МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ ЕКОНОМІЧНИЙ УНІВЕРСИТЕТ
ІМЕНІ ВАДИМА ГЕТЬМАНА
НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ
«ІНСТИТУТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ В ЕКОНОМІЦІ»

Кафедра інформаційних систем в економіці

галузь знань 12 «Інформаційні технології»
спеціальність 122 «Комп'ютерні науки»

Форма навчання: денна

КВАЛІФІКАЦІЙНИЙ БАКАЛАВРСЬКИЙ ПРОЄКТ

на тему

ПРОЄКТУВАННЯ ЕЛЕКТРОННОГО МАГАЗИНУ «ТЕХНО-ТОВАРІВ»

студента Скоропадського Андрія Олександровича _____

Науковий керівник:
д.е.н., професор
_____Ріппа С. П.

Кваліфікаційний бакалаврський проєкт допущений до захисту в Екзаменаційній комісії з атестації здобувачів вищої освіти

завідувач кафедри
_____ Тішков Б.О.

Київ 2024

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ ЕКОНОМІЧНИЙ УНІВЕРСИТЕТ
ІМЕНІ ВАДИМА ГЕТЬМАНА
Навчально-науковий інститут «Інститут інформаційних технологій в економіці»
Кафедра інформаційних систем в економіці

ОСВІТНЬО-ПРОФЕСІЙНА ПРОГРАМА «КОМП'ЮТЕРНІ НАУКИ»

галузь знань	12 Інформаційні технології
спеціальність	122 Комп'ютерні науки

ПОГОДЖЕНО: Керівник проектної групи (гарант) освітньо-професійної програми	ЗАТВЕРДЖУЮ: Завідувач кафедри інформаційних систем в економіці
_____Іванченко Г.Ф. «____» _____202_ р.	_____ Тішков Б.О. «____» _____ 202_ р.

ІНДИВІДУАЛЬНЕ ЗАВДАННЯ
здобувача вищої освіти _Скоропадського Андрія Олександровича_
(прізвище, ім'я, по батькові)
_____**денної**_____ форми навчання
очної (денної), заочної, дистанційної

на підготовку кваліфікаційного бакалаврського проекту
на тему: «ПРОЄКТУВАННЯ ЕЛЕКТРОННОГО МАГАЗИНУ «ТЕХНО-ТОВАРІВ»

Тему затверджено наказом ректора Університету від «__» _____ 2024 р. №529-ст

Кваліфікаційний бакалаврський проект виконується на матеріалах
усіх доступних публікацій та Інтернет джерел

План кваліфікаційного бакалаврського проекту

Розділ І. Характеристика та аналіз предметної галузі
(назва розділу)

Розділ ІІ. Розробка вимог і моделювання інформаційної системи
(назва розділу)

Розділ ІІІ. Проектування та реалізація компонентів системи
(назва розділу)

Об'єкт дослідження: процес організації і управління е-магазином електро-товарів, який включає адміністрування і ведення БД електро-товарів, реєстрацію покупців і персоналу електронного магазину, моніторинг товарів е-магазину, тестування, підтримку різноманітних довідкових і допоміжних функцій в е-торгівлі.

Предмет дослідження: методи та технології організації е-торгівлі електро-товарами, які застосовуються для організації діяльності електронного магазину.

Мета кваліфікаційного бакалаврського проєкту: створення інформаційної системи організації і підтримки процесу Інтернет-торгівлі електро-товарами.

Конкретні завдання, які здобувач повинен виконати для досягнення поставленої мети:

У розділі I

1. Дослідження предметної області. Збір інформації та вивчення матеріалів за тематикою кваліфікаційного бакалаврського проекту.
2. Аналіз літератури та існуючих підходів до організації дистанційного навчання і розробка концепції е-магазину електро-товарів.

У розділі II

1. Аналіз і специфікація вимог до інформаційної системи е-торгівлі.
2. Формулювання функціональних і нефункціональних вимог до ІС е-магазину електро-товарів.
3. Постановка і характеристика задач е-торгівлі електро-товарами і побудова алгоритмів функціонування ІС е-торгівлі.
4. Математичне забезпечення процесів Інтернет-торгівлі електро-товрами.

У розділах III і IV

1. Проектування та реалізація інформаційного забезпечення діяльності е-магазину електро-товарів.
2. Організаційне забезпечення діяльності е-магазину.
3. Технічне забезпечення Інтернет-торгівлі електро-товарами.
4. Результати реалізації компонентів системи е-торгівлі.
5. Розробка тест-кейсів і трасування вимог до системи е-магазину.

Завдання підготував
науковий керівник _____ _____Ріппа Сергій Петрович_____
(підпис) *(ініціали, прізвище)*

«_06_» __лютого____ 2024 р.

Завдання одержав
здобувач _____ _____Скоропадський Андрій Олександрович____
(підпис) *(ініціали, прізвище)*

«_07_» __лютого____ 2024 р.

АНОТАЦІЯ

кваліфікаційного бакалаврського проєкту студента 4 курсу
Навчально-наукового інституту
«Інститут інформаційних технологій в економіці»
Скоропадського Андрія Олександровича, виконаного на тему:
«Проектування електронного магазину техно-товарів»
Київ: кафедра інформаційних систем в економіці, 2024 р.

Кваліфікаційний бакалаврський проект присвячений розробці електронного магазину техно-товарів. Актуальність теми зумовлена стрімким розвитком електронної комерції та зростаючим попитом на електроніку та гаджети в Україні. Проект спрямований на аналіз ринку та потреб цільової аудиторії з метою створення зручного та надійного онлайн-середовища для покупки техно-товарів.

Основна ідея проекту полягає в розробці ефективного інструменту для підтримки продажу, який забезпечить високий рівень задоволення клієнтів та оптимізацію внутрішніх процесів логістики та управління запасами. Для досягнення цієї мети було проведено комплексний аналіз ринку електронних магазинів, досліджено потреби цільової аудиторії та визначено основні функціональні та нефункціональні вимоги до системи.

В рамках проекту розроблено архітектуру додатку, спроектовано базу даних та обрано технологічний стек для фронтенду та бекенду. Для забезпечення швидкості та адаптивності інтерфейсу використано сучасні фреймворки та бібліотеки, такі як Vue.js та Bootstrap. Бекенд системи побудовано на стабільному та безпечному фреймворку Spring Boot, що дозволяє легко масштабувати проект в майбутньому.

Насамперед, в проекті приділяється увага забезпеченню безпеки та захисту персональних даних користувачів. Впроваджено систему аутентифікації та авторизації, а також використано сучасні методи шифрування даних. Крім того, здійснено оптимізацію користувацького досвіду та інтерфейсу для підвищення конверсії відвідувань в покупки, враховуючи принципи

SEO-оптимізації.

Новизна проекту полягає в комплексному підході до створення електронного магазину техно-товарів з урахуванням специфіки українського ринку. Використання сучасного стеку технологій, зокрема Vue.js та Spring Boot, дозволило створити масштабовану, надійну та безпечну платформу з високим рівнем користувацького досвіду. Практична значимість проекту полягає в можливості його використання як основи для створення реального електронного магазину техно-товарів, а також в отриманні практичних навичок розробки сучасних електронних магазинів та аналізу потреб ринку.

РЕФЕРАТ

Кваліфікаційний бакалаврський проєкт 90 сторінки, 5 таблиць, 52 рисунків, перелік джерел посилань з 30 найменувань, 4 додатки.

«Проектування електронного магазину техно-товарів»

Перелік ключових слів: *Електронний магазин, техно-товари, Java, Spring Boot, Vue.js, MySQL, електронна комерція, онлайн-продаж, каталог товарів, оформлення замовлення, оплата, доставка, особистий кабінет, адміністрування, безпека, UML, SySML, Enterprise Architect, тест-кейси.*

Об’єктом розроблення є процес створення прототипу та функціонування електронного магазину техно-товарів в умовах сучасного українського ринку.

Мета кваліфікаційного бакалаврського проекту – розробка функціонального прототипу електронного магазину техно-товарів, який буде відповідати сучасним вимогам ринку, забезпечить високий рівень сервісу для клієнтів та ефективність управління для адміністраторів.

Методами розроблення кваліфікаційного бакалаврського проекту є:

- визначення вимог до системи, аналізу предметної області та розробки архітектури системи;
- застосування методів об'єктно-орієнтованого програмування для проектування та реалізації системи;
- моделювання бізнес-процесів для візуалізації та аналізу основних процесів роботи електронного магазину;
- використання UML-діаграм (Use Case, послідовності, класів, станів) та SysML-діаграм для візуалізації та документування системи;
- прототипування користувацького інтерфейсу в Figma для візуалізації дизайну;

- тестування для перевірки коректності роботи системи та її відповідності вимогам;
- створення інформаційної системи в середовищі розробки IntelliJ IDEA з використанням бази даних MySQL.

Апаратура використана при розробленні кваліфікаційного бакалаврського проєкту є персональний комп’ютер.

Розроблена система може бути використана для створення реального електронного магазину, подальших досліджень в галузі електронної комерції та для навчання студентів. Нововведення та підходи, розроблені в проекті, можуть слугувати базою для подальшого вдосконалення платформи електронного магазину техно-товарів. Включаючи розширення функціоналу, інтеграцію з додатковими сервісами, впровадження нових технологій (наприклад, штучного інтелекту для персоналізації рекомендацій) та оптимізацію користувацького досвіду на основі аналізу поведінки користувачів.

Рік виконання кваліфікаційного бакалаврського проєкту – 2024. Рік захисту кваліфікаційного бакалаврського проєкту – 2024.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ СКОРОЧЕНЬ

БД	База даних
ООП	Об’єктно-орієнтований підхід
ОС	Операційна система
ІС	Інформаційна Система
СКБД	Система керування базами даних
API	Application Programming Interface
BDD	Block definition diagrams
CSS	Cascading Style Sheets
HTML	Hyper Text Markup Language
MVC	Model View Controller
IBD	Internal Block diagram
SQL	Structured Query Language
UML	Unified Modeling Language
SEO	Search Engine Optimization
UI	User Interface
JS	JavaScript
DTO	Data Transfer Object
SysML	Systems Modeling Language
OPD	Object-Process Diagram

ВСТУП

З набуттям обертів електронної комерції, продаж товарів через Інтернет стає не лише зручним, але й переважним способом здійснення торгівлі. Електронний магазин електро-товарів відкриває можливості для швидкого та ефективного споживчого досвіду, дозволяючи користувачам легко порівнювати ціни, отримувати інформацію про товари та читати відгуки. Важливість такого роду магазинів підкреслюється стрімким зростанням української аудиторії та збільшенням попиту на новітню електроніку.

Актуальність теми кваліфікаційної бакалаврської роботи «Проектування електронного магазину техно-товарів» зумовлена необхідністю розробки комплексного рішення, яке відповідало б сучасним вимогам ринку та водночас було б зручним у користуванні як для клієнтів, так і для працівників.

Практичне значення проекту полягає в можливості його застосування для продажу електроніки в Інтернеті, що відкриває шлях для оптимізації торгівлі та розширення бізнесу. Результати цієї роботи мають на меті створення функціонального та конкурентоспроможного електронного магазину з можливістю адмініструвати магазин та розширювати можливості.

Метою даної кваліфікаційної роботи є розробка проєкту електронного магазину техно-товарів, який забезпечить високий рівень користувацького досвіду та відповідає сучасним вимогам ринку. Для досягнення цієї мети необхідно вирішити наступні завдання:

Дослідити предметну область, визначити вимоги до системи.

- проаналізувати предметну область для якої створюється електронний магазин;

- визначити вимоги (бізнес-вимоги, функціональні та не функціональні) та їх специфікації;
- обґрунтувати вибір програмних рішень для розроблення електронного магазину;
- розробити інформаційну модель системи та описати вхідні і вихідні дані;
- розробити функціональну модель системи;
- розробити структурний алгоритм роботи;
- спроєктувати архітектуру електронного магазину;
- спроєктувати базу даних;
- спроєктувати користувацький інтерфейс для електронного магазину;
- скласти тест-кейси для перевірки вимог системи;
- представити прототип в Figma;
- представити проєкт системи в IntelliJ IDEA.

Об’єктом дослідження є процес електронної комерції, який включає в себе онлайн-продаж техно-товарів. Предметом дослідження є інформаційна система електронного магазину, що включає в себе функціональні та нефункціональні вимоги до системи, а також її архітектурні та технологічні аспекти.

Для досягнення поставленої мети було використано ряд методів дослідження, включаючи:

- системний аналіз, який допоміг визначити основні компоненти системи та їх взаємодію;
- моделювання бізнес-процесів дозволило описати потоки даних і процеси в системі;
- проектування баз даних забезпечило ефективне зберігання та обробку інформації.

Наукова новизна дослідження полягає в розробці комплексного проекту електронного магазину техно-товарів, який враховує сучасні тенденції в галузі електронної комерції та забезпечує високий рівень користувацького досвіду.

Отримані результати мають як теоретичне, так і практичне значення. Теоретично, робота узагальнює сучасні підходи до проектування інформаційних систем для електронної комерції, розширює знання про ефективні методи організації бізнес-процесів та управління даними. Практично, розроблена система може бути впроваджена для реального продажу техно-товарів в Інтернеті, що сприятиме розвитку електронної комерції в Україні.

Використані інструментальні засоби: середовище проектування IntelliJ IDEA Ultimate 2023.2.1, СКБД MySQL Workbench 8.0 CE, HTML5, CSS, JavaScript, Vuejs, BootStrap, WebPack, jQuery, Draw.IO, Enterprice Architect, OPCAT, операційна система Windows 10, текстовий редактор Microsoft Word 2016 для підготовки та оформлення пояснювальної записки до кваліфікаційного бакалаврського проєкту.

Структура роботи зумовлена метою і завданнями та складається зі вступу, чотирьох розділів, висновку, переліку джерел посилання та додатків.

РОЗДІЛ 1 ХАРАКТЕРИСТИКА ТА АНАЛІЗ ПРЕДМЕТНОЇ ГАЛУЗІ

1.1 Характеристика предметної галузі та об’єкта дослідження

Предметною галуззю дослідження є електронна комерція (e-commerce), зокрема, її сегмент, пов'язаний з онлайн-продажем техно-товарів.

Предметна галузь характеризується:

- постійним технологічним оновленням на ринку, де регулярно з'являються нові гаджети та технології, що зумовлює необхідність постійного оновлення асортименту та адаптації до вимог сучасного споживача;
- високою інформованістю покупців, насамперед користувачі, які активно вивчають характеристики, порівнюють ціни та читають відгуки перед покупкою, що вимагає від електронних магазинів надання вичерпної та достовірної інформації;
- на конкурентоспроможність електронного магазину впливають такі фактори, як швидка доставка, гнучкі умови оплати, програми лояльності та якісне після продажне обслуговування.

Об'єктом дослідження виступає процес створення та функціонування електронного магазину техно-товарів в умовах сучасного українського ринку, що охоплює етапи від формування концепції та проектування системи до її реалізації, просування та взаємодії з клієнтами.

Предметом дослідження є інформаційна система електронного магазину техно-товарів, яка виступає основою для його ефективної роботи. В рамках кваліфікаційного бакалаврського проєкту будуть розглянуті такі елементи:

- аналіз ринку та визначення цільової аудиторії магазину;

- формування унікальної торгової пропозиції та конкурентних переваг;
- проектування зручної та інтуїтивно зрозумілої структури сайту та каталогу товарів;
- вибір оптимального технологічного стеку для розробки платформи;
- забезпечення безпеки;
- можливість швидко редагувати товарні картки за допомогою адмін панелі;
- розробка стратегії просування електронного магазину.

Дослідження спрямоване на розробку комплексного та конкурентоспроможного електронного магазину техно-товарів, здатного задовольнити потреби вимогливих сучасних покупців та спростити роботу працівникам магазину.

Сучасний світ важко уявити без інтернет-торгівлі, яка швидко еволюціонує та кардинально змінює традиційні способи здійснення покупок. Електронні магазини відкривають доступ до величезного асортименту товарів і послуг, значно перевершуючи можливості фізичних магазинів. Споживачі можуть легко переглядати та купувати будь-який товар, отримувати детальну інформацію про нього та читати відгуки інших користувачів, що робить процес покупки зручним і прозорим [1].

Завдяки електронним магазинам, продаж електро-техніки стає швидким та доступним процесом, який вигідно відрізняється зручністю та доступністю для широкої аудиторії. Основною перевагою таких магазинів є можливість автоматизації процесів пошуку і оплати товарів, що знижує витрати на персонал

та дозволяє вести бізнес з будь-якої точки світу за наявності доступу до Інтернету. Ця гнучкість та ефективність відкриває нові можливості для підприємців та спрощує доступ споживачів до необхідних товарів [1].

Український ринок інтернет-торгівлі перебуває на етапі активного розвитку. Протягом останнього десятиліття кількість інтернет-магазинів значно зросла, а обіг електронної комерції досяг мільярдних значень [2]. Така динаміка сприяє використанню та вдосконаленню існуючих технологій, а також стимулює конкуренцію на ринку, яка вимагає від підприємств не лише мінімізації цін, але й підвищення якості обслуговування.

Однак, інтернет-торгівля має й свої недоліки, зокрема, ризик натрапити на шахраїв, можливі помилки в програмному забезпеченні та труднощі з залученням клієнтів. Часто користувачі відвідують електронні магазини, але не завжди здійснюють покупки, що створює виклики для бізнесу в плані конверсії відвідувачів у покупців. Для подолання цих викликів необхідно забезпечити надійну, зручну та безпечну платформу для онлайн-покупок [22].

Предметною областю нашого кваліфікаційного бакалаврського проєкту є проектування електронного магазину електро-техніки, який забезпечить надійну, зручну та безпечну платформу для здійснення онлайн-покупок.

Мета полягає в тому, щоб навчитися створювати електронний магазин, який відповідатиме сучасним вимогам ринку, забезпечуватиме високий рівень обслуговування та ефективність управління. Завдання проєкту включають дослідження ринку, визначення вимог до системи, вибір програмних рішень, розробку моделі системи, проектування архітектури та бази даних, дизайн інтерфейсу, створення прототипу, розроблення тест-кейсів та перевірки роботоздатності магазину. Результати проекту мають практичне значення, оскільки спроектований електронний магазин може бути використаний для реалізації товарів на ринку інтернет-торгівлі, який також може спростити роботу працівникам.

1.2 Аналіз літературних джерел та практичного досвіду використання ІС і технологій в предметній галузі

В області продажу електро-техніки одні із основних позицій займають такі електронні магазини як Moyo.ua, Citrus.ua, Foxtrot.com.ua., Rozetka.ua Розглянемо їх.

1.2.1 Дослідження існуючих інформаційних систем для обраної предметної галузі.

Загальна характеристика MOYO:

Аналізуючи інформацію про компанію MOYO, хочу відзначити її важливість як одного з провідних гравців на ринку електроніки та побутової техніки в Україні. MOYO – це українська мережа магазинів та інтернет-магазинів, яка надає широкий асортимент споживчих товарів, представлених як в офлайн, так і в онлайн середовищі [33].

Однією з ключових переваг MOYO.UA є використання формату магазину 24/7, що дозволяє клієнтам робити замовлення та забирати їх у зручний для них час. Це особливо актуально в умовах сучасного ритму життя, коли багато людей мають обмежений час для здійснення покупок [33].

Крім того, компанія регулярно проводить різноманітні акції, як щотижневі, так і святкові, що дозволяє клієнтам економити на покупках. Для підвищення лояльності клієнтів MOYO пропонує реєстрацію на сайті, що дає можливість отримувати повідомлення про акції та спеціальні пропозиції.

Особливу увагу заслуговує система обслуговування клієнтів у разі несправності товару. MOYO забезпечує можливість повернення товару, його перевірки, налагодження або заміни [33]. Це свідчить про високий рівень сервісу та прагнення компанії задовольнити потреби своїх клієнтів.

На основі проведеного аналізу, можна зробити висновок, що MOYO.UA успішно поєднує сучасні технології, високий рівень сервісу та орієнтованість на клієнта, що робить її одним з лідерів на ринку електроніки та побутової техніки в Україні.

Загальна характеристика Цитрус:

Аналізуючи загальну характеристику компанії "Цитрус", варто відзначити, що це один з найсучасніших магазинів електроніки та аксесуарів в Україні. Сучасний формат та миттєве обслуговування клієнтів дозволили компанії швидко завоювати довіру на ринку гаджетів та електронних девайсів [34].

На сьогоднішній день в Україні діє понад 60 магазинів мережі "Цитрус", де представлені топові моделі смартфонів, ноутбуків та іншої техніки. Компанія працює в двох основних напрямках: інтернет-магазин розумних гаджетів та мережа офлайн-магазинів [34]. Такий підхід дозволяє клієнтам обирати зручний формат покупок: можна особисто відвідати магазин, щоб побачити товар та вивчити його характеристики, або ж заощадити час і зробити покупку в інтернеті.

Компанія створила зручні умови для кожного клієнта. Покупки на сайті дозволяють забрати техніку в обраному магазині або скористатися послугами різних поштових служб, таких як Укрпошта, Нова Пошта, Meets, Justin, або кур'єрською доставкою. Це забезпечує максимальну зручність та гнучкість для клієнтів [34].

Головна сторінка сайту, яка є відображенням сучасного підходу компанії до онлайн-продажів та забезпечення клієнтів усією необхідною інформацією для вибору та придбання техніки.

"Цитрус" поєднує в собі інноваційні рішення, сучасний формат обслуговування та орієнтованість на клієнта, що робить його одним з лідерів на ринку електроніки та аксесуарів в Україні.

Загальна характеристика Фокстрот*:*

Foxtrot.com.ua – це відома мережа роздрібних магазинів та інтернет-платформа, яка пропонує різноманітні електро-товари та техніку в Україні. Бренд утворився 25 років тому і з того часу активно розвивався, відкривши понад 228 супермаркетів "Фокстрот" по всій країні. За цей час компанія зуміла створити потужний та ефективний магазин, який пропонує широкий асортимент товарів [31].

Асортимент онлайн-магазину Foxtrot.com.ua охоплює всі товарні категорії, доступні в стаціонарних магазинах [31]. Це включає в себе електроніку, побутову техніку, гаджети та інші споживчі товари, що робить магазин універсальним місцем для покупок.

У 2019 році компанія розробила нову стратегію, яка торкнулася всіх аспектів діяльності мережі [31]. Ця стратегія включала оновлення фірмового стилю, впровадження численних варіантів доставки та покращення роботи консультантів. Завдяки цим змінам "Фокстрот" зміг підвищити рівень обслуговування клієнтів, забезпечивши зручність і якість сервісу на найвищому рівні.

Foxtrot.com.ua поєднує багаторічний досвід, сучасні рішення та орієнтацію на потреби клієнтів, що робить його одним з провідних ритейлерів електроніки та техніки в Україні.

Розетка – це найбільший та найпопулярніший інтернет-магазин в Україні, що спеціалізується на продажу електроніки та техніки. Заснований у 2005 році, він швидко став ключовим учасником електронної комерції в країні, завдяки широкому асортименту товарів та високій якості обслуговування [32].

Асортимент товарів, який пропонує Розетка, охоплює різні категорії, включаючи електроніку, техніку для дому та спортивні товари. Кожен товар забезпечується гарантією та можливістю повернення, що забезпечує додаткову безпеку покупок для клієнтів. Участь у клубі Розетка+ надає додаткові переваги, такі як знижки та спеціальні пропозиції для учасників [32].

Загальна характеристика Rozetka:

Онлайн-платформа Розетка вирізняється зручним інтерфейсом та високою технічною якістю сайту, що робить процес покупок комфортним та швидким. Клієнти можуть обирати з різних варіантів доставки, включаючи самовивіз з найближчих пунктів видачі.

Розетка регулярно проводить різноманітні акції та розпродажі, що робить покупки ще вигіднішими для клієнтів [32]. Додатковою перевагою є можливість перегляду рейтингів та відгуків покупців, що допомагає іншим користувачам у виборі товару.

Активна присутність в соціальних мережах та місія забезпечення клієнтів високоякісними товарами та послугами сприяють тому, що Розетка залишається лідером електронної комерції в Україні. Компанія постійно вдосконалюється та адаптується до потреб своїх клієнтів, що підтверджує її статус провідного інтернет-магазину в країні.

Ці платформи є провідними на ринку електро-товарів в Україні та пропонують різноманітні товари, конкуруючи як за асортиментом, так і за обслуговуванням та акційними пропозиціями.

1.2.2 Порівняння інформаційних систем

Результати порівняння програмних систем наведені в табл. 1.1, в якій за рядками стоять характеристики, упорядковані у групи, за стовпцями – системи, а зміст таблиці надає можливість порівняти системи, сайти:

Таблиця 1.1 – Порівняння програмних систем

№	*Характеристика*	*Rozetka*	*Moyo.ua*	*Citrus.ua*	*Foxtrot.com.ua*
Зручність використання систем/сайтів					
1	Інтерфейс та навігація	Зручний інтерфейс, легка навігація	Зручний інтерфейс, легка навігація	Інтуїтивний інтерфейс, зручний пошук	Зручний інтерфейс, легка навігація

2	Мобільна версія	Є мобільний додаток, оптимізований сайт	Є мобільний додаток, оптимізований сайт	Оптимізована мобільна версія	Є мобільний додаток
3	Пошук товарів	Ефективний пошуковий інструмент	Швидкий та ефективний пошук	Зручний пошук товарів	Широкий функціонал пошуку
4	Оформлення замовлення	Простий та зрозумілий процес оформлення	Зручний процес замовлення	Швидке та зручне оформлення	Простий та швидкий процес
5	Можливість відгуків	Легка можливість залишати та переглядати відгуки	Можливість залишати та переглядати відгуки	Збір відгуків, рейтинги	Інтеграція відгуків та рейтингів
6	Опції доставки	Різні варіанти доставки, включаючи самовивіз	Різні варіанти доставки, самовивіз	Доставка товарів, самовивіз	Різні варіанти доставки
7	Клієнтська підтримка	Доступна онлайн та офлайн підтримка	Опції онлайн та телефонної підтримки	Контактні центри, чат-підтримка	Опції онлайн та телефонної підтримки
Подання інформації					
8	Чіткість та зрозумілість інформації	Так	Так	Так	Так
9	Доступність детальної інформації про товар	Так	Так	Так	Так
10	Інформативність описів товарів	Так	Так	Так	Так
11	Наявність відгуків і рейтингів на сайті	Так	Так	Так	Так

Продовження Таблиці 1.1

12	Розміщення відгуків поруч з товарами	Так	Так	Ні	Так
13	Наявність фото та відео товарів	Так	Так	Так	Так
14	Зручність порівняння товарів	Так	Так	Так	Ні
Спілкування (Взаємодія користувача)					
15	Доступність чат-підтримки	Так	Так	Так	Так

16	Відповіді на запитання користувачів відзначені	Так	Так	Так	Так
17	Доступність телефонної підтримки	Так	Так	Так	Так
18	Швидкість відповіді на електронні листи	Так	Ні	Так	Так
19	Наявність онлайн-консультанта на сайті	Так	Так	Так	Так
20	Якість інструкцій та допомоги від сайту	Так	Так	Так	Ні
21	Реакція на негативні відгуки та скарги	Так	Так	Так	Так
Функціональні можливості					
22	Можливість фільтрації товарів	Так	Так	Так	Так
23	Опції порівняння товарів	Так	Так	Так	Так
24	Наявність розширених пошукових опцій	Так	Так	Так	Так
25	Можливість створення списків бажань	Так	Ні	Так	Так
26	Система відслідковування цін на товари	Так	Так	Так	Так

Продовження Таблиці 1.1

27	Автоматичні рекомендації товарів	Так	Так	Так	Так
28	Наявність мобільного додатку	Так	Так	Так	Так
29	Наявність системи	Так	Так	Так	Ні

	відгуків на товари				
30	Можливість замовлення товару без реєстрації	Так	Так	Ні	Так
Інноваційні можливості					
31	Використання розширеної реальності	Так	Ні	Так	Ні
32	Впровадження штучного інтелекту	Так	Так	Ні	Так
33	Роботизоване обслуговування	Ні	Так	Ні	Так
34	Екологічні ініціативи в упаковці	Так	Ні	Ні	Так
35	Використання блокчейн-технологій	Ні	Так	Ні	Так

1.2.3 Позитивні характеристики існуючих систем, що їх слід відтворити у проектних рішеннях

Висновки за результатами аналізу програмних систем (позитивні характеристики існуючих систем, що їх слід відтворити у проектних рішеннях та недоліки існуючих систем та напрями їх удосконалення):

1. Чіткість та зрозумілість інформації:

Всі чотири системи – Rozetka, Moyo.ua, Citrus.ua, Foxtrot.com.ua – відзначаються чітким та доступним поданням інформації для користувачів. Це важливий елемент для забезпечення комфортного взаємодії з електронним магазином.

2. Доступність чат-підтримки:

Усі чотири системи надають можливість взаємодії з чат-підтримкою, що сприяє ефективній комунікації з користувачами та вирішенню їхніх проблем у режимі реального часу.

3. Зручність оформлення замовлення:

Процес оформлення замовлення у всіх системах є простим та зрозумілим, що забезпечує зручність для покупців та слугує прикладом для проектних рішень.

4. Можливість порівняння товарів:

Розетка, Moyo.ua, Citrus.ua, та Foxtrot.com.ua надають користувачам можливість порівняння товарів, що сприяє більш обдуманому вибору товару та підвищує задоволення від покупок.

5. Доступність мобільного додатку:

У всіх чотирьох систем є мобільні додатки, що робить процес покупок більш доступним та зручним для користувачів, які використовують смартфони.

1.2.4 Недоліки існуючих систем та напрями їх удосконалення

Недоліки існуючих систем та напрями їх удосконалення:

1. Наявність розширеної реальності та використання ШІ:

Розетка та Moyo.ua вже використовують розширену реальність та штучний інтелект. Однак, напрямок для удосконалення включає розгортання цих технологій у Citrus.ua та Foxtrot.com.ua для поліпшення користувацького досвіду.

2. Роботизоване обслуговування та використання блокчейн-технологій:

Citrus.ua та Foxtrot.com.ua можуть удосконалити свої системи, впроваджуючи роботизоване обслуговування та використання блокчейн-технологій для покращення ефективності та надійності процесів.

3. Наявність системи відгуків на товари та відповідь на негативні відгуки:

Розетка вже використовує систему відгуків. Проте, системи Citrus.ua та Foxtrot.com.ua можуть збільшити довіру користувачів, вдосконаливши систему відгуків та ефективніше реагуючи на негативні відгуки.

4. Можливість замовлення товару без реєстрації:

Можливість замовлення без обов'язкової реєстрації є позитивною рисою для користувачів. Таку можливість можна було б впровадити у Citrus.ua для зручності користувачів.

5. Екологічні ініціативи в упаковці:

Застосування екологічних ініціатив у роботі з упаковкою є актуальним напрямком для удосконалення в усіх системах, зокрема у Moyo.ua та Foxtrot.com.ua.

Таблиця 1.2 – Розглянуті характеристики існуючих систем та напрями для їх подальшого удосконалення у проектних рішеннях

№	*Характеристика*	*Rozetka*	*Moyo.ua*	*Citrus.ua*	*Foxtrot.com.ua*
1	Чіткість та зрозумілість інформації	Позитивна, варто підтримувати	Позитивна, варто підтримувати	Позитивна, забезпечити сталу чіткість	Позитивна, слід розширювати функціонал
2	Доступність чат-підтримки	Відмінна, можливе розширення	Ефективна, слід підтримувати	Потребує покращень, бажано реалізувати 24/7	Добра, можна вдосконалити інтерфейс
3	Зручність оформлення замовлення	Добра, важливо тримати рівень	Висока, слід підтримувати	Позитивна, можна удосконалити процес	Висока, працювати над автоматизацією
4	Можливість порівняння товарів	Ефективна, варто покращувати	Добра, тримати на високому рівні	Є, але можна розширити функціонал	Висока, можна додавати нові функції
5	Доступність мобільного додатку	Є, працювати над функціоналом	Є, слід вдосконалювати	Є, можна робити оновлення	Є, працювати над швидкодією та функціоналом
6	Використання розширеної реальності	Так, тримати на високому рівні	Ні, слід додавати функціонал	Ні, варто розглядати впровадження	Ні, слід вивчати можливості
7	Впровадження штучного інтелекту	Так, можна подальше вдосконалення	Так, працювати над інтеграцією	Ні, слід досліджувати можливості	Так, вдосконалювати алгоритми
8	Роботизоване обслуговування	Ні, можливо вивчати можливості	Так, можна розширювати функціонал	Ні, важливо вивчати переваги	Так, працювати над автоматизацією
9	Екологічні ініціативи в упаковці	Є, слід покращувати	Немає, можна додавати функціонал	Немає, варто вивчати можливості	Є, слід збільшувати участь у

		та розширювати			екологічних програмах
10	Наявність системи відгуків на товари	Так, важливо тримати високий рівень	Так, слід розглядати можливості	Немає, можна розглядати впровадження	Ні, важливо додавати і вдосконалювати

Продовження Таблиці 1.2

11	Можливість замовлення товару без реєстрації	Так, тримати рівень та покращувати	Так, важливо підтримувати	Ні, можна додавати функціонал	Так, важливо робити процес замовлення простим
12	Використання блокчейн-технологій	Немає, вивчати можливості впровадження	Так, розглядати можливості	Немає, варто вивчати переваги	Так, розглядати можливості інтеграції
13	Використання технологій для зменшення впливу на довкілля	Так, покращувати та розширювати	Ні, варто досліджувати можливості	Ні, важливо досліджувати переваги	Так, розглядати можливості використання новітніх технологій
14	Інноваційні можливості	Так, слід розширювати та досліджувати	Так, важливо підтримувати рівень	Ні, можна розглядати впровадження	Так, важливо бути в лідерах інновацій

Таблиця надає інформацію про позитивні характеристики та недоліки обраних систем, а також напрями для їх удосконалення.

1.3 Розробка концепції інформаційної системи

В рамках кваліфікаційного бакалаврського проекту «Проектування електронного магазину техно-товарів» ключовим завданням є розробка концепції інформаційної системи, яка б забезпечила ефективну реалізацію всіх функціональних вимог та відповідала сучасним стандартам веб-розробки [23].

З метою забезпечення гнучкості, масштабованості та зручності супроводження системи було обрано архітектурний підхід MVC

(Model-View-Controller). Даний підхід дозволяє чітко розмежувати бізнес-логіку (Model), користувацький інтерфейс (View) та логіку управління (Controller), що спрощує розробку, тестування та подальшу модернізацію системи [8].

Для реалізації бекенду проекту було обрано мову програмування Java у поєднанні з фреймворком Spring Boot. Java, будучи мовою зі строгою типізацією та великою кількістю бібліотек, забезпечує створення надійного та високопродуктивного коду [10]. Spring Boot, в свою чергу, дозволяє автоматизувати значну частину конфігурації веб-додатку, спростити роботу з базами даних та інтеграцію з іншими сервісами.

В якості системи управління базами даних (СУБД) було обрано MySQL. Дана СУБД є відкритою, широко розповсюдженою та відрізняється високою продуктивністю, надійністю та підтримкою транзакцій, що є критично важливими факторами для електронного магазину [9].

Для відображення динамічного контенту на стороні сервера та створення шаблонів веб-сторінок буде використано Thymeleaf. Цей механізм шаблонів легко інтегрується з Spring Boot та дозволяє створювати зручні та ефективні веб-сторінки з динамічними даними [11].

Для створення інтерактивного та динамічного користувацького інтерфейсу (UI) електронного магазину буде використано прогресивний JavaScript-фреймворк Vue.js [12]. Цей фреймворк надає розробникам потужний інструментарій для побудови складних веб-додатків, зберігаючи при цьому простоту та легкість в освоєнні.

Vue.js дозволяє створювати гнучкі та легковагові компоненти інтерфейсу, що можуть бути багаторазово використані на різних сторінках магазину, забезпечуючи узгодженість дизайну та прискорюючи розробку. Реактивність Vue.js автоматично оновлює інтерфейс при зміні даних, роблячи його динамічним та інтерактивним [12].

Для структурування та стилізації веб-сторінок буде використано HTML та CSS відповідно [15]. JavaScript забезпечить інтерактивність веб-сторінок та

обробку подій на стороні клієнта. Додатково, для пришвидшення розробки та забезпечення консистентності дизайну буде використано Bootstrap – популярний фреймворк для фронтенду [13].

Поєднання Vuejs, Thymleaf, HTML, CSS, JavaScript та Bootstrap у фронтенді нашого проекту створює сильну основу для розробки інтерактивного, зручного для користувача та адаптивного інтерфейсу електронного магазину [16].

Візуалізація концепції інформаційної системи та прототипування інтерфейсу користувача буде здійснено за допомогою онлайн-сервісу Figma, що дозволяє створювати інтерактивні макети веб-сторінок [18].

Обраний стек технологій та архітектурний підхід дозволять створити електронний магазин, який буде надійним та безпечним завдяки використанню перевірених технологій та забезпечення захисту даних, гнучким та масштабованим завдяки можливості легкого розширення функціоналу та збільшення навантаження та зручним та інтуїтивно зрозумілим, як для адміністраторів, так і для кінцевих користувачів.

В цілому, запропонована концепція інформаційної системи є сучасним та ефективним рішенням для створення конкурентоспроможного електронного магазину техно-товарів.

РОЗДІЛ 2 РОЗРОБКА ВИМОГ І МОДЕЛЮВАННЯ ІНФОРМАЦІЙНОЇ СИСТЕМИ

2.1 Аналіз і специфікація вимог до інформаційної системи

Аналіз і специфікація вимог до інформаційної системи є ключовим етапом в процесі її розробки, оскільки від цього залежить успішність впровадження та функціонування кінцевого продукту. На цьому етапі потрібно дослідити, які саме проблеми повинна вирішити система та як вона буде взаємодіяти з користувачами. Збір вимог включає в себе не тільки технічні аспекти, але й бізнес-цілі, які система має підтримувати.

Розробка ефективної та конкурентоспроможної ІС електронного магазину техно-товарів має ключове значення для визначення функціональних можливостей системи, її обмежень, а також для формування чіткого бачення кінцевого продукту.

Процес аналізу розпочався з формулювання функціональних вимог до системи. До них відносяться: управління каталогом товарів, реалізація процесу оформлення та обробки замовлень, система автентифікації та реєстрації користувачів, управління замовленнями, забезпечення зворотного зв'язку з клієнтами та адміністративна панель [30].

Наступний крок – це визначення нефункціональних вимог, які безпосередньо не пов'язані з функціоналом системи, але впливають на її якість та привабливість для користувачів. До них належать: висока продуктивність та швидкість роботи системи, забезпечення надійності та безпеки даних, інтуїтивно зрозумілий та зручний інтерфейс, кросбраузерність [3, 29].

Важливим аспектом є визначення бізнес-цілей та завдань, які має вирішувати система. Було проведено аналіз поточного стану ринку електронної комерції в Україні, вивчено потреби цільової аудиторії, а також проаналізовано

досвід конкурентів [5].

Не менш важливим є забезпечення безпеки інформаційної системи та захисту даних користувачів. Це передбачає використання сучасних методів аутентифікації, шифрування даних, захист від несанкціонованого доступу.

Функціональні вимоги до системи

Функціональні вимоги визначають основні функції та можливості, які повинна мати система, щоб відповідати потребам користувачів та бізнес-цілям. Вони визначають, що система повинна робити і які операції вона повинна виконувати, коли її запитують користувачі або інші системи.

Для нашого електронного магазину техно-товарів функціональні вимоги передбачають:

1. *Інтерфейс для перегляду каталогу товарів:*

Магазин повинен мати зручний інтерфейс, який дозволяє користувачам швидко знаходити необхідні товари, переглядати їх описи, характеристики та зображення.

2. *Система автентифікації та реєстрації користувачів:*

Для здійснення покупок та зберігання особистої інформації користувачі мають мати можливість створити обліковий запис в магазині, а також увійти в нього з метою перегляду статусу замовлень.

3. *Інтеграція з платіжними системами:*

Магазин повинен підтримувати різні методи оплати, такі як кредитні картки, електронні гаманці, для зручності користувачів та забезпечення безпеки операцій.

4. *Управління замовленнями:*

Система повинна забезпечувати можливість прийому, обробки та відстеження замовлень.

5. *Рекомендаційна система:*

Для покращення користувацького досвіду магазин може використовувати рекомендаційну систему, яка аналізує поведінку користувачів та рекомендує їм товари, які можуть їх зацікавити.

6. *Адміністративна панель:*

Для управління каталогом товарів, замовленнями, користувачами та іншими аспектами магазину адміністраторам має бути доступна спеціальна адміністративна панель.

7. Безпека та конфіденційність:

Магазин повинен гарантувати захист персональних даних користувачів та безпеку фінансових транзакцій, використовуючи захист від несанкціонованого доступу та шифрування даних.

Детальніше переглянути функціональні вимоги можна на діаграмі, що зображена на рис. 2.1 та специфікації вимог зроблені за допомогою менеджера специфікацій (Specification Manager) Enterprise Architect [19], що зображені на рис. 2.2-2-5.

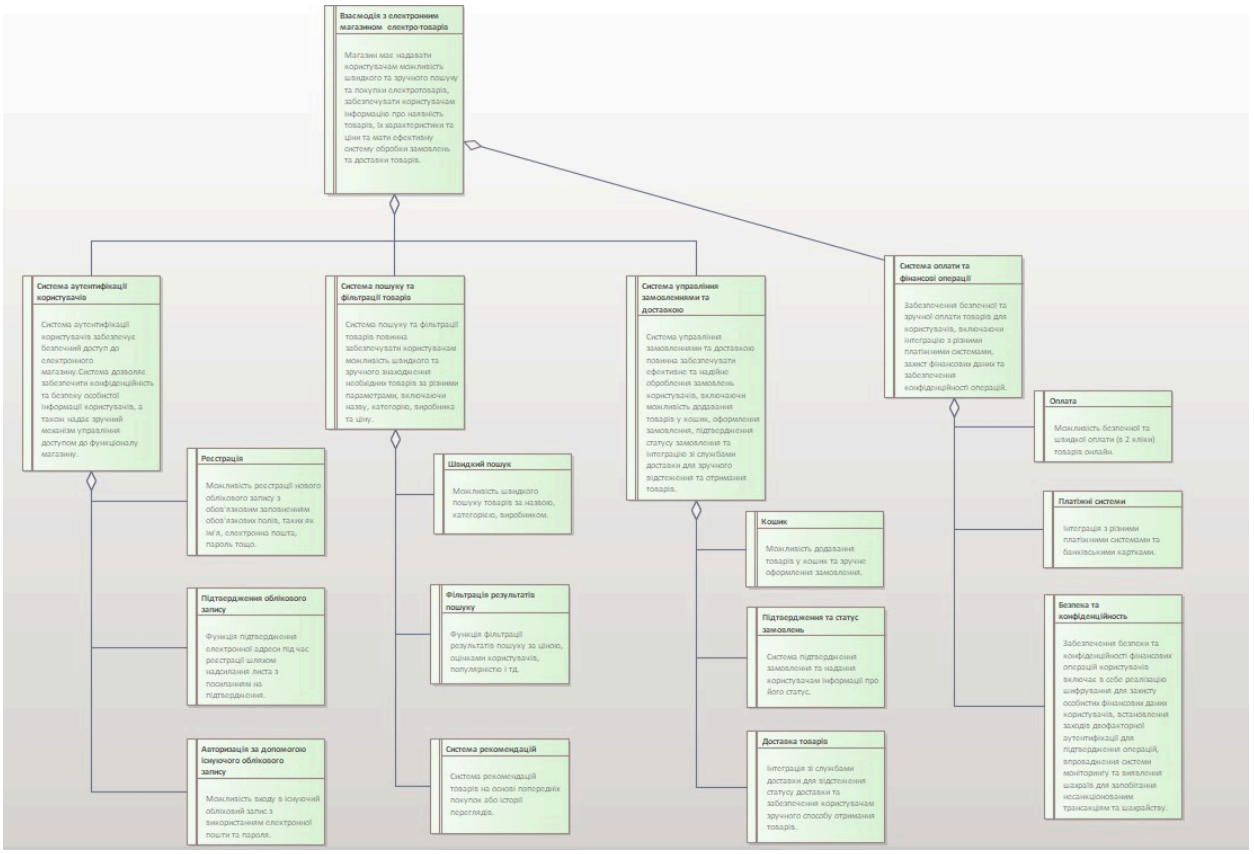


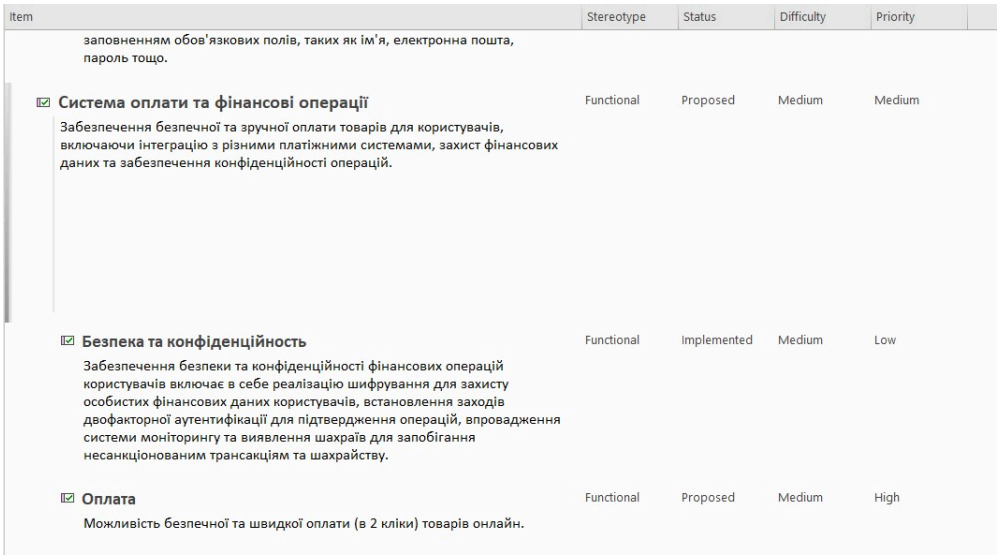
Рисунок 2.1 – Діаграма функціональних вимог

Нефункціональні вимоги до системи

Нефункціональні вимоги описують характеристики системи, такі як її надійність, продуктивність, безпека та інші аспекти, які не є прямо пов'язаними з функціональними можливостями, але впливають на загальну якість та ефективність системи. Для нашого електронного магазину техно-товарів нефункціональні вимоги можуть включати:

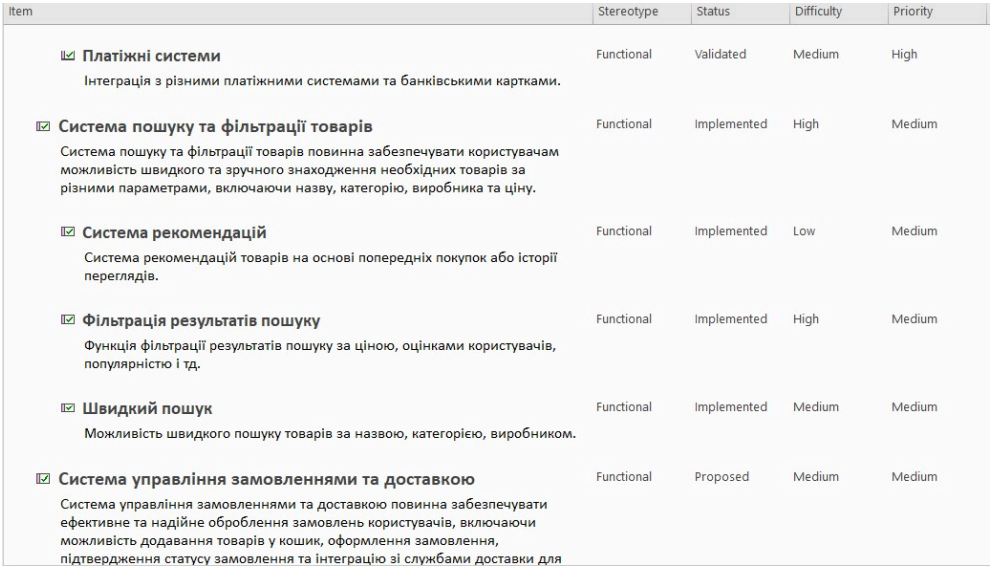
Item	Stereotype	Status	Difficulty	Priority
Взаємодія з електронним магазином електро-товарів Магазин має надавати користувачам можливість швидкого та зручного пошуку та покупки електротоварів, забезпечувати користувачам інформацію про наявність товарів, їх характеристики та ціни та мати ефективну систему обробки замовлень та доставки товарів.	Functional	Approved	Medium	High
Система аутентифікації користувачів Система аутентифікації користувачів забезпечує безпечний доступ до електронного магазину.Система дозволяє забезпечити конфіденційність та безпеку особистої інформації користувачів, а також надає зручний механізм управління доступом до функціоналу магазину.	Functional	Implemented	Medium	High
Авторизація за допомогою існуючого облікового запису Можливість входу в існуючий обліковий запис з використанням електронної пошти та пароля.	Functional	Implemented	Medium	Medium
Підтвердження облікового запису Функція підтвердження електронної адреси під час реєстрації шляхом надсилання листа з посиланням на підтвердження.	Functional	Implemented	Medium	High
Реєстрація Можливість реєстрації нового облікового запису з обов'язковим заповненням обов'язкових полів, таких як ім'я, електронна пошта, пароль тощо.	Functional	Implemented	Medium	High

Рисунок 2.2 – Специфікації вимог «Системи автентифікації користувачів»



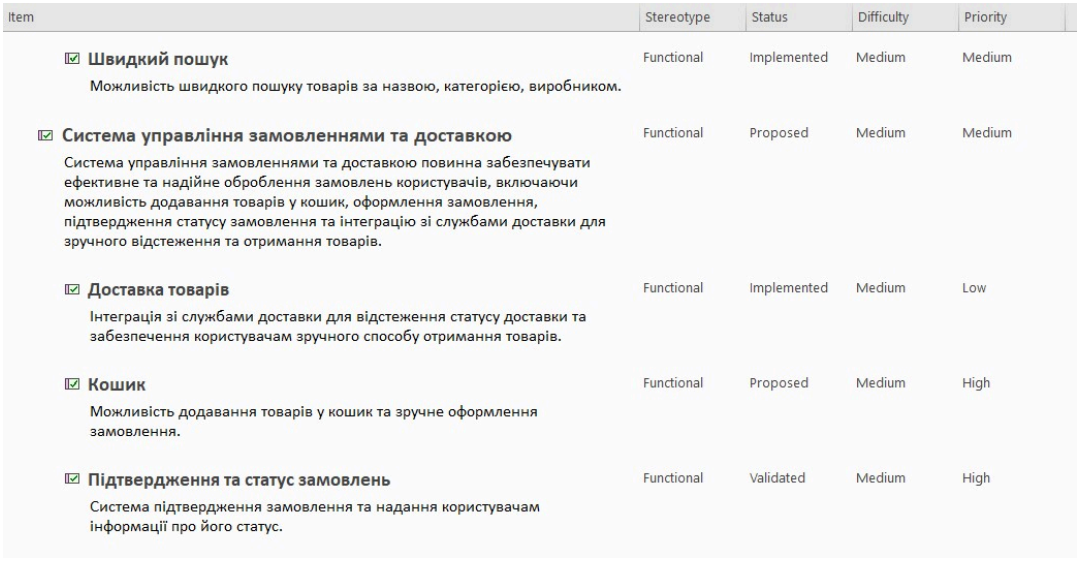
Item	Stereotype	Status	Difficulty	Priority
заповненням обов'язкових полів, таких як ім'я, електронна пошта, пароль тощо.				
Система оплати та фінансові операції Забезпечення безпечної та зручної оплати товарів для користувачів, включаючи інтеграцію з різними платіжними системами, захист фінансових даних та забезпечення конфіденційності операцій.	Functional	Proposed	Medium	Medium
Безпека та конфіденційність Забезпечення безпеки та конфіденційності фінансових операцій користувачів включає в себе реалізацію шифрування для захисту особистих фінансових даних користувачів, встановлення заходів двофакторної аутентифікації для підтвердження операцій, впровадження системи моніторингу та виявлення шахраїв для запобігання несанкціонованим трансакціям та шахрайству.	Functional	Implemented	Medium	Low
Оплата Можливість безпечної та швидкої оплати (в 2 кліки) товарів онлайн.	Functional	Proposed	Medium	High

Рисунок 2.3 – Специфікації вимог «Система оплати та фінансові операції»



Item	Stereotype	Status	Difficulty	Priority
Платіжні системи Інтеграція з різними платіжними системами та банківськими картками.	Functional	Validated	Medium	High
Система пошуку та фільтрації товарів Система пошуку та фільтрації товарів повинна забезпечувати користувачам можливість швидкого та зручного знаходження необхідних товарів за різними параметрами, включаючи назву, категорію, виробника та ціну.	Functional	Implemented	High	Medium
Система рекомендацій Система рекомендацій товарів на основі попередніх покупок або історії переглядів.	Functional	Implemented	Low	Medium
Фільтрація результатів пошуку Функція фільтрації результатів пошуку за ціною, оцінками користувачів, популярністю і тд.	Functional	Implemented	High	Medium
Швидкий пошук Можливість швидкого пошуку товарів за назвою, категорією, виробником.	Functional	Implemented	Medium	Medium
Система управління замовленнями та доставкою Система управління замовленнями та доставкою повинна забезпечувати ефективне та надійне оброблення замовлень користувачів, включаючи можливість додавання товарів у кошик, оформлення замовлення, підтвердження статусу замовлення та інтеграцію зі службами доставки для	Functional	Proposed	Medium	Medium

Рисунок 2.4 – Специфікації вимог «Система пошуку та фільтрації товарів»



Item	Stereotype	Status	Difficulty	Priority
Швидкий пошук Можливість швидкого пошуку товарів за назвою, категорією, виробником.	Functional	Implemented	Medium	Medium
Система управління замовленнями та доставкою Система управління замовленнями та доставкою повинна забезпечувати ефективне та надійне оброблення замовлень користувачів, включаючи можливість додавання товарів у кошик, оформлення замовлення, підтвердження статусу замовлення та інтеграцію зі службами доставки для зручного відстеження та отримання товарів.	Functional	Proposed	Medium	Medium
Доставка товарів Інтеграція зі службами доставки для відстеження статусу доставки та забезпечення користувачам зручного способу отримання товарів.	Functional	Implemented	Medium	Low
Кошик Можливість додавання товарів у кошик та зручне оформлення замовлення.	Functional	Proposed	Medium	High
Підтвердження та статус замовлень Система підтвердження замовлення та надання користувачам інформації про його статус.	Functional	Validated	Medium	High

Рисунок 2.5 – Специфікації вимог «Система управління замовленнями та доставкою»

1. *Продуктивність та час відгуку:*

Система має забезпечувати швидку обробку запитів користувачів. Час завантаження сторінок повинен бути мінімізований, не перевищувати 3-4 секунди.

2. *Надійність та доступність:*

Електронний магазин повинен бути доступним для користувачів 24/7, з мінімальним ризиком простою. Потрібно забезпечити резервне копіювання даних та ефективні механізми відновлення при збоях.

3. Масштабованість:

Система має бути розроблена з урахуванням потенційного зростання кількості користувачів та товарних позицій. Магазин повинен легко адаптуватися до збільшеного навантаження без втрати продуктивності.

4. Безпека:

Забезпечення високого рівня безпеки користувацьких даних та фінансових транзакцій є обов'язковим. Це включає шифрування даних та несанкціонованих доступів.

5. Інтуїтивність та зручність користування:

Інтерфейс магазину повинен бути інтуїтивно зрозумілим, з простим та логічним навігаційним меню, що сприяє покращенню загального досвіду користувачів.

Загальні нефункціональні вимоги можна оглянути на рис. 2.6 та специфікації вимог зроблені за допомогою менеджера специфікацій (Specification Manager) Enterprise Architect [19], що зображені на рис. 2.7-2-9.

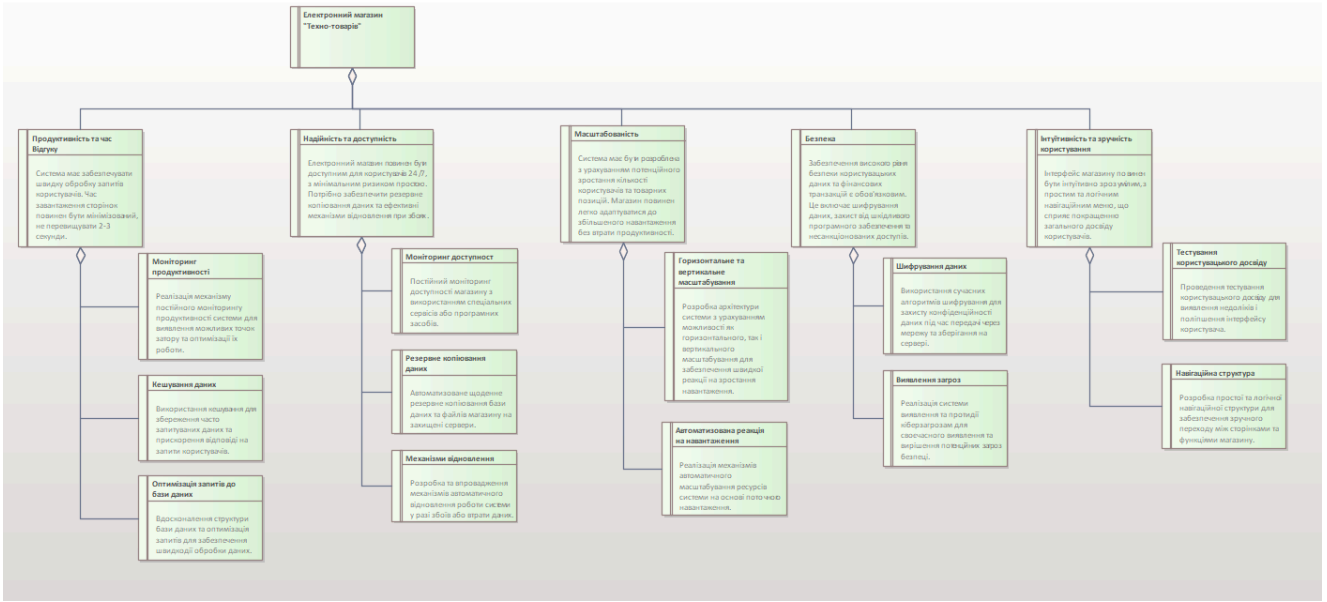


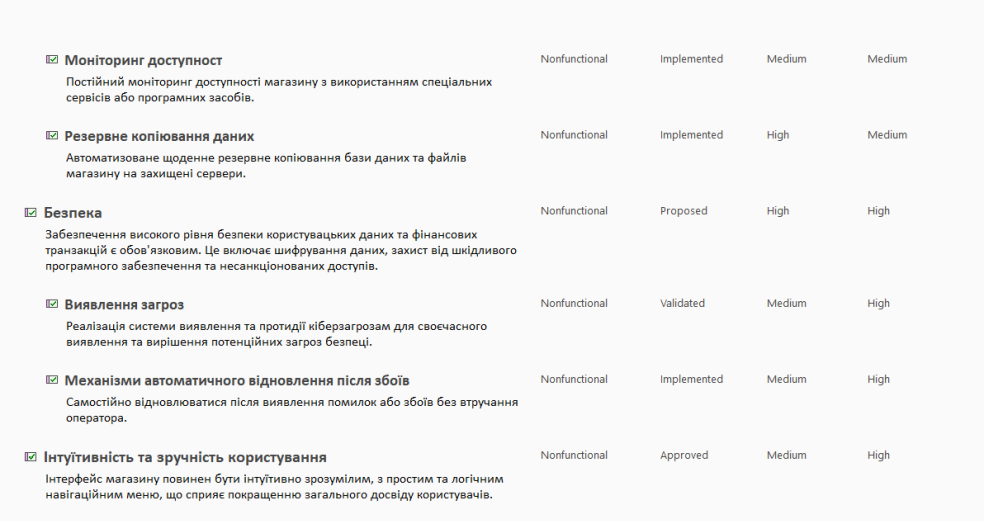
Рисунок 2.6 – Діаграма нефункціональних вимог

Електронний магазин "Техно-товарів"	interface	Approved	High	High
Контекстна довідка та підказки Наявність підтримки користувачів під час використання.	NonfunctionalRequire...	Implemented	Medium	Low
Персоналізація інтерфейсу Можливість налаштування інтерфейсу під індивідуальні потреби користувачів.	NonfunctionalRequire...	Implemented	Medium	Medium
Тестування користувацького досвіду Проведення тестування користувацького досвіду для виявлення недоліків і поліпшення інтерфейсу користувача.	Nonfunctional	Proposed	Medium	High
Шифрування даних Використання сучасних алгоритмів шифрування для захисту конфіденційності даних під час передачі через мережу та зберігання на сервері.	Nonfunctional	Implemented	Medium	High
Продуктивність та час Відгуку Система має забезпечувати швидку обробку запитів користувачів. Час завантаження сторінок повинен бути мінімізований, не перевищувати 2-3 секунди.	Nonfunctional	Validated	Medium	High

Рисунок 2.7 – Специфікації нефункціональних вимог «Продуктивність та час відгуку та надійність, доступність»

Бізнес-вимоги до системи

Визначимо основні бізнес-вимоги до системи електронного магазину техно-товарів:



Моніторинг доступност Постійний моніторинг доступності магазину з використанням спеціальних сервісів або програмних засобів.	Nonfunctional	Implemented	Medium	Medium
Резервне копіювання даних Автоматизоване щоденне резервне копіювання бази даних та файлів магазину на захищені сервери.	Nonfunctional	Implemented	High	Medium
Безпека Забезпечення високого рівня безпеки користувацьких даних та фінансових транзакцій є обов'язковим. Це включає шифрування даних, захист від шкідливого програмного забезпечення та несанкціонованих доступів.	Nonfunctional	Proposed	High	High
Виявлення загроз Реалізація системи виявлення та протидії кіберзагрозам для своєчасного виявлення та вирішення потенційних загроз безпеці.	Nonfunctional	Validated	Medium	High
Механізми автоматичного відновлення після збоїв Самостійно відновлюватися після виявлення помилок або збоїв без втручання оператора.	Nonfunctional	Implemented	Medium	High
Інтуїтивність та зручність користування Інтерфейс магазину повинен бути інтуїтивно зрозумілим, з простим та логічним навігаційним меню, що сприяє покращенню загального досвіду користувачів.	Nonfunctional	Approved	Medium	High

Рисунок 2.8 – Специфікації нефункціональних вимог «Масштабованість та безпека даних»

Механізми автоматичного відновлення після збоїв Самостійно відновлюватися після виявлення помилок або збоїв без втручання оператора.	Validate	Implemented	Medium	Low
Навігаційна структура Розробка простої та логічної навігаційної структури для забезпечення зручного переходу між сторінками та функціями магазину.	Nonfunctional	Validated	Medium	High
Масштабованість Система має бути розроблена з урахуванням потенційного зростання кількості користувачів та товарних позицій. Магазин повинен легко адаптуватися до збільшеного навантаження без втрати продуктивності.	Nonfunctional	Proposed	Medium	Medium
Автоматизована реакція на навантаження Реалізація механізмів автоматичного масштабування ресурсів системи на основі поточного навантаження.	Nonfunctional	Validated	Medium	High
Кешування даних для покращення продуктивності Використання механізмів кешування для збереження проміжних результатів запитів або часто використовуваних даних.	Nonfunctional	Implemented	Medium	Medium

Рисунок 2.9 – Специфікації нефункціональних вимог «Інтуїтивність та зручність користування та масштабованість»

1. Забезпечення зручного та ефективного управління асортиментом товарів, включаючи:

- можливість додавати, редагувати та вилучати товарні позиції з каталогу;
- ведення комплексної інформації про товар (опис, характеристики, ціни, наявність на складі);
- групування товарів за категоріями, фільтрація та пошук.

2. Забезпечення безпечної та конфіденційної обробки платежів та персональних даних клієнтів:

- інтеграція з надійними платіжними шлюзами;
- дотримання стандартів безпеки даних (PCI DSS);
- захист конфіденційної інформації користувачів.[6]

3. Підвищення задоволеності та лояльності покупців:

- зручний та інтуїтивно-зрозумілий інтерфейс для покупок;
- можливість перегляду історії замовлень, повернення товарів;
- система знижок, акцій та бонусних програм.

4. Максимізація прибутку:

- впровадження ефективних маркетингових стратегій (електронні розсилки, реклама в соцмережах);
- оптимізація ланцюжка поставок та управління запасами;
- аналітика продажів для прийняття рішень щодо ціноутворення, асортименту.[4]

5. Забезпечення автоматизованого збору та аналізу статистичних даних:

- відстеження ключових показників ефективності (KPI);
- створення аналітичних звітів та візуалізація даних.[4]

Також, необхідно врахувати вимоги до масштабованості, безпеки, продуктивності системи та браузерами для максимального охоплення цільової аудиторії.

Головною бізнес-ціллю електронного магазину електронних товарів є збільшення обсягів продажів та розширення частки ринку. Це можна досягти завдяки залученню більшої кількості покупців через покращення клієнтського досвіду та підвищення упізнаваності бренду.

Важливим є також підвищення рентабельності та прибутковості бізнесу. Оптимізація бізнес-процесів, логістики та управління запасами дозволить скоротити операційні витрати.

Не менш вагомою ціллю є забезпечення ефективних операцій та безперебійної роботи магазину. Це передбачає впровадження інноваційних рішень, автоматизацію процесів, а також постійний моніторинг та аналіз ключових показників ефективності.

На рис. 2.10 проілюстровано діаграму бізнес-вимог та бізнес-цілей кожної особи яка взаємодіє з магазином. Основними особами, що взаємодіють з системою є тестувальники (розробники), продавці та адміністратори магазину.

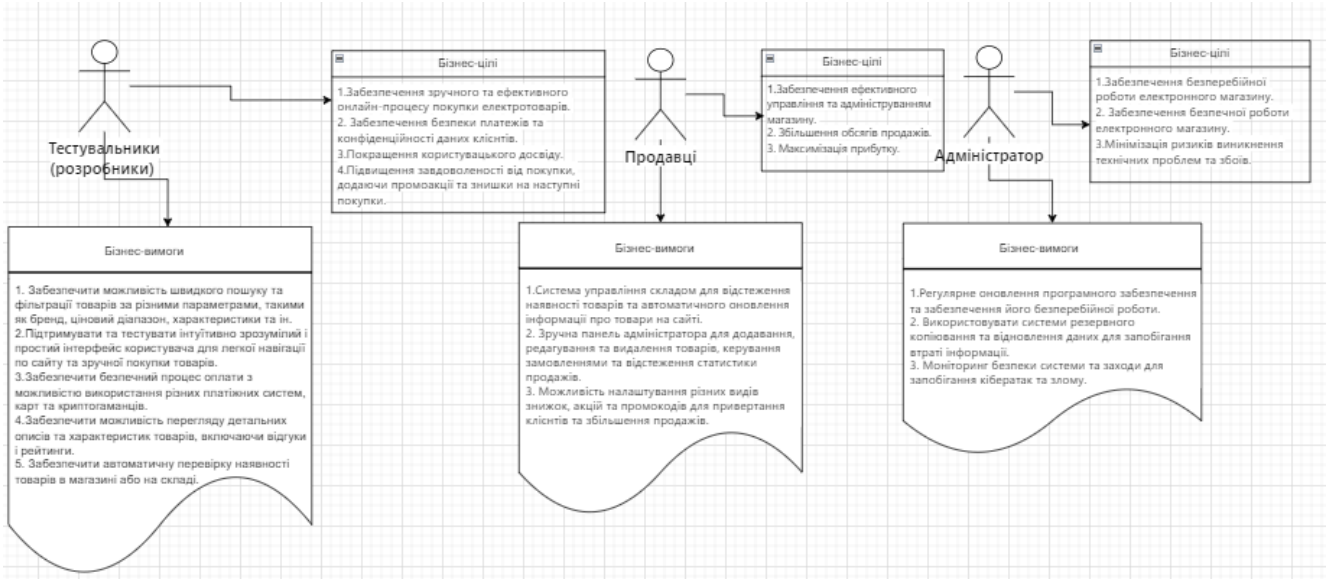


Рисунок 2.10 – Діаграма визначення бізнес-цілей та бізнес-вимог

2.2 Обґрунтування методології проєктування та функціональна модель задачі

Створення електронного магазину – це динамічний процес, що вимагає гнучкості та швидкої реакції на зміни. Ітеративний характер розробки, необхідність постійного вдосконалення та адаптації до вимог ринку зумовили вибір гнучкої методології розробки (Agile) для даного проекту.[7]

Agile – це не просто набір інструментів, а філософія розробки, яка ставить на перше місце людський фактор. Основний фокус робиться на створенні робочого продукту, який приносить реальну користь.[7]

Agile передбачає активну співпрацю з замовником (клієнтами), який виступає не просто спостерігачем, а повноцінним учасником процесу розробки. Гнучкість до змін – ще один важливий принцип Agile. Зміни розглядаються не як проблема, а як можливість для покращення та вдосконалення продукту.

Використання Agile дозволить швидко реагувати на нові вимоги та зміни на ринку, забезпечити прозорість розробки та залученість замовника на всіх етапах, а також оперативно виправляти помилки та вдосконалювати продукт.

Функціональна модель розроблюваного електронного магазину техно-товарів базується на принципах об'єктно-орієнтованого програмування (ООП). Надає можливість представити систему як сукупність взаємопов'язаних об'єктів, кожен з яких має свою чітко визначену роль та функціональність.

Серед ключових об'єктів системи можна виділити "Користувача". Він отримує можливість реєструватися, авторизуватися, переглядати каталог товарів, здійснювати пошук за заданими критеріями, додавати товари до кошика, оформлювати замовлення, відслідковувати його статус та змінювати свої персональні дані. Інформація про кожен товар зберігається в об'єкті "Товар", який містить його назву, детальний опис, фотографії, ціну, інформацію про наявність та основні характеристики. Для зручності навігації та пошуку товари групуються за категоріями та підкатегоріями, що реалізується за допомогою об'єкта "Категорія". Об'єкт "Замовлення" містить всю необхідну інформацію про замовлення користувача, включаючи перелік товарів, їх кількість, загальну суму, обраний спосіб оплати та доставки, а також поточний статус замовлення. Об'єкт "Доставка" надає користувачам можливість обрати зручний спосіб доставки та відстежувати його статус.

Типові сценарії використання системи ілюструють її функціональність та взаємодію між об'єктами. Наприклад, сценарій "Пошук товару" починається з введення користувачем ключових слів у пошуковий рядок. Система аналізує запит та відображає список товарів, що відповідають заданим критеріям. Користувач може додатково сортувати та фільтрувати результати пошуку для

швидкого знаходження потрібного товару. Сценарій "Додавання товару до кошика та оформлення замовлення" демонструє процес купівлі товару. Користувач додає вибрані товари до кошика, де система автоматично розраховує загальну суму замовлення. Далі користувач вводить контактну інформацію та адресу доставки, обирає спосіб оплати та доставки, після чого підтверджує замовлення. Система формує замовлення та передає його на обробку. Сценарій "Адміністрування магазину" демонструє функціональність, доступну адміністратору системи. Після авторизації адміністратор може додавати, редагувати та видаляти товари, категорії, керувати користувачами, а також переглядати статистику замовлень, аналізувати дані про продажі та популярні товари.

Розроблена функціональна модель забезпечує створення зручного та інтуїтивно зрозумілого електронного магазину техно-товарів, який задовольнятиме потреби користувачів та надасть ефективні інструменти для управління асортиментом, замовленнями та клієнтською базою.

2.3 Моделювання предметної галузі

Основна мета моделювання предметної області електронного магазину техно-товарів полягає в створенні чіткої та повної моделі процесів, об'єктів та взаємозв'язків, що складають систему. Аналіз ринку та визначення потреб цільової аудиторії дозволяють вивчити ринок електротоварів, визначити основних учасників та їхні вимоги.

За допомогою моделювання визначаються всі основні функції та можливості, які має мати веб-сайт магазину. Це включає управління товарами, замовленнями, оплатою, доставкою, а також взаємодію з клієнтами та адміністраторами. Моделювання допомагає розробити оптимальну структуру веб-сайту та інтерфейс, який буде зрозумілим та зручним для користувачів.

Оптимізація процесів внутрішньої логістики та управління товарними запасами передбачає розробку ефективної системи управління замовленнями, оплатою та доставкою товарів, що сприяє підвищенню продуктивності та зниженню витрат. Моделювання дозволяє передбачити інтеграцію з різними платіжними системами та розробити стратегію SEO-оптимізації для підвищення ефективності інтернет-маркетингу.

Головний процес OPD діаграми, "Управління електронним магазином техно-товарів", визначає основні функції системи, які включають споживачів результатів, таких як клієнти, адміністратори та система оплати. Учасники процесу включають цих користувачів, а також службу підтримки.[20] Результати процесу – це оформлені замовлення, оплачені замовлення та відправлені товари. Модель предметної області подано ієрархією об’єктно-процесних діаграм мовою моделювання OPM (див. рис. 2.13-2.18).

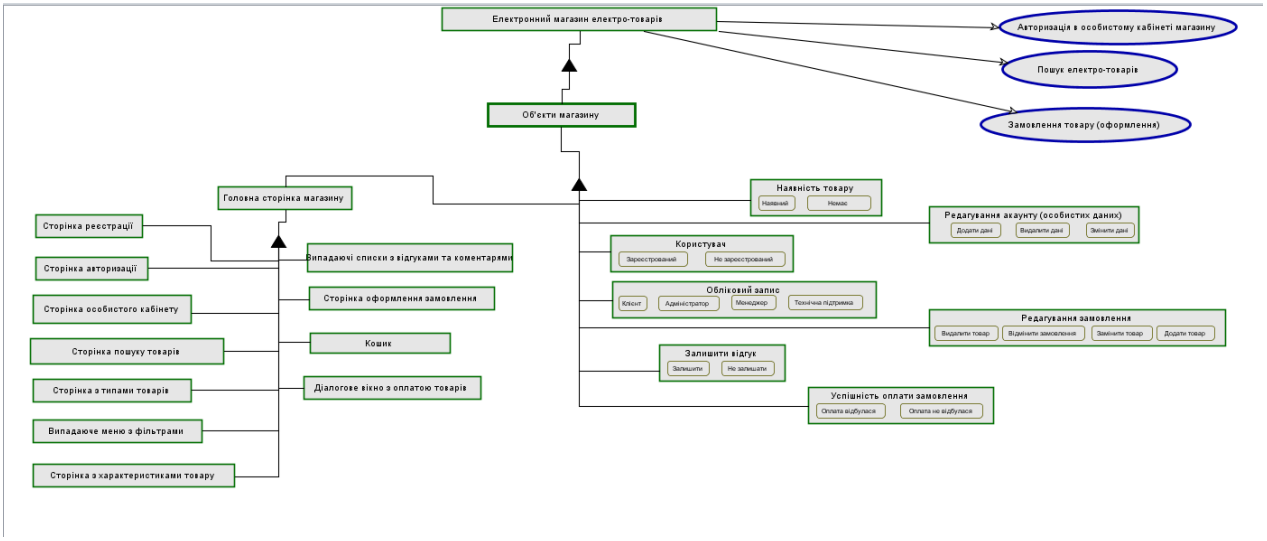


Рисунок 2.11 – Головна діаграма

На діаграмах наступного рівня деталізується функціонал системи через розгортання підпроцесів та асоційованих об'єктів. Діаграма "Авторизація в особистому кабінеті магазину" охоплює процеси авторизації та реєстрації користувачів. Об'єкти включають форми авторизації та реєстрації, а також

особистий кабінет користувача, який забезпечує доступ до персоналізованих сервісів та інформації.

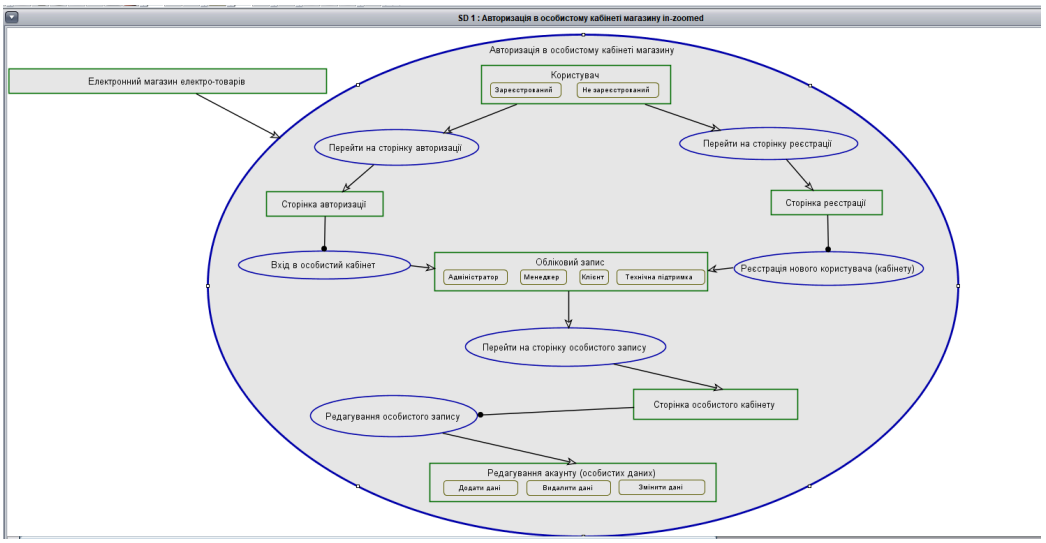


Рисунок 2.12 – Діаграма «Авторизація в особистому кабінеті магазину in-zoomed»

Другий підпроцес, "Пошук доступного номеру", описує процеси пошуку товарів за різними критеріями. Він включає пошукові форми та результати пошуку, що дозволяє користувачам швидко знаходити потрібні товари.

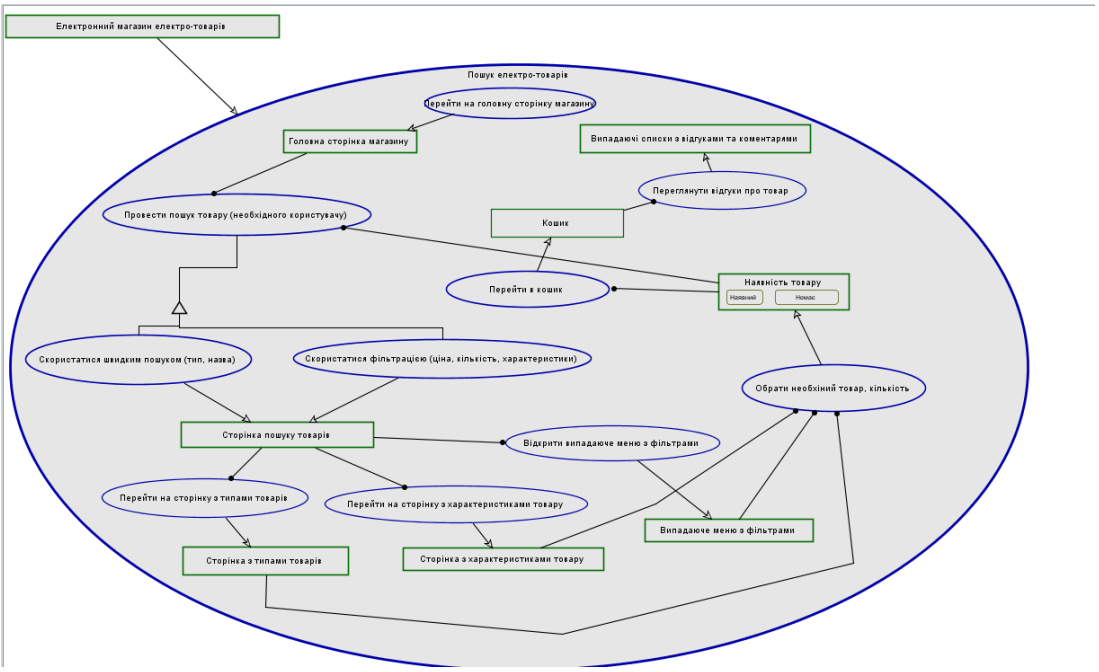


Рисунок 2.13 – Діаграма «Пошук техно-товарів in-zoomed»

Третій підпроцес, "Замовлення товару (оформлення)", описує процеси оформлення та оплати замовлення. Він охоплює об'єкти, такі як кошик, форму замовлення та систему оплати, забезпечуючи користувачам зручність та безпеку під час здійснення покупок.

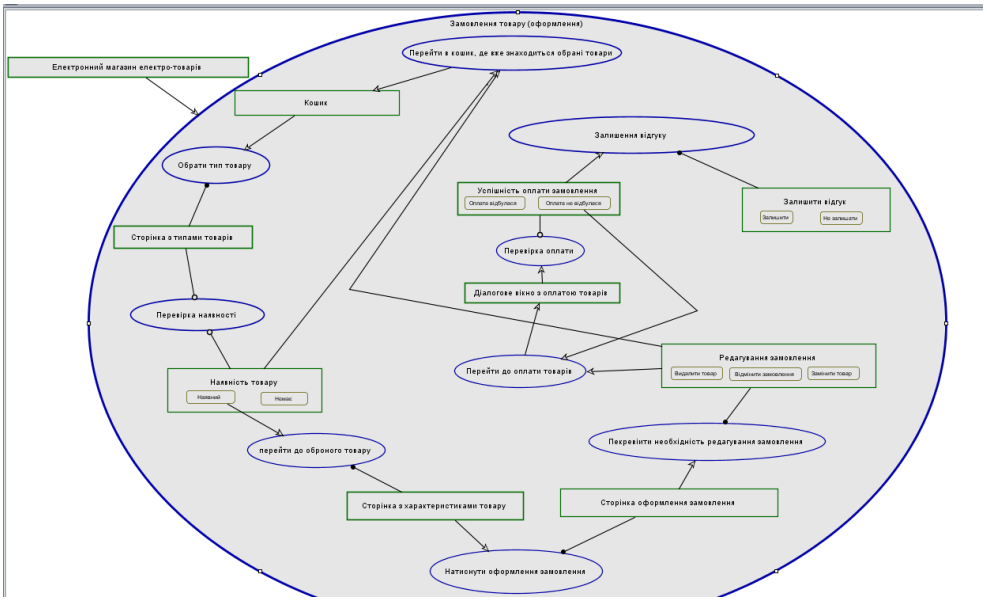


Рисунок 2.14 – Діаграма «Замовлення товару, оформлення in-zoomed»

Механізм розгортання (огляд всіх об’єктів магазину) відображає взаємозв'язки між об'єктами системи, допомагаючи зрозуміти їхню структуру та взаємодію.

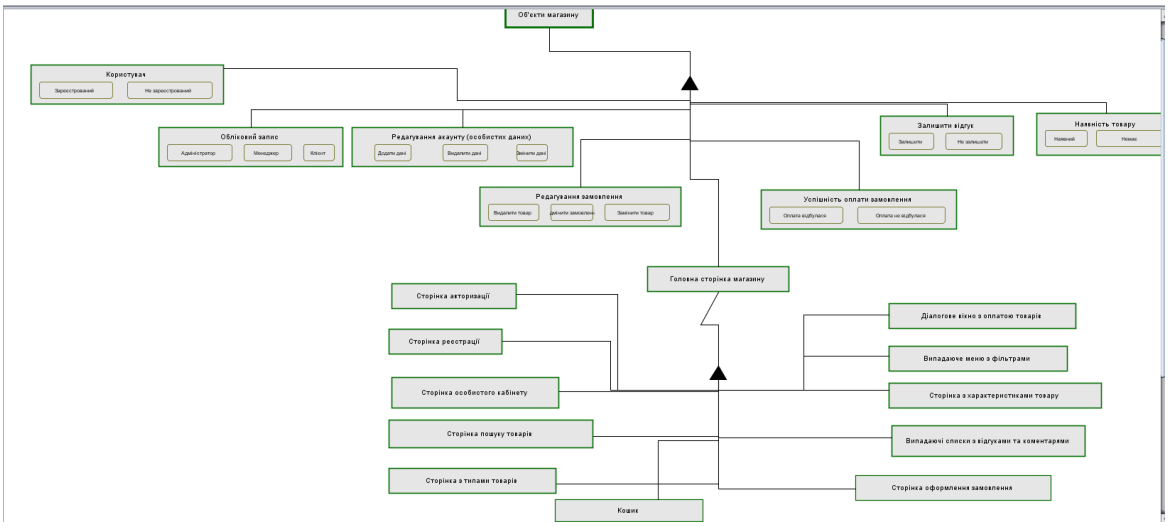


Рисунок 2.15 – Механізм розгортання (огляд всіх об’єктів магазину)

Для візуалізації та детального опису процесів електронного магазину було застосовано методологію моделювання OPCAT. Створена ієрархія OPD-діаграм

відображає повну картину взаємодії процесів та об'єктів системи, включаючи їх стани та переходи. Такий рівень деталізації дозволив чітко сформулювати та задокументувати функціональні вимоги до проектованого електронного магазину.

2.4 Характеристика задачі

Задача нашого кваліфікаційного бакалаврського проєкту полягає у створенні ефективного, зручного для користувача та технічно сучасного електронного магазину з техно-товарів. Проект передбачає проектування веб-платформи, яка дозволить користувачам переглядати асортимент товарів, вибирати необхідні продукти, робити замовлення та здійснювати платежі онлайн. Основною метою є створення інтуїтивно зрозумілого та простого у використанні інтерфейсу, який водночас буде містити всі необхідні функції для здійснення покупок.

Основні функціональні вимоги до магазина включають (детальніше розглянуто в підрозділі 2.1.1 функціональні вимоги до системи):

- каталог товарів;
- система кошика та оформлення замовлень;
- реєстрація та управління користувацькими обліковими записами;
- інтеграція з платіжними системами;
- логістика та управління замовленнями.

Процес роботи електронного магазину:

- перегляд та вибір товарів;
- оформлення замовлення;
- обробка та доставка замовлення;
- після продажне обслуговування.

Характеристика нефункціональних вимог (детальніше розглянуто в підрозділі 2.1.2 нефункціональні вимоги до системи):

- продуктивність та час відгуку;
- надійність та доступність;
- масштабованість;
- безпека;
- інтуїтивність та зручність користування.

В рамках розробки інформаційної системи електронного магазину техно-товарів, система управління базами даних (СУБД) забезпечує надійне та ефективне зберігання та управління всіма даними магазину.

MySQL виступає централізованим сховищем для всіх даних, пов'язаних з роботою електронного магазину. Це включає детальну інформацію про товари, дані про користувачів, інформацію про замовлення, маркетингові дані, логи та звіти про роботу системи. Кожна з цих категорій даних є важливою для забезпечення повноцінної роботи електронного магазину та надання користувачам якісного сервісу.

Інтеграція з MySQL дає змогу реалізувати важливі функції електронного магазину. Зберігання історії покупок та уподобань користувачів дає змогу

надавати їм персоналізовані рекомендації, що підвищує ефективність маркетингових кампаній та лояльність клієнтів. Централізоване зберігання даних про замовлення спрощує їх обробку, дозволяє відстежувати статус доставки та забезпечує прозорість взаємодії з клієнтами.[9] Насамперед, збережені дані можна використовувати для аналізу ефективності роботи магазину, виявлення тенденцій та прийняття обґрунтованих бізнес-рішень.

Інформаційна модель електронного магазину електро-товарів відображає шлях даних та взаємодій між трьома головними складовими: користувачами, які здійснюють покупки, системою, що обробляє інформацію та забезпечує функціонування магазину, та адміністраторами, що керують його роботою.

Початковою точкою взаємодії є реєстрація користувача, що відкриває доступ до персоналізованих функцій магазину. Далі користувач може переглядати каталог, додавати товари до кошика, обирати спосіб оплати та доставки, а також оформлювати замовлення. Кожна дія користувача ініціює обробку інформації системою та виконання відповідних операцій, таких як резервування товару, обробка платежу та формування замовлення.

Система інформує користувачів про статус їхніх замовлень за допомогою повідомлень. Підтвердження замовлення, інформація про оплату та відстеження доставки роблять процес покупки прозорим та зрозумілим.

Неменше важливою складовою інформаційної моделі є можливість для користувачів залишати відгуки та оцінювати товари. Інформація допомагає іншим покупцям приймати рішення та сприяє покращенню асортименту магазину.

Адміністратори магазину мають доступ до розширеного функціоналу для ефективного управління. Вони можуть оновлювати каталог товарів, контролювати запаси, обробляти замовлення, відповідати на запити користувачів та вирішувати питання, пов'язані з поверненням товарів або виникненням проблем.

Інформаційна модель відображає, як дії користувачів та адміністраторів трансформуються у конкретні операції з даними в базі даних. Така модель дозволяє оптимізувати всі бізнес-процеси електронного магазину, зробити його роботу ефективнішою та забезпечити високий рівень сервісу для клієнтів.

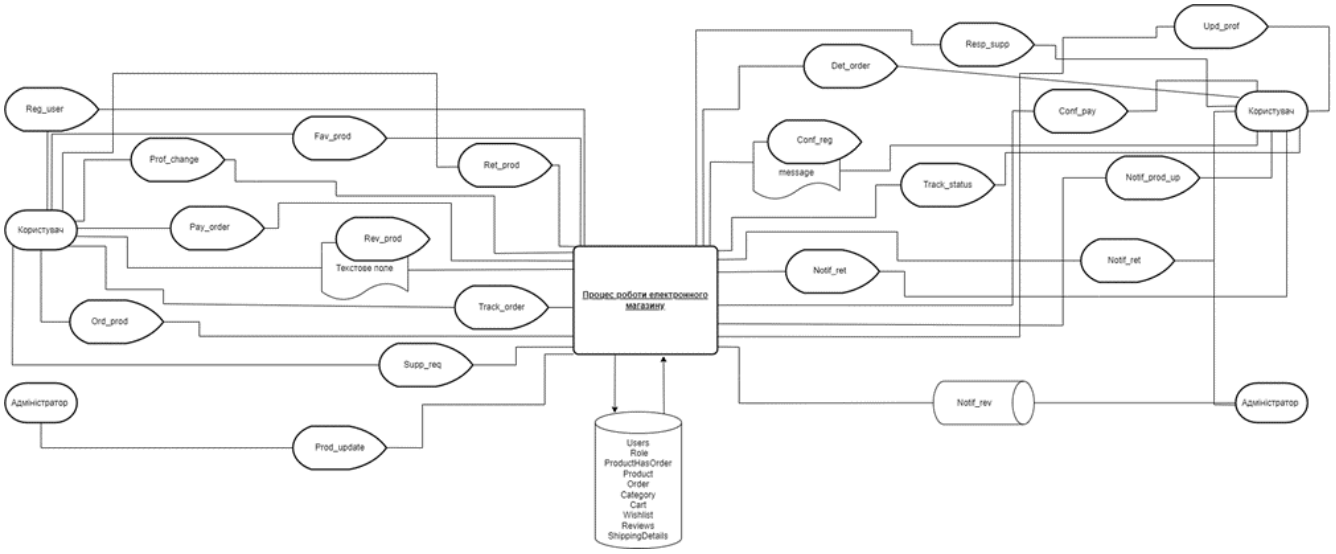


Рисунок 2.16 – Інформаційна модель задач

Вихідна інформація

Перелік і опис вихідних повідомлень наведено у табл. 2.1.

Таблиця 2.1 – Повідомлення вихідної інформації

№ з/п	***Назва вихідного повідомлення***	***Ідентифікатор***	***Форма подання***	***Термін видання і частота надходження***	***Отримувач інформації***
1	Підтвердження реєстрації	Conf_reg	Електрона форма/повідомлення	Після реєстрації	Користувач
2	Деталі замовлення	Det_order	Електрона форма	Після оформлення замовлення	Користувач
3	Підтвердження оплати	Conf_pay	Електрона форма	Після оплати	Користувач

4	Повідомлення про відгук	Notif_rev	Файл	Після залишення відгуку	Адміністратор
5	Повідомлення про повернення	Notif_ret	Електронний лист/ Електрона форма	Після запиту на повернення	Адміністратор/ Користувач
6	Оновлення даних профілю	Upd_prof	Електрона форма	Після зміни профілю	Користувач
7	Повідомлення про зміни у списку бажаного	Notif_fav	Електрона форма	Після додавання в "Вибране"	Користувач
8	Відповіді технічної підтримки	Resp_supp	Електрона форма	Після отримання запиту	Користувач
9	Повідомлення про оновлення товарів	Notif_prod_up	Електронний лист/ Електрона форма	За розкладом/За потребою	Користувач
10	Інформація про статус замовлення	Track_status	Електрона форма	Після запиту на відстеження	Користувач

Вихідна інформація включає різноманітні типи повідомлень та даних, які магазин надсилає користувачам або адміністраторам для інформування, підтвердження дій чи зворотного зв'язку.

Перший ключовий аспект – це повідомлення, пов'язані з користувацькими діями, такі як підтвердження реєстрації користувача, деталі замовлень, підтвердження оплати, та повідомлення про статус замовлення. Ці повідомлення, ідентифіковані як Conf_reg, Det_order, Conf_pay, та Track_status відповідно, зазвичай надсилаються через електронний лист або продемонстровані на електронній формі користувача і відіграють важливу роль у забезпеченні прозорого та ефективного процесу покупки для користувачів.

Другий аспект включає повідомлення для внутрішнього використання, такі як повідомлення про відгуки на товари (Rev_prod), запити на повернення товарів (Ret_prod), оновлення даних профілю користувача (Prof_change), а також відповіді на запити технічної підтримки (Supp_req).

В таблиці включені спеціальні ідентифікатори для повідомлень про

оновлення товарів (Prod_update) від адміністраторів, а також інформаційні повідомлення про вибрані товари (Fav_prod), що сприяє покращенню взаємодії з користувачами та підтримці їх інтересу до продукції магазину.

Вхідна інформація

Основні типи вхідних даних, які необхідні для функціонування та управління електронним магазином, дозволяючи організувати та систематизувати процеси взаємодії з користувачами та адміністрацією. Перелік і опис вхідних повідомлень наведено у табл. 2.2.

Таблиця 2.2 – Повідомлення вхідної інформації

№ з/п	***Назва вхідного повідомлення***	***Ідентифікатор***	***Форма подання***	***Термін і частота надходження***	***Джерело***
1	Реєстрація користувача	Reg_user	Електронна форма	По запиту	Користувач
2	Замовлення товару	Ord_prod	Електронна форма	По запиту	Користувач
3	Оплата замовлення	Pay_order	Електронна форма	При замовленні	Користувач
4	Відгук на товар	Rev_prod	Електронна форма, Текстове поле	По запиту	Користувач
5	Запит на повернення товару	Ret_prod	Електронна форма	По запиту	Користувач
6	Особистий кабінет - зміна даних	Prof_change	Електронна форма	По запиту	Користувач
7	Вподобаний товар (Вибране)	Fav_prod	Електронна форма/Кнопка	По запиту	Користувач
8	Запити технічної підтримки	Supp_req	Електронна форма/Email	По запиту	Користувач
9	Оновлення асортименту товарів	Prod_update	Адміністративний інтерфейс(Електронна форма)	За розкладом/За потребою	Адміністратор

10	Запити на відстеження замовлення	Track_order	Електронна форма	По запиту	Користувач

Таблиця вхідної інформації відіграє роль у забезпеченні ефективної та гладкої взаємодії з користувачами. Вона включає ряд важливих типів повідомлень та запитів, які надходять від користувачів або через веб-інтерфейс, що є основою для різних операцій у магазині.

Першим аспектом є реєстрація користувача (ідентифікована як Reg_user), де користувачі подають свої дані через електронну форму на сайті. Це включає інформацію, таку як ім'я, електронна пошта, та пароль, і відбувається по запиту користувача. Другий важливий елемент – це замовлення товару (Ord_prod), що також реалізується через електронну форму. Користувачі вибирають товари, додають їх до кошика та оформлюють замовлення.

Оплата замовлення (Pay_order) – ще один критичний аспект, де користувачі використовують електронні платіжні системи для завершення покупки. Це відбувається при кожному замовленні та є ключовим для завершення процесу продажу.

В таблиці присутній елемент відгуку на товар (Rev_prod), де користувачі можуть залишати свої враження та оцінки товарів через текстове поле на сайті. Це важливо для забезпечення зворотного зв'язку та покращення якості товарів і послуг.

Запит на повернення товару (Ret_prod) дозволяє користувачам подавати заявки на повернення або обмін товарів, що також здійснюється через електронну форму.

Додаткові аспекти, такі як зміни в особистому кабінеті користувача (Prof_change), запити на технічну підтримку (Supp_req), а також оновлення асортименту товарів (Prod_update) та запити на відстеження замовлення (Track_order), забезпечують додатковий рівень взаємодії та зв'язку з користувачами, допомагаючи в управлінні замовленнями та покращенні якості

обслуговування.

2.5 Математичне забезпечення та алгоритм функціонування системи

Математичний опис

Математичний опис задачі проектування електронного магазину з електронних товарів зосереджується на моделюванні та аналізі ключових бізнес-процесів, задіяних у функціонуванні магазину (детальніше наведено в підрозділі 2.1.3 Бізнес-вимоги до системи).

Для цього використовуються математичні поняття та методи, які допомагають у формалізації та оптимізації цих процесів.

1. Моделювання попиту:

Використання статистичних методів для прогнозування попиту на різні товари. Використовується формула прогнозування попиту, яка може включати часові ряди, регресійний аналіз або методи машинного навчання.

Формула:

$$D(t) = a + b \times t + \sum_{i=1}^{n} c_i \times f_i(t) \qquad (2.1)$$

де, D(t)– прогнозований попит у час t;

a,b,c – статистичні параметри;

$f_i(t)$ – актори, що впливають на попит.

2. Оптимізація запасів:

Використання методів оптимізації для визначення оптимального рівня запасів. Можуть бути застосовані моделі EOQ (економічна кількість замовлення) або JIT (точно вчасно).

Формула:

$$EOQ = \frac{2 \times D \times S}{H} \quad (2.2)$$

де, EOQ – економічна кількість замовлення;

D – річний попит;

S – витрати на розміщення замовлення;

H – витрати на утримання одиниці товару на складі.

3. Аналіз кошика покупок:

Використання методів аналізу асоціацій та шаблонів покупок для виявлення спільно купованих товарів. Це може допомогти в оптимізації маркетингових стратегій та управління асортиментом.

Формула:

$$P \times (A \cup B) = P \times (A) \times P(B|A) \quad (2.3)$$

де, $P \times (A \cup B)$ – ймовірність купівлі товарів A та B разом;

$P \times (A)$ – відповідна ймовірність;

$P(B|A)$ – відповідна ймовірність.

4. Моделювання Доходів:

Використання фінансових моделей для прогнозування доходів від продажів. Сумарний дохід може бути розрахований як сума доходів від кожного проданого товару.

Формула:

$$Revenue = \sum_{i=1}^{n} (P_i \times Q_i) \quad (2.4)$$

де, P_i – ціна товару;

Q_i – кількість проданих одиниць товару i.

Математичне моделювання та розрахунки стають незамінним інструментом для оптимізації ключових бізнес-процесів електронного магазину. За їх допомогою формується ефективна стратегія управління товарними

запасами, розробляються дієві маркетингові кампанії та складаються точні фінансові плани. Аналіз даних, прогнозування ключових показників та прийняття зважених управлінських рішень – усе це стає можливим завдяки використанню математичних моделей та формул.

Алгоритм розв'язання задачі

Алгоритм розв'язання задачі продемонстровано на рис 2.17.

Робота над алгоритмом розв'язання задачі для електронного магазину техно-товарів була зосереджена на досягненні двох ключових цілей: оптимізації всіх комерційних процесів та створенні максимально комфортного середовища для користувачів.

Першим кроком на шляху користувача до здійснення покупки є реєстрація в системі. Цей процес передбачає введення особистих даних та створення унікального профілю, що служить основою для персоналізації досвіду взаємодії з магазином. Після реєстрації користувач може авторизуватися в системі, що забезпечує безпеку його даних, а також відкриває доступ до історії покупок та інших персоналізованих функцій, таких як списки улюблених товарів чи відстеження замовлень.

Центральним елементом алгоритму є функціональність пошуку та вибору товарів. Інтуїтивно зрозумілий інтерфейс, розширені можливості фільтрації та сортування, а також наявність детальної інформації про кожен товар роблять цей процес максимально зручним для користувача. Додавання товару до кошика та подальше оформлення замовлення проходить у кілька простих кроків, де користувач може обрати зручний спосіб оплати та доставки.

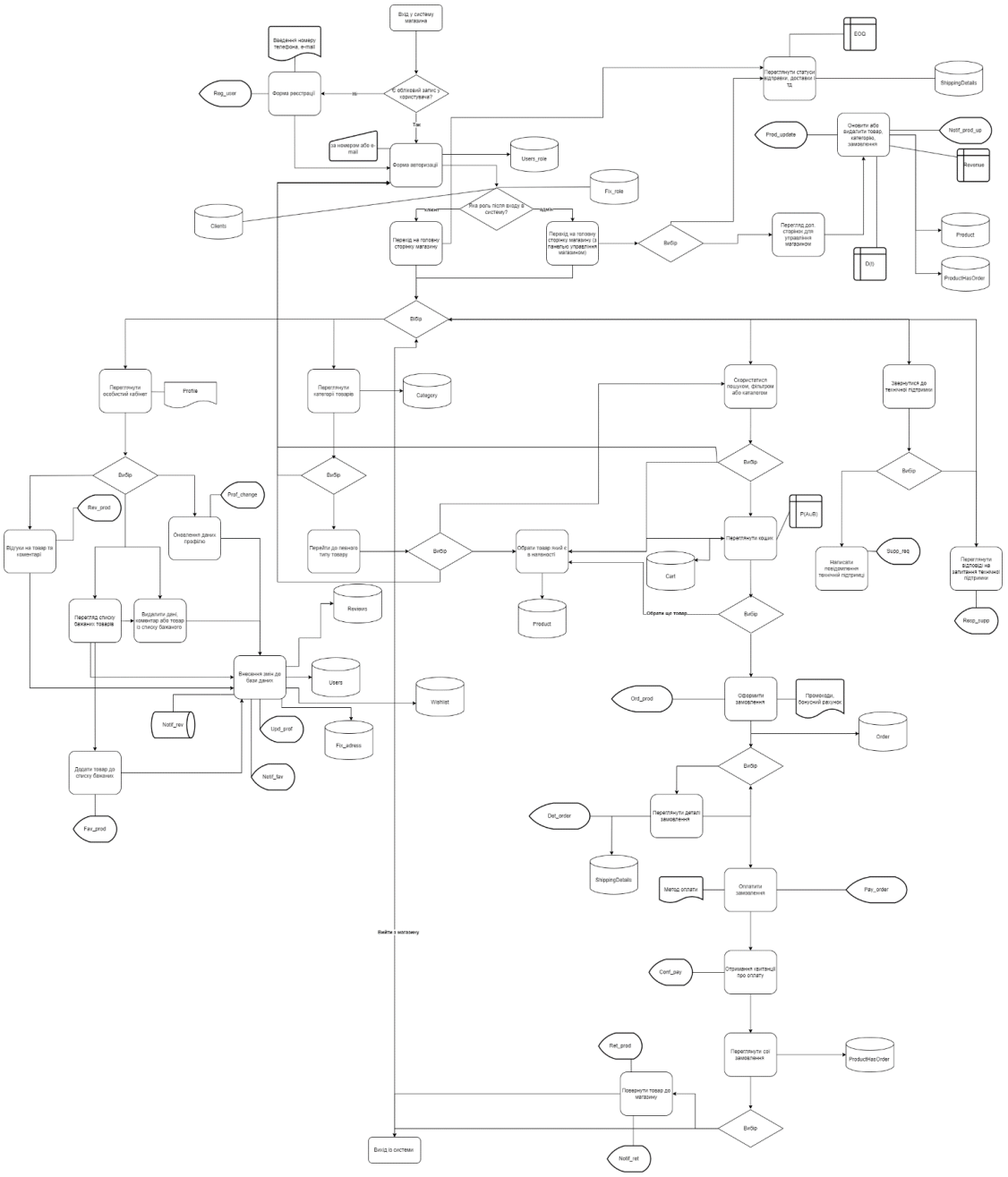


Рисунок 2.17 – Алгоритм розв’язання задачі функціонування та роботи, процесу обробки даних електронного магазину техно-товарів

Після підтвердження замовлення системою ініціюється етап його обробки. Цей етап включає автоматичне резервування товару на складі, обробку платежу за допомогою інтегрованих платіжних шлюзів, формування необхідної документації та передачу замовлення на доставку обраною користувачем службою.

Важливою складовою алгоритму є адміністративний інтерфейс, що надає інструменти для ефективного управління магазином. Адміністратори системи отримують можливість контролювати статуси замовлень, обробляти запити на повернення товарів, аналізувати статистику продажів та відгуки користувачів. Оперативна обробка запитів та врахування зворотного зв'язку від користувачів дозволяє підтримувати високий рівень сервісу та постійно вдосконалювати роботу магазину.

Розроблений алгоритм формує міцну основу для створення електронного магазину електротоварів, який буде відповідати сучасним вимогам ринку.

2.3 Моделювання інформаційної системи

Моделювання поведінки системи електронного магазину техно-товарів – це етап проектування, який дозволяє візуалізувати взаємодію користувача з платформою та визначити ключові сценарії використання. В основі цього процесу лежить аналіз прецедентів, що описують типові дії користувачів та реакцію системи на них.

Першочерговим завданням є забезпечення зручної навігації та пошуку товарів. Користувачі повинні мати можливість легко орієнтуватися в каталозі, використовувати фільтри та сортування для швидкого пошуку товарів за різними критеріями, такими як бренд, ціна, характеристики тощо. Важливим аспектом є також надання повної та структурованої інформації про кожен товар, включаючи якісні фотографії, детальний опис, відгуки покупців та інформацію про умови доставки та оплати.

Зручність використання кошика також є важливим елементом системи. Користувачі повинні мати можливість додавати товари до кошика, змінювати їх кількість, видаляти непотрібні позиції та переглядати загальну суму замовлення. Процес оформлення замовлення має бути максимально простим та

зрозумілим, вимагаючи від користувача мінімум зусиль для введення необхідної інформації та вибору зручних опцій оплати та доставки.

Особистий кабінет користувача надає додаткові можливості для управління замовленнями, відстеження статусу доставки, збереження списків улюблених товарів та редагування персональних даних. Адміністративна частина системи повинна надавати інструменти для управління асортиментом товарів, обробки замовлень, контролю за платежами та доставкою, а також комунікації з користувачами та вирішення можливих проблем.

Детальне опрацювання кожного прецеденту використання дозволяє створити логічну та ефективну систему електронного магазину, яка буде зручною як для користувачів, так і для адміністраторів.

Прецеденти описують основні дії, які можуть здійснювати користувачі в нашому електронному магазині техно-товарів.

Для роботи із електронним магазином електро-товарів можна виділити такі дійові особи:

- покупець (клієнт);
- адміністратор.

Взаємодія дійових осіб та функціональних можливостей системи відображена на діаграмі прецедентів (див. рис. 2.18).

Продемонстрована діаграма прецедентів для електронного магазину техно-товарів включає такі основні випадки використання включають:

1. Замовлення товару:

Клієнт вибирає товар, додає його до кошика та оформляє замовлення. Інтернет-магазин обробляє замовлення, обробляє платіж, надсилає товар на склад, а потім до клієнта через службу доставки. Є альтернативний потік для випадків скасування замовлення, відхилення платежу або пошкодження товару.

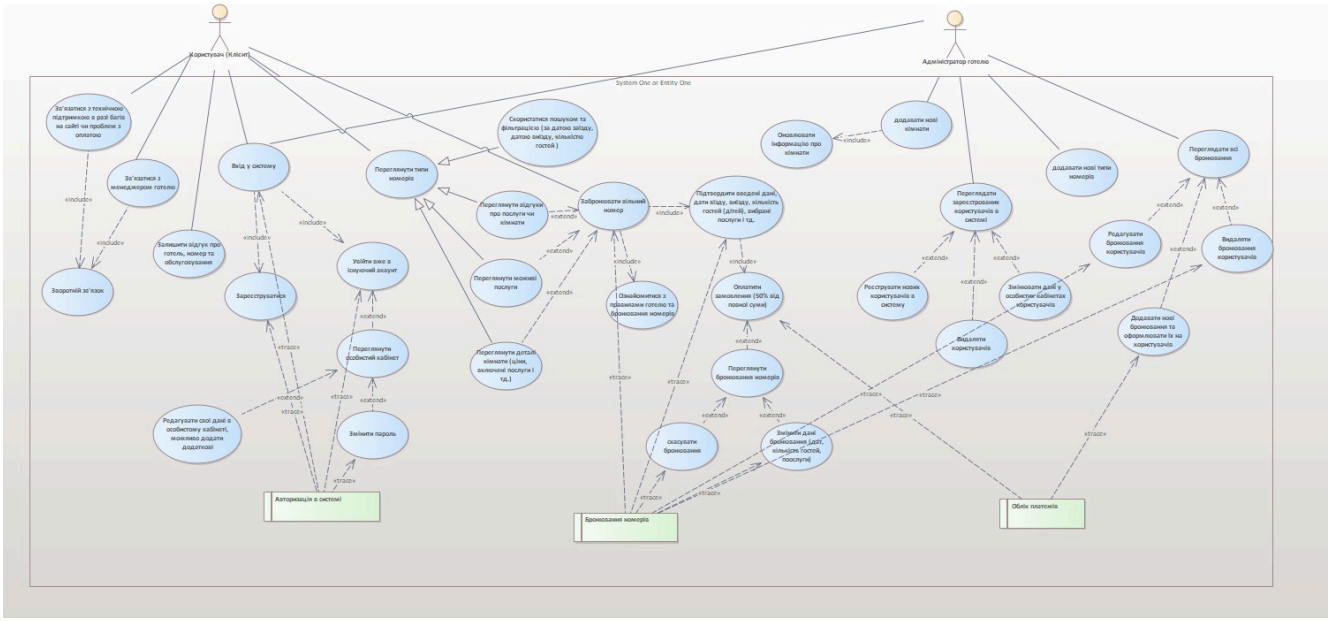


Рисунок 2.18 – Діаграма прецедентів

2. Повернення товару:

Клієнт звертається до Інтернет-магазину для повернення товару, а магазин відшкодовує платіж, якщо товар знаходиться в задовільному стані. Є альтернативний потік, якщо магазин відмовляється прийняти повернення.

3. Технічна підтримка:

Клієнт звертається за технічною підтримкою до Інтернет-магазину, який допомагає вирішити проблеми. Є альтернативний потік у випадку не вирішення проблеми.

Додаткові прецеденти включають створення облікового запису, перегляд історії замовлень, оплату частинами та залишення відгуків про товари. Відносини між акторами включають взаємодію клієнта з магазином, системою обробки платежів, складом та службою доставки.

Також зроблено матриці відповідності вимог і прецедентів, які продемонстровані на рис. 2.19-2.20.

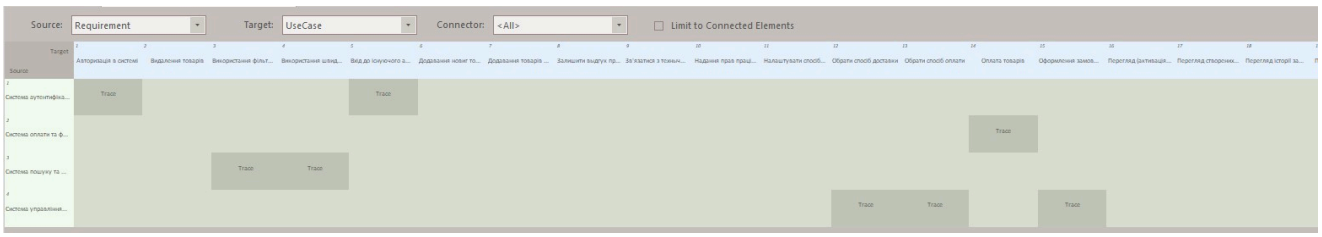


Рисунок 2.19 – Матриця відповідності вимог і прецедентів

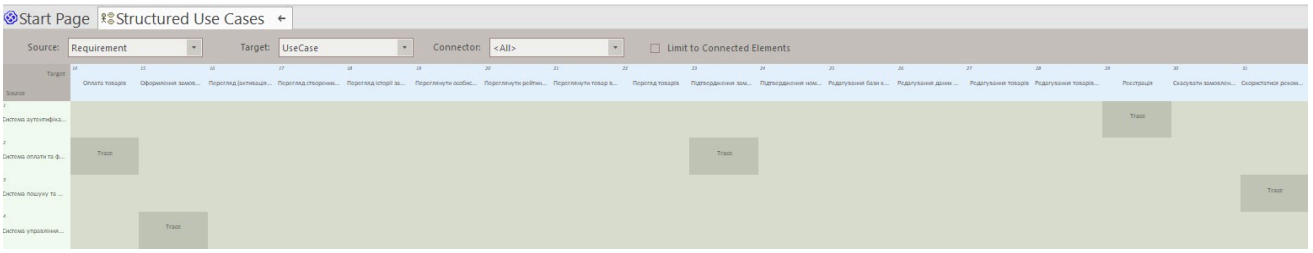


Рисунок 2.20 – Матриця відповідності вимог і прецедентів

Створені текстові сценарії, які детально описують варіанти використання системи, щоб забезпечити зрозумілість та чіткість у розумінні функціональності.

Для цього створюються "нормальні" сценарії, що описують типовий хід подій, а також альтернативні сценарії, які відображають можливі відхилення від "нормального" сценарію. Використали інструмент – розробник сценаріїв (Scenario Builder), для створення деталізованих описів, урахування попередніх та наступних умов, а також обмежень. Кроки кожного сценарію прив'язані до елементів моделі.

Далі наведено приклади поведінки системи залежних від дій користувача на діаграмах послідовностей (див. рис. 2.21-2.24).

На нижченаведеній діаграмі послідовності для електронного магазину техно-товарів зображені взаємодії між користувачем та системою при здійсненні покупки.

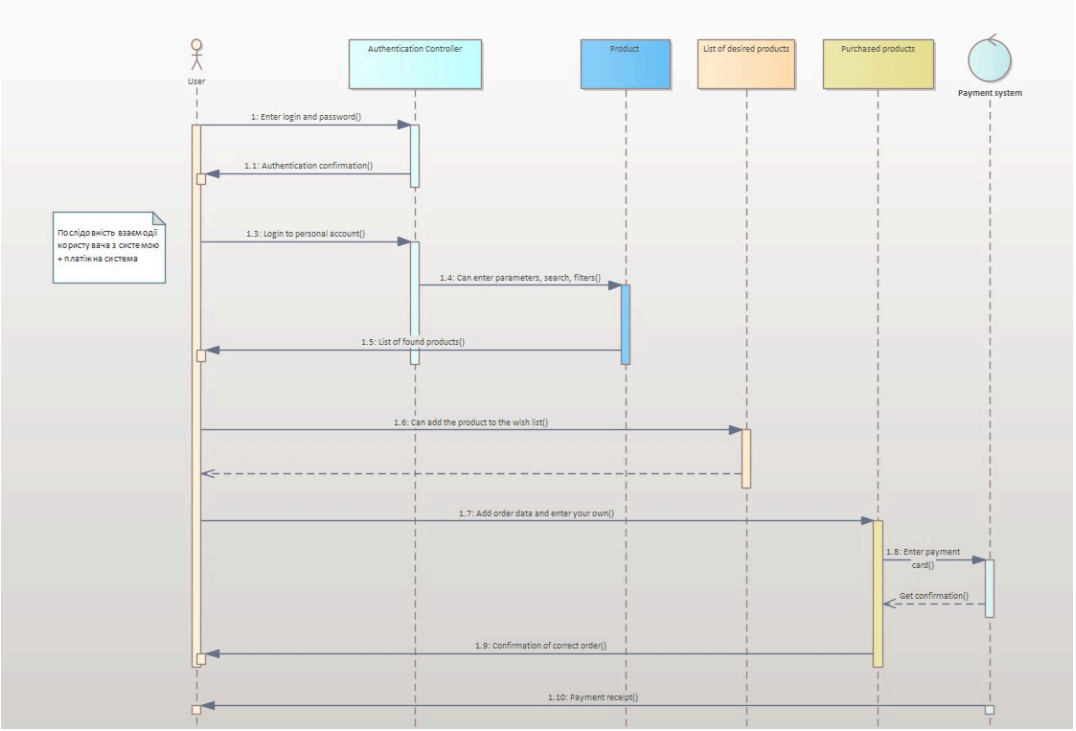


Рисунок 2.21 – Послідовність взаємодії користувача з системою

Користувач:

- починає процес із реєстрації на сайті, вводячи логін та пароль;
- отримує підтвердження реєстрації;
- входить в особистий кабінет.

Товари:

- користувач переглядає список товарів, можливо використовуючи параметри пошуку та фільтри;
- додає товари до списку бажань;
- список бажаних товарів;
- користувач переглядає свій список бажаних товарів.

Придбати товари:

- користувач робить замовлення, вибираючи товари із списку бажань або каталогу;
- після цього користувач отримує дані замовлення.

Платіжна система:

- замовлення передається до платіжної системи, де користувач може оплатити замовлення за допомогою платіжної картки;
- по завершенню оплати користувач отримує квитанцію про оплату.

Діаграма дозволяє зрозуміти логіку взаємодії в системі та оцінити користувацький досвід при роботі з електронним магазином.

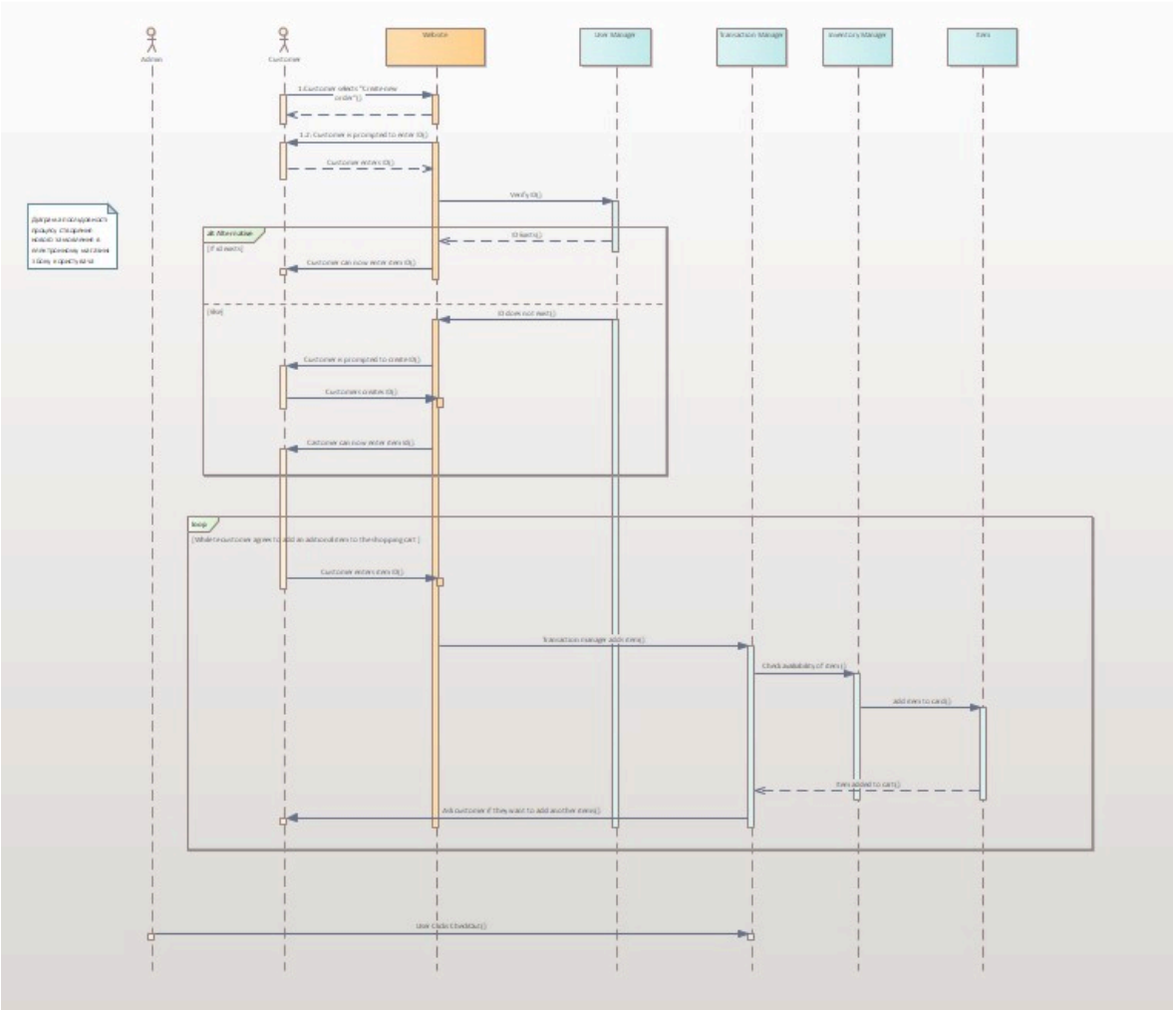
На діаграмі представлено взаємодію між користувачем, електронним магазином, менеджером користувачів, менеджером транзакцій та адміністратором інвентарю в процесі створення замовлення.

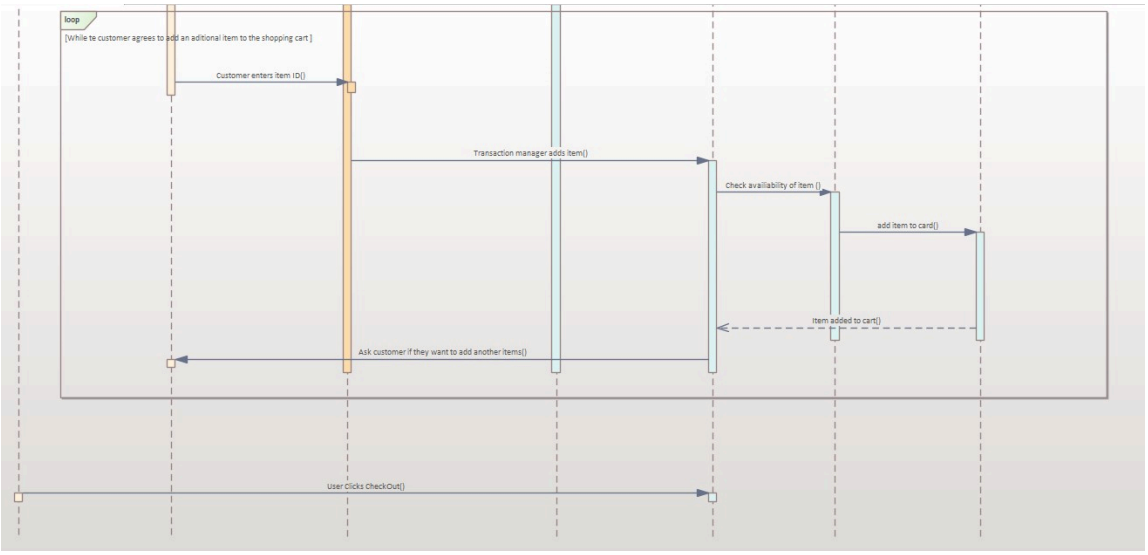
1. *Створення нового замовлення:*

- клієнт вибирає опцію "Створити нове замовлення" на веб-сайті.

2. *Введення ідентифікатора користувача (ID):*

- клієнту пропонується ввести свій ID;
- якщо ID існує, клієнт може продовжити введенням ID товару;
- якщо ID не існує, клієнту пропонується створити новий ID, після чого він може вводити ID товару.





Рисунки 2.22-2.23 – Діаграма послідовності процесу створення нового замовлення в електронному магазині з боку користувача

3. *Введення ID товару:*

- клієнт вводить ID товару в циклі (додаючи декілька товарів).

4. Додавання товару до замовлення:

− менеджер транзакцій перевіряє наявність товару і додає його до кошика.

5. Перевірка кошика та завершення замовлення:

− клієнту запропоновано додати ще товари до кошика. Якщо клієнт вирішує завершити покупку, він натискає "Checkout".

6. Залучення адміністратора:

− адміністратор залучений у процес на різних етапах, наприклад, для допомоги користувачу або обробки замовлення.

Опис діаграми послідовності створення акаунту в електронному магазині (див рис. 2.29).

Рисунок 2.24 – Діаграма послідовності процесу створення нового користувача, де запити ініціює адміністратор магазину

Актори:

1. Адміністратор магазину:

Користувач, який має право створювати нові акаунти в магазині.

2. Система створення акаунтів:

Система, яка відповідає за створення нових акаунтів у магазині.

Послідовність кроків:

1. адміністратор магазину ініціює запит на створення акаунту:

- адміністратор магазину входить до системи створення акаунтів;
- адміністратор магазину вводить інформацію про нового користувача, включаючи ім'я, прізвище, адресу електронної пошти та пароль;
- адміністратор магазину надсилає запит на створення акаунту в систему створення акаунтів.

2. система створення акаунтів перевіряє інформацію про користувача:

- система створення акаунтів перевіряє, чи є введена інформація про користувача правильною та повною;
- якщо інформація про користувача є правильною та повною, система створення акаунтів переходить до наступного кроку;
- якщо інформація про користувача є невірною або неповною, система створення акаунтів відображає відповідне повідомлення адміністратору магазину.

3. система створення акаунтів створює акаунт:

- система створення акаунтів створює новий акаунт для користувача в базі даних магазину;

- система створення акаунтів надсилає електронний лист із підтвердженням новому користувачеві.

4. новий користувач активує свій акаунт:

- новий користувач відкриває електронний лист із підтвердженням, який він отримав від системи створення акаунтів;
- новий користувач переходить за посиланням в електронному листі, щоб активувати свій акаунт;
- новий користувач може увійти до свого акаунту та почати користуватися магазином.

Альтернативні потоки:

- запит на створення акаунту може бути відхилений системою створення акаунтів, якщо інформація про користувача є невірною або неповною;
- новий користувач може не активувати свій акаунт.

2.2.1 Моделювання структури системи

В основі розробки електронного магазину техно-товарів лежить сучасний архітектурний підхід MVC (Model-View-Controller). Цей підхід дозволяє структурувати систему, розділяючи її на три взаємопов'язані частини: модель даних (Model), що відповідає за зберігання та обробку даних, представлення (View), що формує користувацький інтерфейс, та контролер (Controller), що обробляє запити користувачів та керує логікою додатку [8].

Для візуалізації архітектури системи та її складових елементів використовується діаграма класів (рис.2.25). Ця діаграма дозволяє наочно представити ключові класи системи, їх атрибути, методи та взаємозв'язки, що сприяє кращому розумінню її структури та принципів роботи.

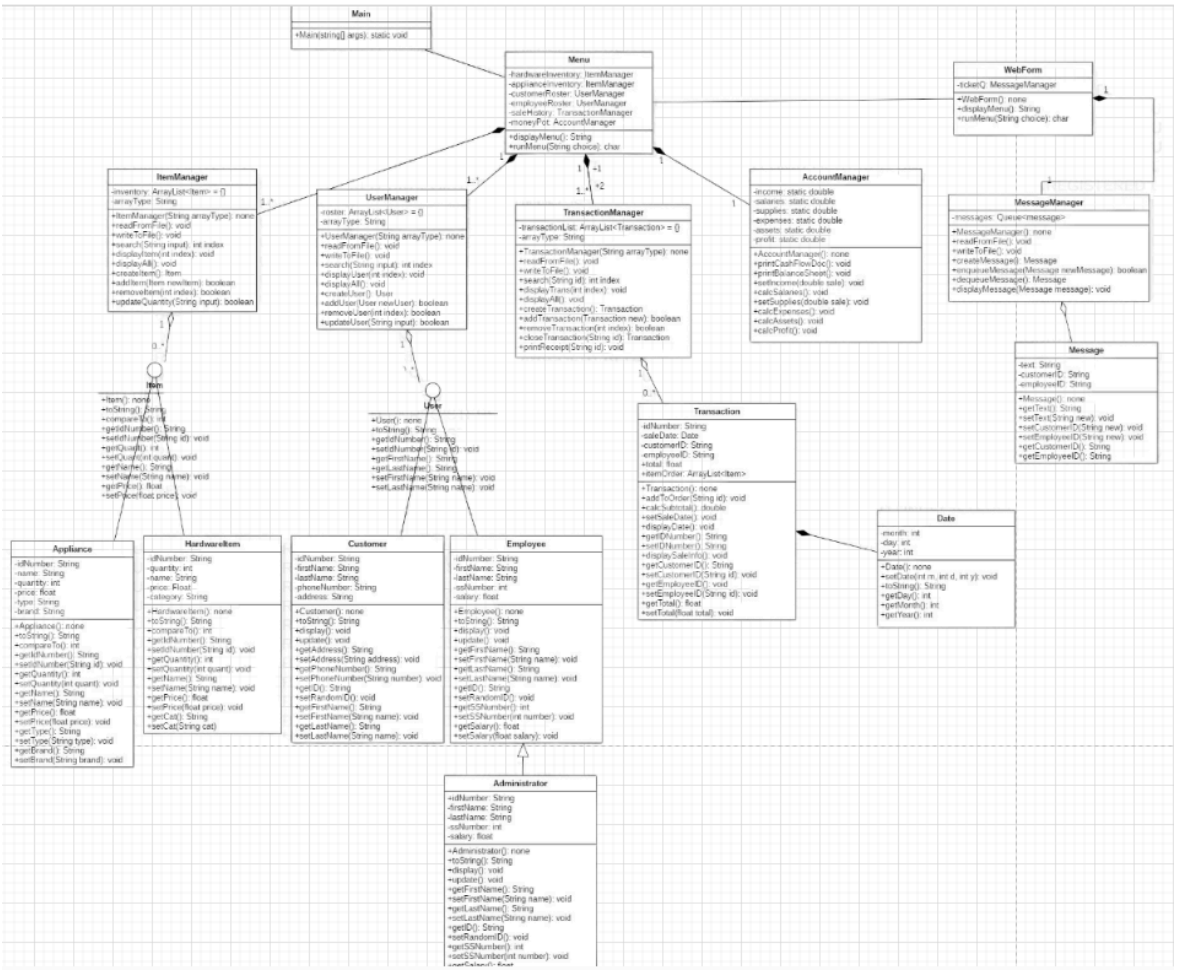


Рисунок 2.25 – Діаграма класів (MVC)

На діаграмі класів електронного магазину можна виділити кілька основних категорій класів. Класи сутностей репрезентують ключові об'єкти предметної області, такі як "Товар", "Користувач", "Замовлення" тощо. Кожен такий клас має певний набір атрибутів, що описують його властивості, та методів, що реалізують його поведінку. Класи-контролери відповідають за обробку запитів від користувачів та взаємодію з класами сутностей для виконання заданих дій.

Діаграма класів відображає багаторівневу архітектуру електронного

магазину. Класи сутностей формують рівень доступу до даних, класи-контролери – рівень бізнес-логіки, а класи користувацького інтерфейсу (View) – рівень презентації. Взаємозв'язки між класами на діаграмі відображають залежності та взаємодії між ними, демонструючи, як різні частини системи співпрацюють для реалізації її функціональності.

Також, наведено приклад стислого фрагмента роботи класів в електронному магазині «техно-товарів» в Enterprise Architec (рис. 2.31) в системі моделює структуру з точки зору головних об'єктів, які в ній присутні, їхні взаємовідносини та властивості.

Класи:

- Product – клас представляє продукт, який продається в магазині. Він містить такі атрибути:
  - productID: унікальний ідентифікатор продукту;
  - name: назва продукту;
  - description: детальний опис продукту;
  - price: ціна продукту;
  - category: категорія продукту;
  - brand: бренд продукту;
  - inventory: кількість продукту на складі.
- Order – клас представляє замовлення, розміщене в магазині. Він містить такі атрибути:
  - orderID: унікальний ідентифікатор замовлення;

- customerID: ідентифікатор клієнта, який розмістив замовлення;
- orderDate: дата та час розміщення замовлення;
- orderStatus: поточний статус замовлення (наприклад, "Очікує", "Обробляється", "Відправлено");
- orderItems: Список елементів замовлення, кожен з яких представляє продукт і його кількість.

- OrderItem – клас представляє елемент замовлення. Він містить такі атрибути:
  - orderItemID: унікальний ідентифікатор елемента замовлення;
  - orderID: ідентифікатор замовлення, до якого належить елемент;
  - productID: ідентифікатор продукту, який включено до елемента замовлення;
  - quantity: кількість продукту, включеного до елемента замовлення;
  - unitPrice: ціна продукту на момент розміщення замовлення.
- Customer – клас представляє клієнта магазину. Він містить такі атрибути:
  - customerID: унікальний ідентифікатор клієнта;
  - firstName: ім'я клієнта;
  - lastName: прізвище клієнта;
  - email: адреса електронної пошти клієнта;

- address: адреса доставки клієнта.

Зв'язки:

- Product:
  - 1:N зв'язок з OrderItem – продукт може бути включено до кількох замовлень.
- Order:
  - 1:N зв'язок з OrderItem – замовлення містить кілька елементів замовлення;
  - 1:1 зв'язок з Customer – замовлення розміщує один клієнт.
  - OrderItem:
  - 1:1 зв'язок з Product – елемент замовлення посилається на один продукт;
  - 1:1 зв'язок з Order – елемент замовлення належить до одного замовлення.
- Payment – клас представляє процес оплати замовлення. Він містить такі атрибути:
  - paymentID: унікальний ідентифікатор платежу;
  - orderID: ідентифікатор замовлення, для якого здійснено платіж;
  - paymentMethod: спосіб оплати, використаний для замовлення (наприклад, "кредитна картка", "PayPal");

- paymentStatus: поточний статус платежу (наприклад, "Очікує", "Підтверджено", "Відхилено").

- Shipping – представляє процес доставки замовлення. Він містить такі атрибути:
  - shippingID: унікальний ідентифікатор доставки;
  - orderID: ідентифікатор замовлення, для якого здійснюється доставка;
  - shippingMethod: спосіб доставки, обраний для замовлення (наприклад, "Безкоштовна доставка", "Експрес-доставка");
  - shippingAddress: адреса доставки замовлення;
  - shippingStatus: поточний статус доставки (наприклад, "Очікує", "Відправлено", "Доставлено").
- User – клас представляє зареєстрованого користувача магазину. Він містить такі атрибути:
  - userID: унікальний ідентифікатор користувача;
  - username: ім'я користувача для входу в систему;
  - password: пароль користувача;
  - firstName: ім'я користувача;
  - lastName: прізвище користувача;
  - email: адреса електронної пошти користувача;

- address: адреса доставки користувача.

- Review – клас представляє відгук про продукт, написаний користувачем. Він містить такі атрибути:
  - reviewID: унікальний ідентифікатор відгуку;
  - productID: ідентифікатор продукту, про який йдеться у відгуку;
  - userID: ідентифікатор користувача, який написав відгук;
  - rating: рейтинг продукту від користувача (від 1 до 5 зірок);
  - reviewText: текст відгуку користувача.

Зв'язки:

- Order:
  - 1:1 зв'язок з Payment – замовлення може мати лише один платіж;
  - 1:1 зв'язок з Shipping – замовлення може мати лише одну доставку.
- User:
  - 1:N зв'язок з Order – користувач може розмістити кілька замовлень;
  - 1:N зв'язок з Review – користувач може написати кілька відгуків.
- Product:
  - 1:N зв'язок з Review – продукт може мати кілька відгуків.

Product Image – клас, який представляє зображення продукту.
Wishlist – клас, який представляє список бажань користувача.
Cart – клас, який представляє кошик користувача.

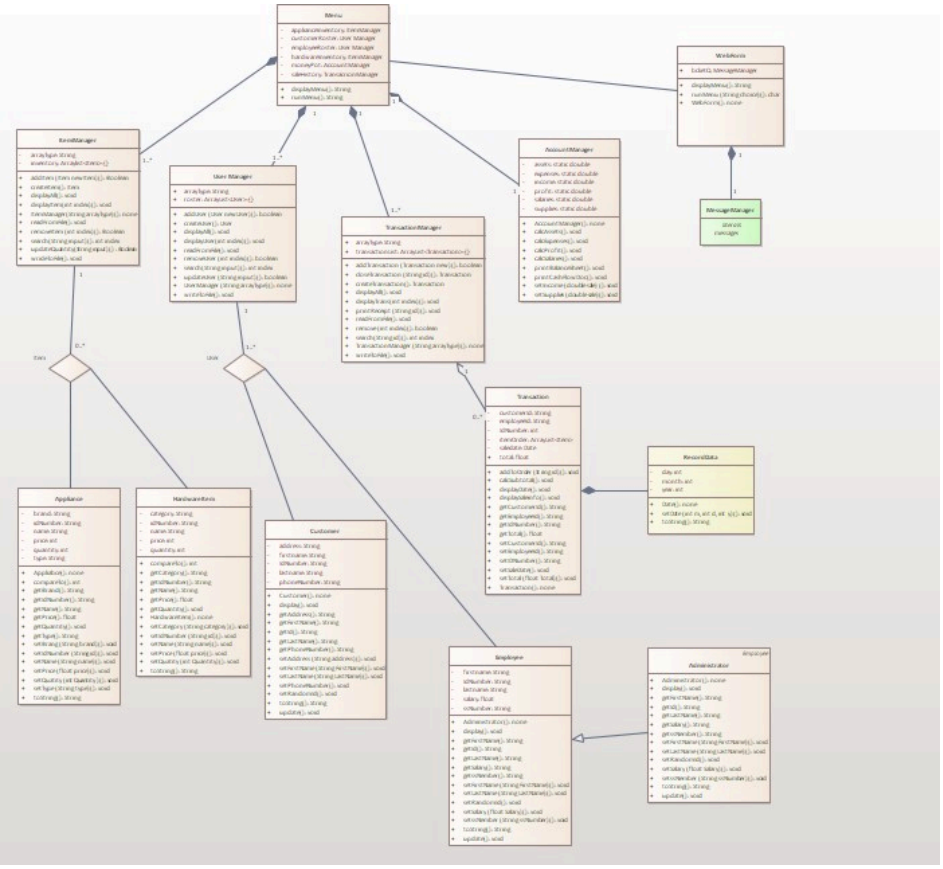


Рисунок 2.26 – Діаграма класів магазину, спроектована в Enterprise Architect

Для візуалізації внутрішньої структури електронного магазину техно-товарів та деталізації його складових елементів використовуються діаграми визначення блоків у форматі SysML.

Ключові блоки системи електронного магазину техно-товарів та їх призначення:

1. *Модуль управління товарами:*

Блок відповідає за всебічну роботу з каталогом товарів, включаючи додавання нових позицій, редагування існуючих, контроль наявності на складі, а також ведення та оновлення характеристик товарів.

2. *Модуль обробки замовлень:*

Блок забезпечує безперебійну обробку всіх етапів роботи з замовленнями - від моменту їх оформлення користувачем до підтвердження, обробки платежу, формування заявки на доставку та інформування користувача про статус замовлення.

3. Модуль управління клієнтами:

Блок відповідає за взаємодію з користувачами магазину, включаючи процеси реєстрації, авторизації, управління персональними даними, а також надання доступу до історії замовлень.

4. Модуль адміністрування:

Блок надає інструменти для централізованого управління системою магазину. Включаючи в себе налаштування параметрів роботи, моніторинг ключових показників, управління користувачами та їхніми правами доступу, а також забезпечення загальної безпеки та стабільності платформи.

Використання діаграм визначення блоків SysML дозволяє візуалізувати архітектуру електронного магазину та зрозуміти взаємозв'язки між його основними компонентами. (див. рис. 2.27).

На рисунку 2.28 зображено IBD діаграму, що описує внутрішні блоки електронного магазину, їх зв'язки та потоки даних між ними. Кожен блок представляє ключову функціональну частину магазину, а зв'язки демонструють, як ці блоки взаємодіють один з одним.

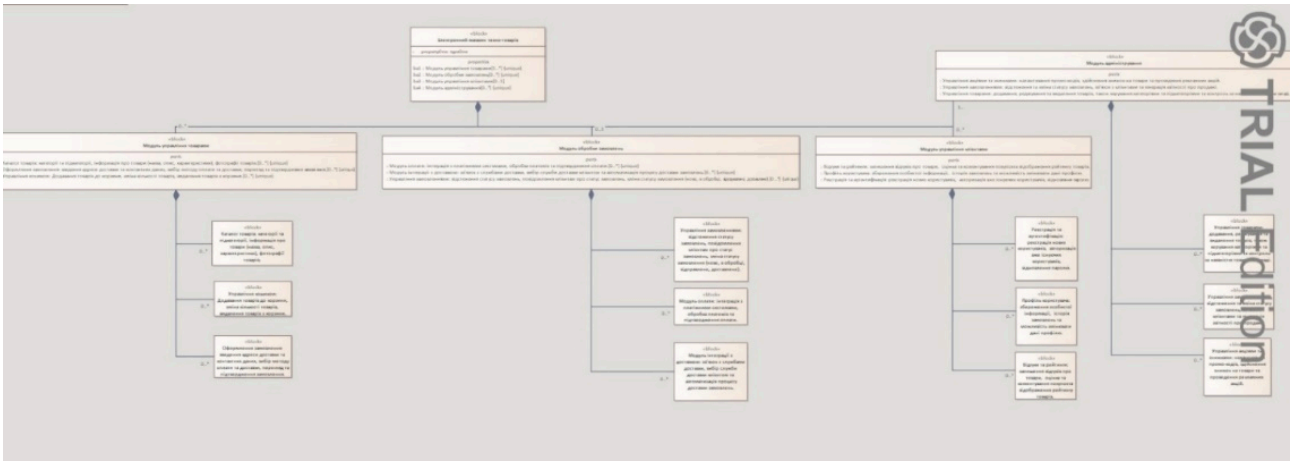


Рисунок 2.27 – Діаграма визначення блоків для подання внутрішньої структури системи (BDD)

Блоки:

1. Інтерфейс користувача (UI):

Блок відповідає за взаємодію користувача з магазином. Він включає елементи, такі як сторінки продуктів, кошик оформлення замовлення та особистий кабінет.

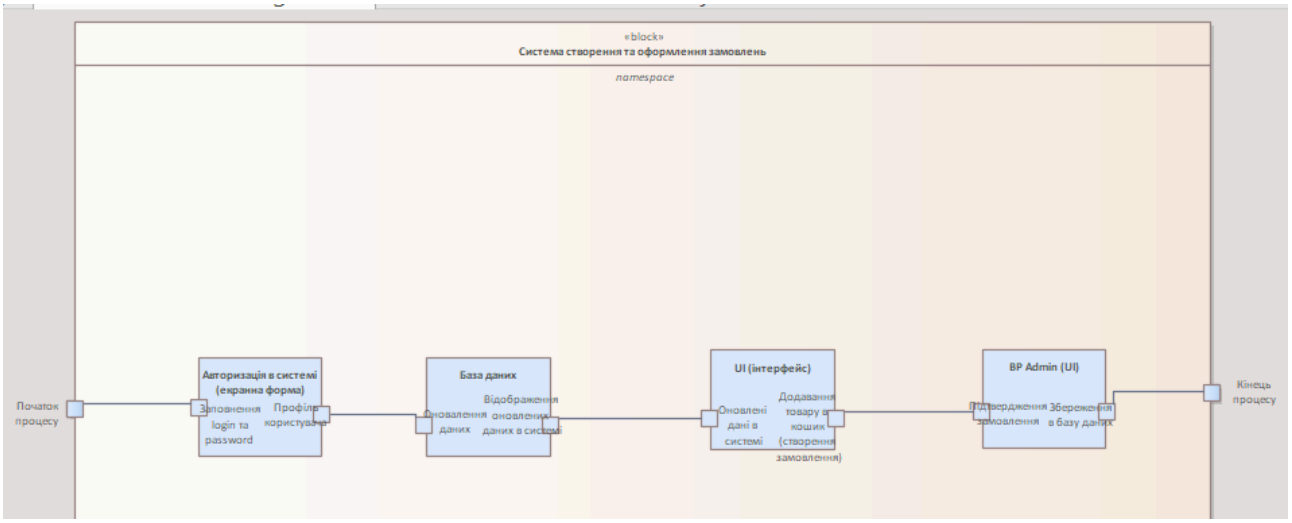


Рисунок 2.28 – Процес оформлення замовлення (робота магазину) (IBD)

2. *База даних:*

Блок зберігає всю інформацію про продукти, замовлення, користувачів та інші дані, необхідні для роботи магазину.

3. *Система авторизації:*

Блок відповідає за аутентифікацію та авторизацію користувачів, а також за управління їхніми профілями.

4. *Система обробки платежів:*

Блок відповідає за безпечну обробку платежів за замовлення, здійснені користувачами.

5. *Система управління замовленнями:*

Блок відповідає за обробку замовлень, включаючи прийняття замовлень, оновлення статусу замовлень та відстеження доставки.

6. *Система управління каталогом:*

Блок відповідає за управління каталогом продуктів, включаючи додавання, редагування та видалення продуктів, а також за підтримку актуальності інформації про продукти.

7. Система рекомендацій:

Блок пропонує користувачам персоналізовані рекомендації продуктів на основі їхньої історії покупок та переглядів.

Зв'язки:

UI блок зв'язаний з базою даних, щоб отримувати інформацію про продукти, замовлення та користувачів, а також для оновлення інформації про користувачів та замовлення. Система авторизації блок зв'язаний з UI, щоб отримувати дані для входу в систему від користувачів, а також з базою даних для перевірки автентичності користувачів та управління їхніми профілями. Система обробки платежів блок зв'язаний з UI, щоб отримувати інформацію про платіж від користувачів, а також з базою даних, щоб оновлювати статус платежів за замовлення. Система управління замовленнями блок зв'язаний з UI, щоб отримувати інформацію про нові замовлення від користувачів, а також з базою даних, щоб оновлювати інформацію про замовлення, відстежувати доставку та генерувати повідомлення про замовлення. Система управління каталогом блок зв'язаний з базою даних, щоб отримувати та оновлювати інформацію про продукти. Система рекомендацій блок зв'язаний з UI, щоб отримувати інформацію про перегляди та покупки користувачів, а також з базою даних, щоб отримувати інформацію про продукти та замовлення.

Потоки даних:

Користувач вводить дані для входу в систему. Дані для входу в систему надсилаються з UI до системи авторизації. Система авторизації перевіряє автентичність користувача. Система авторизації перевіряє дані для входу в систему в базі даних. Користувач переглядає продукти та інформація про продукти отримується з бази даних та відображається в UI. Користувач додає продукт до кошика та інформація про продукт та кількість додається до кошика

в UI. Користувач оформляє замовлення та інформація про замовлення надсилається з UI до системи управління замовленнями. Система управління замовленнями підтверджує замовлення. Система управління замовленнями оновлює статус замовлення в базі даних і надсилає повідомлення про підтвердження користувачеві. Користувач здійснює платіж та інформація про платіж надсилається з UI до системи обробки платежів.

2.2.2 Моделювання користувацького інтерфейсу

На рисунку 2.29 зображено приклади користувацького інтерфейсу магазину, створені в Enterprise Architect. Наведено взаємодію всіх інтерфейсів, можливості використання адмін панелі, сторінки з вибором товарів, пошуком та фільтрацією та сторінки особистого кабінету.

Інтерфейс сторінки з вибором товарів, пошуком та фільтрацією призначений для навігації користувачів по каталозі товарів, пошуку конкретних товарів та застосування фільтрів для полегшення пошуку. Він містить різноманітні елементи інтерфейсу, такі як поле пошуку, фільтри за категоріями або параметрами товарів, кнопки для вибору товару або переходу на сторінку товару для подальшого перегляду деталей.

Інтерфейс сторінки особистого кабінету призначений для взаємодії користувача з особистим кабінетом в електронному магазині. Він містить різноманітні функції, такі як перегляд та редагування особистої інформації користувача, перегляд історії замовлень, керування адресами доставки або способами оплати, зміна паролю тощо. Цей інтерфейс дозволяє користувачам керувати своїм особистим обліковим записом та здійснювати різноманітні дії, пов'язані з їх особистими налаштуваннями та історією.

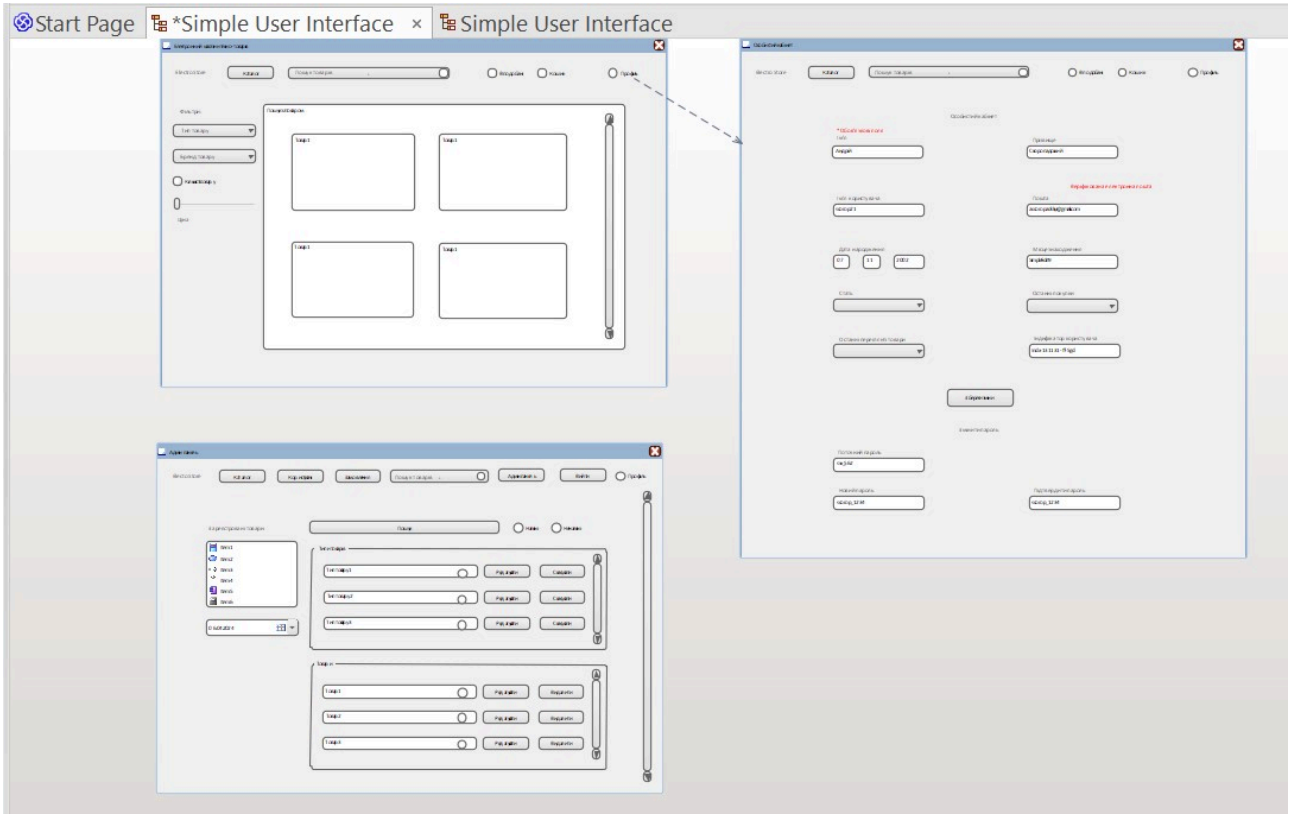


Рисунок 2.29 – Взаємодія всіх інтерфейсів змодельованих в Enterprise Architect

Інтерфейс сторінки адмін-панелі призначений для адміністраторів або управлінців електронного магазину технічних товарів і забезпечує їм доступ до різноманітних інструментів та функцій для управління магазином. Він містить різні розділи і опції, які дозволяють адміністраторам керувати каталогом товарів, цінами, акціями, аналізувати продажі, керувати замовленнями, управляти користувачами та їхніми правами, налаштовувати опції доставки та оплати, керувати сторінками контенту та іншими параметрами магазину.

2.2.3 Прототипування користувацьких інтерфейсів

Для наочної демонстрації користувацьких інтерфейсів та візуалізації основних сценаріїв взаємодії з електронним магазином було розроблено інтерактивний прототип за допомогою векторного онлайн-сервісу Figma.[18]

Прототип включає в себе як клієнтську частину, де відтворено ключові етапи покупки товару (перегляд каталогу, пошук, додавання товару до кошика,

оформлення замовлення), так і адміністраторську частину, що демонструє інструменти управління магазином (додавання та редагування товарів, обробка замовлень, перегляд статистики тощо).

Клікабельний прототип дозволяє оцінити зручність та інтуїтивність інтерфейсу, а також зрозуміти логіку роботи системи. Детальні зображення сторінок прототипу наведено у «Додатку В».

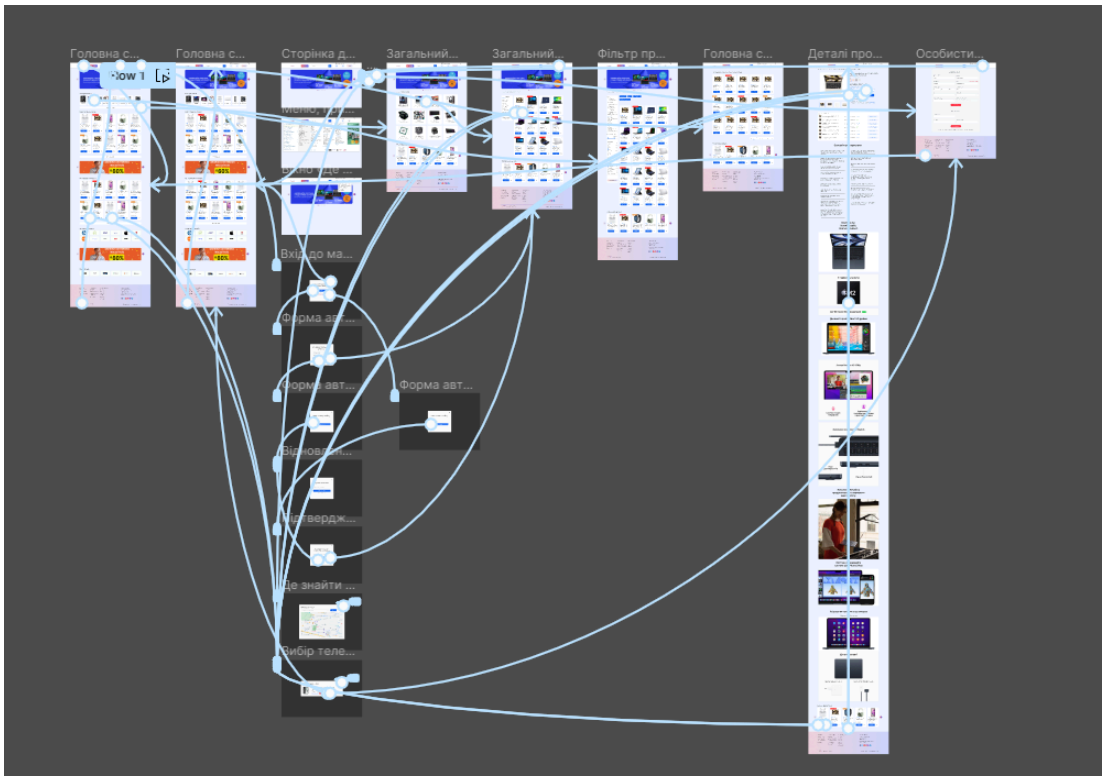


Рисунок 2.30 – Клікабельний прототип електронного магазину «Техно-товарів» розроблений у Figma

Головна сторінка електронного магазину розроблена з фокусом на інтуїтивності, зручності та візуальній привабливості. Навіть незареєстровані користувачі отримують доступ до ряду корисних функцій та інформації.

Динамічні слайдери з анімацією демонструють прототипи товарів, надаючи користувачам наочне уявлення про асортимент магазину. Форма авторизації дозволяє швидко увійти до системи або створити новий обліковий запис. У випадку втрати пароля, користувач може легко його відновити за допомогою спеціальної форми.

Після авторизації користувачеві відкривається доступ до особистого кабінету, де він може редагувати свої дані, переглядати історію покупок та користуватися іншими персоналізованими опціями. Випадне меню забезпечує зручну навігацію по сайту, надаючи швидкий доступ до ключових розділів та функцій.

Каталог товарів структуровано за категоріями та підкатегоріями, що дозволяє швидко знайти потрібну продукцію. Для деяких категорій передбачено окремі секції з новинками, хітами продажів та акційними пропозиціями.

Для зручності пошуку товарів реалізовано можливість фільтрації за різними параметрами та сортування за ціною. Додавання товару до обраного або кошика супроводжується анімаційними ефектами, що робить взаємодію з інтерфейсом більш приємною.

Комплексна структура сторінок, продумана система навігації та інтерактивні елементи спрямовані на створення максимально зручного та ефективного досвіду онлайн-шопінгу для кожного користувача.

Також, був проведений аналіз функціонала системи та узгодження з вимогами (див. табл. 2.3).

Таблиця 2.3 – Порівняння програмних систем

Priority	*User Story*	*Результат*
1	Користувач магазину хоче мати можливість швидко знаходити необхідні товари та переглядати їхні основні характеристики.	Розробка дизайну інтуїтивного і ефективного інтерфейсу пошуку та категоризації товарів для швидкого доступу до продуктів та їхніх характеристик.
2	Користувач хоче мати можливість зручно вибирати товари за фільтрами та сортувати їх за різними критеріями.	Дизайн сторінок з розширеним функціоналом фільтрації та сортування для надання користувачам можливості налаштовувати свої переваги та швидко вибирати необхідні товари.
3	Для користувача важливо бачити всі категорії та підкатегорії товарів, щоб швидко знаходити потрібний товар.	Створено відведену сторінку для категорій товарів, після якої можна перейти в підкатегорію для зручності пошуку товару за категоріями.
4	Дозволити користувачам легко шукати потрібні товари завдяки пошуку (за	Майже на кожній сторінці реалізований пошук, для швидкого знаходження товарів

	брендом товару, за назвою, за категорією і тд.)	або бреднів товарів, якщо користувач вже знає що хоче.
5	Підвищити візуальну привабливість сайту за допомогою анімації.	Інтеграція анімаційних ефектів на різних сторінках та елементах меню для поліпшення користувацького досвіду. Анімація кнопок, при наведення миші на картинку товару, інший ракурс товару і тд.
6	Користувач також має переглядати відгуки про товари та вибирати тільки якісні продукти.	Вдосконалення розділу з відгуками, включаючи розширений функціонал відгуків, рейтингів та фотографій користувачів.
7	Клієнт бажає залишити відгук, побажання чи повідомити про помилки	Створено додаткові сторінки для зворотного зв'язку та бажаним способом (гаряча лінія телефону чи електронна пошта).
8	Надання користувачу зручного та безпечного процесу онлайн-оплати товарів.	Покращення системи онлайн-оплати, включаючи нові методи платежів та додаткові заходи безпеки для забезпечення комфорту та надійності операцій.[24]
9	Користувач хоче отримувати персоналізовані рекомендації та акційні пропозиції, а також можливість переглядати тільки акційні товари, товари топ продажу або новинки.	Розробка системи персоналізованих рекомендацій та акцій, яка адаптується до індивідуальних потреб та історії покупок користувача. Також, фільтри за акційними товарами, товарами топ продажу або новинками.
10	Користувач повинен мати особистий кабінет для зберігання та відстеження своїх особистих даних та історії покупок, останні перегляди товарів та змогою змінити пароль.	Розробка функціонального особистого кабінету з можливістю управління даними, перегляду історії покупок та налаштування особистих параметрів, перегляд останніх збережених товарів та змогу змінити пароль до особистого кабінету.
11	Користувач повинен мати змогу швидко додавати та оформлювати замовлення, напиклад в 2 кліки.	Розроблений дизайн швидкого додавання до корзини та оплати в 2 кліки.
12	Користувач повинен мати можливість авторизації та реєстрації за номером телефону для забезпечення безпеки та швидкості входу на сайт.	Створення зручного дизайну системи авторизації та реєстрації, включаючи швидкий та безпечний вхід за номером телефону та можливість керування обліковим записом (також в разі потреби відновлення паролю за номером телефону).

Під час розробки електронного магазину для реалізації збереження даних, було використано реляційну БД MySQL. Для побудови схеми БД було використано СКБД MySQL Workbench.

Користувач із сервером зв'язуються за допомогою HTTP-запитів. Серед них є:

- запит на реєстрацію нового користувача;
- запит на авторизацію (отримання одноразового коду для авторизації);
- запит на авторизацію за певною роллю;
- запит на отримання особистої інформації користувача;
- запит на редагування особистих даних користувача в особистому кабінеті;
- запит на перегляд останніх покупок;
- запит на пошук за категоріями товарів;
- запит на пошук за брендом товару;
- запит на пошук за назвою товару;
- запит на пошук за фільтрами (ціна, акції, розстрочка і тд.);
- запит на отримання даних з кошику;
- запит на перегляд товарів в кошику;
- запит на видалення товарів в кошику;
- запит на створення відгуків та коментарів про товари;
- запит на видалення відгуків та коментарів про товари;

- запит на додавання товарів в обране;
- запит на видалення товарів з обраного;
- запит на створення нових категорій, підкатегорій товарів;
- запит на редагування категорій, підкатегорій товарів;
- запит на видалення категорій, підкатегорій товарів;
- запит на створення нових товарів;
- запит на редагування товарів;
- запит на видалення товарів;
- запит на перегляд адміністратором зареєстрованих користувачів;
- запит на видалення користувача адміністратором;
- запит на аналіз продажу певних товарів;
- запит на аналіз продажу певних категорій товарів;
- запит на аналіз додавання до обраного товарів користувачем;
- запит для аналітики найпопулярнішої категорії товарів;
- запит для аналітики найпопулярнішого товару;
- запит для аналітики загальної суми замовлень;

- запит на звернення в разі проблем до технічної підтримки.

РОЗДІЛ 3 ПРОЄКТУВАННЯ ТА РЕАЛІЗАЦІЯ КОМПОНЕНТІВ СИСТЕМИ

3.1 Інформаційне забезпечення

В процесі проектування інформаційної системи електронного магазину техно-товарів важливим кроком став вибір оптимальної системи управління базами даних (СУБД). Після аналізу різних варіантів, враховуючи специфіку проекту та його вимоги, було обрано MySQL – популярну реляційну СУБД з відкритим кодом.[9]

MySQL приваблює широкою популярністю, великою та активною спільнотою розробників, що гарантує доступність документації, підтримки та різноманітних інструментів. Висока продуктивність MySQL робить її придатною для обробки великої кількості транзакцій в реальному часі, що критично важливо для електронного магазину. Не менш важливим фактором є забезпечення належного рівня безпеки даних, адже система зберігає персональну інформацію користувачів та обробляє фінансові операції.[9]

Хоча MySQL має певні обмеження у порівнянні з більш потужними комерційними СУБД, такими як Oracle чи Microsoft SQL Server, вона задовольняє потреби даного проекту. MySQL демонструє хорошу масштабованість та гнучкість, а також пропонує легкий вхідний поріг та швидку налаштування, що робить її ідеальним вибором для стартапів та проектів середнього розміру.

Порівняння з іншими популярними СУБД, такими як PostgreSQL чи NoSQL-системами на кшталт MongoDB, показало, що MySQL пропонує оптимальний баланс між функціональністю, продуктивністю, надійністю, доступністю підтримки та вартістю для даного проекту.

Окрім вибору СУБД, важливою складовою проектування інформаційної системи є розробка зручного та інтуїтивно зрозумілого користувацького

інтерфейсу. Основна мета – забезпечити ефективну взаємодію користувача з магазином. Це досягається за допомогою продуманих вхідних форм для реєстрації, авторизації, оформлення замовлень, а також інформативних вихідних форм, що надають дані про замовлення, оплату та доставку.

Підхід до UX/UI ґрунтується на принципах простоти, зручності та безпеки. Мінімалістичний дизайн, швидка робота інтерфейсу та надійний захист персональних даних – це ключові аспекти, що формують позитивний досвід взаємодії користувачів з електронним магазином.

Візуалізація моделі бази даних здійснена в середовищі MySQL Workbench та представлена на рисунку 3.1.

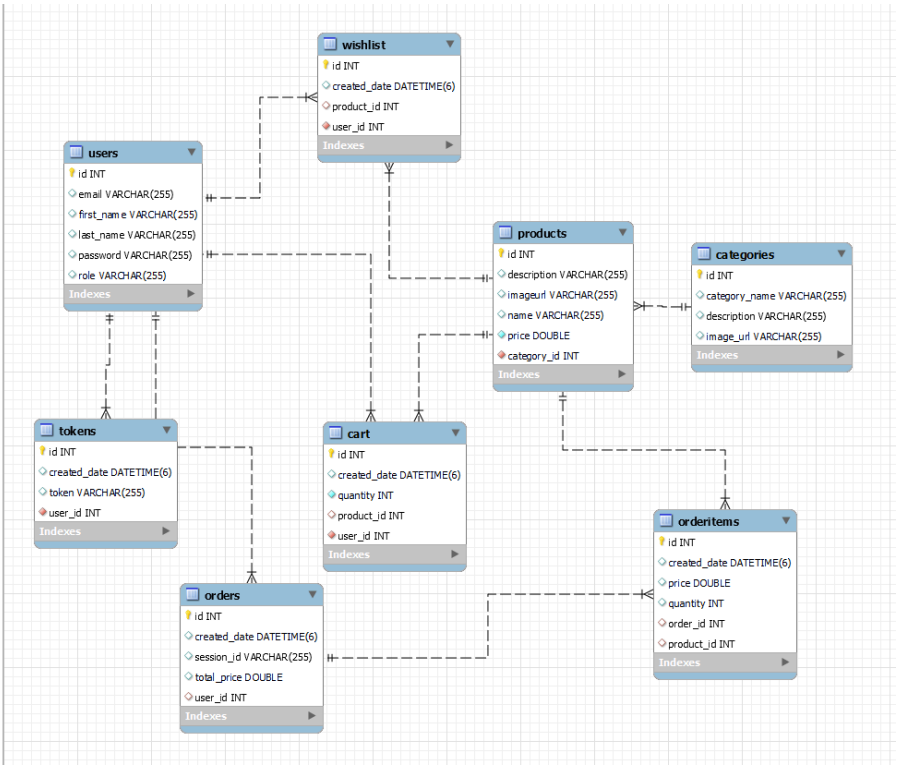


Рисунок 3.1 – Er-діаграма створена в MySQL Workbench

Результат перетворення ER-моделі в схему бази даних на основі СУБД MySQL та вміст файлу згенерованої моделі даних представлено у «Додатку Г».

Специфікація вхідних та вихідних даних електронного магазину електро-товарів представлена на рис. 3.2-3.9.

Table Name: users

Column Name	Datatype	PK	NN	UQ	B	UN	ZF	AI	G	Default/Express
id	INT	☑	☑	☐	☐	☐	☐	☑	☐	
email	VARCHAR(255)	☐	☐	☐	☐	☐	☐	☐	☐	NULL
first_name	VARCHAR(255)	☐	☐	☐	☐	☐	☐	☐	☐	NULL
last_name	VARCHAR(255)	☐	☐	☐	☐	☐	☐	☐	☐	NULL
password	VARCHAR(255)	☐	☐	☐	☐	☐	☐	☐	☐	NULL
role	VARCHAR(255)	☐	☐	☐	☐	☐	☐	☐	☐	NULL

Рисунок 3.2 – Специфікація таблиці «users»

Table Name: tokens

Column Name	Datatype	PK	NN	UQ	B	UN	ZF	AI	G	Default/Expres
id	INT	☑	☑	☐	☐	☐	☐	☑	☐	
created_date	DATETIME(6)	☐	☐	☐	☐	☐	☐	☐	☐	NULL
token	VARCHAR(255)	☐	☐	☐	☐	☐	☐	☐	☐	NULL
user_id	INT	☐	☑	☐	☐	☐	☐	☐	☐	

Рисунок 3.3 – Специфікація таблиці «tokens»

Table Name: products

Column Name	Datatype	PK	NN	UQ	B	UN	ZF	AI	G	Default/Expressi
id	INT	☑	☑	☐	☐	☐	☐	☑	☐	
description	VARCHAR(255)	☐	☐	☐	☐	☐	☐	☐	☐	NULL
imageurl	VARCHAR(255)	☐	☐	☐	☐	☐	☐	☐	☐	NULL
name	VARCHAR(255)	☐	☐	☐	☐	☐	☐	☐	☐	NULL
price	DOUBLE	☐	☑	☐	☐	☐	☐	☐	☐	
category_id	INT	☐	☑	☐	☐	☐	☐	☐	☐	

Рисунок 3.4 – Специфікація таблиці «products»

Table Name: categories

Column Name	Datatype	PK	NN	UQ	B	UN	ZF	AI	G	Defau
id	INT	☑	☑	☐	☐	☐	☐	☑	☐	
category_name	VARCHAR(255)	☐	☐	☐	☐	☐	☐	☐	☐	NULL
description	VARCHAR(255)	☐	☐	☐	☐	☐	☐	☐	☐	NULL
image_url	VARCHAR(255)	☐	☐	☐	☐	☐	☐	☐	☐	NULL

Рисунок 3.5 – Специфікація таблиці «categories»

Table Name: cart

Column Name	Datatype	PK	NN	UQ	B	UN	ZF	AI	G	Default/I
id	INT	☑	☑	☐	☐	☐	☐	☑	☐	
created_date	DATETIME(6)	☐	☐	☐	☐	☐	☐	☐	☐	NULL
quantity	INT	☐	☑	☐	☐	☐	☐	☐	☐	
product_id	INT	☐	☐	☐	☐	☐	☐	☐	☐	NULL
user_id	INT	☐	☑	☐	☐	☐	☐	☐	☐	

Рисунок 3.6 – Специфікація таблиці «cart»

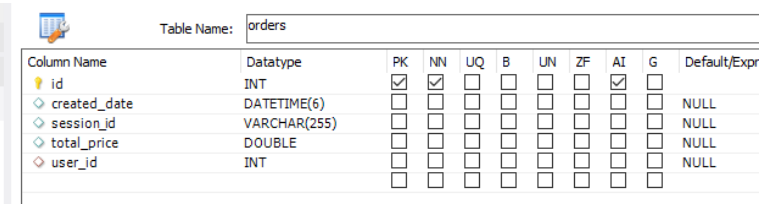


Рисунок 3.7 – Специфікація таблиці «orders»

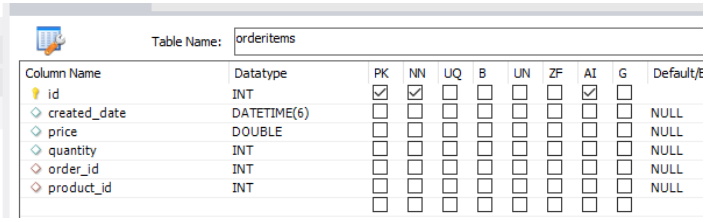


Рисунок 3.8 – Специфікація таблиці «orderitems»

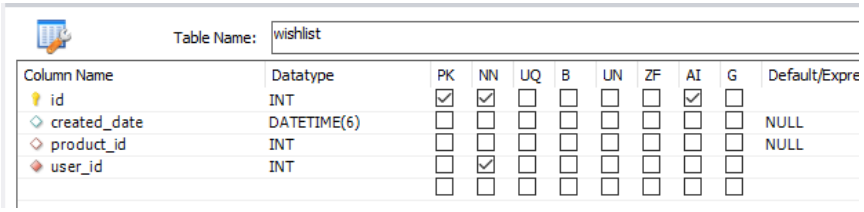


Рисунок 3.9 – Специфікація таблиці «wishlist»

База даних для електронного магазину техно-товарів демонструє структуровану систему для зберігання та управління ключовою інформацією, необхідною для функціонування платформи. Вона складається з восьми взаємопов'язаних таблиць, кожна з яких відіграє свою унікальну роль.

Таблиця "users" є основою для ідентифікації користувачів та зберігання їхніх персональних даних, включаючи email, ім'я, прізвище, пароль та роль в системі. Таблиця "tokens" забезпечує безпеку та контроль доступу до облікових записів користувачів, зберігаючи токені аутентифікації. Особисті вподобання користувачів відображаються в таблиці "wishlist", де зберігаються списки бажаних товарів, пов'язуючи ідентифікатор користувача з ідентифікатором товару.

Таблиця "cart" відображає вміст кошика кожного користувача, зберігаючи інформацію про вибрані товари та їх кількість. Інформація про оформлені

замовлення зберігається в таблиці "orders", яка містить дату створення, ідентифікатор сесії, загальну вартість замовлення та ідентифікатор користувача. Детальна інформація про товари, що входять до складу кожного замовлення, представлена в таблиці "orderItems".

Таблиця "products" є центральним елементом бази даних, зберігаючи всю необхідну інформацію про товари, включаючи їх назву, опис, посилання на зображення, ціну та приналежність до певної категорії. Інформація про категорії товарів зберігається в таблиці "categories", що дозволяє структурувати каталог товарів та спростити навігацію для користувачів.

3.2 Організаційне забезпечення

Організаційна структура електронного магазину може представлена у вигляді ієрархічної схеми, що відображає взаємозв'язок між різними відділами та посадами. Ключовими підрозділами є: відділ продажів, відповідальний за обробку замовлень та взаємодію з клієнтами; відділ маркетингу, що займається просуванням магазину та залученням нових клієнтів; відділ логістики, що забезпечує своєчасну доставку товарів; відділ IT, що підтримує роботу інформаційної системи та веб-сайту.

Не менш важливим елементом організаційного забезпечення є опис основних бізнес-процесів, таких як обробка замовлень, управління запасами, обробка платежів, організація доставки, обробка повернень тощо. Детальний опис кожного процесу включає в себе послідовність дій, відповідальних осіб, необхідні документи та інформаційні системи, що залучені до його виконання (продемонстровано в підрозділі 2.1.3).

Організація ефективних інформаційних потоків є необхідною умовою для злагодженої роботи електронного магазину. Вона передбачає встановлення чітких правил збору, обробки, зберігання та передачі інформації між різними відділами та співробітниками.

Необхідно вжити заходів для захисту персональних даних користувачів, забезпечення безпеки фінансових операцій та запобігання несанкціонованому доступу до інформаційної системи.[24]

1. Захист персональних даних:

Шифрування даних – вся конфіденційна інформація, така як паролі, номери кредитних карток та адреси, повинна зберігатися в зашифрованому вигляді.

Контроль доступу – встановлення різних рівнів доступу до даних для співробітників в залежності від їхніх обов'язків та ролей в системі.

2. Безпека фінансових операцій:

Використання надійних платіжних шлюзів – співпраця з перевіреними платіжними системами, що забезпечують безпеку та надійність фінансових транзакцій.

Двофакторна аутентифікація – підвищення рівня безпеки при проведенні платежів за допомогою додаткового підтвердження операції через SMS або мобільний додаток.[24]

Моніторинг підозрілої активності – впровадження систем моніторингу та аналізу фінансових операцій для виявлення та запобігання шахрайству.

3. Запобігання несанкціонованому доступу:

Використання надійних паролів та двофакторної аутентифікації для доступу до адміністративної панелі та інших чутливих розділів системи.

Регулярне оновлення програмного забезпечення – встановлення останніх версій програмного забезпечення, що містять виправлення вразливостей та покращення безпеки.

Далі на рисунку 3.10 продемонстрована діаграма станів. Стани в які може переходити система при взаємодії з користувачем показано на UML-діаграмі станів.

Рисунок 3.10 – Statechart diagram (діаграма станів) – послідовність станів замовлення в процесі його обробки в електронному магазині

1. *Підтвердити замовлення (Confirm order):*

Процес починається з підтвердження замовлення. Це перший стан, в якому перебуває замовлення після його створення.

2. *Підготувати замовлення (Prepare order):*

Після підтвердження замовлення переходить у стан підготовки, де можливо відбувається комплектація товарів або інші процеси підготовки замовлення до відправлення.

3. *Оплатити замовлення (Pay for the order):*

У цей час замовлення може бути оплачене. Це критичний етап, який зазвичай має бути завершений, перш ніж замовлення може бути відправлене.

4. *Відхилити замовлення (Cancel the order):*

Є також можливість, що замовлення буде відхилене на будь-якому етапі перед його оплатою.

5. *Доставити замовлення (Deliver the order):*

Після оплати замовлення переходить у стан доставки, де воно перебуває до моменту вручення клієнту.

Діаграма активності/дії представляє процеси, через які проходить користувач при використанні електронного магазину електро-товарів, від реєстрації до оформлення замовлення (див. рис. 3.12).

4. Реєстрація користувача:

Процес починається з реєстрації користувача, де він може зареєструватися або ввійти в систему, якщо вже має аккаунт.

5. Перегляд та пошук товарів:

Після входу, користувач може переглянути або пошукати товари. Також є можливість переглянути статус свого аккаунту.

Якщо користувач хоче вийти з системи, він може вибрати опцію.

6. Додавання товарів до кошика:

Користувач може додавати товари до свого кошика та переглядати його вміст.

У кошику можна змінювати кількість товарів або видаляти їх з кошика.

7. Оформлення покупки:

Після того як користувач завершив вибір товарів, він може перейти до оформлення покупки, натиснувши.

8. Оплата:

Процес оплати включає інтеграцію з платіжними системами та можливо з POS (Point of Sale) системами.

9. Доставка та обробка:

Після оплати здійснюється обробка замовлення та організовується доставка.

10.Підтвердження замовлення:

Для завершення, користувач отримує підтвердження замовлення, після чого процес закінчується.

Кожен крок представлений у вигляді блоку, що описує дію або рішення. Стрілки показують напрямок потоку від одного кроку до наступного, а ромби представляють точки рішення, де користувач або система мають вибрати між декількома шляхами дій.

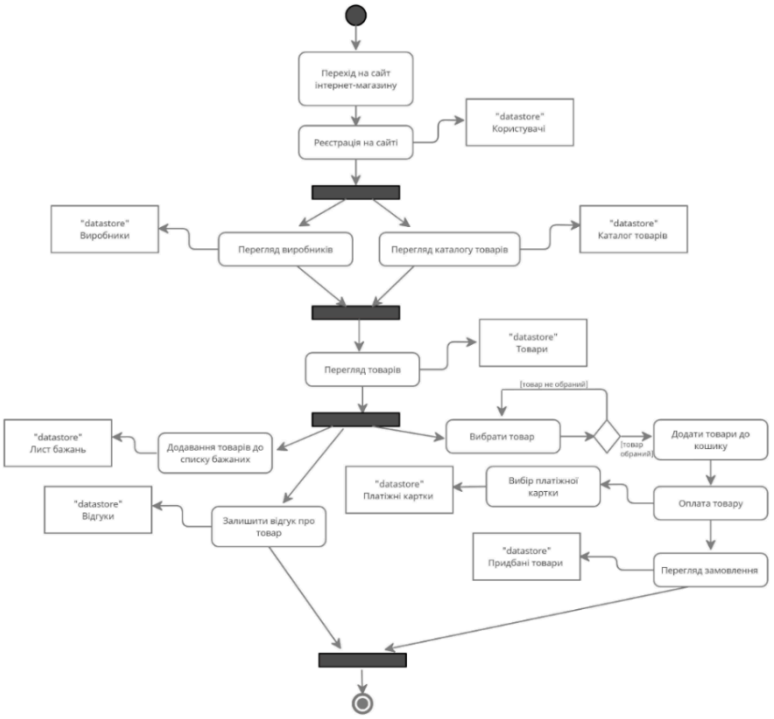


Рисунок 3.11 – Діаграма активності представляє процеси, через які проходить користувач при використанні електронного магазину техно-товарів

3.3 Програмне забезпечення

Для створення електронного магазину техно-товарів ми обрали потужний та сучасний стек технологій, який дозволить реалізувати всі заплановані функції та забезпечити високу продуктивність, безпеку та зручність користування.

Головною частиною магазину стане надійна та перевірена база даних MySQL. Висока продуктивність MySQL дозволить обробляти численні запити та транзакції в режимі реального часу, гарантуючи швидку роботу магазину навіть при високому навантаженні.

Логіка роботи магазину буде реалізована на мові програмування Java з використанням фреймворку Spring Boot. Java – це потужна та універсальна мова, що дозволяє створювати складні та надійні веб-додатки.[10]

Для розробки клієнтської частини було обрано комплексний набір сучасних технологій, що забезпечують високу продуктивність, зручність користування та естетичну привабливість інтерфейсу. В якості основи для створення інтерактивного та динамічного інтерфейсу використовується прогресивний Js-фреймворк Vue.js, що дозволяє розробляти гнучкі та масштабовані компонентні веб-додатки. Для пришвидшення розробки та забезпечення візуальної консистентності застосовується фронтенд-фреймворк Bootstrap, що надає широкий спектр готових компонентів інтерфейсу. Фундаментальні технології веб-розробки — HTML, CSS та JavaScript — використовуються для структурування, візуального оформлення та надання інтерактивності сторінкам магазину.[14] Webpack забезпечує оптимізацію та збірку фронтенд-коду, що позитивно впливає на швидкість завантаження сторінок.[17]

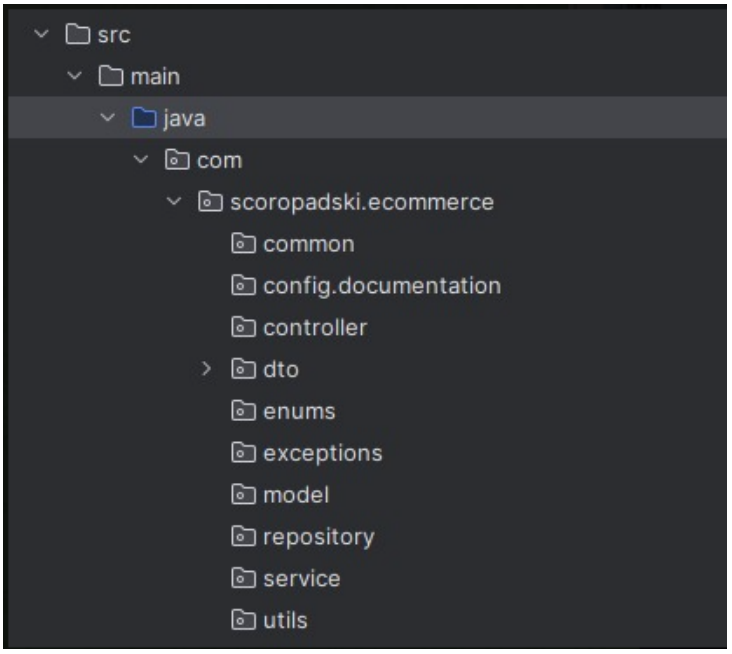


Рисунок 3.12 – Структура packages back-end в середовищі Intelij Idea

Для демонстрації практичної реалізації бекенд-логіки електронного магазину техно-товарів в «Додатку Б» до цього проекту наведено фрагменти коду контролерів. Ці фрагменти ілюструють принципи обробки запитів від користувачів, взаємодії з базою даних MySQL та формування відповідей для клієнтської частини магазину.

Код контролерів демонструє використання фреймворку Spring Boot для спрощення розробки та забезпечення ефективної роботи з даними. Він також показує використання різних методів HTTP ("GET", "PUT", "POST", "PATCH", "DELETE", "OPTIONS") для обробки різних типів запитів від клієнтів. Аналіз коду контролерів дозволяє зрозуміти принципи функціонування бекенд-частини магазину та взаємодію між різними компонентами системи.

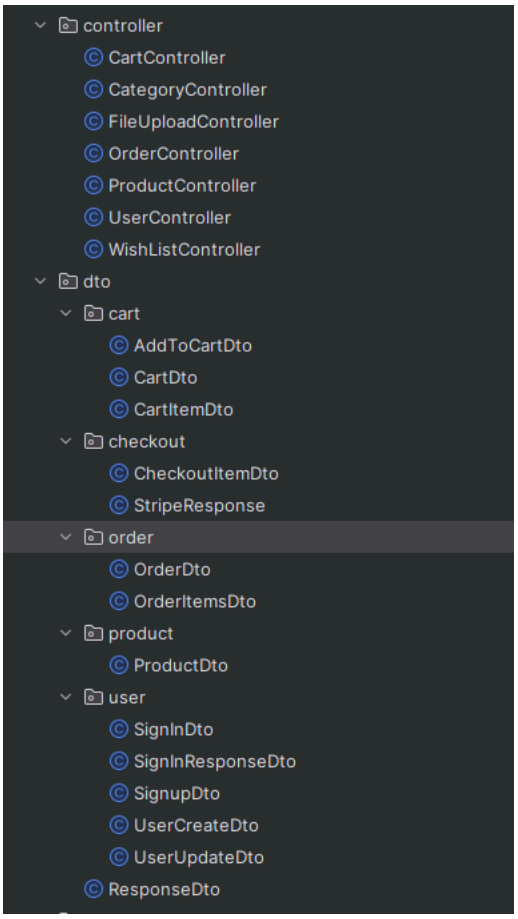


Рисунок 3.13 – Структура back-end частини (Controllers, DTO) в середовищі Intelij Idea

На рисунку 3.13 представлена структура пакету controller та dto (Data Transfer Object) в проекті електронного магазину. Кожен контролер відповідає за певний аспект функціональності магазину, наприклад, CartController керує операціями з кошиком, ProductController – з товарами, UserController – з користувачами, а OrderController - з замовленнями.

DTO використовуються для передачі даних між різними частинами додатку. Наприклад, AddToCartDto містить інформацію, необхідну для додавання товару до кошика, а OrderDto – для оформлення замовлення. Така структура забезпечує чіткий розподіл відповідальності та полегшує розробку та підтримку коду.

Структура демонструє використання принципів об'єктно-орієнтованого програмування та архітектурного підходу MVC для створення чіткої та організованої структури програмного коду [8].

MySQL, обрана як система керування базами даних (СУБД) для цього проекту, служить надійним сховищем для всієї інформації, необхідної для ефективного функціонування електронного магазину. Включаючи дані про товари, користувачів, замовлення та інші важливі аспекти. Детальний опис структури бази даних, розробленої з урахуванням специфічних вимог проекту, представлено в «Додатку Г».

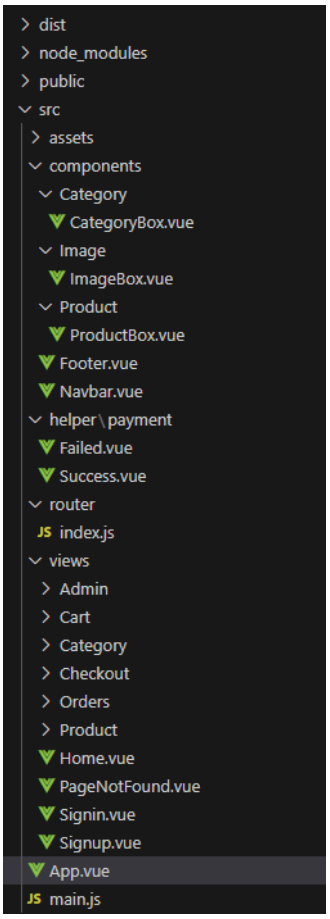


Рисунок 3.14 – Структура front-end частини в середовищі Visual Studio Code

Фронтенд частина електронного магазину, розроблена на Vue.js, має чітку та логічну структуру, що сприяє ефективній розробці та підтримці коду [12]. Всі файли проекту організовані в папки, що відображають їх призначення. Папка dist містить згенеровані файли, готові до розгортання на веб-сервері, включаючи CSS, JavaScript та зображення [16]. Папка node_modules зберігає всі залежності проекту, встановлені за допомогою менеджера пакетів npm. Статичні файли, які копіюються в папку dist без змін, розташовуються в папці public.

Фрагменти коду для front-end частини, а саме «HTML код головної сторінки магазину», «Фрагменти CSS-коду» та «Фрагменти JS-коду» представлено в «Додатку Б».

Основний вихідний код проекту знаходиться в папці src. Вона поділена на кілька підпапок для кращої організації. Папка assets містить статичні ресурси, такі як зображення або шрифти. Папка components містить компоненти Vue.js,

які є основними будівельними блоками користувацького інтерфейсу. Кожен компонент має свій файл .vue, що містить HTML-розмітку, CSS-стилі та JavaScript-логіку.[26] Папка helper містить допоміжні функції та утиліти, що використовуються в різних частинах проекту.

Конфігурація маршрутизації додатку, що визначає відповідність між URL-адресами та компонентами Vue.js, знаходиться в папці router. Папка views містить компоненти Vue.js, що відповідають за відображення окремих сторінок додатку. Файл main.js є точкою входу до додатку, він ініціалізує Vue.js та підключає необхідні компоненти та плагіни.

Структура проекту забезпечує чіткий розподіл коду за його функціональністю, що робить фронтенд-частину електронного магазину більш зрозумілою та легкою в підтримці.

На рисунку 3.15 зображена діаграма компонентів, де показано структуру взаємодії клієнтів та адміністратора з різними частинами електронної системи магазину. Ось основні елементи діаграми:

Клієнт (зліва):

Розпочинає взаємодію з системою через інтерфейс реєстрації ("Interface" Реєстрація). Після реєстрації можливий вхід до системи через відповідний інтерфейс ("Interface" Вхід до системи). Далі користувачу доступний модуль реєстрації (Модуль реєстрації), який веде до головної сторінки ("Interface" Головна сторінка). З головної сторінки клієнт може перейти до форми замовлення ("Interface" Форма замовлення).

Модулі обробки замовлення (у центрі):

Замовлення товару здійснюється через вказану форму, після чого інформація передається в модуль замовлення (Замовлення товару).

Товар може бути вибраний із каталогу (Каталог товарів), який доступний як клієнту, так і адміністратору. Оплата товару (Оплата товару) теж є частиною процесу замовлення.

Адміністратор (зправа):

Має доступ до заявки на бронювання (Заявка на бронювання), що, судячи з контексту, пов'язано з обробкою замовлень або управлінням запасами.

Рахунок на оплату (праворуч внизу):

Після замовлення товару інформація передається для генерації рахунку (Рахунок на оплату).

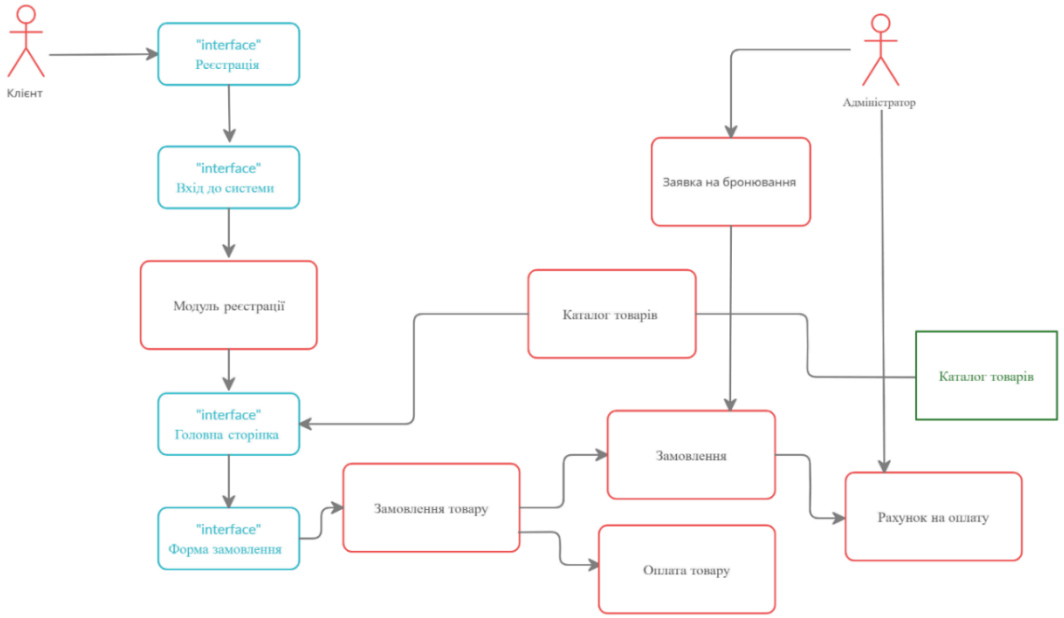


Рисунок 3.15 – Діаграма компонентів електронного магазину електро-товарів – ілюстрацією компоненти кожного апаратного та програмного вузла

3.4 Технічне забезпечення

Діаграма розгортання, представлена на рисунку 3.16, візуалізує архітектуру електронного магазину техно-товарів, розділяючи її на три ключові компоненти: фронтенд, бекенд на Spring Boot та сервер з базою даних MySQL.

Фронтенд представлений як окремий блок, що містить всі необхідні елементи для взаємодії з користувачем. Сюди входять:

- шаблони (Templates) – визначають структуру та зовнішній вигляд сторінок магазину, реалізовані за допомогою HTML, CSS та JavaScript;

- компоненти системи (Components of system) – це модулі JavaScript-коду, стилів CSS та інших ресурсів, що формують інтерактивні елементи інтерфейсу, розроблені за допомогою Vue.js;
- сервіси (Services): забезпечують обробку запитів від користувачів, здійснюють взаємодію з бекендом та оновлюють дані на сторінках магазину.

Бекенд на Spring Boot – це серверна частина додатку, що відповідає за обробку бізнес-логіки та взаємодію з базою даних. Його складові:

- MVC-контролери реалізують шаблон проектування Model-View-Controller, керуючи обміном даними між моделлю, що представляє дані та бізнес-логіку, та представленням, що формує користувацький інтерфейс;
- модель електронного магазину (Model of Electro-shop) містить класи та об'єкти, що відображають сутності предметної області (товар, користувач, замовлення тощо) та реалізують бізнес-правила магазину;
- сервіси (Service) містять бізнес-логіку магазину, таку як обробка замовлень, управління кошиком та інтеграція з платіжними системами;
- DAO (Data Access Object) або репозиторії (Repository) забезпечують абстракцію доступу до бази даних, спрощуючи взаємодію з MySQL та приховуючи деталі реалізації.

Сервер – це фізичний або віртуальний сервер, на якому розгорнуто базу даних MySQL.

- MySQL Main_Database – реляційна база даних, що зберігає всі дані електронного магазину, такі як інформація про товари, користувачів, замовлення, категорії тощо.

Взаємодія між компонентами системи відбувається за допомогою мережевих протоколів. Фронтенд надсилає запити до бекенду, який обробляє їх, взаємодіє з базою даних через DAO або репозиторії, та повертає відповідь фронтенду. Така розподілена архітектура забезпечує гнучкість, масштабованість та надійність електронного магазину.

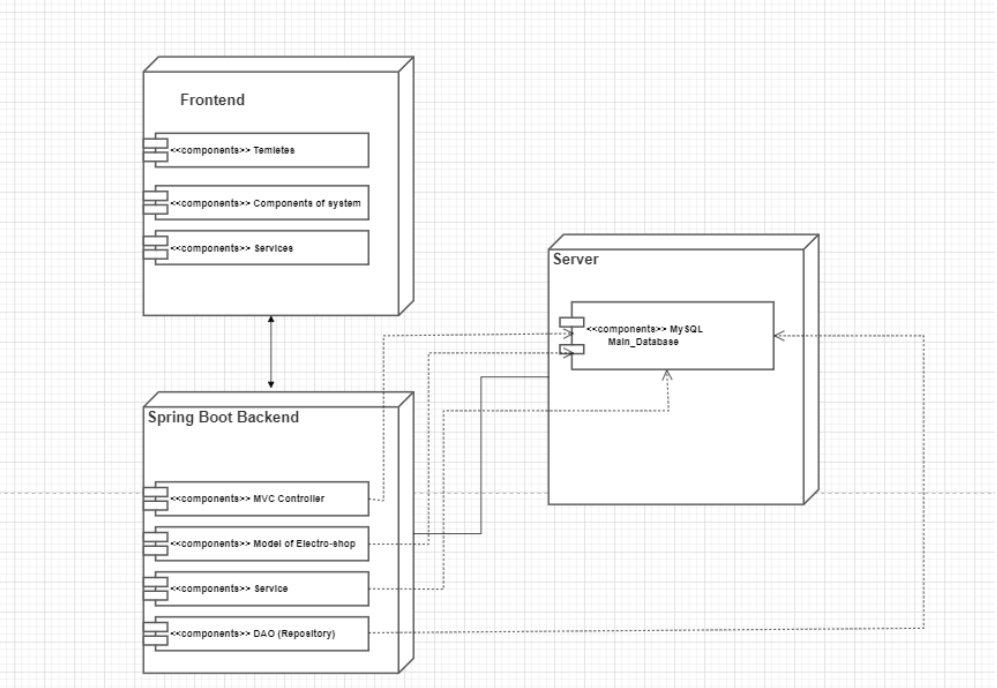


Рисунок 3.16 – Діаграма розгортання (Deployment)

3.5 Результати реалізації інформаційної системи

В результаті виконання кваліфікаційного бакалаврського проекту «Проектування електронного магазину техно-товарів» було розроблено комплексну ІС, що відповідає сучасним вимогам ринку електронної комерції та забезпечує зручність як для користувачів, так і для адміністраторів.

Проект розпочався з ґрунтовного аналізу предметної області та вивчення існуючих рішень на ринку електронних магазинів техно-товарів в Україні. Було визначено основні функціональні та нефункціональні вимоги до системи, а також сформульовано бізнес-цілі проекту, що лягли в основу подальшої розробки.

Для забезпечення гнучкості, масштабованості та зручності супроводження системи було обрано архітектурний підхід MVC (Model-View-Controller) та гнучку методологію розробки (Agile). Це дозволило ефективно організувати процес розробки, швидко реагувати на зміни вимог та забезпечити активну участь замовника на всіх етапах проекту.

Була розроблена детальна інформаційна модель системи, що включає в себе діаграми прецедентів, діаграми поведінки системи (діаграми послідовності та діаграми активності), а також описи типових сценаріїв використання. Ця модель надає повне уявлення про функціональність магазину та дозволяє зрозуміти, як різні компоненти системи взаємодіють між собою.

Особливу увагу було приділено проектуванню бази даних. В якості СУБД було обрано MySQL, що відрізняється надійністю, швидкістю роботи та масштабованістю. Була розроблена детальна ER-діаграма, що відображає структуру таблиць бази даних та зв'язки між ними, а також створено SQL-скрипт для її генерації.

Для реалізації бекенд частини системи була використана мова програмування Java та фреймворк Spring Boot. Spring Boot значно спростив розробку та розгортання додатку, надавши готові рішення для роботи з базою даних, безпекою, веб-сервісами та іншими важливими аспектами [11].

Фронтенд частина системи була реалізована за допомогою JavaScript-фреймворку Vue.js. Він дозволив створити інтерактивний та зручний користувацький інтерфейс, використовуючи компонентний підхід. Для пришвидшення розробки та забезпечення візуальної консистентності було задіяно фреймворк Bootstrap, а також фундаментальні технології веб-розробки - HTML, CSS та JavaScript [14]. Webpack використовувався для збірки та оптимізації фронтенд-коду, а Figma допомогла створити інтерактивний прототип інтерфейсу [17].

Для забезпечення якості системи було розроблено тестову документацію, що включає в себе модульні, інтеграційні, системні та приймальні тести, а також тести-сценарії.

Реалізований електронний магазин техно-товарів надає користувачам широкий спектр можливостей для зручного та безпечного онлайн-шопінгу. Завдяки продуманій структурі бази даних та інтуїтивно зрозумілому інтерфейсу, клієнти можуть легко знаходити потрібні товари, додавати їх до кошика, оформлювати замовлення та відслідковувати їх статус в режимі реального часу.

Перед початком покупок користувачам необхідно створити обліковий запис, що дозволяє зберігати персональну інформацію, історію замовлень та формувати списки бажаних товарів. Каталог товарів чітко структурований за категоріями та підкатегоріями, а також оснащений потужним пошуком, що дозволяє швидко знаходити потрібні товари за назвою, брендом, характеристиками тощо.

Процес оформлення замовлення максимально спрощений та зрозумілий - користувач обирає спосіб оплати та доставки, вказує адресу та підтверджує замовлення.

Адміністратори магазину мають доступ до розширеного функціоналу для ефективного управління всіма аспектами роботи магазину. Вони можуть додавати, редагувати та видаляти товари, керувати категоріями, обробляти замовлення, контролювати платежі та доставку, а також взаємодіяти з користувачами.

Створений електронний магазин техно-товарів є конкурентоспроможним рішенням, здатним задовольнити потреби сучасних покупців та забезпечити ефективність бізнес-процесів.

РОЗДІЛ 4 ТЕСТУВАННЯ І ВЕРИФІКАЦІЯ ПРОЄКТНИХ РІШЕНЬ

4.1 Розробка тест-кейсів

У процесі проектування магазину була зроблена розробка тест-кейсів різних видів для проектованої системи. Кожен вид тест-кейсів має своє призначення та важливість у процесі тестування програмного забезпечення [21].

Модульні тест-кейси використовуються для перевірки правильності роботи окремих модулів або компонентів програми. Вони дозволяють ізолювати та тестувати окремі частини програми, сприяючи виявленню помилок на ранніх етапах розробки.

Модульні тест-кейси:

- тестування функції додавання товару до кошика;
- перевірка коректності обробки даних введених користувачем у формі реєстрації;
- тестування алгоритму фільтрації товарів за певними критеріями.

Інтеграційні тест-кейси необхідні для перевірки взаємодії між різними модулями або компонентами програми після їх об'єднання. Вони допомагають виявити проблеми, що виникають при інтеграції програми, та перевіряють правильність обміну даними та інші взаємодії.

Інтеграційні тест-кейси:

- перевірка взаємодії між модулем управління товарами та модулем обробки замовлень;
- тестування взаємодії між інтерфейсом користувача та базою даних.

Системні тест-кейси призначені для перевірки функціональності програми як єдності, включаючи всі її компоненти та взаємодії між ними. Вони перевіряють відповідність програми вимогам та коректність її роботи в реальних умовах.

Системні тест-кейси:

- тестування процесу замовлення товару від початку до кінця, включаючи оплату та доставку;
- перевірка взаємодії між різними компонентами системи під час виконання різних операцій.

Приймальні тест-кейси використовуються для перевірки відповідності програми вимогам та очікуванням клієнта. Вони перевіряються в контексті реального середовища використання та допомагають визначити, чи відповідає програма очікуванням користувачів.

Приймальні тест-кейси:

- тестування користувальницького інтерфейсу з погляду зручності та інтуїтивності;
- перевірка коректності відображення інформації про товари та їх характеристики.

Тести-сценарії використовуються для тестування поведінки програми в конкретних ситуаціях або за конкретними умовами. Вони дозволяють перевірити, як програма взаємодіє з користувачем або з іншими системами та виявити непередбачені або несподівані поведінки.

Тести-сценарії:

- сценарій перевірки процесу оформлення замовлення, включаючи випадки успішного та неуспішного завершення;
- тестування сценарію пошуку товарів з використанням різних критеріїв фільтрації.

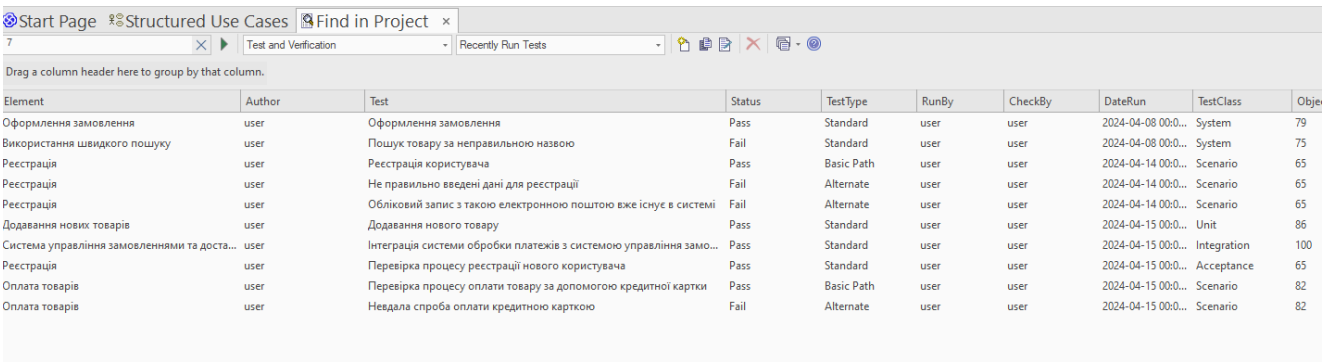


Рисунок 4.1 – Розроблені тест кейси в Enterprise Architect

У «Додатку А» наведено тестова документація, підготовлена засобами Enterprise Architect.

4.2 Трасування вимог до системи

Трасування вимог – це ключовий процес в розробці програмного забезпечення, який забезпечує відповідність реалізованої функціональності початковим задумам та потребам користувачів. Він дозволяє відстежувати вимоги протягом всього життєвого циклу розробки – від їх визначення до реалізації та тестування.

В контексті електронного магазину техно-товарів трасування вимог дозволяє чітко зрозуміти, як різні функціональні блоки системи взаємодіють між собою та задовольняють потреби користувачів. Наприклад, вимога "Замовлення товару" розкривається через низку прецедентів використання: вибір товару, додавання його до кошика, оформлення замовлення, вибір способу оплати та доставки. Кожен з цих прецедентів може бути деталізований за допомогою діаграм поведінки системи та описаний у вигляді сценаріїв використання.

Діаграма трасування вимог візуалізує зв'язки між різними вимогами, прецедентами та компонентами системи, використовуючи спеціальні типи зв'язків, такі як "Trace", "Realization" та "Abstraction".

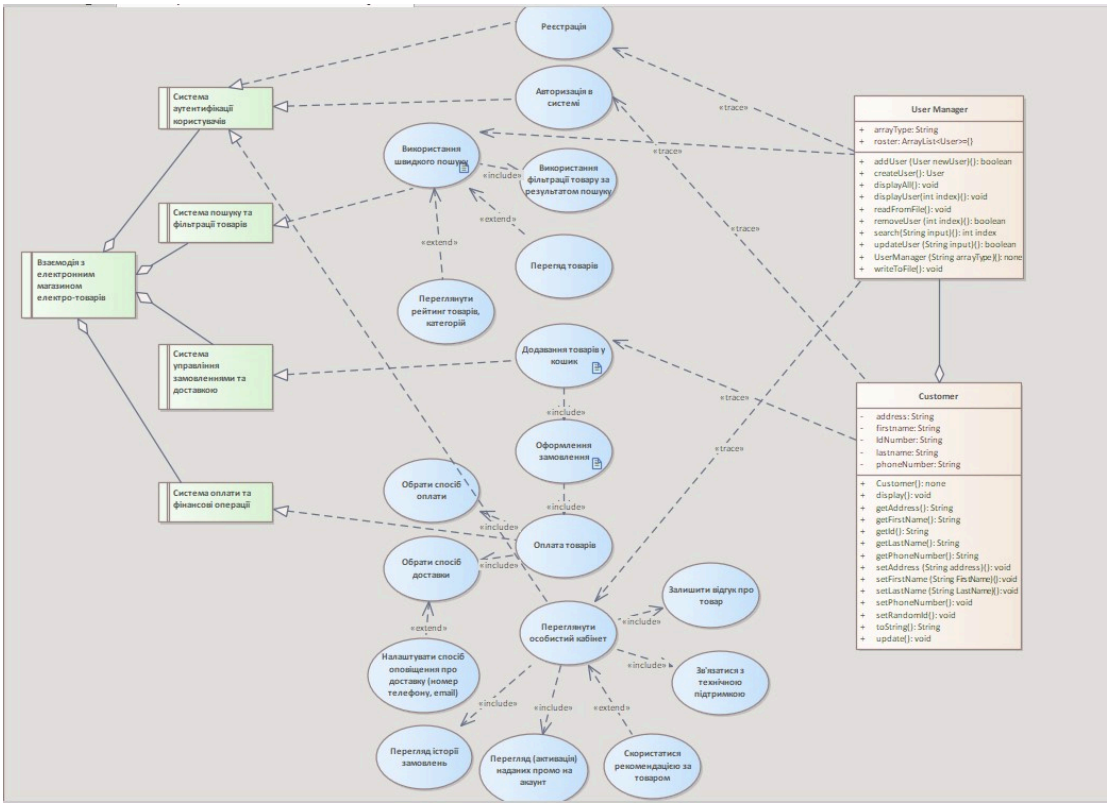
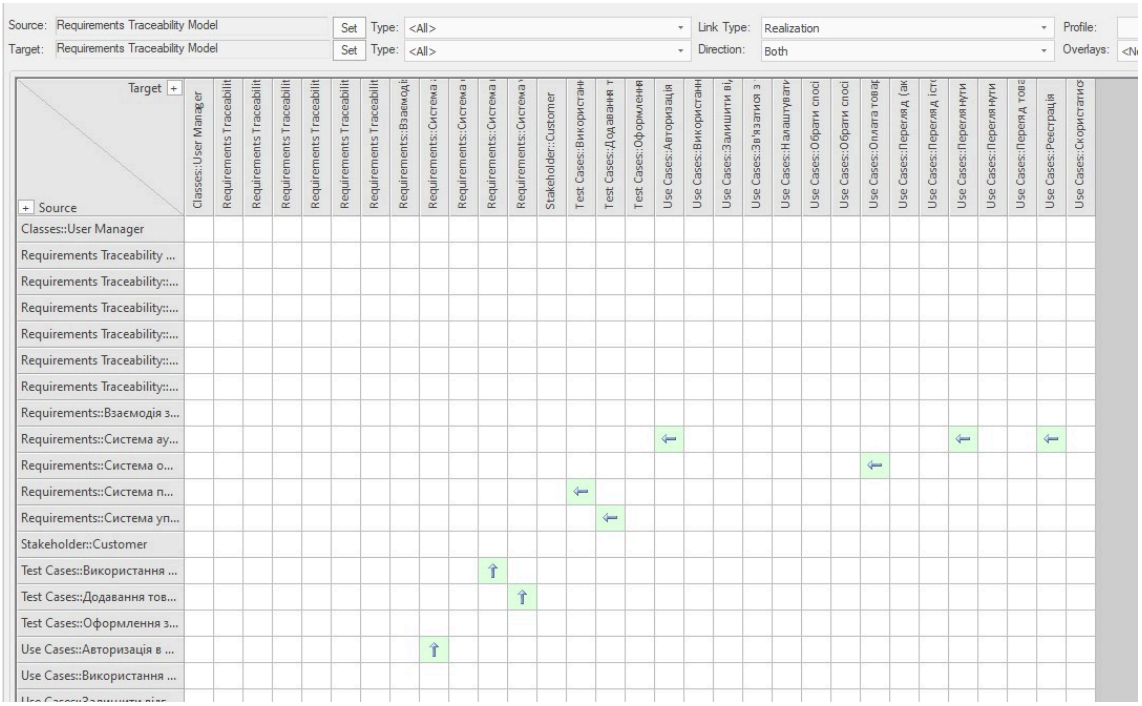
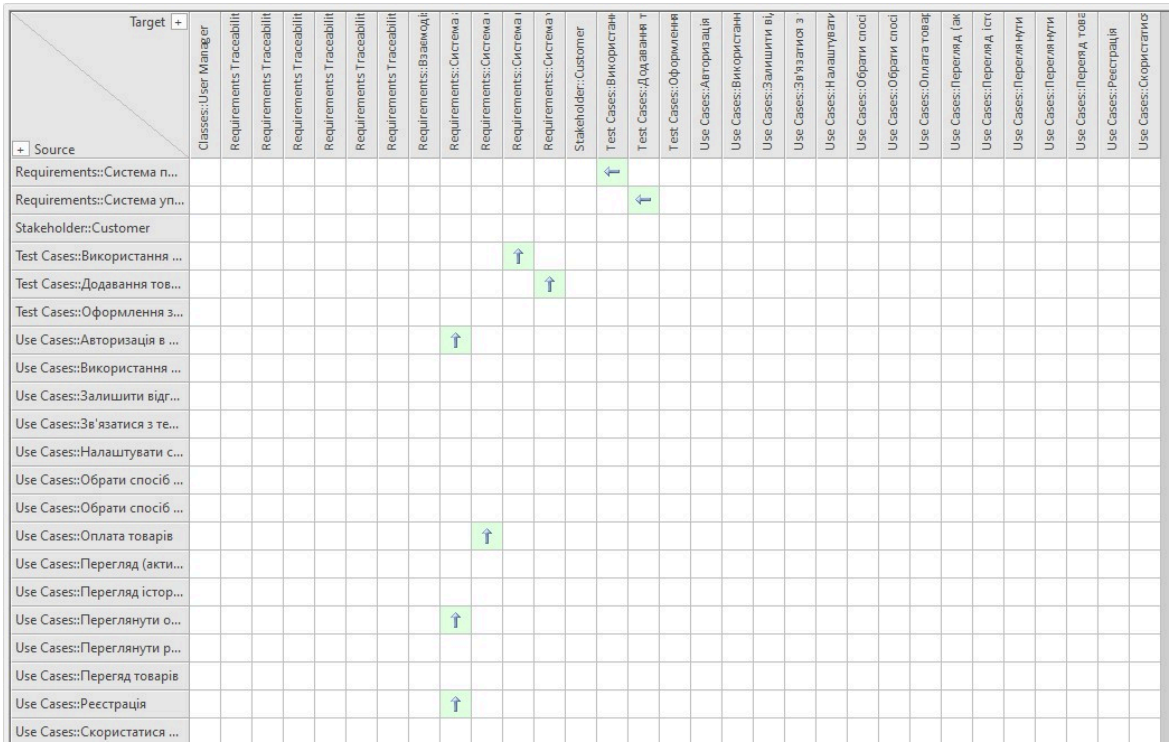


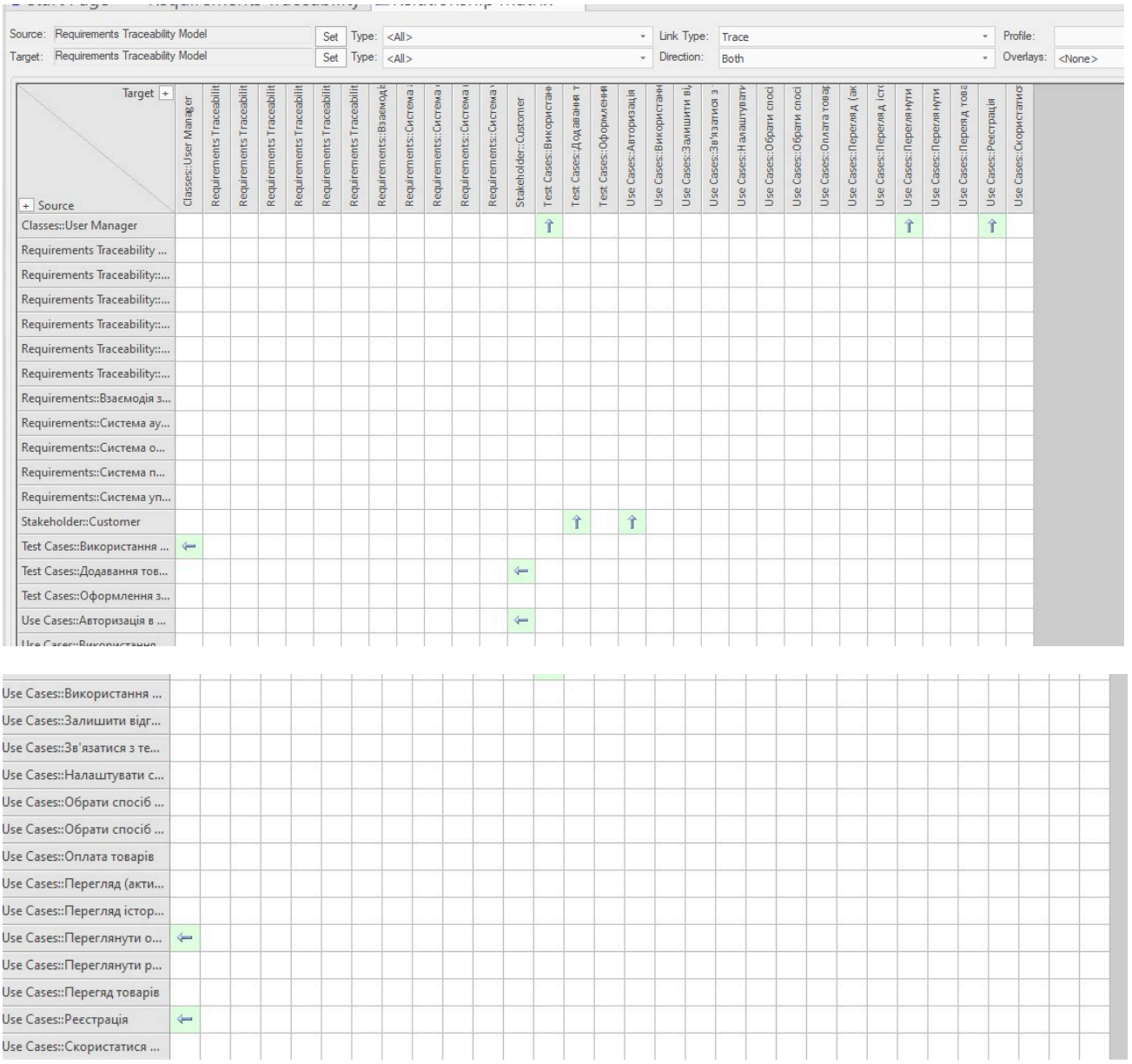
Рисунок 4.2 – Діаграма трасування в системі

Матриці зв'язків доповнюють діаграму трасування вимог, надаючи додаткову інформацію про відповідність між різними елементами системи. Вони дозволяють наочно побачити, які тестові сценарії покривають ті чи інші вимоги, які елементи користувацького інтерфейсу задіяні в реалізації конкретних прецедентів, та забезпечити повне тестування всієї функціональності магазину (див. рис. 4.3-4.6).





Рисунки 4.3-4.4 – Матриця взаємозв’язків в системі «Realization»



Рисунки 4.5-4.6 – Матриця взаємозв’язків в системі «Trace»

ВИСНОВКИ

Кваліфікаційний бакалаврський проект "Проектування електронного магазину техно-товарів" завершився успішною розробкою комплексної інформаційної системи, що відповідає сучасним вимогам ринку електронної комерції та здатна задовольнити зростаючі потреби українських споживачів. В ході роботи було виконано низку важливих етапів, охоплюючи аналіз, проектування, реалізацію та тестування системи.

Спочатку було проведено детальне дослідження ринку електронних магазинів техно-товарів в Україні. Проаналізувавши досвід провідних магазинів, таких як Rozetka, Moyo.ua, Citrus.ua та Foxtrot.com.ua, визначено основні тенденції, потреби цільової аудиторії та конкурентні переваги успішних платформ. Інформація лягла в основу формування ключових вимог до проектованої системи.

Наступним етапом стало визначення цільової аудиторії магазину та її потреб. Враховуючи специфіку ринку техно-товарів, було виділено сегменти користувачів з різним рівнем досвіду в онлайн-шопінгу, а також з різними потребами та очікуваннями від електронного магазину. На основі отриманих даних сформульовано перелік функціональних та нефункціональних вимог до системи, включаючи можливості перегляду каталогу товарів, пошуку та фільтрації, додавання товарів до кошика, оформлення замовлення, вибору способу оплати та доставки, відстеження статусу замовлення, управління особистим кабінетом, системи відгуків та рейтингу, а також адміністрування магазину. Нефункціональні вимоги охоплювали такі аспекти, як продуктивність, надійність, безпека, масштабованість та зручність використання системи.

Для візуалізації основних процесів та об'єктів системи, взаємозв'язків між ними, а також вхідних та вихідних даних була розроблена інформаційна модель системи за допомогою мови моделювання ОРМ. Додатково, взаємодія користувачів з системою була змодельована за допомогою діаграм прецедентів

(Use Case Diagram).

Важливим етапом стало проектування бази даних для зберігання даних електронного магазину. За допомогою MySQL Workbench було створено ER-діаграму бази даних, що відображає структуру таблиць та зв'язки між ними, а також згенеровано SQL-скрипт для її створення.

Бекенд частина системи була реалізована на мові програмування Java з використанням фреймворку Spring Boot, що знач но спростило процес розробки та розгортання додатку та надало готові рішення для роботи з базою даних, безпекою, веб-сервісами та іншими важливими аспектами. Фронтенд частина системи була реалізована за допомогою JavaScript-фреймворку Vue.js, що дозволив створити динамічний та зручний інтерфейс з використанням компонентного підходу. Для пришвидшення розробки та забезпечення візуальної консистентності було задіяно фреймворк Bootstrap.

Наступним етапом стала розробка тестової документації та проведення тестування системи. Для перевірки коректності роботи системи та її відповідності вимогам було створено різні типи тестів: модульні, інтеграційні, системні, приймальні та тести-сценарії. Для демонстрації відповідності реалізованої функціональності початковим вимогам було здійснено трасування вимог до системи, використовуючи діаграму трасування вимог та матриці зв'язків.

В результаті виконання проекту було створено прототип електронного магазину техно-товарів, який може бути використаний як база для створення реального магазину з урахуванням конкретних потреб бізнесу. Проект продемонстрував ефективне застосування сучасних технологій та методологій розробки, що дозволяє створити конкурентоспроможний продукт на ринку електронної комерції.

Напрямки подальшого розвитку проекту включають розширення функціоналу магазину (наприклад, впровадження системи персональних рекомендацій та розширеної аналітики продажів), інтеграцію з додатковими

сервісами (соціальні мережі, платіжні системи), впровадження нових технологій (штучний інтелект, машинне навчання) та покращення користувацького досвіду на основі аналізу поведінки користувачів.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Електронна комерція в Україні: тенденції та перспективи розвитку. Міністерство цифрової трансформації України. URL: https://thedigital.gov.ua/ (дата звернення: 10.01.2024).
2. Ринок електроніки та гаджетів в Україні: аналіз та прогнози. GfK Ukraine. URL: https://www.gfk.com/uk/ (дата звернення: 11.01.2024).
3. UX/UI дизайн для електронних магазинів: кращі практики. UX Collective. URL: https://uxdesign.cc/ (дата звернення: 13.01.2024).
4. SEO оптимізація для електронних магазинів: повний гайд. Search Engine Journal. URL: https://www.searchenginejournal.com/ (дата звернення: 14.01.2024).
5. Безпека електронних платежів: стандарти та технології. PCI Security Standards Council. URL: https://www.pcisecuritystandards.org/ (дата звернення: 21.01.2024).
6. Закон України "Про захист персональних даних". Верховна Рада України. URL: https://zakon.rada.gov.ua/laws/main/2297-17 (дата звернення: 26.01.2024).
7. Agile Software Development: Principles, Patterns, and Practices. Robert C. Martin. URL: https://www.amazon.com/Agile-Software-Development-Principles-Practices/dp/0135974445 (дата звернення: 26.01.2024).
8. The MVC Pattern - Model View Controller Architecture and Frameworks Explained. freeCodeCamp. URL: https://www.freecodecamp.org/news/the-model-view-controller-pattern-mvc-architecture-and-frameworks-explained/ (дата звернення: 26.01.2024).
9. MySQL Documentation. MySQL. URL: https://dev.mysql.com/doc/ (дата звернення: 26.01.2024).

10. Java Oracle. URL: https://docs.oracle.com/javase/tutorial/ (дата звернення: 27.01.2024).
11. Spring Boot Documentation. Spring.io. URL: https://spring.io/projects/spring-boot (дата звернення: 27.01.2024).
12. Vue.js Vue.js. URL: https://vuejs.org/ (дата звернення: 27.01.2024).
13. Bootstrap Getbootstrap.com. URL: https://getbootstrap.com/ (дата звернення: 28.01.2024).
14. HTML Tutorial. [Електронний ресурс] // W3Schools. – URL: https://www.w3schools.com/html/ (дата звернення: 29.01.2024).
15. CSS Tutorial. [Електронний ресурс] // W3Schools. – URL: https://www.w3schools.com/css/ (дата звернення: 29.01.2024).
16. JavaScript [Електронний ресурс] // W3Schools. – URL: https://www.w3schools.com/js/ (дата звернення: 29.01.2024).
17. Webpack [Електронний ресурс] // Webpack.js.org. – URL: https://webpack.js.org/ (дата звернення: 30.01.2024).
18. Figma [Електронний ресурс] // Figma.com. – URL: https://help.figma.com/ (дата звернення: 30.01.2024).
19. Enterprise Architect User Guide. [Електронний ресурс] // Sparx Systems. – URL: https://sparxsystems.com/products/ea/user-guide.html (дата звернення: 01.02.2024).
20. Object-Process Methodology (OPM). [Електронний ресурс] // OPCAT. – URL: http://www.opcat.com/opm.htm (дата звернення: 02.02.2024).
21. Тестування програмного забезпечення: види, методи, інструменти. [Електронний ресурс] // Software Testing Help. – URL: https://www.softwaretestinghelp.com/ (дата звернення: 04.02.2024).
22. E-commerce в Україні 2023: головні тренди та прогнози розвитку. [Електронний ресурс] // AIN.UA. – URL: https://ain.ua/2023/02/17/e-commerce-v-ukrayini-2023-golovni-trendy-ta-prognozy-rozvytku/ (дата звернення: 04.02.2024).

23. Як створити успішний електронний магазин: практичні поради. [Електронний ресурс] // Shopify. – URL: https://www.shopify.com/ (дата звернення: 08.03.2024).
24. Безпека веб-додатків: захист від загроз. [Електронний ресурс] // OWASP. – URL: https://owasp.org/ (дата звернення: 08.03.2024).
25. Microservices Architecture: A Complete Guide. [Електронний ресурс] // DZone. – URL: https://dzone.com/articles/microservices-architecture-a-complete-guide (дата звернення: 22.03.2024).
26. JavaScript Promises: An Introduction. [Електронний ресурс] // freeCodeCamp. – URL: https://www.freecodecamp.org/news/javascript-promises-an-introduction/ (дата звернення: 22.03.2024).
27. Best Practices for REST API Design. [Електронний ресурс] // RESTful API. – URL: https://restfulapi.net/ (дата звернення: 27.04.2024).
28. Accessibility in Web Design: A Beginner's Guide. [Електронний ресурс] // A11Y Project. – URL: https://a11yproject.com/ (дата звернення: 15.05.2024).
29. Performance Optimization for Web Applications. [Електронний ресурс] // Google Developers. – URL: https://developers.google.com/web/fundamentals/performance/ (дата звернення: 18.05.2024).
30. The Complete Guide to User Story Mapping. [Електронний ресурс] // Jeff Patton. – URL: https://jpattonassociates.com/user-story-mapping/ (дата звернення: 31.05.2024).
31. Foxtrot [Електронний ресурс] – Режим доступу до ресурсу: https://www.foxtrot.com.ua/ (дата звернення: 18.01.2024).
32. Rozetka [Електронний ресурс] – Режим доступу до ресурсу: https://rozetka.com.ua/ua/ (дата звернення: 18.01.2024).
33. MOYO [Електронний ресурс] – Режим доступу до ресурсу: https://www.moyo.ua/ua/ (дата звернення: 18.01.2024).

34. Цитрус [Електронний ресурс] – Режим доступу до ресурсу: https://www.ctrs.com.ua/ (дата звернення: 18.01.2024).

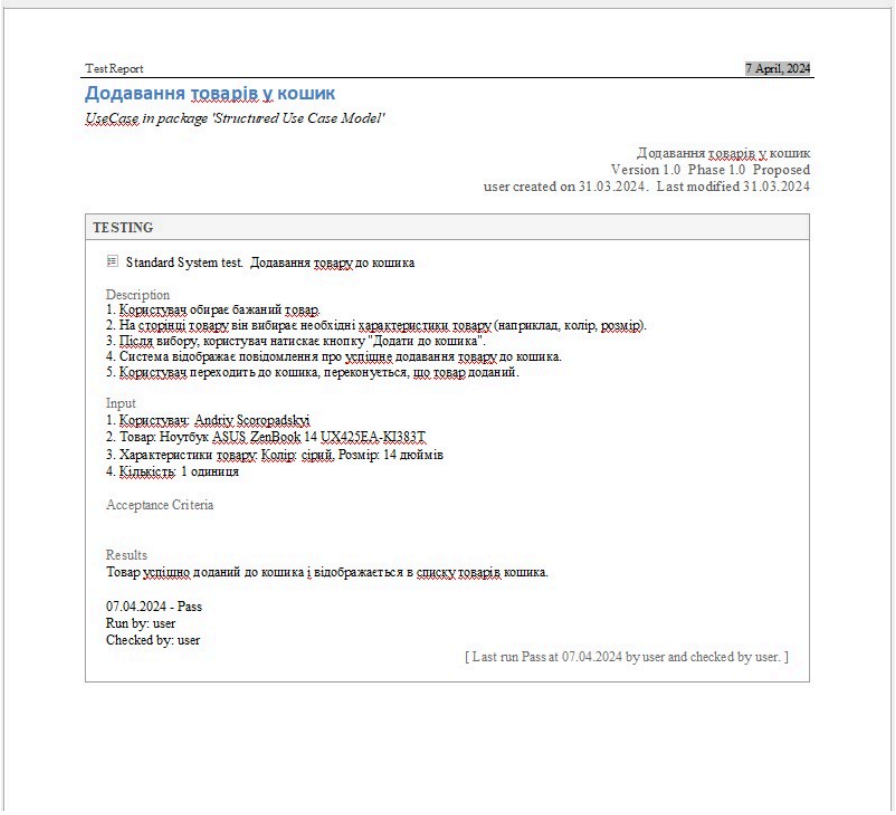
ДОДАТКИ

ДОДАТОК А

Тестова документація, підготовлена засобами Enterprise Architect

А.1 Системні тести

Розроблені тестові звіти (Test Report):



Test Report 7 April, 2024

Додавання товарів у кошик

UseCase in package 'Structured Use Case Model'

Додавання товарів у кошик
Version 1.0 Phase 1.0 Proposed
user created on 31.03.2024. Last modified 31.03.2024

TESTING

Standard System test. Додавання товару до кошика

Description
1. Користувач обирає бажаний товар.
2. На сторінці товару він вибирає необхідні характеристики товару (наприклад, колір, розмір).
3. Після вибору, користувач натискає кнопку "Додати до кошика".
4. Система відображає повідомлення про успішне додавання товару до кошика.
5. Користувач переходить до кошика, переконується, що товар доданий.

Input
1. Користувач: Andriy Scoropadskyi
2. Товар: Ноутбук ASUS ZenBook 14 UX425EA-KI383T
3. Характеристики товару: Колір: сірий, Розмір: 14 дюймів
4. Кількість: 1 одиниця

Acceptance Criteria

Results
Товар успішно доданий до кошика і відображається в списку товарів кошика.

07.04.2024 - Pass
Run by: user
Checked by: user

[Last run Pass at 07.04.2024 by user and checked by user.]

Рис. А.1.1 – Системний тест-кейс «Додавання товарів у кошик»

Test Report 8 April, 2024

Оформлення замовлення

UseCase in package 'Structured Use Case Model'

Оформлення замовлення
Version 1.0 Phase 1.0 Proposed
user created on 31.03.2024. Last modified 31.03.2024

TESTING

Standard System test. Оформлення замовлення

Description
1. Користувач переходить до сторінки кошика і натискає кнопку "Оформити замовлення".
2. Він обирає метод доставки та вводить адресу доставки.
3. Користувач вибирає метод оплати та вводить відповідні дані (наприклад, реквізити банківської карти).
4. Перевіряє замовлення та натискає кнопку "Підтвердити замовлення".
5. Система відображає підтвердження замовлення з деталями.

Input
1. Адреса доставки: вул. Івасюка, 10, м. Київ, Україна
2. Метод оплати: Банківська карта Monobank
3. Номер банківської карти: 1234 5678 9012 3456
4. Термін дії: 12/24
5. CVV-код: 123

Acceptance Criteria

Results
Замовлення успішно оформлене і відображається підтвердження з деталями замовлення.

08.04.2024 - Pass
Run by: user
Checked by: user

[Last run Pass at 08.04.2024 by user and checked by user.]

Рис. А.1.2 – Системний тест-кейс «Оформлення замовлення»

Test Report 8 April, 2024

Використання швидкого пошуку

UseCase in package 'Structured Use Case Model'

Використання швидкого пошуку
Version 1.0 Phase 1.0 Proposed
user created on 31.03.2024. Last modified 31.03.2024

TESTING

Standard System test. Пошук товару за неправильною назвою

Description
1.Користувач вводить неправильну назву товару в поле пошуку.
2.Натискає кнопку "Пошук".

Input
1. Користувач вводить, "Samsang".

Acceptance Criteria

Results
Система відображає повідомлення про відсутність результатів для введеної назви товару. Немає результатів, пов'язаних з неправильною назвою.

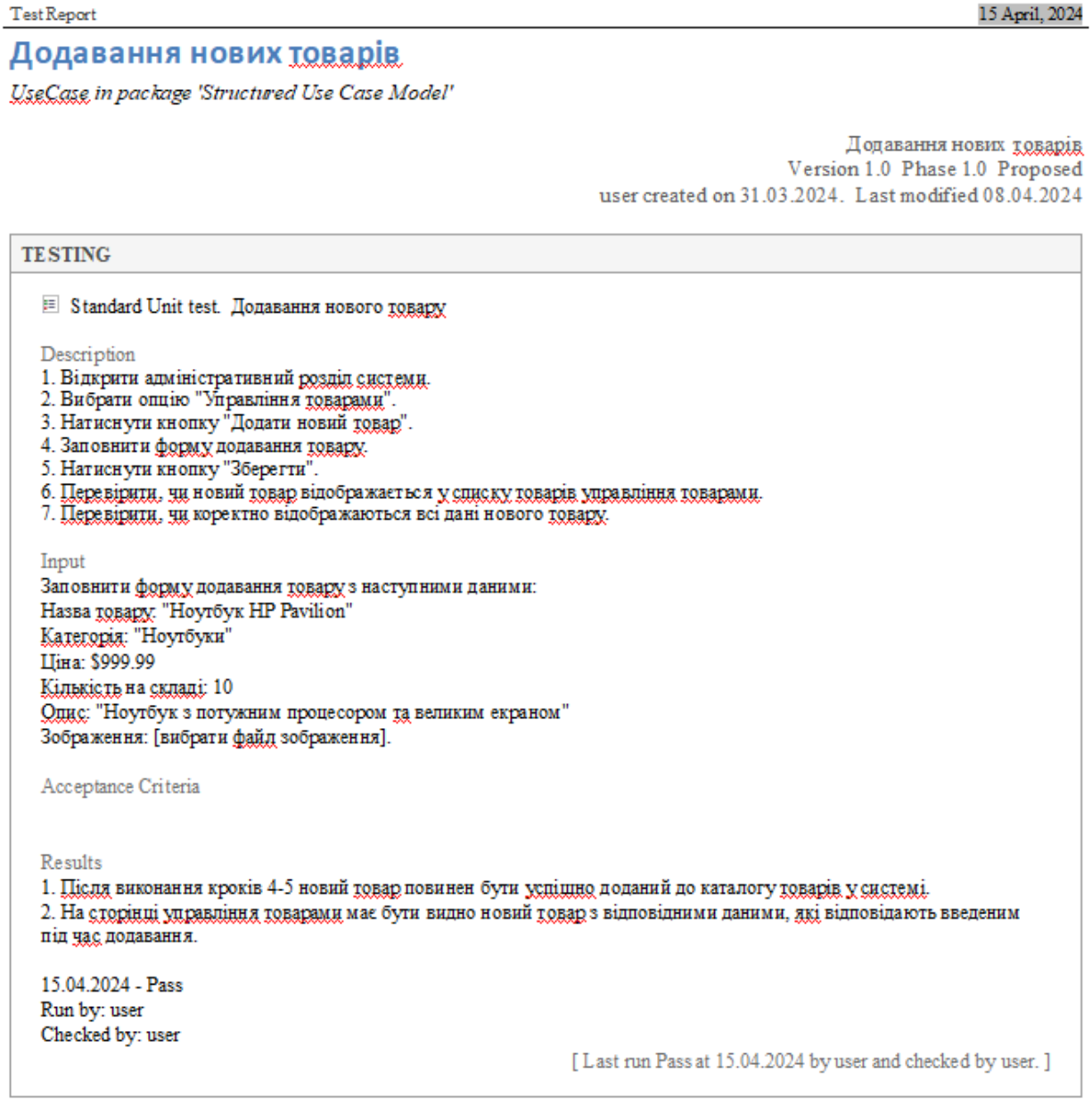
08.04.2024 - Fail
Run by: user
Checked by: user

[Last run Fail at 08.04.2024 by user and checked by user.]

Рис. А.1.3 – Системний тест-кейс «Вхід в систему»

А.2 Модульні тести

Розроблені тестові звіти (Test Report):



Test Report 15 April, 2024

Додавання нових товарів

UseCase in package 'Structured Use Case Model'

Додавання нових товарів
Version 1.0 Phase 1.0 Proposed
user created on 31.03.2024. Last modified 08.04.2024

TESTING

Standard Unit test. Додавання нового товару

Description
1. Відкрити адміністративний розділ системи.
2. Вибрати опцію "Управління товарами".
3. Натиснути кнопку "Додати новий товар".
4. Заповнити форму додавання товару.
5. Натиснути кнопку "Зберегти".
6. Перевірити, чи новий товар відображається у списку товарів управління товарами.
7. Перевірити, чи коректно відображаються всі дані нового товару.

Input
Заповнити форму додавання товару з наступними даними:
Назва товару: "Ноутбук HP Pavilion"
Категорія: "Ноутбуки"
Ціна: $999.99
Кількість на складі: 10
Опис: "Ноутбук з потужним процесором та великим екраном"
Зображення: [вибрати файл зображення].

Acceptance Criteria

Results
1. Після виконання кроків 4-5 новий товар повинен бути успішно доданий до каталогу товарів у системі.
2. На сторінці управління товарами має бути видно новий товар з відповідними даними, які відповідають введеним під час додавання.

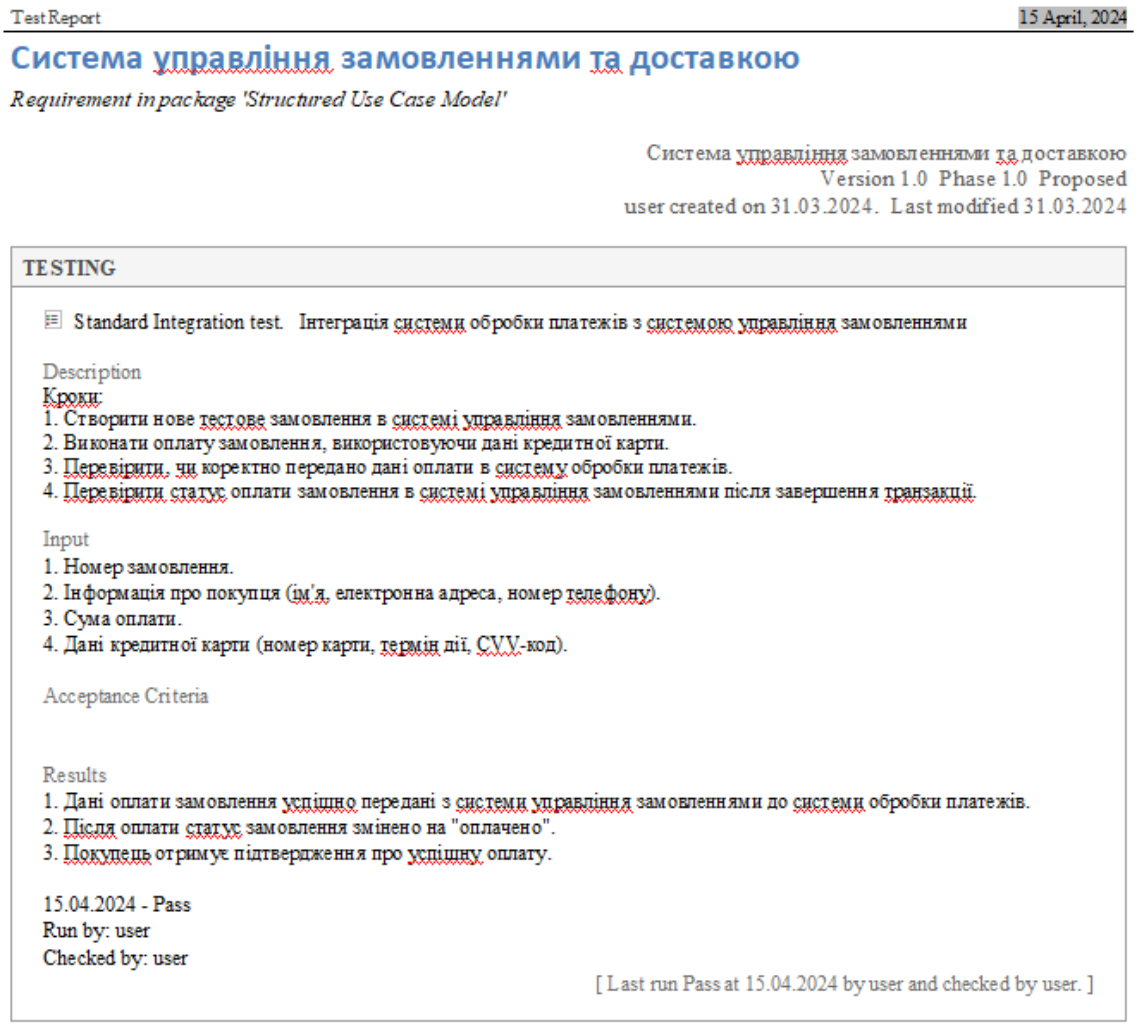
15.04.2024 - Pass
Run by: user
Checked by: user

[Last run Pass at 15.04.2024 by user and checked by user.]

Рис. А.2.1 – Модульний тест-кейс «Додавання нового товару»

А.3 Інтеграційні тести

Розроблені тестові звіти (Test Report):



Test Report 15 April, 2024

Система управління замовленнями та доставкою

Requirement in package 'Structured Use Case Model'

Система управління замовленнями та доставкою
Version 1.0 Phase 1.0 Proposed
user created on 31.03.2024. Last modified 31.03.2024

TESTING

Standard Integration test. Інтеграція системи обробки платежів з системою управління замовленнями

Description
Кроки:
1. Створити нове тестове замовлення в системі управління замовленнями.
2. Виконати оплату замовлення, використовуючи дані кредитної карти.
3. Перевірити, чи коректно передано дані оплати в систему обробки платежів.
4. Перевірити статус оплати замовлення в системі управління замовленнями після завершення транзакції.

Input
1. Номер замовлення.
2. Інформація про покупця (ім'я, електронна адреса, номер телефону).
3. Сума оплати.
4. Дані кредитної карти (номер карти, термін дії, CVV-код).

Acceptance Criteria

Results
1. Дані оплати замовлення успішно передані з системи управління замовленнями до системи обробки платежів.
2. Після оплати статус замовлення змінено на "оплачено".
3. Покупець отримує підтвердження про успішну оплату.

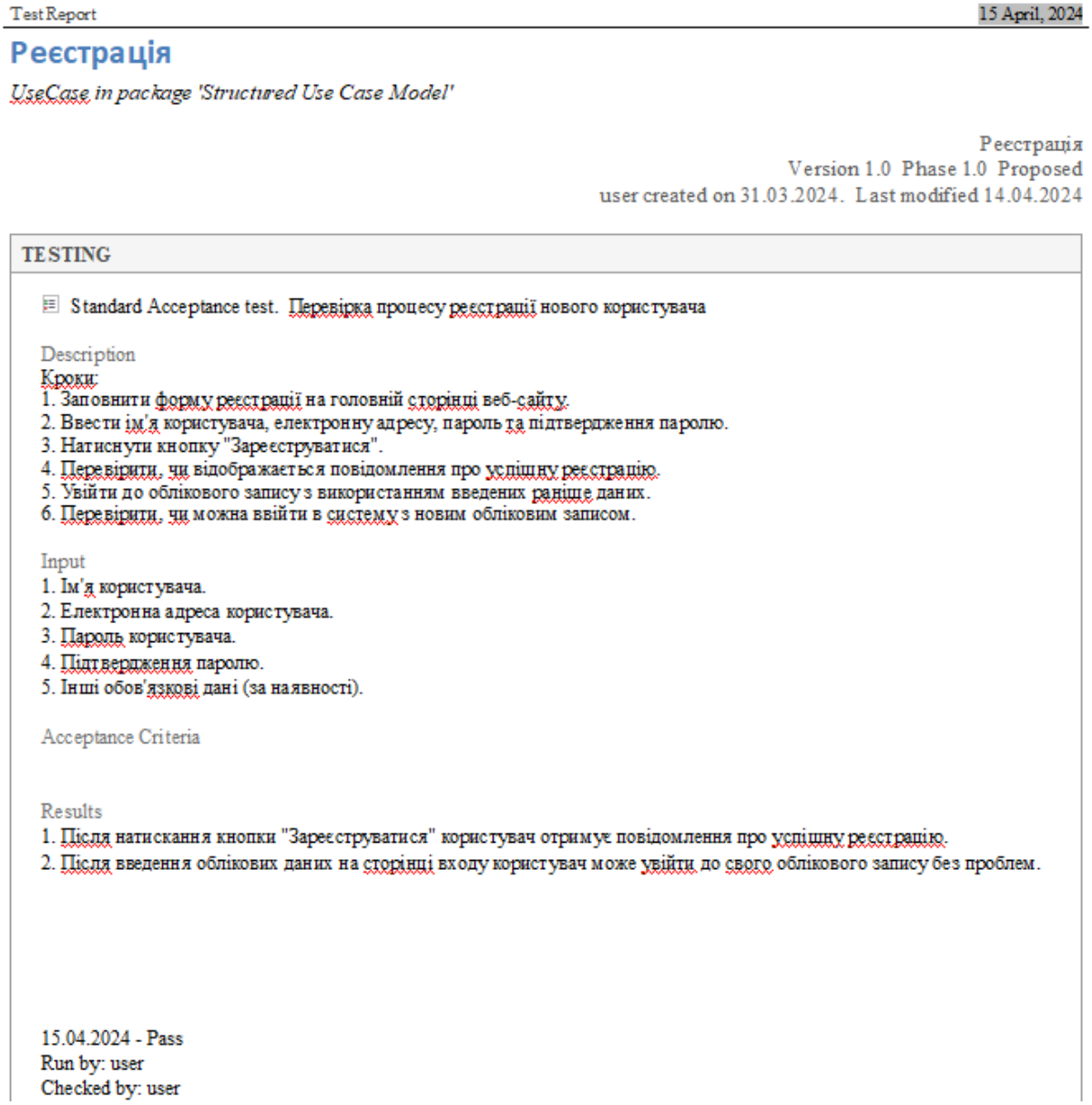
15.04.2024 - Pass
Run by: user
Checked by: user

[Last run Pass at 15.04.2024 by user and checked by user.]

Рис. А.3.1 – Інтеграційні тест-кейс «Інтеграція системи обробки платежів з системою управління замовленнями»

А.4 Приймальні тести

Розроблені тестові звіти (Test Report):



Test Report 15 April, 2024

Реєстрація

UseCase in package 'Structured Use Case Model'

Реєстрація
Version 1.0 Phase 1.0 Proposed
user created on 31.03.2024. Last modified 14.04.2024

TESTING

Standard Acceptance test. Перевірка процесу реєстрації нового користувача

Description
Кроки:
1. Заповнити форму реєстрації на головній сторінці веб-сайту.
2. Ввести ім'я користувача, електронну адресу, пароль та підтвердження паролю.
3. Натиснути кнопку "Зареєструватися".
4. Перевірити, чи відображається повідомлення про успішну реєстрацію.
5. Увійти до облікового запису з використанням введених раніше даних.
6. Перевірити, чи можна ввійти в систему з новим обліковим записом.

Input
1. Ім'я користувача.
2. Електронна адреса користувача.
3. Пароль користувача.
4. Підтвердження паролю.
5. Інші обов'язкові дані (за наявності).

Acceptance Criteria

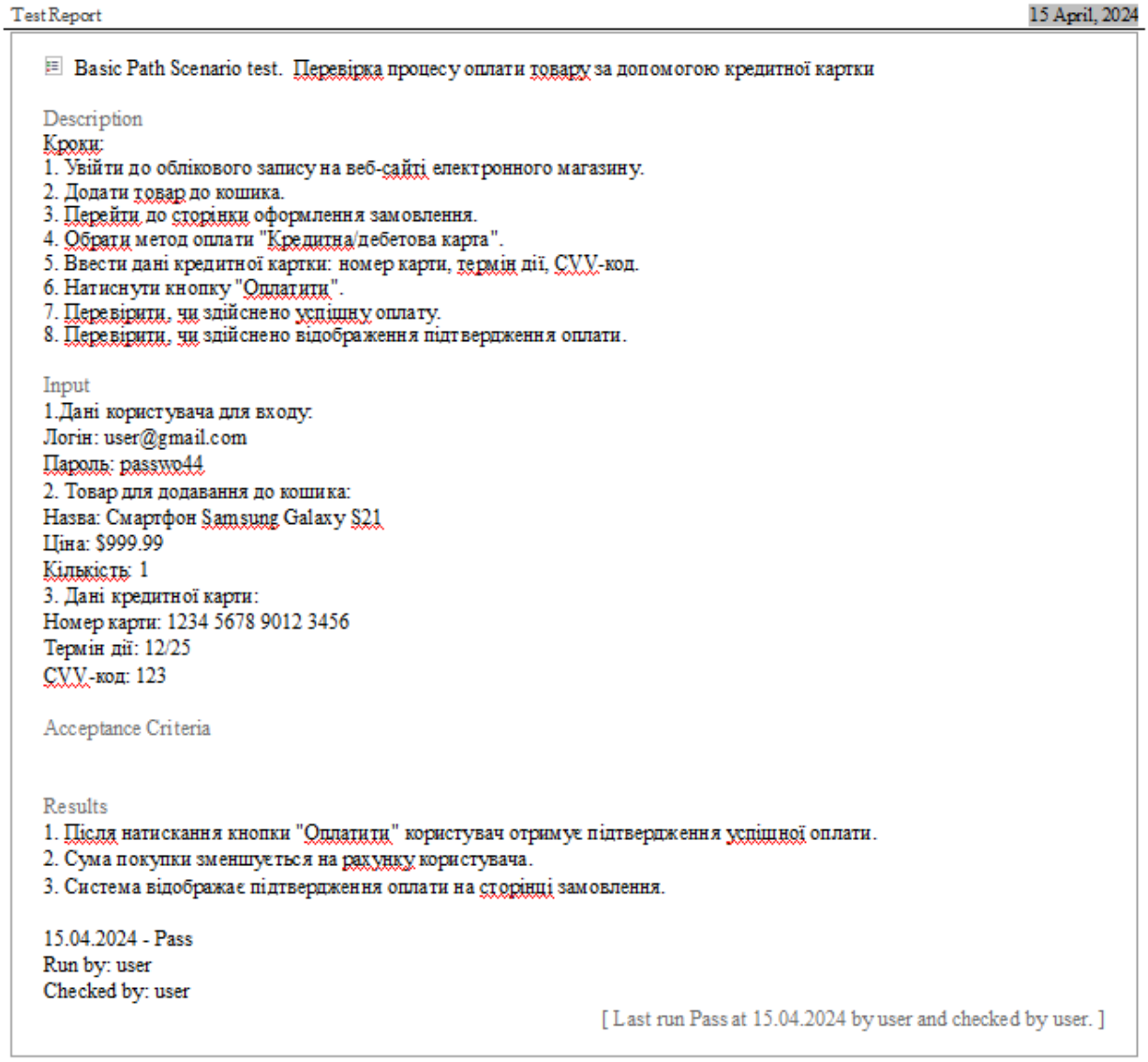
Results
1. Після натискання кнопки "Зареєструватися" користувач отримує повідомлення про успішну реєстрацію.
2. Після введення облікових даних на сторінці входу користувач може увійти до свого облікового запису без проблем.

15.04.2024 - Pass
Run by: user
Checked by: user

Рис. А.4.1 – Приймальний тест-кейс «Перевірка процесу реєстрації нового користувача»

А.5 Тести-сценарії

Розроблені тестові звіти (Test Report):



Test Report 15 April, 2024

Basic Path Scenario test. Перевірка процесу оплати товару за допомогою кредитної картки

Description
Кроки:
1. Увійти до облікового запису на веб-сайті електронного магазину.
2. Додати товар до кошика.
3. Перейти до сторінки оформлення замовлення.
4. Обрати метод оплати "Кредитна/дебетова карта".
5. Ввести дані кредитної картки: номер карти, термін дії, CVV-код.
6. Натиснути кнопку "Оплатити".
7. Перевірити, чи здійснено успішну оплату.
8. Перевірити, чи здійснено відображення підтвердження оплати.

Input
1.Дані користувача для входу:
Логін: user@gmail.com
Пароль: passwo44
2. Товар для додавання до кошика:
Назва: Смартфон Samsung Galaxy S21
Ціна: $999.99
Кількість: 1
3. Дані кредитної карти:
Номер карти: 1234 5678 9012 3456
Термін дії: 12/25
CVV-код: 123

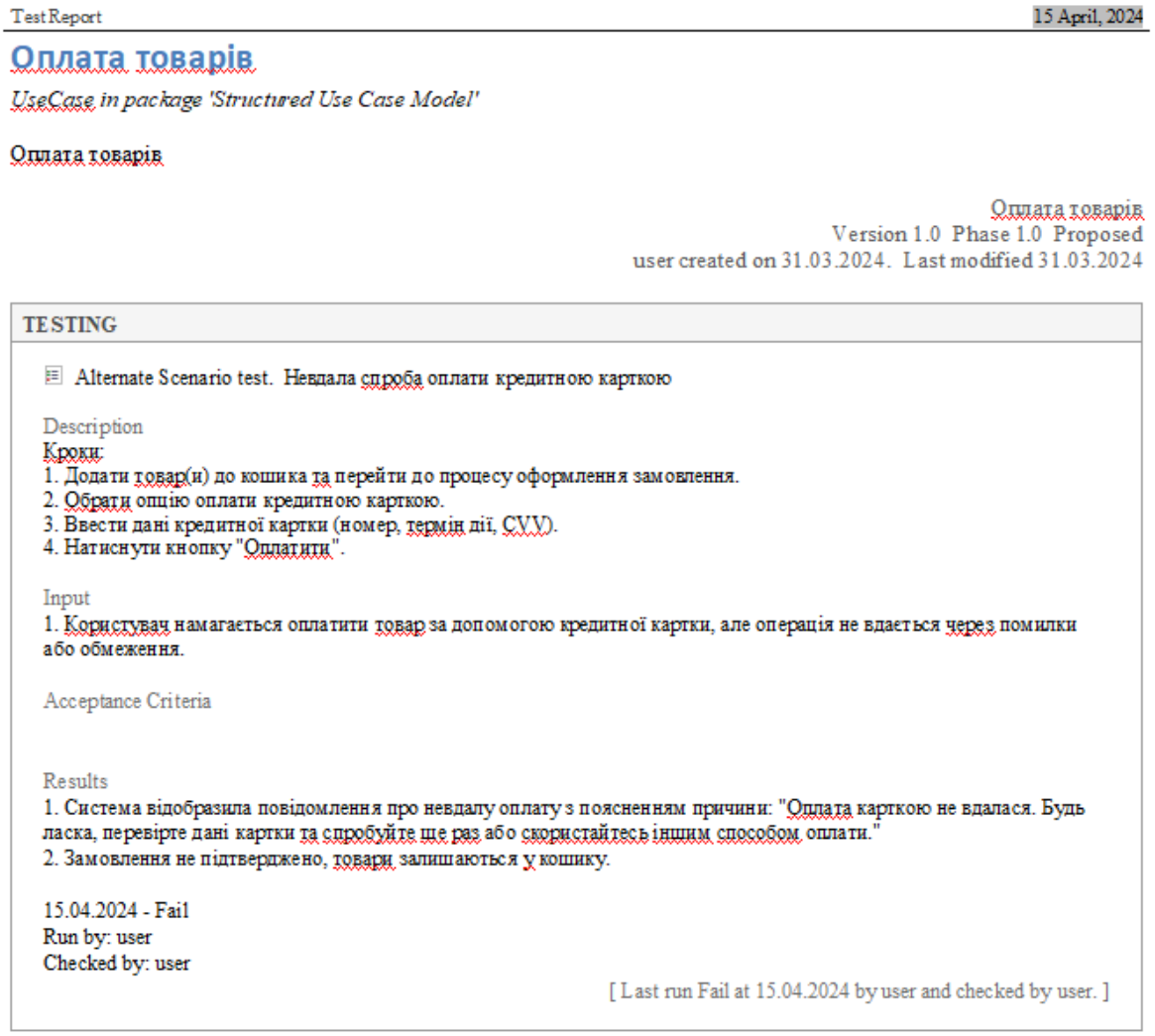
Acceptance Criteria

Results
1. Після натискання кнопки "Оплатити" користувач отримує підтвердження успішної оплати.
2. Сума покупки зменшується на рахунку користувача.
3. Система відображає підтвердження оплати на сторінці замовлення.

15.04.2024 - Pass
Run by: user
Checked by: user

[Last run Pass at 15.04.2024 by user and checked by user.]

Рис. А.5.1 – Тести-сценарії «Перевірка процесу оплати товару за допомогою кредитної картки»



Test Report 15 April, 2024

Оплата товарів

UseCase in package 'Structured Use Case Model'

Оплата товарів

Оплата товарів
Version 1.0 Phase 1.0 Proposed
user created on 31.03.2024. Last modified 31.03.2024

TESTING

Alternate Scenario test. Невдала спроба оплати кредитною карткою

Description
Кроки:
1. Додати товар(и) до кошика та перейти до процесу оформлення замовлення.
2. Обрати опцію оплати кредитною карткою.
3. Ввести дані кредитної картки (номер, термін дії, CVV).
4. Натиснути кнопку "Оплатити".

Input
1. Користувач намагається оплатити товар за допомогою кредитної картки, але операція не вдається через помилки або обмеження.

Acceptance Criteria

Results
1. Система відобразила повідомлення про невдалу оплату з поясненням причини: "Оплата карткою не вдалася. Будь ласка, перевірте дані картки та спробуйте ще раз або скористайтесь іншим способом оплати."
2. Замовлення не підтверджено, товари залишаються у кошику.

15.04.2024 - Fail
Run by: user
Checked by: user

[Last run Fail at 15.04.2024 by user and checked by user.]

Рис. А.5.2 – Тести-сценарії (Альтернативний сценарій) «Невдала спроба оплати кредитною карткою (негативний випадок)»

ДОДАТОК Б

Фрагменти коду, що було створено для реалізації бекенду магазину (контролери)

```
// CategoryController
@RestController
@RequestMapping("/category")
@CrossOrigin("http://localhost:8001")

public class CategoryController {

	@Autowired
	private CategoryService categoryService;

	@GetMapping("/")
	@CrossOrigin("http://localhost:8001")
   public ResponseEntity<List<Category>> getCategories() {
      List<Category> body = categoryService.listCategories();
      return new ResponseEntity<List<Category>>(body, HttpStatus.OK);
   }

	@PostMapping("/create")
	@CrossOrigin("http://localhost:8001")
	public ResponseEntity<ApiResponse> createCategory(@Valid @RequestBody Category category) {
		if (Helper.notNull(categoryService.readCategory(category.getCategoryName()))) {
			return new ResponseEntity<ApiResponse>(new ApiResponse(false, "category already exists"),
HttpStatus.CONFLICT);
		}
		categoryService.createCategory(category);
		return new ResponseEntity<ApiResponse>(new ApiResponse(true, "created the category"), HttpStatus.CREATED);
	}

	@PostMapping("/update/{categoryID}")
	@CrossOrigin("http://localhost:8001")
	public ResponseEntity<ApiResponse> updateCategory(@PathVariable("categoryID") Integer categoryID, @Valid @RequestBody
Category category) {
		// Check to see if the category exists.
		if (Helper.notNull(categoryService.readCategory(categoryID))) {
			// If the category exists then update it.

	categoryService.updateCategory(categoryID, category);
			return new ResponseEntity<ApiResponse>(new ApiResponse(true, "updated the category"), HttpStatus.OK);
		}

		// If the category doesn't exist then return a response of unsuccessful.
		return new ResponseEntity<ApiResponse>(new ApiResponse(false, "category does not exist"),
HttpStatus.NOT_FOUND);
	}
}
}
// ProductController
@RestController
@RequestMapping("/product")
@CrossOrigin("http://localhost:8001")
public class ProductController {
   @Autowired ProductService productService;
   @Autowired
   CategoryService categoryService;

   @GetMapping("/")
   @CrossOrigin("http://localhost:8001")
   public ResponseEntity<List<ProductDto>> getProducts() {
      List<ProductDto> body = productService.listProducts();
```

```
    return new ResponseEntity<List<ProductDto>>(body, HttpStatus.OK);
}
```

```
    @PostMapping("/add")
    @CrossOrigin("http://localhost:8001")
    public ResponseEntity<ApiResponse> addProduct(@RequestBody ProductDto productDto) {
        Optional<Category> optionalCategory = categoryService.readCategory(productDto.getCategoryId());
        if (!optionalCategory.isPresent()) {
            return new ResponseEntity<ApiResponse>(new ApiResponse(false, "category is invalid"), HttpStatus.CONFLICT);
        }
        Category category = optionalCategory.get();
        productService.addProduct(productDto, category);
        return new ResponseEntity<ApiResponse>(new ApiResponse(true, "Product has been added"), HttpStatus.CREATED);
    }

    @PostMapping("/update/{productID}")
    @CrossOrigin("http://localhost:8001")
    public ResponseEntity<ApiResponse> updateProduct(@PathVariable("productID") Integer productID, @RequestBody @Valid ProductDto
productDto) {

        Optional<Category> optionalCategory = categoryService.readCategory(productDto.getCategoryId());
        if (!optionalCategory.isPresent()) {
            return new ResponseEntity<ApiResponse>(new ApiResponse(false, "category is invalid"), HttpStatus.CONFLICT);
        }
        Category category = optionalCategory.get();
        productService.updateProduct(productID, productDto, category);
        return new ResponseEntity<ApiResponse>(new ApiResponse(true, "Product has been updated"), HttpStatus.OK);
    }
}
```

// UserController

```
@RequestMapping("user")
@CrossOrigin("http://localhost:8001")
//@CrossOrigin(origins = "*", allowedHeaders = "*")
@RestController
public class UserController {
    @Autowired
    UserRepository userRepository;
    @Autowired
    AuthenticationService authenticationService;
    @Autowired
    UserService userService;
    @GetMapping("/all")
    @CrossOrigin("http://localhost:8001")
    public List<User> findAllUser(@RequestParam("token") String token) throws AuthenticationFailException {
        authenticationService.authenticate(token);
        return userRepository.findAll();
    }
    @PostMapping("/signup")
    @CrossOrigin("http://localhost:8001")
    public ResponseDto Signup(@RequestBody SignupDto signupDto) throws CustomException {
        return userService.signUp(signupDto);
    }
    @PostMapping("/signIn")
    @CrossOrigin("http://localhost:8001")
    public SignInResponseDto Signup(@RequestBody SignInDto signInDto) throws CustomException {
        return userService.signIn(signInDto);
    }
    @PostMapping("/createUser")
    @CrossOrigin("http://localhost:8001")
    public ResponseDto updateUser(@RequestParam("token") String token, @RequestBody UserCreateDto userCreateDto)
            throws CustomException, AuthenticationFailException {
        authenticationService.authenticate(token);
        return userService.createUser(token, userCreateDto);
    }
}
```

// CartController

```
@RestController
@RequestMapping("/cart")
@CrossOrigin("http://localhost:8001")
public class CartController {
    @Autowired
    private CartService cartService;
    @Autowired
    private ProductService productService;
```

```
    @Autowired
    private AuthenticationService authenticationService;
    @PostMapping("/add")
    @CrossOrigin("http://localhost:8001")
    public ResponseEntity<ApiResponse> addToCart(@RequestBody AddToCartDto addToCartDto,
                                         @RequestParam("token") String token) throws AuthenticationFailException, ProductNotExistException {
        authenticationService.authenticate(token);
        User user = authenticationService.getUser(token);
        Product product = productService.getProductById(addToCartDto.getProductId());
        System.out.println("product to add"+  product.getName());
        cartService.addToCart(addToCartDto, product, user);
        return new ResponseEntity<ApiResponse>(new ApiResponse(true, "Added to cart"), HttpStatus.CREATED);
    }
    @GetMapping("/")
    @CrossOrigin("http://localhost:8001")
    public ResponseEntity<CartDto> getCartItems(@RequestParam("token") String token) throws AuthenticationFailException {
        authenticationService.authenticate(token);
        User user = authenticationService.getUser(token);
        CartDto cartDto = cartService.listCartItems(user);
        return new ResponseEntity<CartDto>(cartDto,HttpStatus.OK);
    }
    @PutMapping("/update/{cartItemId}")
    @CrossOrigin("http://localhost:8001")
    public ResponseEntity<ApiResponse> updateCartItem(@RequestBody @Valid AddToCartDto cartDto,
                                           @RequestParam("token") String token) throws AuthenticationFailException,ProductNotExistException {
        authenticationService.authenticate(token);
        User user = authenticationService.getUser(token);
        Product product = productService.getProductById(cartDto.getProductId());
        authenticationService.authenticate(token);
        return userService.createUser(token, userCreateDto);
    }
}
// CartController
@RestController
@RequestMapping("/cart")
@CrossOrigin("http://localhost:8001")
public class CartController {
    @Autowired
    private CartService cartService;
    @Autowired
    private ProductService productService;
    @Autowired
    private AuthenticationService authenticationService;
    @PostMapping("/add")
    @CrossOrigin("http://localhost:8001")
    public ResponseEntity<ApiResponse> addToCart(@RequestBody AddToCartDto addToCartDto,
                                         @RequestParam("token") String token) throws AuthenticationFailException, ProductNotExistException {
        authenticationService.authenticate(token);
        User user = authenticationService.getUser(token);
        Product product = productService.getProductById(addToCartDto.getProductId());
        System.out.println("product to add"+  product.getName());
        cartService.addToCart(addToCartDto, product, user);
        return new ResponseEntity<ApiResponse>(new ApiResponse(true, "Added to cart"), HttpStatus.CREATED);
    }
    @GetMapping("/")
    @CrossOrigin("http://localhost:8001")
    public ResponseEntity<CartDto> getCartItems(@RequestParam("token") String token) throws AuthenticationFailException {
        authenticationService.authenticate(token);
        User user = authenticationService.getUser(token);
        CartDto cartDto = cartService.listCartItems(user);
        return new ResponseEntity<CartDto>(cartDto,HttpStatus.OK);
    }
    @PutMapping("/update/{cartItemId}")
    @CrossOrigin("http://localhost:8001")
    public ResponseEntity<ApiResponse> updateCartItem(@RequestBody @Valid AddToCartDto cartDto,
                                           @RequestParam("token") String token) throws AuthenticationFailException,ProductNotExistException {
        authenticationService.authenticate(token);
        User user = authenticationService.getUser(token);
        Product product = productService.getProductById(cartDto.getProductId());

cartService.updateCartItem(cartDto, user,product);
```

```
        return new ResponseEntity<ApiResponse>(new ApiResponse(true, "Product has been updated"), HttpStatus.OK);
    }
    @DeleteMapping("/delete/{cartItemId}")
    @CrossOrigin("http://localhost:8001")
    public ResponseEntity<ApiResponse> deleteCartItem(@PathVariable("cartItemId") int itemID,@RequestParam("token") String token) throws
AuthenticationFailException, CartItemNotExistException {
        authenticationService.authenticate(token);
        int userId = authenticationService.getUser(token).getId();
        cartService.deleteCartItem(itemID, userId);
        return new ResponseEntity<ApiResponse>(new ApiResponse(true, "Item has been removed"), HttpStatus.OK);
    }
}
// OrderController
@RestController
@RequestMapping("/order")
@CrossOrigin("http://localhost:8001")
public class OrderController {
    @Autowired
    private OrderService orderService;
    @Autowired
    private AuthenticationService authenticationService;
    // place order after checkout
    @PostMapping("/add")
    @CrossOrigin("http://localhost:8001")
    public ResponseEntity<ApiResponse> placeOrder(@RequestParam("token") String token, @RequestParam("sessionId") String sessionId)
            throws AuthenticationFailException {
        // validate token
        authenticationService.authenticate(token);
        // retrieve user
        User user = authenticationService.getUser(token);
        // place the order
        orderService.placeOrder(user, sessionId);
        return new ResponseEntity<>(new ApiResponse(true, "Order has been placed"), HttpStatus.CREATED);
    }
    // get all orders
    @GetMapping("/")
    @CrossOrigin("http://localhost:8001")
    public ResponseEntity<List<Order>> getAllOrders(@RequestParam("token") String token) throws AuthenticationFailException {
        // validate token
        authenticationService.authenticate(token);
// retrieve user
        User user = authenticationService.getUser(token);
        // get orders
        List<Order> orderDtoList = orderService.listOrders(user);
        return new ResponseEntity<>(orderDtoList, HttpStatus.OK);
    }
    // get orderitems for an order
    @GetMapping("/{id}")
    @CrossOrigin("http://localhost:8001")
    public ResponseEntity<Object> getOrderById(@PathVariable("id") Integer id, @RequestParam("token") String token)
            throws AuthenticationFailException {
        // validate token
        authenticationService.authenticate(token);
        try {
            Order order = orderService.getOrder(id);
            return new ResponseEntity<>(order,HttpStatus.OK);
        }
        catch (OrderNotFoundException e) {
            return new ResponseEntity<>(e.getMessage(),HttpStatus.NOT_FOUND);
        }
    }
}
// WishListController
@RestController
@RequestMapping("/wishlist")
@CrossOrigin("http://localhost:8001")
public class WishListController {
        @Autowired
        private WishListService wishListService;
        @Autowired
        private AuthenticationService authenticationService;
```

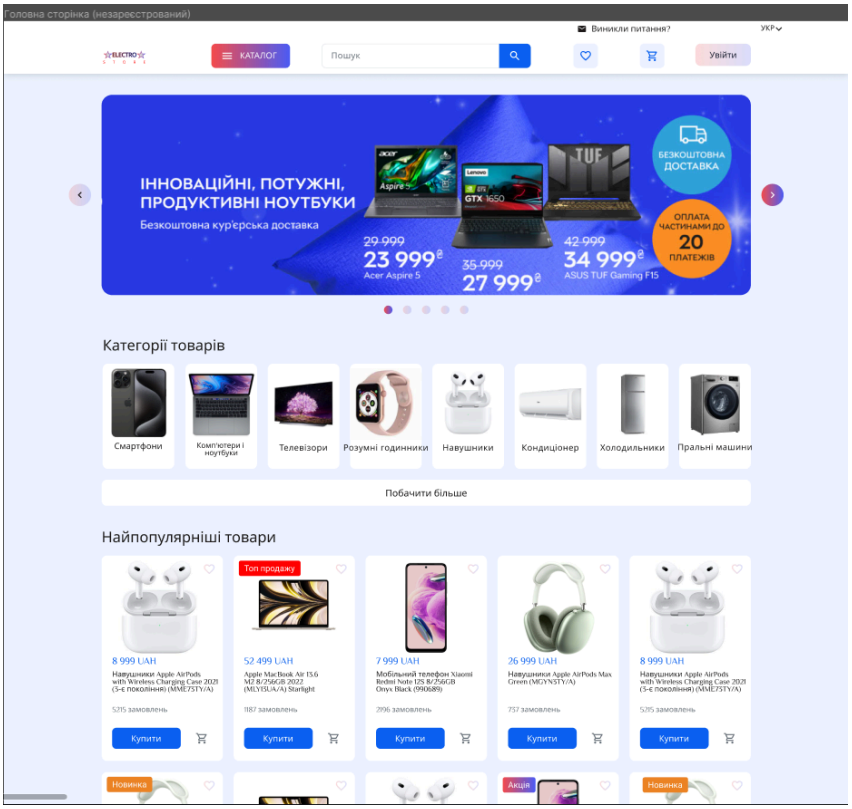
```
    @GetMapping("/{token}")
    @CrossOrigin("http://localhost:8001")
    public ResponseEntity<List<ProductDto>> getWishList(@PathVariable("token") String token) {
        int user_id = authenticationService.getUser(token).getId();
        List<WishList> body = wishListService.readWishList(user_id);
        List<ProductDto> products = new ArrayList<ProductDto>();
        for (WishList wishList : body) {
            products.add(ProductService.getDtoFromProduct(wishList.getProduct()));
        }
        return new ResponseEntity<List<ProductDto>>(products, HttpStatus.OK);
    }
    @PostMapping("/add")
    @CrossOrigin("http://localhost:8001")
    public ResponseEntity<ApiResponse> addWishList(@RequestBody Product product, @RequestParam("token") String token) {
        authenticationService.authenticate(token);
        User user = authenticationService.getUser(token);
        WishList wishList = new WishList(user, product);
        wishListService.createWishlist(wishList);
        return new ResponseEntity<ApiResponse>(new ApiResponse(true, "Add to wishlist"), HttpStatus.CREATED);
    }
}
// FileUploadController
@RestController
@RequestMapping("/fileUpload")
@CrossOrigin("http://localhost:8001")
public class FileUploadController {
  @Autowired
  FIleStoreService fileStoreService;
  @PostMapping("/")
  @CrossOrigin("http://localhost:8001")
  public String handleFileUpload(@RequestParam("file") MultipartFile file) {
    return fileStoreService.store(file);
  }
  @GetMapping("/")
  @CrossOrigin("http://localhost:8001")
  public ResponseEntity<List<FileInfo>> getListFiles() {
    List<FileInfo> fileInfos = fileStoreService.loadAll().map(path -> {
      String filename = path.getFileName().toString();
      String url = MvcUriComponentsBuilder
          .fromMethodName(FileUploadController.class, "getFile", path.getFileName().toString()).build().toString();
      return new FileInfo(filename, url);
    }).collect(Collectors.toList());
    Stream<Path> pathStream = fileStoreService.loadAll();
    return ResponseEntity.status(HttpStatus.OK).body(fileInfos);
  }
  @GetMapping("/files/{filename:.+}")
  @CrossOrigin("http://localhost:8001")
  public ResponseEntity<Resource> getFile(@PathVariable String filename) {
    Resource file = fileStoreService.load(filename);
    return ResponseEntity.ok()
        .header(HttpHeaders.CONTENT_DISPOSITION, "attachment; filename=\"" + file.getFilename() + "\"").body(file);

  }
}
```

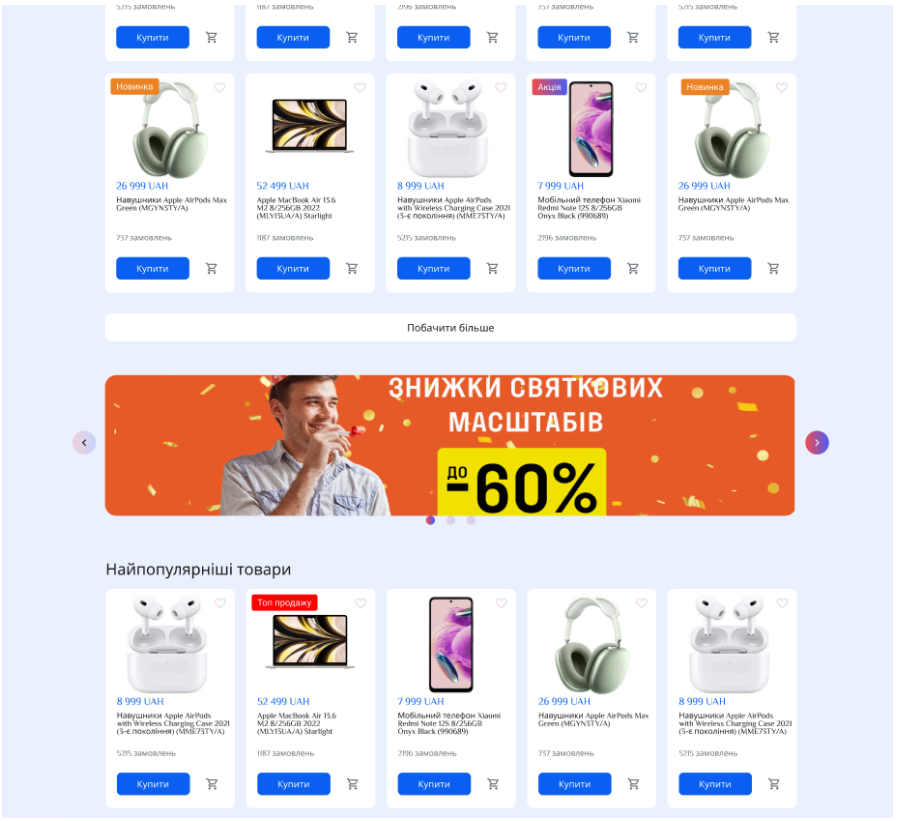
HTML код головної сторінки магазину

```
<template>
  <div id="home">
    <!-- Page Wrapper -->
    <div id="background-div" class="page-holder bg-cover">
      <div class="container py-5">
        <header class="text-left text-white py-5">
          <h3 class="mb-4 rounded"><a href="#start-shopping" class="bg-white px-2 py-2 rounded" id="heading">Твоя техніка мрії - ближче, ніж ти думаєш! Починай купувати прямо зараз!</a></h3>
          <p id="content" class="lead mb-0 bg-dark p-1 rounded">Відкрий для себе світ інноваційних гаджетів та електроніки з Electro-store.
            Приєднуйтесь до спільноти техно-ентузіастів та отримуйте ексклюзивні пропозиції!</p>
        </header>
      </div>
    </div>
    <div id="start-shopping" class="container">
      <div class="row">
        <div class="col-12 text-left">
          <h2 class="pt-3">Найпопулярніші категорії</h2>
        </div>
      </div>
      <div class="row">
        <div v-for="index in this.category_size" :key="index" class="col-md-6 col-xl-4 col-12 pt-3  justify-content-around d-flex">
          <CategoryBox :category="categories[index-1]">
          </CategoryBox>
        </div>
      </div>
    </div>
    <hr>
    <div class="container">
      <div class="row">
        <div class="col-12 text-left">
          <h2 class="pt-3">Кращі товари</h2>
        </div>
      </div>
      <div class="row">
        <div v-for="index in this.product_size" :key="index" class="col-md-6 col-xl-4 col-12 pt-3  justify-content-around d-flex">
          <ProductBox :product="products[index-1]">
          </ProductBox>
        </div>
      </div>
    </div>
  </div>
</template>
```

ДОДАТОК В

Сторінки прототипів магазину зроблені у Figma





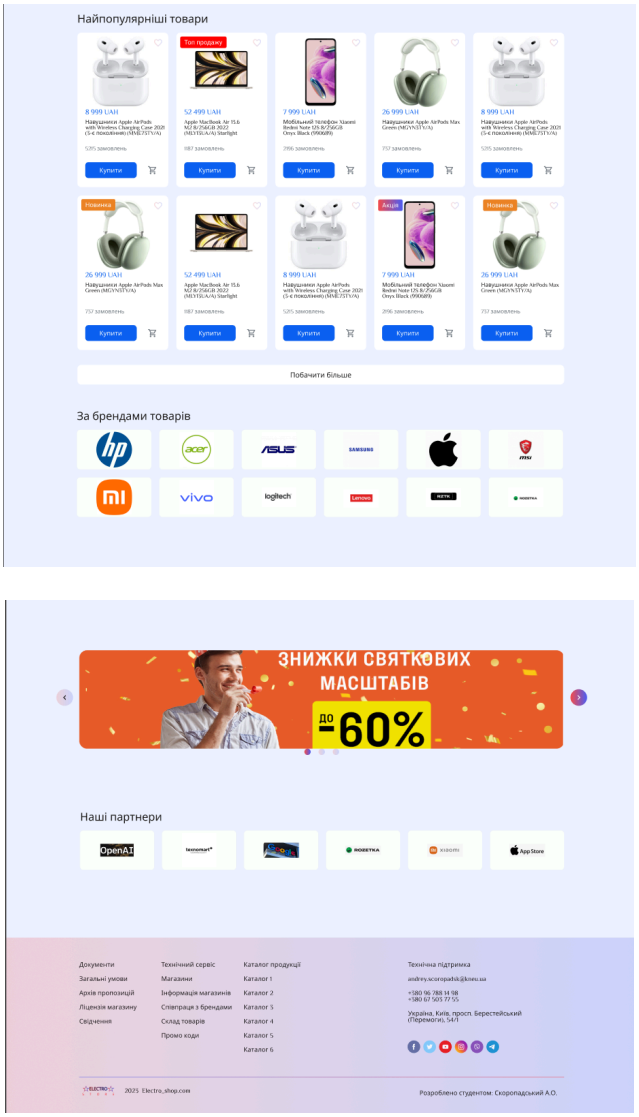


Рис. В.1 – Головна сторінка магазину, коли користувач не зареєстрований

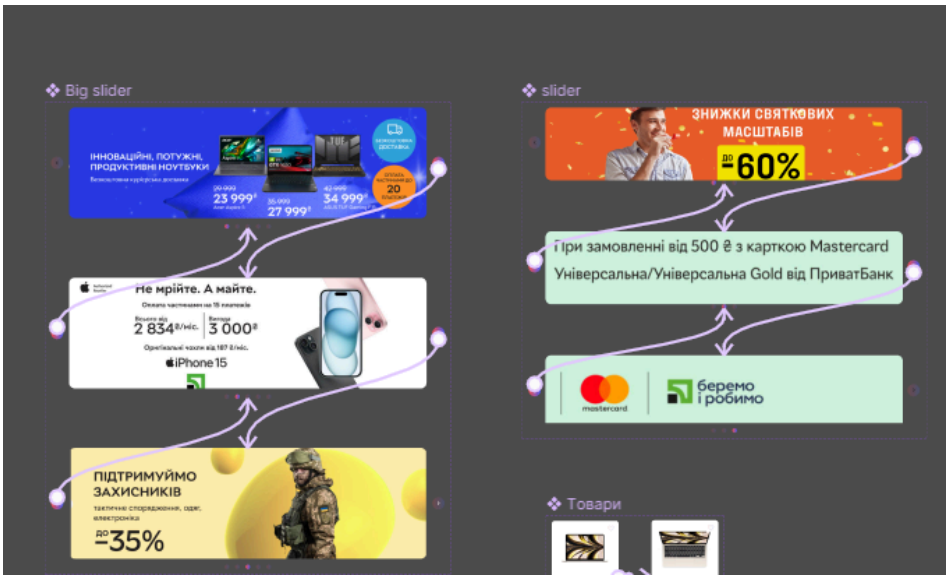


Рис. В.2 – Компонети для слайдерів

Прототипи товарів, з динамічною анімацією:

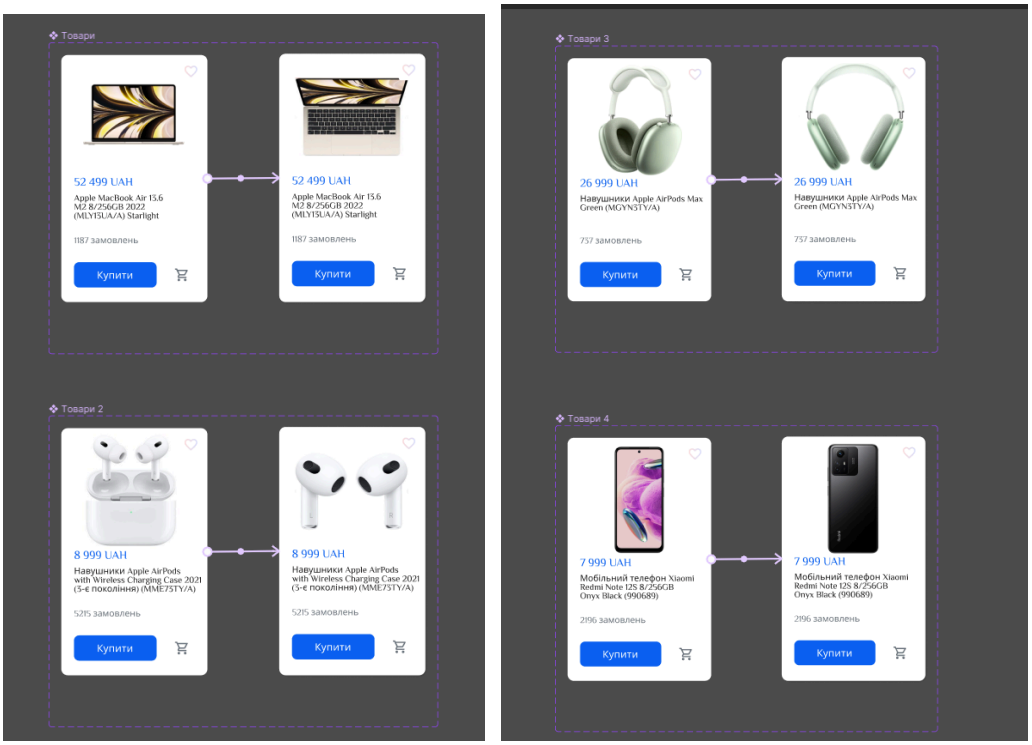


Рис. В.3-В.4 – Компонети для товарів, де при наведенні мижкою буде змінюватися ракурс товару на ішну картинку

Рис. В.5 – Прототипи для категорій та підкатегорій

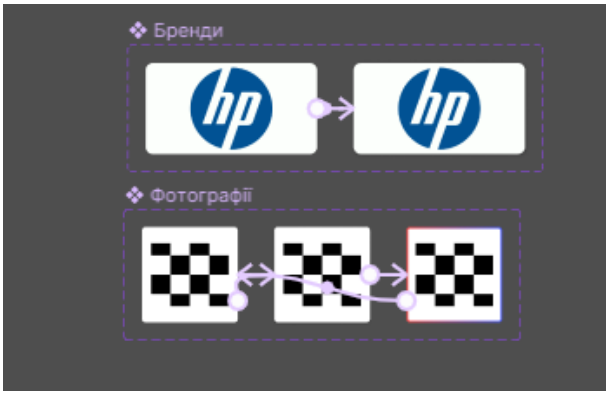
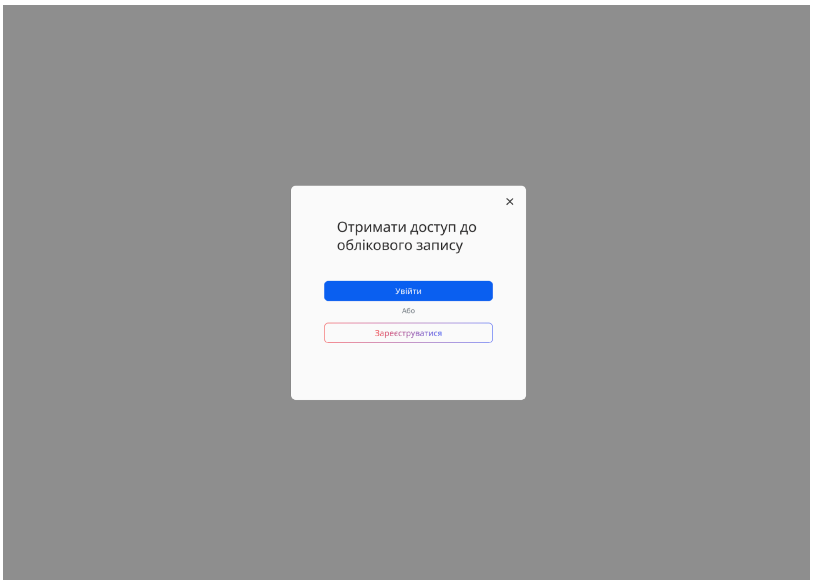


Рис. В.6 – Прототипи для брендів, партнерів



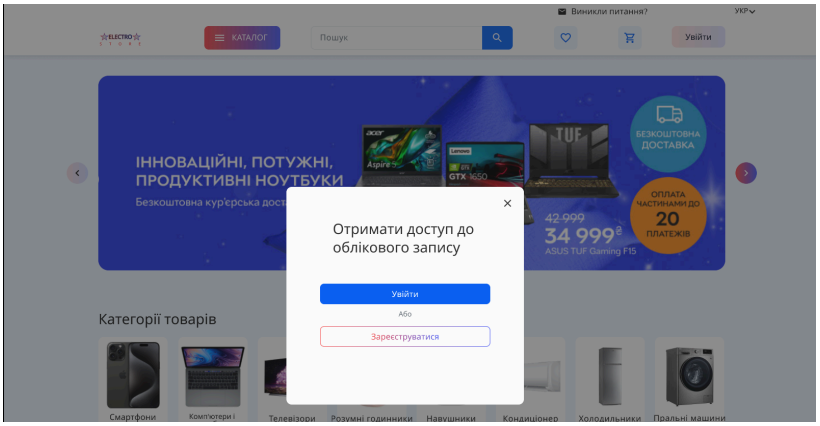


Рис. В.7-В.8 – Реалізована електронна форма авторизації, при натискання на кнопку “Увійти”, буди випадати форма з авторизацією чи реєстрацією

Далі, при наступних діях буде також вискакувати форма для взаємодії: Натискаючи на кнопку “Увійти”:

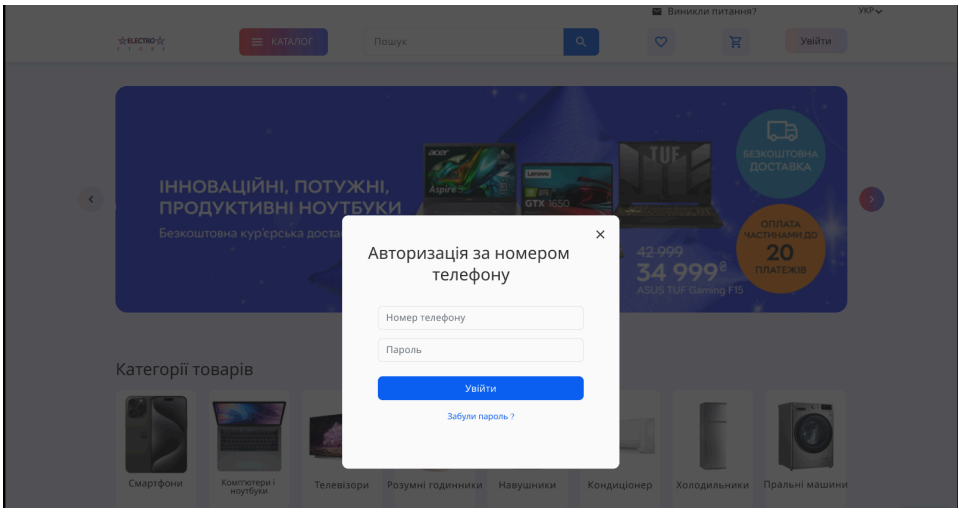


Рис. В.9 – Авторизація в систему

Якщо користувач не пам’ятає пароль:

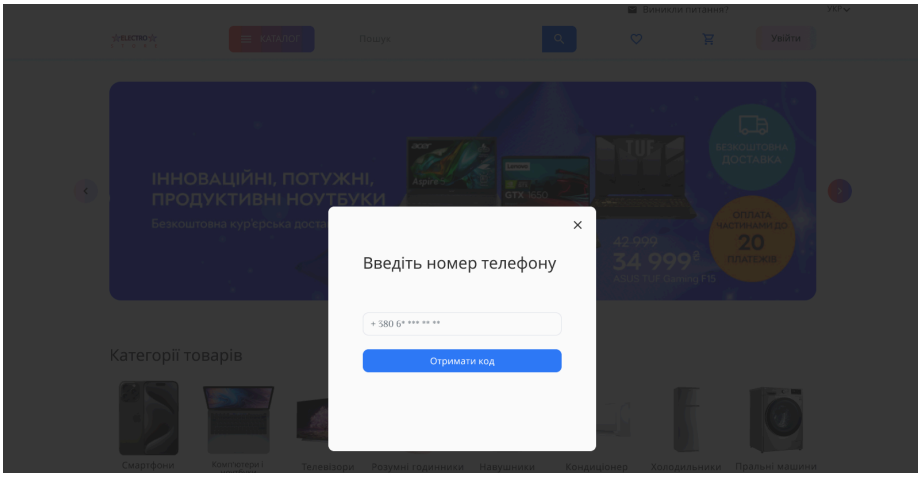


Рис. В.10 – Форма, якщо користувач забув пароль від акаунту

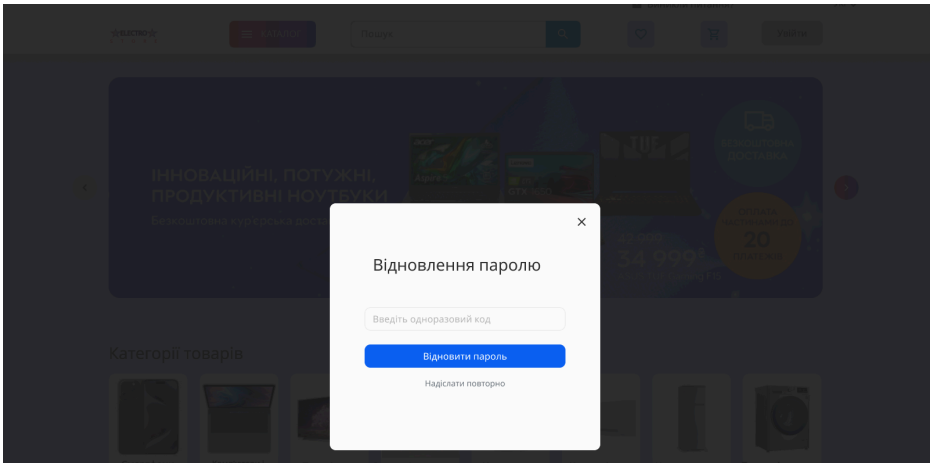


Рис. В.11 – Після введення одноразового коду користувач попаде до системи

При реєстрації нового користувача, йому необхідно ввести свій номер телефону:

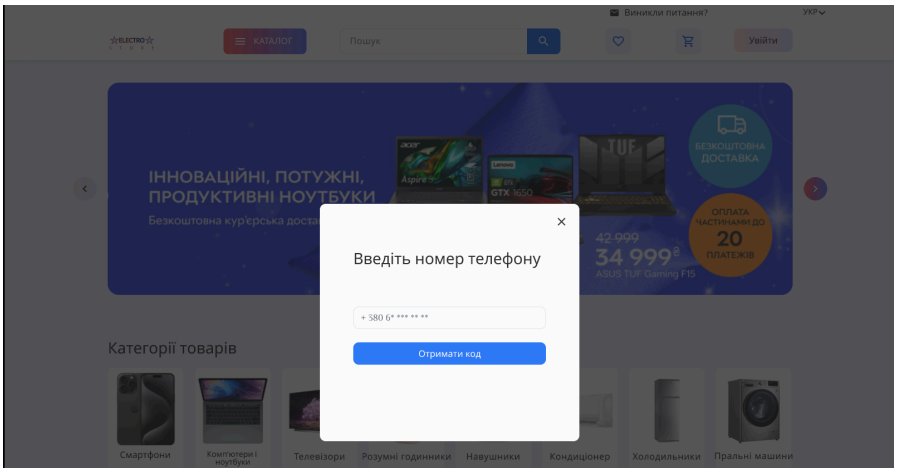


Рис. В.12 – Реєстрація в систему

Далі, необхідно ввести одноразовий пароль до нашого магазину, після чого заповнити дані в особистому кабінеті, та створивши свій власний пароль, якщо пароль не отримано, форма оновиться і прийде новий пароль:

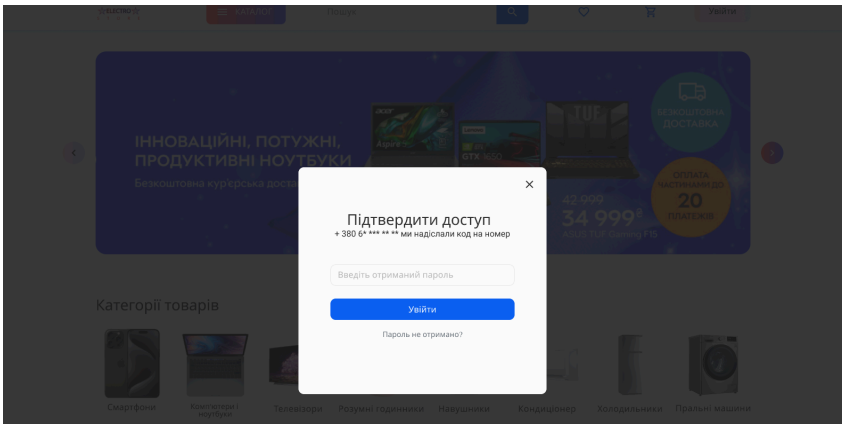
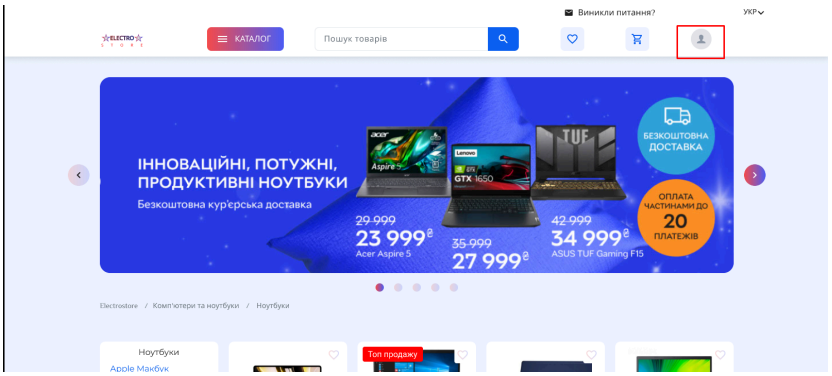


Рис. В.13 – Вхід в систему з одноразовим паролем для реєстрації



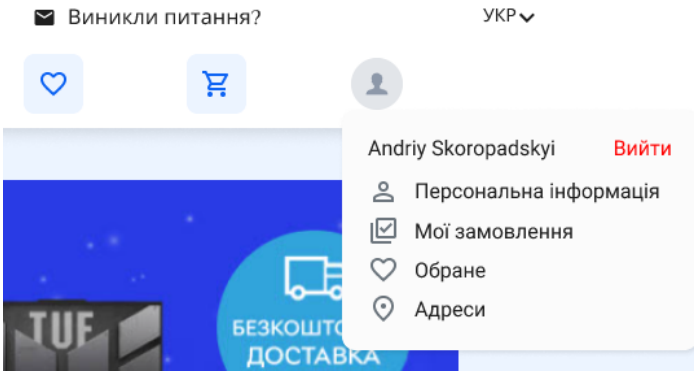


Рис. В.14-В.15 – Авторизований користувач та випадне меню

Після входу в аккаунт, або реєстрації нового, нам стає доступним особистий кабінет та подальші операції.

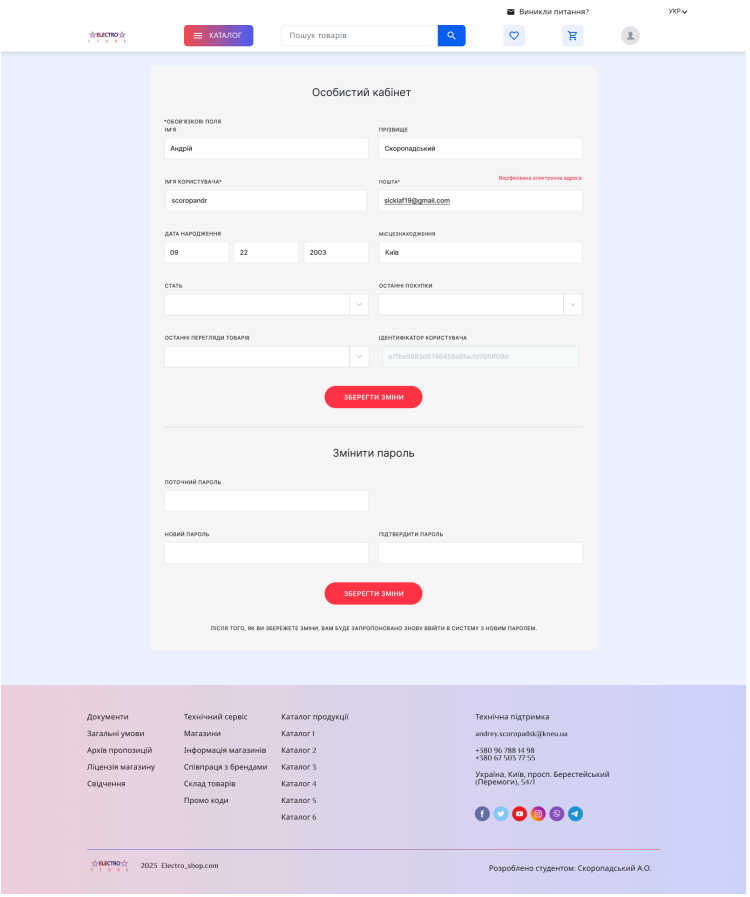


Рис. В.16 – Особистий кабінет користувача, де він може редагувати дані, переглядати останні розглянуті товари, остані покупки, підтверджувати пошту та змінювати пароль до особистого кабінету

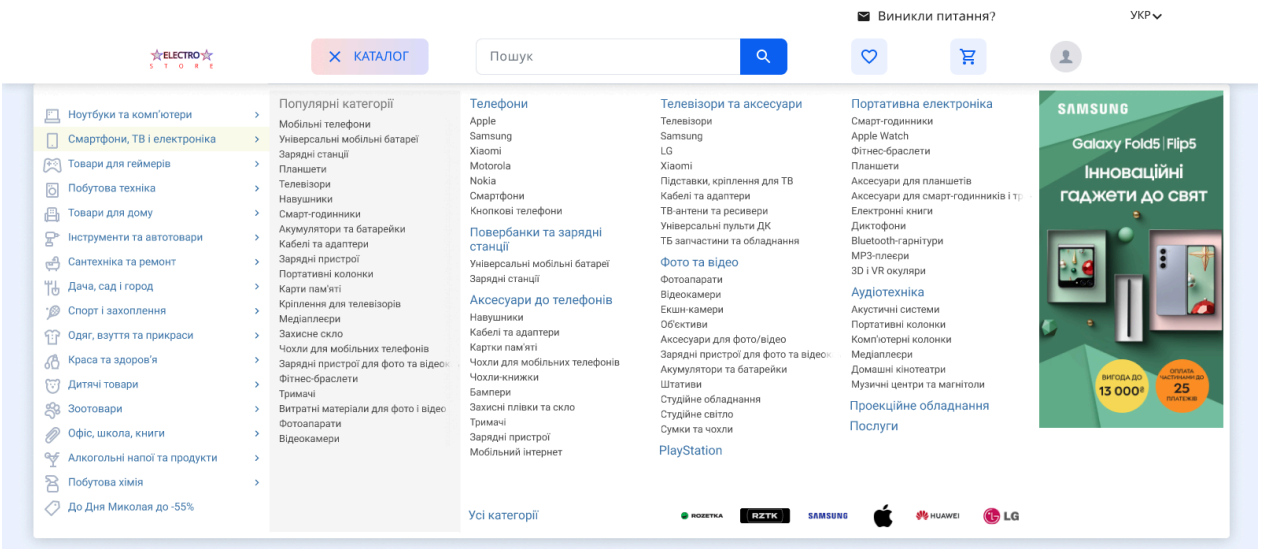


Рис. В.17 – Випадне меню, при натискані кнопки каталогу, а також анімована кнопка “Каталог”

Структура сторінки з категоріями товарів спроектована для забезпечення зручності користувачів та ефективного пошуку необхідних товарів. Зверху також видно ієрархію, після написання на категорію, відкриватися інші підкатегорії.

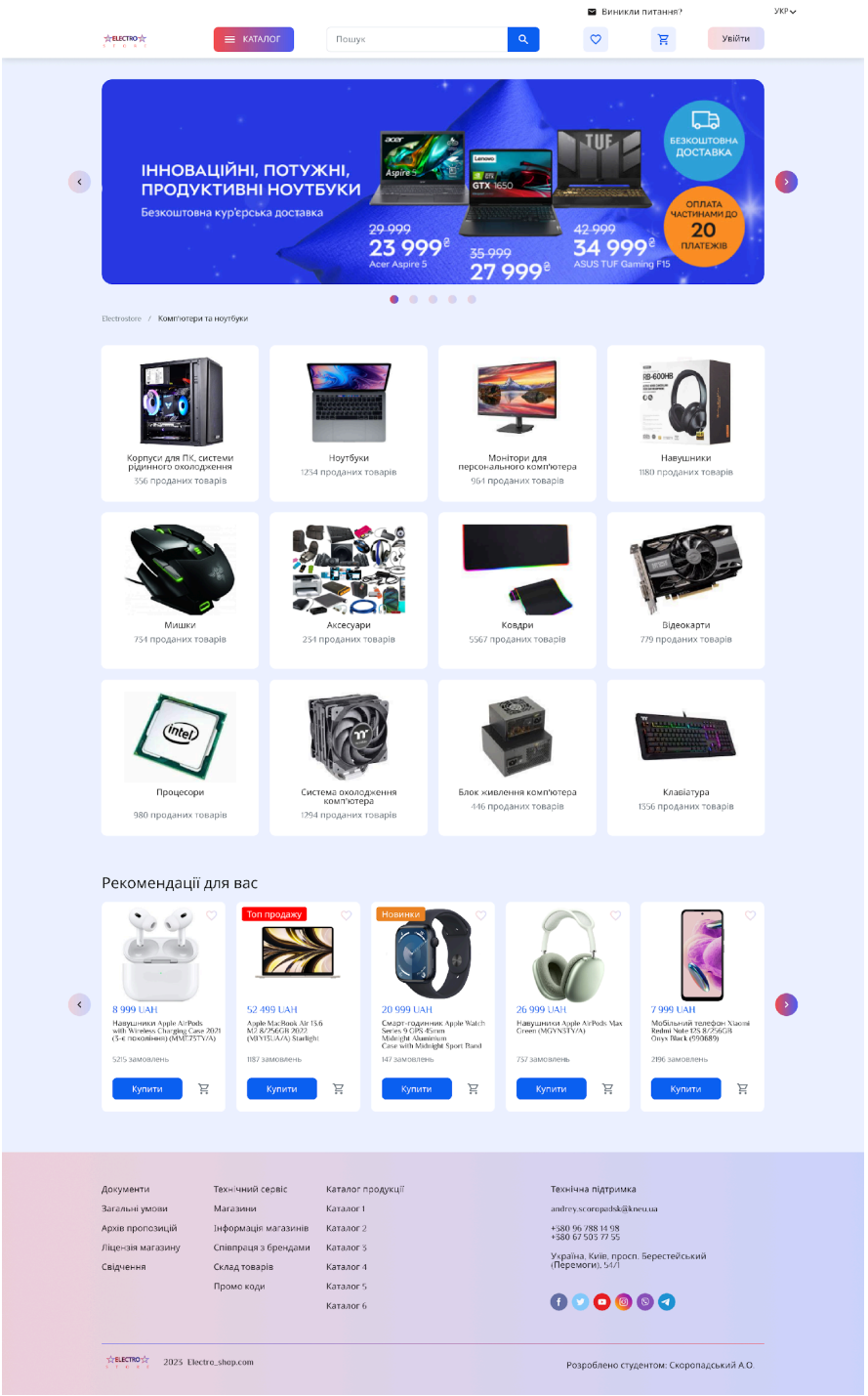


Рис. В.18 – Категорії товарів

Також, для товарів на головній сторінці зроблено підкатегорії:

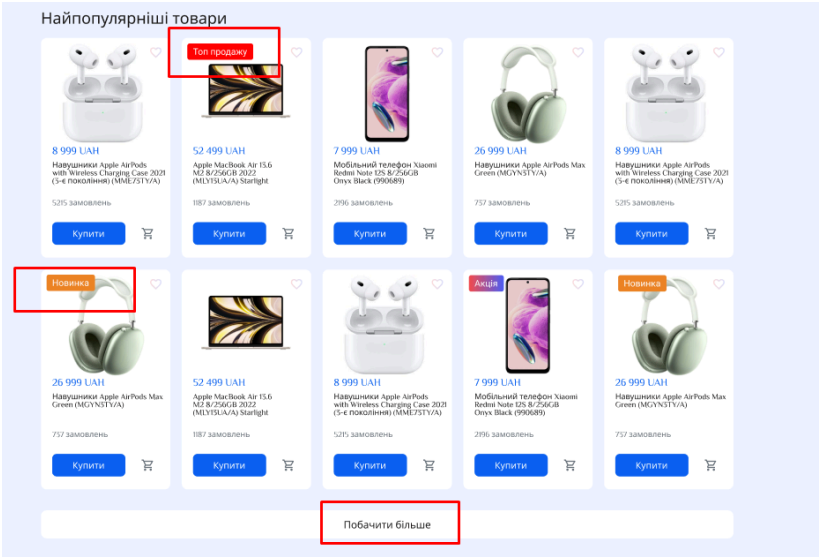


Рис. В.19 – Найпопулярніші товари

Аніміція для додавання в “Обране”:

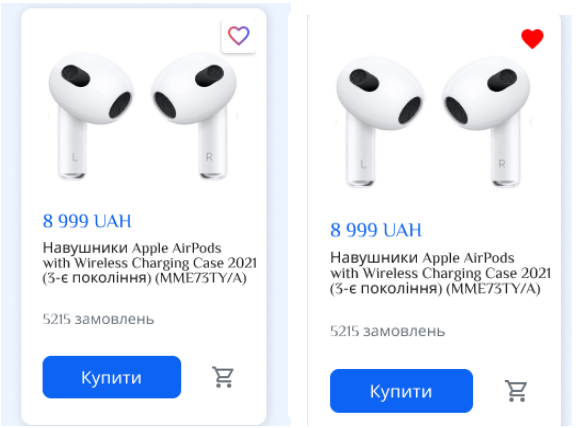


Рис. В.22 – Додати товар в обране

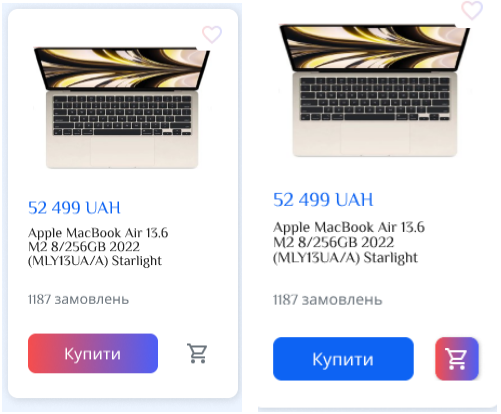


Рис. В.20-В.21 – Додати товар в корзину та купити в 2 кліки

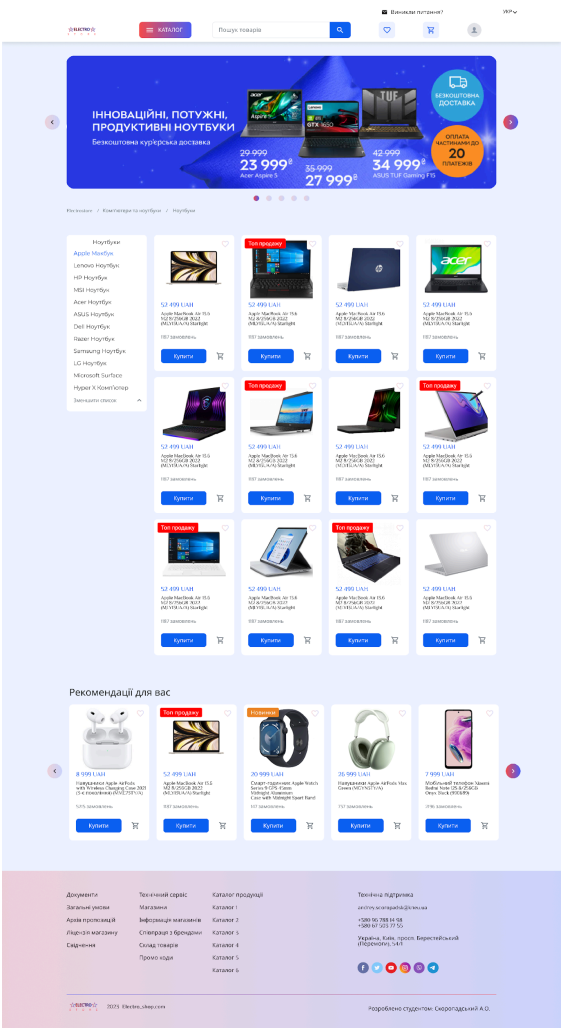


Рис. В.22 – Підкатегорія “Ноутбуки” з роздвижним листом з видами, брендами ноутбуків, можливо попасти на цю сторіку через пошук, наприклад ввести “Ноутбуки”, чи “Hp ноутбук”, чи просто назву товару

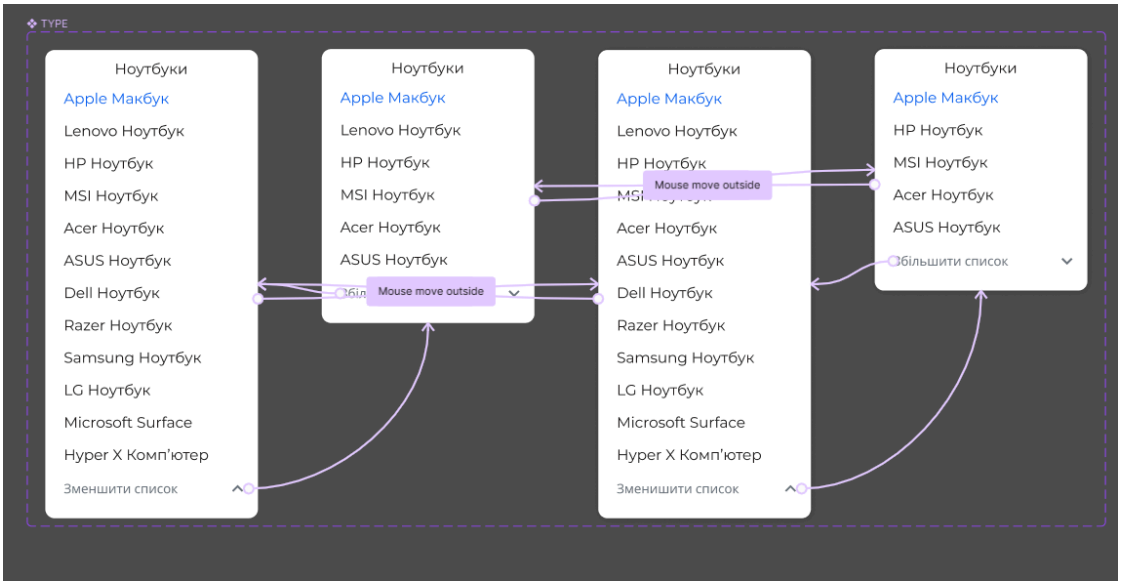


Рис. В.23 – Компонети та анімація роздвижного листа брендів ноутбуків

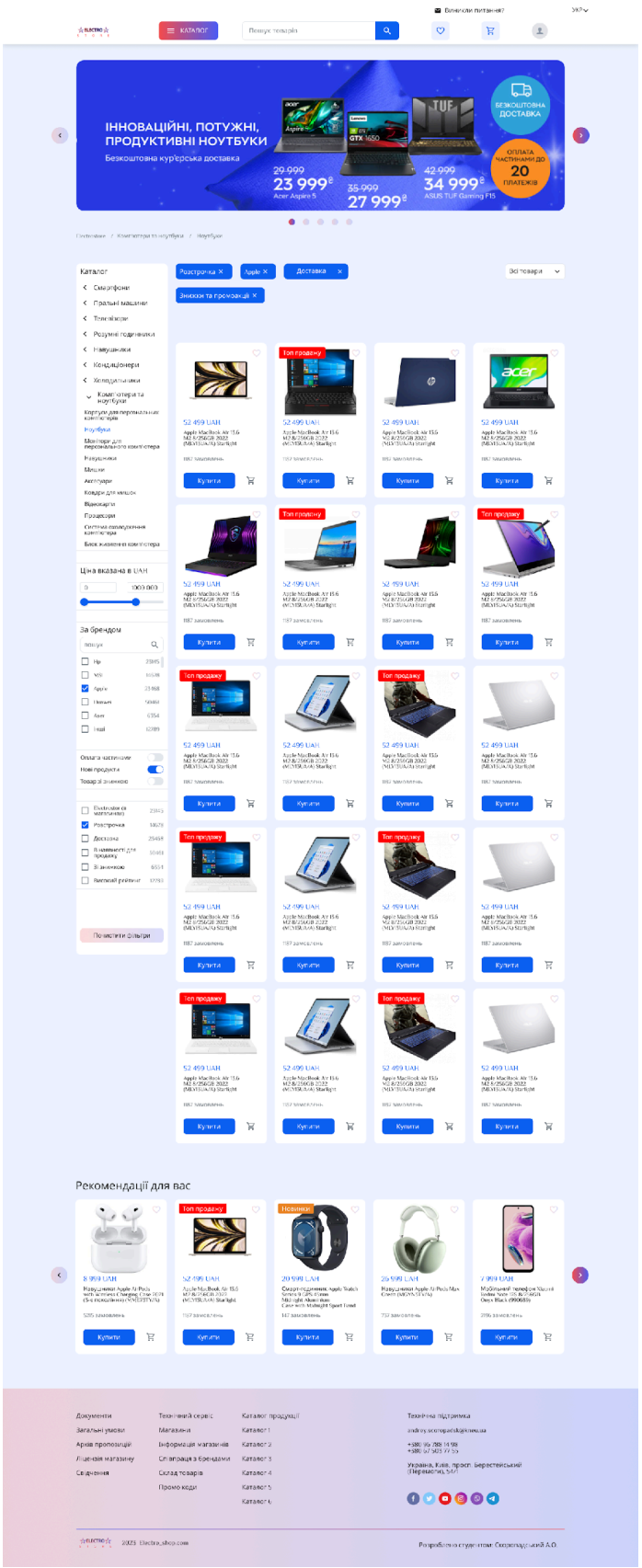


Рис. В.24 – Сторінка вже з обраною фільрацією (з параметром цін, за брендом, за типом товару, акційний, з доставкою і тд.)

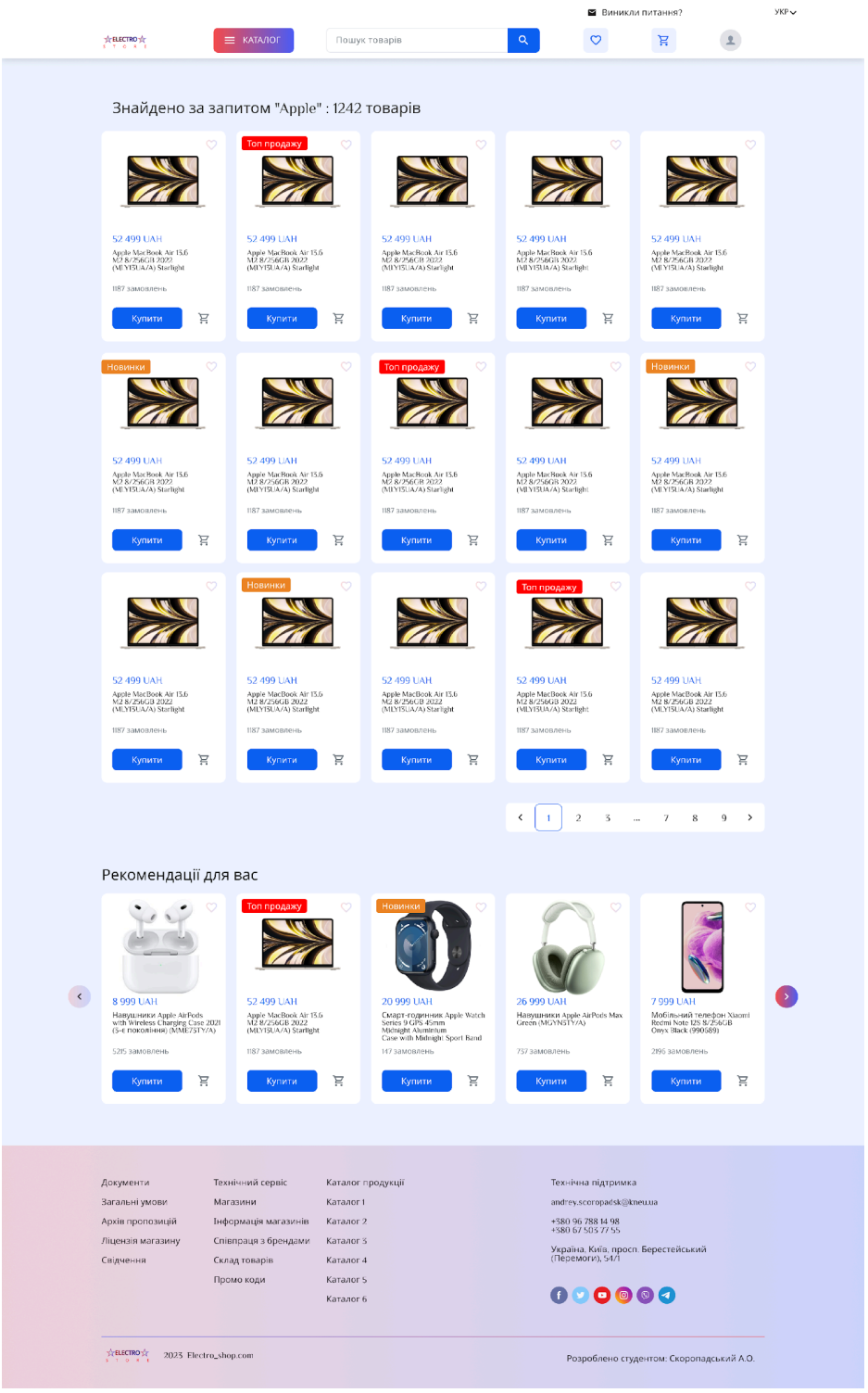


Рис. В.25 – Пошук за брендом товару, наприклад “Apple”

Сторінка з обраним товаром і інформацією про товар (сторінка дуже велика, ному в декілька скринів):

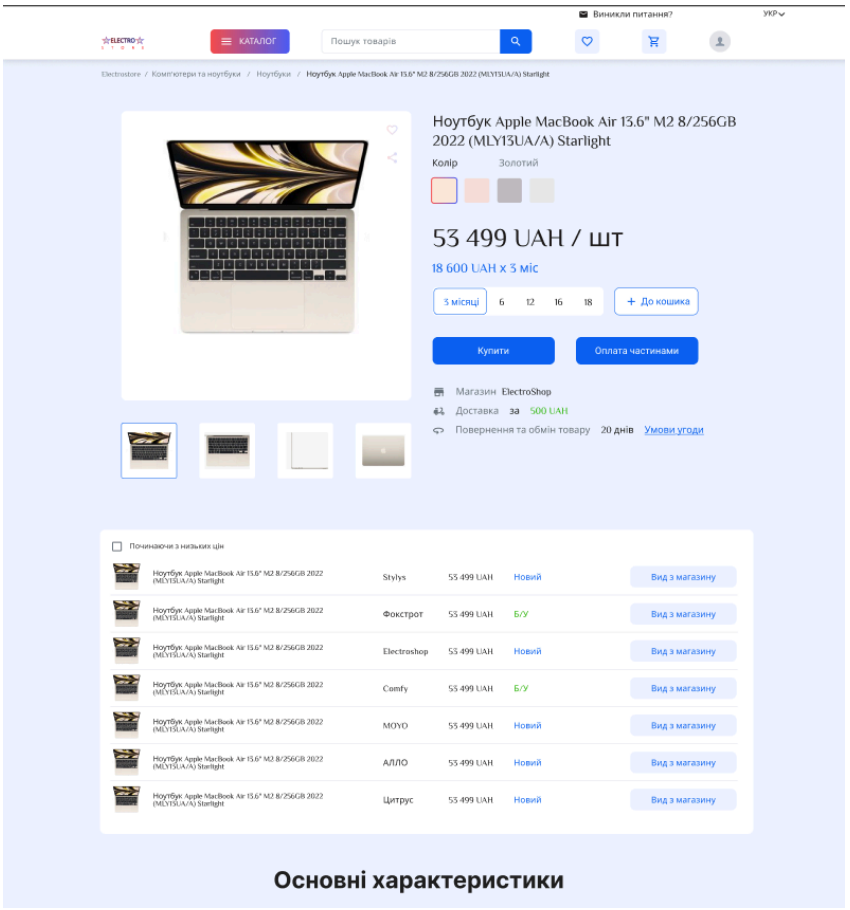


Рис. В.26 – Сторінка опису товару, де можна купити, в розстрочку, порівняти з ціною інших магазинів, обрати колір та коплектацію товару, та додати в кошик

Також, для спрощення перегляду інформації на цій сторінці реалізовано 2 скроли, щоб швидко переміщатися до потрібної користувачеві інформації.

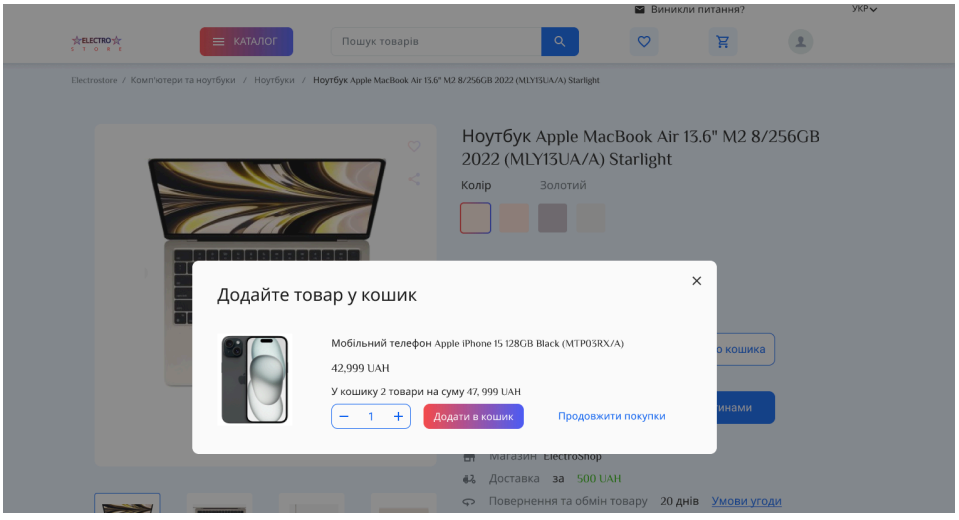


Рис. В.27 – Випадна форма додавання товару до кошику

При необхідності користувачу знайти фізничний магазин, можна відкрити в випадаючому меню “Адреси” і тоді з’явиться форма з мапою чи списком тп є пошук:

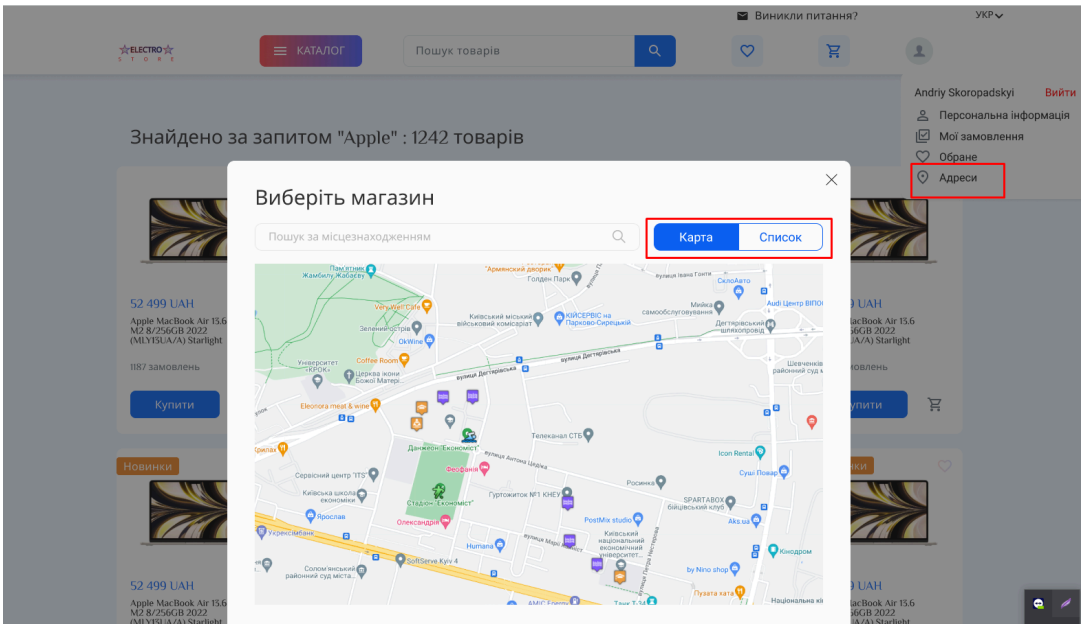


Рис. В.28 – Випадна форма з “Адреси”, де можна подивитися на мапу магазинів або на список з назвами вулиць

ДОДАТОК Г

Результат перетворення ER-моделі в модель даних СУБД MySQL

Згенерований код:

```
SET @OLD_UNIQUE_CHECKS=@@UNIQUE_CHECKS, UNIQUE_CHECKS=0;
SET @OLD_FOREIGN_KEY_CHECKS=@@FOREIGN_KEY_CHECKS, FOREIGN_KEY_CHECKS=0;
SET @OLD_SQL_MODE=@@SQL_MODE, SQL_MODE='ONLY_FULL_GROUP_BY,STRICT_TRANS_TABLES,NO_ZERO_IN_DATE,NO_ZERO_DATE,ERROR_FOR_DIVISION_BY_ZERO,NO_ENGINE_SUBSTITUTION';
-- -----------------------------------------------------
-- Schema mydb
-- -----------------------------------------------------
SHOW WARNINGS;
-- -----------------------------------------------------
-- Schema ecommerce
-- -----------------------------------------------------
DROP SCHEMA IF EXISTS `ecommerce` ;

-- -----------------------------------------------------
-- Schema ecommerce
-- -----------------------------------------------------
CREATE SCHEMA IF NOT EXISTS `ecommerce` DEFAULT CHARACTER SET utf8mb4 COLLATE utf8mb4_0900_ai_ci ;
SHOW WARNINGS;
USE `ecommerce` ;
-- -----------------------------------------------------
-- Table `cart`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `cart` ;
SHOW WARNINGS;
CREATE TABLE IF NOT EXISTS `cart` (
  `id` INT NOT NULL AUTO_INCREMENT,
  `created_date` DATETIME(6) NULL DEFAULT NULL,
  `quantity` INT NOT NULL,
  `product_id` INT NULL DEFAULT NULL,
  `user_id` INT NOT NULL,
  PRIMARY KEY (`id`),
  CONSTRAINT `FKg5uhi8vpsuy0lgloxk2h4w5o6`
    FOREIGN KEY (`user_id`)
    REFERENCES `users` (`id`),
  CONSTRAINT `FKpu4bcbluhsxagirmbdn7dilm5`
    FOREIGN KEY (`product_id`)
    REFERENCES `products` (`id`))
ENGINE = InnoDB
DEFAULT CHARACTER SET = utf8mb4
COLLATE = utf8mb4_0900_ai_ci;
SHOW WARNINGS
CREATE INDEX `FKpu4bcbluhsxagirmbdn7dilm5` ON `cart` (`product_id` ASC) VISIBLE;
SHOW WARNINGS;
```

```
CREATE INDEX `FKg5uhi8vpsuy0lgloxk2h4w5o6` ON `cart` (`user_id` ASC) VISIBLE;
SHOW WARNINGS;
-- -----------------------------------------------------
-- Table `categories`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `categories` ;
SHOW WARNINGS;
CREATE TABLE IF NOT EXISTS `categories` (
  `id` INT NOT NULL AUTO_INCREMENT,
  `category_name` VARCHAR(255) NULL DEFAULT NULL,
  `description` VARCHAR(255) NULL DEFAULT NULL,
  `image_url` VARCHAR(255) NULL DEFAULT NULL,
  PRIMARY KEY (`id`))
ENGINE = InnoDB
DEFAULT CHARACTER SET = utf8mb4
COLLATE = utf8mb4_0900_ai_ci;
SHOW WARNINGS;
-- -----------------------------------------------------
-- Table `orderitems`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `orderitems` ;
SHOW WARNINGS;
CREATE TABLE IF NOT EXISTS `orderitems` (
  `id` INT NOT NULL AUTO_INCREMENT,
  `created_date` DATETIME(6) NULL DEFAULT NULL,
  `price` DOUBLE NULL DEFAULT NULL,
  `quantity` INT NULL DEFAULT NULL,
  `order_id` INT NULL DEFAULT NULL,
  `product_id` INT NULL DEFAULT NULL,
  PRIMARY KEY (`id`),
  CONSTRAINT `FKm3mp87f5ygbbfuqfdhc09y9a`
    FOREIGN KEY (`order_id`)
    REFERENCES `orders` (`id`),
  CONSTRAINT `FKqu7dfdpbhltsgxfq2ahkdnyv5`
    FOREIGN KEY (`product_id`)
    REFERENCES `products` (`id`))
ENGINE = InnoDB
DEFAULT CHARACTER SET = utf8mb4
COLLATE = utf8mb4_0900_ai_ci;
SHOW WARNINGS;
CREATE INDEX `FKm3mp87f5ygbbfuqfdhc09y9a` ON `orderitems` (`order_id` ASC) VISIBLE;
SHOW WARNINGS;
CREATE INDEX `FKqu7dfdpbhltsgxfq2ahkdnyv5` ON `orderitems` (`product_id` ASC) VISIBLE;
SHOW WARNINGS;
-- -----------------------------------------------------
-- Table `orders`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `orders` ;
SHOW WARNINGS;
CREATE TABLE IF NOT EXISTS `orders` (
  `id` INT NOT NULL AUTO_INCREMENT,;
  `created_date` DATETIME(6) NULL DEFAULT NULL,
  `session_id` VARCHAR(255) NULL DEFAULT NULL,
```

```
  `total_price` DOUBLE NULL DEFAULT NULL,
  `user_id` INT NULL DEFAULT NULL,
  PRIMARY KEY (`id`),
  CONSTRAINT `FK32ql8ubntj5uh44ph9659tiih`
    FOREIGN KEY (`user_id`)
    REFERENCES `users` (`id`))
ENGINE = InnoDB
DEFAULT CHARACTER SET = utf8mb4
COLLATE = utf8mb4_0900_ai_ci;
SHOW WARNINGS;
CREATE INDEX `FK32ql8ubntj5uh44ph9659tiih` ON `orders` (`user_id` ASC) VISIBLE;
SHOW WARNINGS;
-- -----------------------------------------------------
-- Table `products`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `products` ;
SHOW WARNINGS;
CREATE TABLE IF NOT EXISTS `products` (
  `id` INT NOT NULL AUTO_INCREMENT,
  `description` VARCHAR(255) NULL DEFAULT NULL,
  `imageurl` VARCHAR(255) NULL DEFAULT NULL,
  `name` VARCHAR(255) NULL DEFAULT NULL,
  `price` DOUBLE NOT NULL,
  `category_id` INT NOT NULL,
  PRIMARY KEY (`id`),
  CONSTRAINT `FKog2rp4qthbtt2lfyhfo32lsw9`
    FOREIGN KEY (`category_id`)
    REFERENCES `categories` (`id`))
ENGINE = InnoDB
DEFAULT CHARACTER SET = utf8mb4
COLLATE = utf8mb4_0900_ai_ci;
SHOW WARNINGS;
CREATE INDEX `FKog2rp4qthbtt2lfyhfo32lsw9` ON `products` (`category_id` ASC) VISIBLE;
SHOW WARNINGS;
-- -----------------------------------------------------
-- Table `tokens`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `tokens` ;
SHOW WARNINGS;
CREATE TABLE IF NOT EXISTS `tokens` (
  `id` INT NOT NULL AUTO_INCREMENT,
  `created_date` DATETIME(6) NULL DEFAULT NULL,
  `token` VARCHAR(255) NULL DEFAULT NULL,
  `user_id` INT NOT NULL,
  PRIMARY KEY (`id`),
  CONSTRAINT `FK2dylsfo39lgjyqml2tbe0b0ss`
    FOREIGN KEY (`user_id`)
    REFERENCES `users` (`id`))
ENGINE = InnoDB
AUTO_INCREMENT = 2
DEFAULT CHARACTER SET = utf8mb4
COLLATE = utf8mb4_0900_ai_ci;
SHOW WARNINGS;
```

```
CREATE INDEX `FK2dylsfo39lgjyqml2tbe0b0ss` ON `tokens` (`user_id` ASC) VISIBLE;
SHOW WARNINGS;
-- -----------------------------------------------------
-- Table `users`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `users` ;
SHOW WARNINGS;
CREATE TABLE IF NOT EXISTS `users` (
  `id` INT NOT NULL AUTO_INCREMENT,
  `email` VARCHAR(255) NULL DEFAULT NULL,
  `first_name` VARCHAR(255) NULL DEFAULT NULL,
  `last_name` VARCHAR(255) NULL DEFAULT NULL,
  `password` VARCHAR(255) NULL DEFAULT NULL,
  `role` VARCHAR(255) NULL DEFAULT NULL,
  PRIMARY KEY (`id`))
ENGINE = InnoDB
AUTO_INCREMENT = 2
DEFAULT CHARACTER SET = utf8mb4
COLLATE = utf8mb4_0900_ai_ci;
SHOW WARNINGS;
-- -----------------------------------------------------
-- Table `wishlist`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `wishlist` ;
SHOW WARNINGS;
CREATE TABLE IF NOT EXISTS `wishlist` (
  `id` INT NOT NULL AUTO_INCREMENT,
  `created_date` DATETIME(6) NULL DEFAULT NULL,
  `product_id` INT NULL DEFAULT NULL,
  `user_id` INT NOT NULL,
  PRIMARY KEY (`id`),
  CONSTRAINT `FK6p7qhvy1bfkri13u29x6pu8au`
    FOREIGN KEY (`product_id`)
    REFERENCES `products` (`id`),
  CONSTRAINT `FKtrd6335blsefl2gxpb8lr0gr7`
    FOREIGN KEY (`user_id`)
    REFERENCES `users` (`id`))
ENGINE = InnoDB
DEFAULT CHARACTER SET = utf8mb4
COLLATE = utf8mb4_0900_ai_ci;
SHOW WARNINGS;
CREATE INDEX `FK6p7qhvy1bfkri13u29x6pu8au` ON `wishlist` (`product_id` ASC) VISIBLE;
SHOW WARNINGS;
CREATE INDEX `FKtrd6335blsefl2gxpb8lr0gr7` ON `wishlist` (`user_id` ASC) VISIBLE;
SHOW WARNINGS;
SET SQL_MODE=@OLD_SQL_MODE;
SET FOREIGN_KEY_CHECKS=@OLD_FOREIGN_KEY_CHECKS;
SET UNIQUE_CHECKS=@OLD_UNIQUE_CHECKS;
COLLATE = utf8mb4_0900_ai_ci;
SHOW WARNINGS;
CREATE INDEX `FK2dylsfo39lgjyqml2tbe0b0ss` ON `tokens` (`user_id` ASC) VISIBLE;
SHOW WARNINGS;
-- -----------------------------------------------------
```

```
-- Table `users`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `users` ;
SHOW WARNINGS;
CREATE TABLE IF NOT EXISTS `users` (
  `id` INT NOT NULL AUTO_INCREMENT,
  `email` VARCHAR(255) NULL DEFAULT NULL,
  `first_name` VARCHAR(255) NULL DEFAULT NULL,
  `last_name` VARCHAR(255) NULL DEFAULT NULL,
  `password` VARCHAR(255) NULL DEFAULT NULL,
  `role` VARCHAR(255) NULL DEFAULT NULL,
  PRIMARY KEY (`id`))
ENGINE = InnoDB
AUTO_INCREMENT = 2
DEFAULT CHARACTER SET = utf8mb4
COLLATE = utf8mb4_0900_ai_ci;
SHOW WARNINGS;
-- -----------------------------------------------------
-- Table `wishlist`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `wishlist` ;
SHOW WARNINGS;
CREATE TABLE IF NOT EXISTS `wishlist` (
  `id` INT NOT NULL AUTO_INCREMENT,
  `created_date` DATETIME(6) NULL DEFAULT NULL,
  `product_id` INT NULL DEFAULT NULL,
  `user_id` INT NOT NULL,
  PRIMARY KEY (`id`),
  CONSTRAINT `FK6p7qhvy1bfkri13u29x6pu8au`
    FOREIGN KEY (`product_id`)
    REFERENCES `products` (`id`),
  CONSTRAINT `FKtrd6335blsefl2gxpb8lr0gr7`
    FOREIGN KEY (`user_id`)
    REFERENCES `users` (`id`))
ENGINE = InnoDB
DEFAULT CHARACTER SET = utf8mb4
COLLATE = utf8mb4_0900_ai_ci;
SHOW WARNINGS;
CREATE INDEX `FK6p7qhvy1bfkri13u29x6pu8au` ON `wishlist` (`product_id` ASC) VISIBLE;
SHOW WARNINGS;
CREATE INDEX `FKtrd6335blsefl2gxpb8lr0gr7` ON `wishlist` (`user_id` ASC) VISIBLE;
SHOW WARNINGS;
SET SQL_MODE=@OLD_SQL_MODE;
SET FOREIGN_KEY_CHECKS=@OLD_FOREIGN_KEY_CHECKS;
SET UNIQUE_CHECKS=@OLD_UNIQUE_CHECKS;
```