понять. Чинний від 01.10.2007. Київ : ДЕРЖСПОЖИВСТАНДАРТ УКРАЇНИ, 2008. 34 с. (дата звернення: 15.05.2025).
6. ДСТУ ISO/IEC 27001:2015. МЕТОДИ ЗАХИСТУ СИСТЕМИ УПРАВЛІННЯ ІНФОРМАЦІЙНОЮ БЕЗПЕКОЮ. На заміну ДСТУ ISO/IEC 27001:2010 ; чинний від 01.01.2017. Київ : ДП «УкрНДНЦ», 2016. 28 с. URL: https://www.assistem.kiev.ua/doc/dstu_ISO-IEC_27001_2015.pdf (дата звернення: 15.05.2025).

7. ДСТУ EN 12464-1:2016. Світло та освітлення. Освітлення робочих місць. Частина 1. Внутрішні робочі місця.. Чинний від 01.12.2017. Київ : ДП «УкрНДНЦ», 2018. 47 с. (дата звернення: 15.05.2025).
8. ДБН В.2.5-28-2006. ПРИРОДНЕ І ШТУЧНЕ ОСВІТЛЕННЯ. На заміну СНиП II-4-79 ; чинний від 15.05.2006. Київ : Мінбуд України, 2006. 96 с.
9. ECMAScript 2022 Language Specification. *ECMAScript*. 01.01.2022. URL: https://262.ecma-international.org/13.0/ (дата звернення: 15.05.2025).
10. ДСТУ Б В.2.5-82:2016. ЕЛЕКТРОБЕЗПЕКА В БУДІВЛЯХ І СПОРУДАХ Вимоги до захисних заходів від ураження електричним струмом. На заміну ДБН В.2.5-27-2006 ; чинний від 01.04.2017. Київ : ДП "УкрНДНЦ", 2016. 110 с. URL: http://www.tsatu.edu.ua/ettp/wp-content/uploads/sites/25/dstu-b-v.2.5-82-2016-elektrobezpeka-v-budivljah-i-sporudah-1.pdf (дата звернення: 15.05.2025).
11. Методичні рекомендації до виконання кваліфікаційної роботи на здобуття освітнього ступеня «бакалавр» спеціальності 122 «Комп’ютерні науки» освітньо-професійної програми «Інформаційні системи та штучний інтелект» денної форми здобуття освіти [Електрон. ресурс] / уклад. С. В. Грибков, Н. В. Ліманська, М. П. Костіков. – К.: НУХТ, 2025. – 43 с.
12. ДСТУ ISO/IEC 25051:2016. ВИМОГИ ДО ЯКОСТІ СИСТЕМ І ПРОГРАМНИХ ЗАСОБІВ ТА ЇЇ ОЦІНЮВАННЯ (SQuaRE). На заміну на заміну ДСТУ ISO/IEC 25051:2015 ; чинний від 27.12.2016. Київ : Не є офіційним виданням., 2017. 15 с. (дата звернення: 15.05.2025).
13. ДСТУ ISO/IEC/IEEE 29119-1:2017. ІНЖЕНЕРІЯ СИСТЕМ І ПРОГРАМНИХ ЗАСОБІВ ТЕСТУВАННЯ ПРОГРАМНИХ ЗАСОБІВ. На заміну НА ЗАМІНУ ДСТУ ISO/IEC/IEEE 29119-1:2015 ; чинний від 19.12.2017. Київ : Не є офіційним виданням., 2018. 20 с. (дата звернення: 15.05.2025).
14. ДСТУ EN ISO 9241-210:2022. Ергономіка взаємодії людина-система.. На заміну На заміну ДСТУ EN ISO 9241-210:2019 ; чинний від 28.12.2022. Київ : Не є офіційним виданням. 21 с. (дата звернення: 15.05.2025).

15. ДСТУ 3974-2000. СИСТЕМА РОЗРОБЛЕННЯ ТА ПОСТАВЛЕННЯ ПРОДУКЦІЇ НА ВИРОБНИЦТВО. На заміну - ; чинний від 27.11.2000. Київ : ДЕРЖСТАНДАРТ УКРАЇНИ, 2001. 48 с. (дата звернення: 15.05.2025).
16. - T. React Native vs Flutter vs Native - Let’s Talk Engines. *Youtube*. *Theo - t3.gg*. 10.07.2022. URL: https://www.youtube.com/watch?v=3_FcxGCCnUs (дата звернення: 15.05.2025).
17. @ Theo - t3.gg. My Final Flutter Video. *Youtube*. *Theo - t3.gg*. 16.08.2023. URL: https://www.youtube.com/watch?v=tTGWfXPKxf4 (дата звернення: 15.05.2025).
18. Style your React Native apps using Tailwind CSS. *Nativewind*. *Style your React Native apps using Tailwind CSS.*. URL: https://www.nativewind.dev/ (дата звернення: 15.05.2025).
19. Rapidly build modern websites without ever leaving your HTML.. *Tailwindcss*. 05.02.2025. URL: https://tailwindcss.com/ (дата звернення: 15.05.2025).
20. @ Theo - t3.gg. Mobile Devs Hate Servers. Expo Wants To Fix That.. *Youtube*. *Theo - t3.gg*. 30.01.2024. URL: https://www.youtube.com/watch?v=2P0q1EdH_oQ (дата звернення: 15.05.2025).
21. Introduction to Expo Router. *Docs.expo*. *Introduction to Expo Router*. URL: https://docs.expo.dev/router/introduction/ (дата звернення: 15.05.2025).
22. API Routes. *Docs.expo*. *API Routes*. URL: https://docs.expo.dev/router/reference/api-routes/ (дата звернення: 15.05.2025).
23. ДСТУ ISO/IEC/IEEE 26512:2018. ІНЖЕНЕРІЯ СИСТЕМ І ПРОГРАМНИХ ЗАСОБІВ. На заміну - ; чинний від 18.12.2018. Київ : Не є офіційним виданням. 20 с. (дата звернення: 15.05.2025).
24. Про нас - Itbox. *Itbox*. *Про нас*. URL: https://www.itbox.ua/ (дата звернення: 15.05.2025).
25. ДСТУ 2293-99. ОХОРОНА ПРАЦІ. На заміну - ; чинний від 01.01.2000. Київ : Не є офіційним виданням., 1999. 17 с. (дата звернення: 15.05.2025).

26. ДСТУ 7237:2011. СИСТЕМА СТАНДАРТІВ БЕЗПЕКИ ПРАЦІ ЕЛЕКТРОБЕЗПЕКА. На заміну на заміну ДСТУ IEC 60050-604:2004 ; чинний від 02.02.2011. Київ : ДЕРЖСТАНДАРТ УКРАЇНИ, 2011. 12 с. (дата звернення: 15.05.2025).
27. ДСТУ 7238:2011. СИСТЕМА СТАНДАРТІВ БЕЗПЕКИ ПРАЦІ. Чинний від 02.02.2011. Київ : ДЕРЖСТАНДАРТ УКРАЇНИ, 2011. 9 с. (дата звернення: 15.05.2025).
28. Introduction. *React Native*. *Introduction*. 15.05.2025. URL: https://reactnative.dev/docs/getting-started (дата звернення: 15.05.2025).
29. JavaScript Guide. *MDN Web Docs*. URL: https://developer.mozilla.org/en-US/docs/Web/JavaScript/Guide (дата звернення: 15.05.2025).
30. React Documentation. *React*. URL: https://react.dev/ (дата звернення: 15.05.2025).

ДОДАТКИ

Додаток А. Схема організаційної структури ITBOX

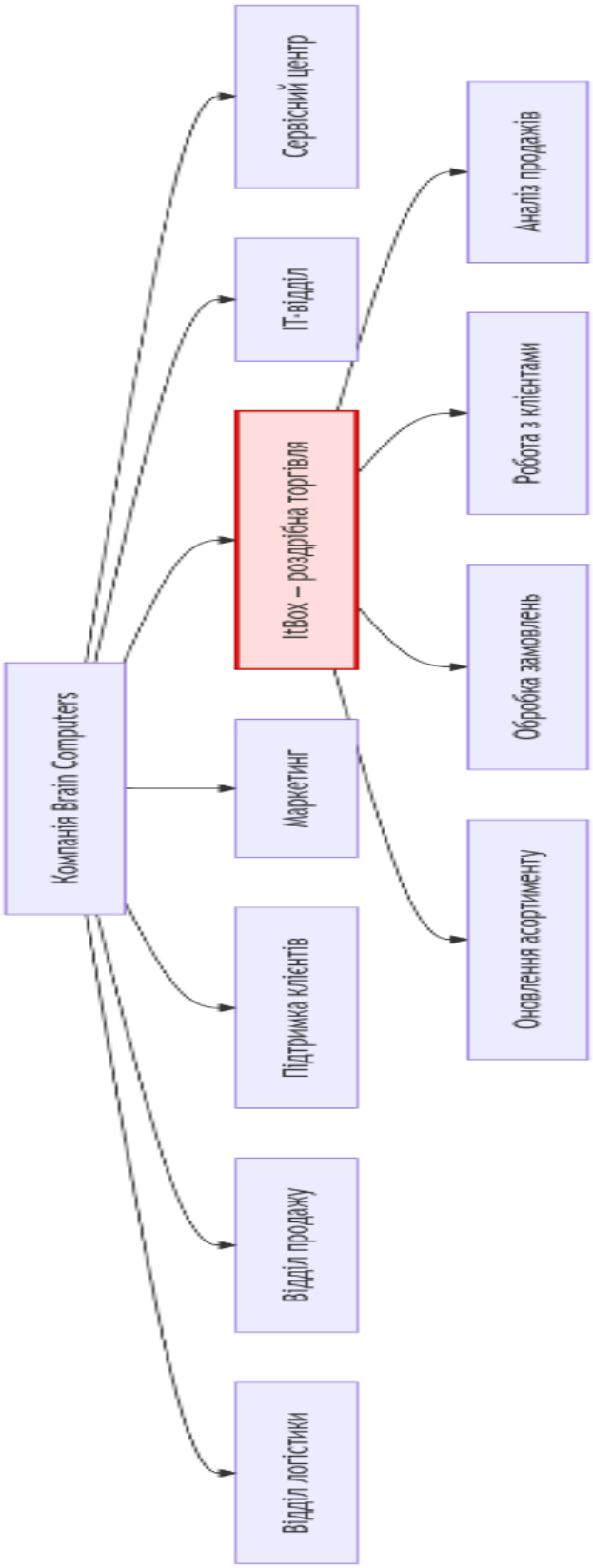
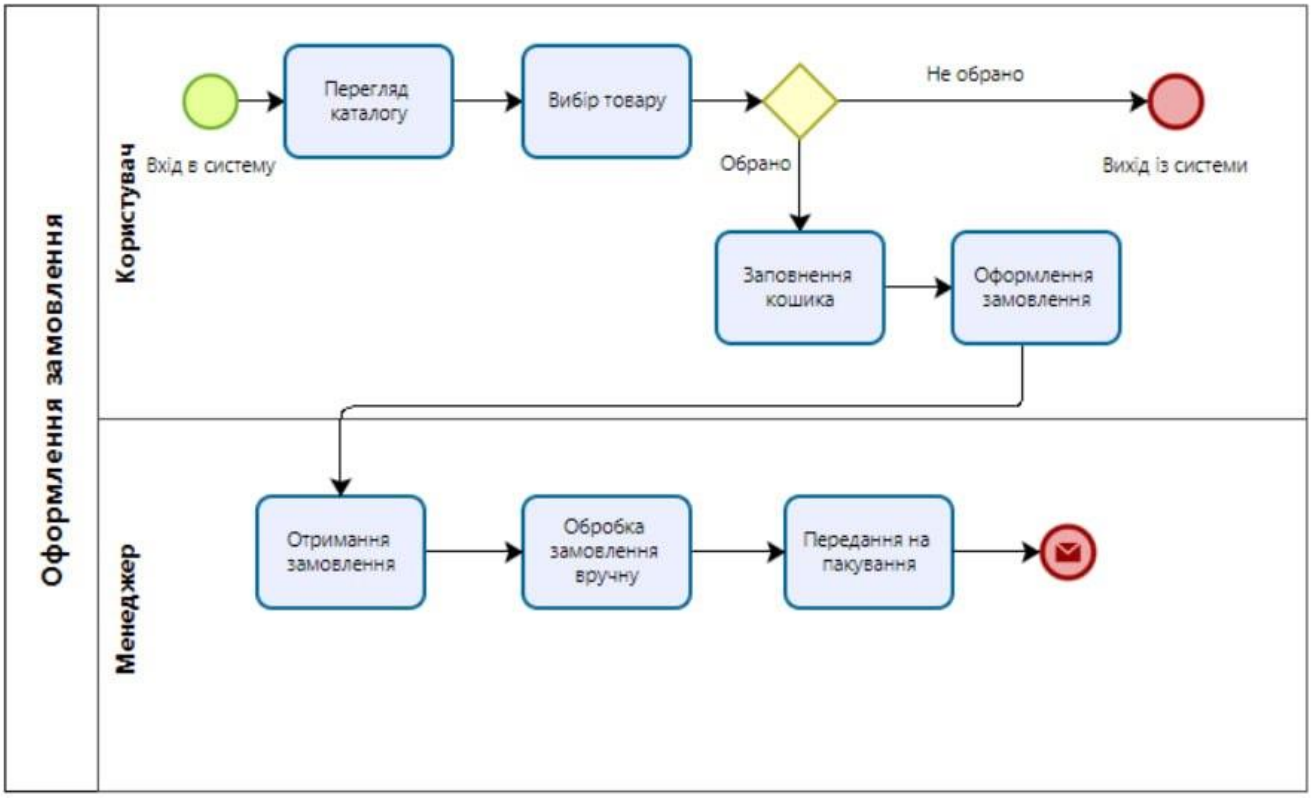
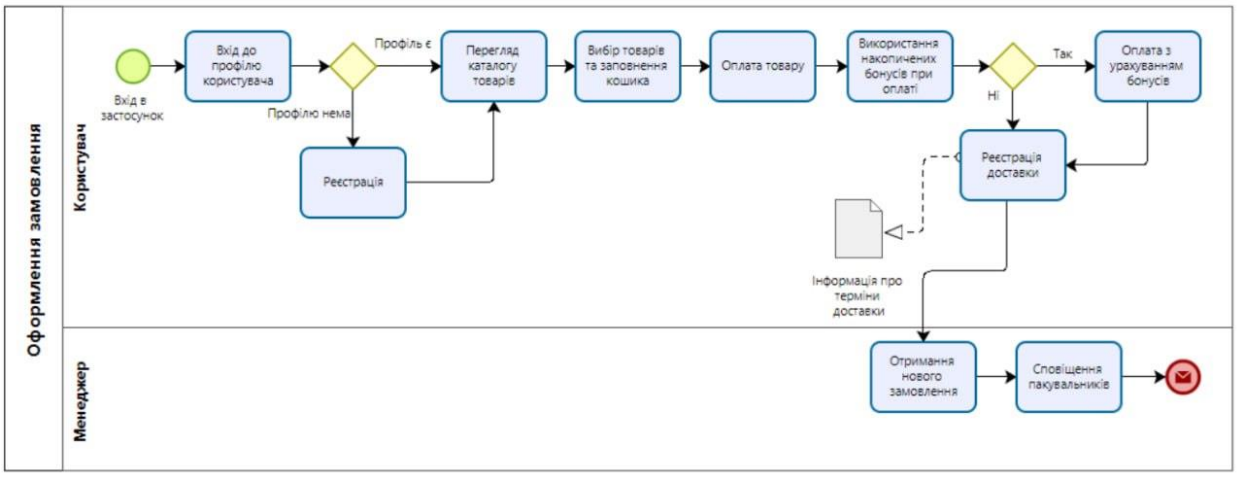


Рисунок А.1 - Структура підприємства з виділенням підрозділу автоматизації

Додаток Б. Модель бізнес-процесу оформлення замовлень в нотації BPMN

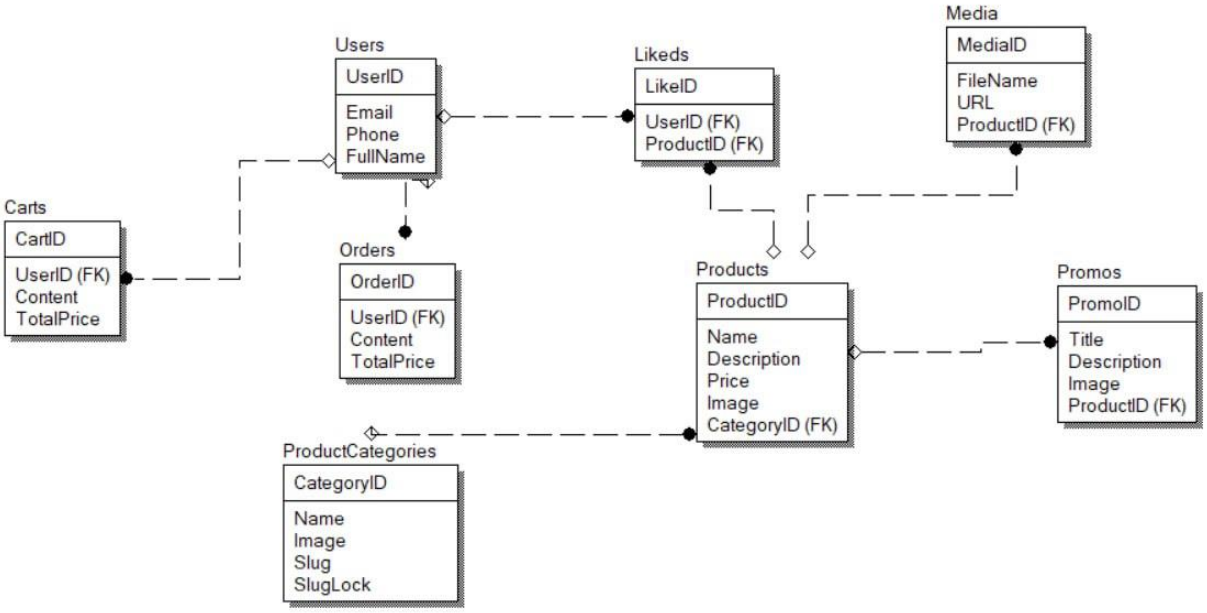


Додаток Б.1 – Модель бізнес-процесу оформлення замовлення – статусу TO BE



Додаток Б.2 – Модель бізнес-процесу оформлення замовлення – статус AS - IS

Додаток В. Логічна модель бази даних



Додаток В.1 – Логічна модель бази даних

Додаток Г. Структура бази даних в СУБД MongoDB

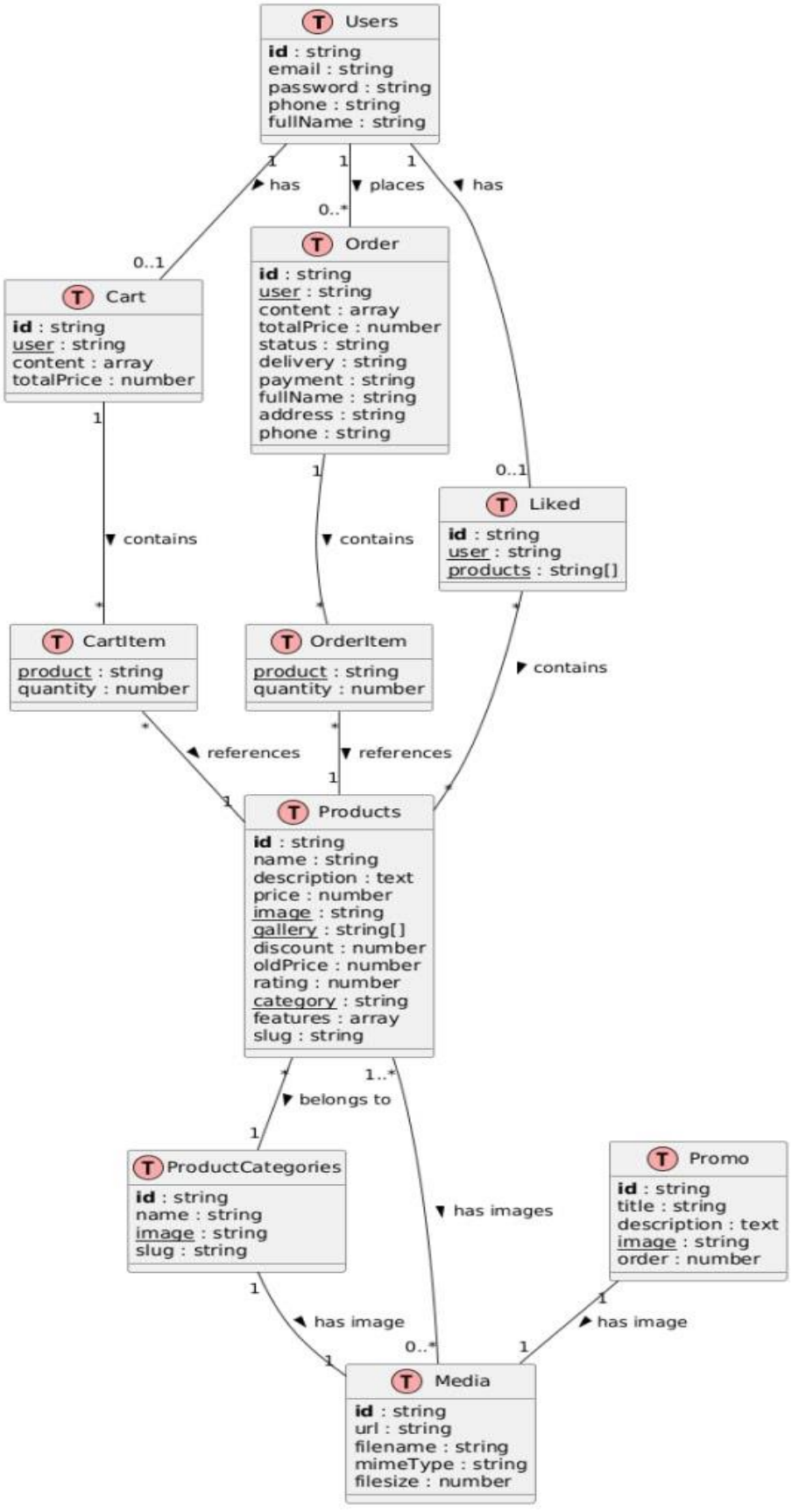


Рисунок Г.5 –Структура бази даних у MongoDB

Додаток Д. Фрагменти функціоналу у вигляді коду

```
const addToCart = useCallback(async (product: Product, quantity: number = 1) => {
    const currentItems = [...cartItems];
    const existingItemIndex = currentItems.findIndex((item) => item.id === product.id);
    let updatedItems: CartItem[];
    if (existingItemIndex > -1) {
      const updatedExistingItem = {
        ...currentItems[existingItemIndex],
        quantity: currentItems[existingItemIndex].quantity + quantity,
      };
      currentItems[existingItemIndex] = updatedExistingItem;
      updatedItems = currentItems;
    } else {
      const newItem: CartItem = {
        ...product,
        quantity: quantity,
      };
      updatedItems = [...currentItems, newItem];
    }
    setCartItems(updatedItems);
    await saveLocalCart(updatedItems, cartId);
    if (user?.id && token) {
      const updatedId = await updateCartInAPI(updatedItems, cartId);
      if (updatedId !== cartId) {
        setCartId(updatedId);
        await saveLocalCart(updatedItems, updatedId);
      }
    }
  }, [cartItems, cartId, updateCartInAPI, saveLocalCart, user, token]);
```

```
  const removeFromCart = useCallback(async (productId: string) => {
    const updatedItems = cartItems.filter((item) => item.id !== productId);
    setCartItems(updatedItems);
    const updatedCartId = updatedItems.length === 0 ? null : cartId;
    await saveLocalCart(updatedItems, updatedCartId);
    if (user?.id && token) {
      if (updatedItems.length === 0 && cartId) {
        await deleteCartFromAPI(cartId);
        setCartId(null);
      } else if (cartId) {
        await updateCartInAPI(updatedItems, cartId);
      }
    } else {
      setCartId(updatedCartId);
    }
  }, [cartItems, cartId, updateCartInAPI, saveLocalCart, user, token,
 deleteCartFromAPI]);

const updateQuantity = useCallback(async (productId: string, quantity: number) => {
    if (quantity <= 0) {
      await removeFromCart(productId);
      return;
    }
    const updatedItems = cartItems.map((item) =>
      item.id === productId ? { ...item, quantity: quantity } : item
    );
    setCartItems(updatedItems);
    await saveLocalCart(updatedItems, cartId);
    if (user?.id && token && cartId) {
      await updateCartInAPI(updatedItems, cartId);
```

```
        }
      }, [cartItems, cartId, updateCartInAPI, saveLocalCart, removeFromCart, user,
    token]);
  const clearCart = useCallback(async () => {
      setCartItems([]);
      setCartId(null);
      await saveLocalCart([], null);
      if (user?.id && token && cartId) {
        await deleteCartFromAPI(cartId);
      }
    }, [cartId, saveLocalCart, user, token, deleteCartFromAPI]);
    const getItemQuantity = useCallback((productId: string): number => {
      const item = cartItems.find((item) => item.id === productId);
      return item ? item.quantity : 0;
    }, [cartItems]);
    const totalCartItems = useMemo(() => {
      return cartItems.reduce((sum, item) => sum + item.quantity, 0);
    }, [cartItems]);
    const totalCartPrice = useMemo(() => {
      return cartItems.reduce(
        (sum, item) => sum + (item.price || 0) * item.quantity,
        0
      );
    }, [cartItems]);

useEffect(() => {
   const initializeCart = async () => {
     if (initializationCompleted.current) {
      console.log('Cart initialization already completed, skipping');
      return;
```

```
}
const { items: localItems, savedCartId } = await loadLocalCart();
console.log('Initial load state:', {
  localItems: localItems.length,
  savedCartId,
  userLoggedIn: !!user?.id,
  token: !!token
});
setCartItems(localItems);
setCartId(savedCartId);
setLocalCartLoaded(true);
if (!user?.id || !token) {
  console.log('No authenticated user, using only local cart');
  initializationCompleted.current = true;
  return;
}
console.log('User logged in, checking for server cart');
const serverCart = await loadCartFromAPI();
if (serverCart) {
  console.log(`Found server cart: ${serverCart.id}`);
  const serverItems: CartItem[] = serverCart.content
    .filter(item => item.product && typeof item.product === 'object')
    .map((item): CartItem => ({
      ...(item.product as Product),
      quantity: item.quantity,
    }));
  if (savedCartId && savedCartId === serverCart.id) {
    console.log('Using server cart as source of truth (IDs match)');
    setCartId(serverCart.id);
    setCartItems(serverItems);
```

```
        await saveLocalCart(serverItems, serverCart.id);
      }
      else if (savedCartId && savedCartId !== serverCart.id) {
        console.log('Cart ID mismatch - merging and keeping server cart');
        const mergedItems = [...localItems];
        serverItems.forEach(serverItem => {
          const existingItemIndex = mergedItems.findIndex(item => item.id ===
serverItem.id);
          if (existingItemIndex === -1) {
            mergedItems.push(serverItem);
          }
        });
        await updateCartInAPI(mergedItems, serverCart.id);
        setCartId(serverCart.id);
        setCartItems(mergedItems);
        await saveLocalCart(mergedItems, serverCart.id);
      }
      else if (localItems.length > 0) {
        console.log('Merging local items into server cart');
        const mergedItems = [...localItems];
        serverItems.forEach(serverItem => {
          const existingItemIndex = mergedItems.findIndex(item => item.id ===
serverItem.id);
          if (existingItemIndex === -1) {
            mergedItems.push(serverItem);
          }
        });
        await updateCartInAPI(mergedItems, serverCart.id);
        setCartId(serverCart.id);
        setCartItems(mergedItems);
```

```
          await saveLocalCart(mergedItems, serverCart.id);
        }
        else {
          console.log('Using server cart (no local items)');
          setCartId(serverCart.id);
          setCartItems(serverItems);
          await saveLocalCart(serverItems, serverCart.id);
        }
      }
      else if (localItems.length > 0) {
        console.log('Creating server cart with local items');
        const newCartId = await createNewCartInAPI(localItems);
        if (newCartId) {
          setCartId(newCartId);
          await saveLocalCart(localItems, newCartId);
        }
      }
      initializationCompleted.current = true;
    };
    if (user !== undefined) {
      initializeCart();
    }
  }, [user, token, loadLocalCart, loadCartFromAPI, saveLocalCart, updateCartInAPI,
createNewCartInAPI]);


const handleCheckout = async () => {
        const isValid = validateData();
        if (!isValid) {
            alert('Будь ласка, заповніть всі поля коректно.');
```

```
    return;
}
try{
    console.log('Checkout data:', {
        user: user?.id,
        content: cartItems.map(item => ({
            product: item.id,
            quantity: item.quantity,
        })),
        totalPrice:0,
        delivery: checkoutData?.deliveryOption,
        payment: checkoutData?.paymentOption,
        fullName,
        phone,
        address,
    });
    await fetch(`${PAYLOAD_API_URL}/order`,{
        method: 'POST',
        headers: {
            'Content-Type': 'application/json',
        },
        body: JSON.stringify({
            user: user?.id,
            content: cartItems.map(item => ({
                product: item.id,
                quantity: item.quantity,
            })),
            totalPrice:0,
            delivery: checkoutData?.deliveryOption,
            payment: checkoutData?.paymentOption,
```

```
                fullName,
                phone,
                address,
            }),
        })
        .then(async (res) => {
            if(!res.ok) {
                const errorData = await res.json();
                console.error('Error response:', errorData);
                throw new Error('Network response was not ok');
            }
            return res.json();
        })
        .then(data => {
            if(data?.doc?.id) {
                alert('Замовлення успішно оформлено!');
                clearCart();
                router.push('/');
            } else {
                alert('Сталася помилка під час оформлення замовлення. Спробуйте ще
раз.');
            }
        });
    }
    catch(e) {
        console.error('Checkout error:', e);
        alert('Сталася помилка під час оформлення замовлення. Спробуйте ще
раз.');
    }
}
```

```
const loadLikedFromAPI = useCallback(async (): Promise<Liked | null> => {
  if (!user?.id || !token) {
    console.log('No user logged in or no token available, not fetching from API');
    return null;
  }
  console.log('Loading liked products from Payload for user:', user.id);
  setIsLoading(true);
  setError(null);
  const query = stringify({
    where: {
      'user.equals': user.id,
    },
    depth: 2,
    limit: 1,
  }, { addQueryPrefix: true });
  try {
    const response = await fetch(`${PAYLOAD_API_URL}/liked${query}`, {
      method: 'GET',
      headers: {
        Authorization: `JWT ${token}`,
        'Content-Type': 'application/json',
      },
    });
    if (!response.ok) {
      if (response.status === 404) {
        console.log('No liked collection found for user (404).');
        return null;
      } else if (response.status === 403 || response.status === 401) {
```

```
          console.error('Authorization error loading liked products:',
response.statusText);
          setError('Authorization error when loading liked products.');
          return null;
        } else {
          console.error('Failed to load liked products:', response.status,
response.statusText);
          throw new Error(`Failed to load liked products (status: ${response.status})`);
        }
      }
      const data = await response.json();
      if (data.docs && data.docs.length > 0) {
        const fetchedLiked = data.docs[0] as Liked;
        console.log('Liked collection loaded:', fetchedLiked.id);
        return fetchedLiked;
      }
      return null;
    } catch (err: any) {
      console.error('Error fetching liked products:', err);
      setError(err.message || 'Error loading liked products.');
      return null;
    } finally {
      setIsLoading(false);
    }
  }, [user?.id, token]);
```